# A Subpolynomial Approximation Algorithm for Graph Crossing Number in Low-Degree Graphs*

Julia Chuzhoy† Zihan Tan‡

June 29, 2022

**Abstract**

We consider the classical Minimum Crossing Number problem: given an $n$-vertex graph $G$, compute a drawing of $G$ in the plane, while minimizing the number of crossings between the images of its edges. This is a fundamental and extensively studied problem, whose approximability status is widely open. In all currently known approximation algorithms, the approximation factor depends polynomially on $\Delta$ – the maximum vertex degree in $G$. The best current approximation algorithm achieves an $O(n^{1/2-\epsilon} \cdot \mathrm{poly}(\Delta \cdot \log n))$-approximation, for a small fixed constant $\epsilon$, while the best negative result is APX-hardness, leaving a large gap in our understanding of this basic problem. In this paper we design a randomized $O\left(2^{O((\log n)^{7/8} \log\log n)} \cdot \mathrm{poly}(\Delta)\right)$-approximation algorithm for Minimum Crossing Number. This is the first approximation algorithm for the problem that achieves a subpolynomial in $n$ approximation factor (albeit only in graphs whose maximum vertex degree is subpolynomial in $n$).

In order to achieve this approximation factor, we design a new algorithm for a closely related problem called Crossing Number with Rotation System, in which, for every vertex $v \in V(G)$, the circular ordering, in which the images of the edges incident to $v$ must enter the image of $v$ in the drawing is fixed as part of input. Combining this result with the recent reduction of [Chuzhoy, Mahabadi, Tan '20] immediately yields the improved approximation algorithm for Minimum Crossing Number. We introduce several new technical tools, that we hope will be helpful in obtaining better algorithms for the problem in the future.

*Extended Abstract to appear in STOC 2022.

†Toyota Technological Institute at Chicago. Email: cjulia@ttic.edu. Supported in part by NSF grant CCF-2006464.

‡Computer Science Department, University of Chicago. Email: zihantan@uchicago.edu. Supported in part by NSF grant CCF-2006464.

# Contents

# 1 Introduction

We study the classical Minimum Crossing Number (MCN) problem: given an $n$-vertex graph $G$, compute a drawing of $G$ in the plane while minimizing the number of its crossings. Here, a drawing $\varphi$ of a graph $G$ is a mapping, that maps every vertex $v \in V(G)$ to some point $\varphi(v)$ in the plane, and every edge $e = (u, v) \in E(G)$ to a continuous simple curve $\varphi(e)$, whose endpoints are $\varphi(u)$ and $\varphi(v)$. For a vertex $v \in V(G)$ and an edge $e \in E(G)$, we refer to $\varphi(v)$ and to $\varphi(e)$ as the *images* of $v$ and of $e$, respectively. We require that, for every vertex $v$ and edge $e$, $\varphi(v) \in \varphi(e)$ only if $v$ is an endpoint of $e$. We also require that, if some point $p$ belongs to the images of three or more edges, then it must be the image of a shared endpoint of these edges. A *crossing* in a drawing $\varphi$ of $G$ is a point that belongs to the images of two edges of $G$, and is not their common endpoint. The *crossing number* of a graph $G$, denoted by $\mathsf{OPT}_{\mathsf{cr}}(G)$, is the minimum number of crossings in any drawing of $G$ in the plane.

The MCN problem was initially introduced by Turán [Tur77] in 1944, and has been extensively studied since then (see, e.g., [Chu11, CMS11, CH11, CS13, KS17, KS19, CMT20], and also [RS09, PT00, Mat02, Vrt, Sch12] for excellent surveys). The problem is of interest to several communities, including, for example, graph theory and algorithms, and graph drawing. As such, much effort was invested into studying it from different angles. But despite all this work, most aspects of the problem are still poorly understood.

In this paper we focus on the algorithmic aspect of MCN. Since the problem is NP-hard [GJ83], and it remains NP-hard even in cubic graphs [Hli06, Cab13], it is natural to consider approximation algorithms for it. Unfortunately, the approximation ratios of all currently known algorithms depend polynomially on $\Delta$, the maximum vertex degree of the input graph. To the best of our knowledge, no non-trivial approximation algorithms are known for the general setting, where $\Delta$ may be arbitrarily large. One of the most famous results in this area, the Crossing Number Inequality, by Ajtai, Chvátal, Newborn and Szemerédi [ACNS82] and by Leighton [Lei83], shows that, for every graph $G$ with $|E(G)| \geq 4|V(G)|$, the crossing number of $G$ is $\Omega(|E(G)|^3/|V(G)|^2)$. Since the problem is most interesting when the crossing number of the input graph is low, it is reasonable to focus on low-degree graphs, where the maximum vertex degree $\Delta$ is bounded by either a constant, or a slowly-growing (e.g. subpolynomial) function of $n$. While we do not make such an assumption explicitly, like in all previous work, the approximation factor that we achieve also depends polynomially on $\Delta$.

Even in this setting, there is still a large gap in our understanding of the problem's approximability, and the progress in closing this gap has been slow. On the negative side, only APX-hardness is known [Cab13, AMS07], that holds even in cubic graphs. On the positive side, the first non-trivial approximation algorithm for MCN was obtained by Leighton and Rao in their seminal paper [LR99]. Given as input an $n$-vertex graph $G$, the algorithm computes a drawing of $G$ with at most $O((n + \mathsf{OPT}_{\mathsf{cr}}(G)) \cdot \Delta^{O(1)} \log^4 n)$ crossings. This bound was later improved to $O((n+\mathsf{OPT}_{\mathsf{cr}}(G)) \cdot \Delta^{O(1)} \log^3 n)$ by [EGS02], and then to $O((n + \mathsf{OPT}_{\mathsf{cr}}(G)) \cdot \Delta^{O(1)} \log^2 n)$ following the improved approximation algorithm of [ARV09] for Sparsest Cut. Note that all these algorithms only achieve an $O(n \,\mathrm{poly}(\Delta \log n)))$-approximation factor. However, their performance improves significantly when the crossing number of the input graph is large. A sequence of papers [CMS11, Chu11] provided an improved $\tilde{O}(n^{0.9} \cdot \Delta^{O(1)})$-approximation algorithm for MCN, followed by a more recent sequence of papers by Kawarabayashi and Sidiropoulos [KS17, KS19], who obtained an $\tilde{O}\left(\sqrt{n} \cdot \Delta^{O(1)}\right)$-approximation algorithm. All of the above results follow the same high-level algorithmic framework, and it was shown by Chuzhoy, Madan and Mahabadi [CMM16] (see [Chu16] for an exposition) that this framework is unlikely to yield a better than $O(\sqrt{n})$-approximation. The most recent result, by Chuzhoy, Mahabadi and Tan [CMT20], obtained an $\tilde{O}(n^{1/2-\epsilon} \cdot \mathrm{poly}(\Delta))$-approximation algorithm for some small fixed constant $\epsilon > 0$. This result was achieved by proposing a new algorithmic framework for the problem, that departs from the previous approach. Specifically, [CMT20] reduced the MCN problem to another problem, called

Minimum Crossing Number with Rotation System (MCNwRS) that we discuss below, which appears somewhat easier than the MCN problem, and then provided an algorithm for approximately solving the MCNwRS problem.

Our main result is a randomized $O\left(2^{O((\log n)^{7/8}\log\log n)}\cdot\Delta^{O(1)}\right)$-approximation algorithm for MCN. In order to achieve this result, we design a new algorithm for the MCNwRS problem that achieves significantly stronger guarantees than those of [CMT20]. This algorithm, combined with the reduction of [CMT20], immediately implies the improved approximation for the MCN problem. We also design several new technical tools that we hope will eventually lead to further improvements. We now turn to discuss the MCNwRS problem.

In the Minimun Crossing Number with Rotation System (MCNwRS) problem, the input consists of a multigraph $G$, and, for every vertex $v\in V(G)$, a circular ordering $\mathcal{O}_v$ of edges that are incident to $v$, that we call a *rotation* for vertex $v$. The set $\Sigma=\{\mathcal{O}_v\}_{v\in V(G)}$ of all such rotations is called a *rotation system* for graph $G$. We say that a drawing $\varphi$ of $G$ *obeys* the rotation system $\Sigma$, if, for every vertex $v\in V(G)$, the images of the edges in $\delta_G(v)$ enter the image of $v$ in the order $\mathcal{O}_v$ (but the *orientation* of the ordering can be either clock-wise or counter-clock-wise). In the MCNwRS problem, given a graph $G$ and a rotation system $\Sigma$ for $G$, the goal is to compute a drawing $\varphi$ of $G$ that obeys the rotation system $\Sigma$ and minimizes the number of edge crossings. For an instance $I=(G,\Sigma)$ of the MCNwRS problem, we denote by $\mathsf{OPT}_{\mathsf{cnwrs}}(I)$ the value of the optimal solution for $I$, that is, the smallest number of crossings in any drawing of $G$ that obeys $\Sigma$. The results of [CMT20] show the following reduction from MCN to MCNwRS: suppose there is an efficient (possibly randomized) algorithm for the MCNwRS problem, that, for every instance $I=(G,\Sigma)$, produces a solution whose expected cost is at most $\alpha(m)\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)$, where $m=|E(G)|$. Then there is a randomized $O(\alpha(n)\cdot\mathrm{poly}(\Delta\cdot\log n))$-approximation algorithm for the MCN problem. Our main technical result is a randomized algorithm, that, given an instance $I=(G,\Sigma)$ of MCNwRS, with high probability produces a solution to instance $I$ with at most $2^{O((\log m)^{7/8}\log\log m)}\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(G,\Sigma)+m)$ crossings, where $m=|E(G)|$. Combining this with the result of [CMT20], we immediately obtain a randomized $O\left(2^{O((\log n)^{7/8}\log\log n)}\cdot\mathrm{poly}(\Delta)\right)$-approximation algorithm for the MCN problem.

The best previous algorithm for the MCNwRS problem, due to [CMT20], is a randomized algorithm, that, given an instance $I=(G,\Sigma)$ of the problem, with high probability produces a solution with at most $\tilde{O}\left((\mathsf{OPT}_{\mathsf{cnwrs}}(G,\Sigma)+m)^{2-\epsilon}\right)$ crossings, where $\epsilon=1/20$. A variant of MCNwRS was previously studied by Pelsmajer et al. [PSŠ11], where for each vertex $v$ of the input graph $G$, both the rotation $\mathcal{O}_v$ of its incident edges, and the orientation of this rotation (say clock-wise) are fixed. They showed that this variant of the problem is also NP-hard, and provided an $O(n^4)$-approximation algorithm with running time $O(m^n\log m)$, where $n=|V(G)|$ and $m=|E(G)|$. They also obtained approximation algorithms with improved guarantees for some special families of graphs.

We introduce a number of new technical tools, that we discuss in more detail in Section 1.2. Some of these tools require long and technically involved proofs, which resulted in the large length of the paper. We view these tools as laying a pathway towards obtaining better algorithms for the Minimum Crossing Number problem, and it is our hope that these tools will eventually be streamlined and that their proofs will be simplified, leading to a better understanding of the problem and cleaner and simpler algorithms. We believe that some of these tools are interesting in their own right.

## 1.1 Our Results

Throughout this paper, we allow graphs to have parallel edges (but not self-loops); graphs with no parallel edges are explicitly called simple graphs. For convenience, we will assume that the input to the MCN problem is a simple graph, while graphs serving as inputs to the MCNwRS problem may

have parallel edges. The latter is necessary in order to use the reduction of [CMT20] between the two problems. Note that the number of edges in a graph with parallel edges may be much higher than the number of vertices. Our main technical contribution is an algorithm for the MCNwRS problem, that is summarized in the following theorem.

**Theorem 1.1** *There is an efficient randomized algorithm, that, given an instance $I = (G, \Sigma)$ of* MCNwRS *with $|E(G)| = m$, computes a drawing of $G$ that obeys the rotation system $\Sigma$. The number of crossings in the drawing is w.h.p. bounded by $2^{O((\log m)^{7/8} \log\log m)} \cdot (\mathsf{OPT}_{\mathsf{cnwrs}}(I) + m)$.*

We rely on the following theorem from [CMT20] in order to obtain an approximation algorithm for the MCN problem.

**Theorem 1.2 (Theorem 1.3 in [CMT20])** *There is an efficient algorithm, that, given an $n$-vertex graph $G$ with maximum vertex degree $\Delta$, computes an instance $I = (G', \Sigma)$ of the* MCNwRS *problem, with $|E(G')| \leq O\left(\mathsf{OPT}_{\mathsf{cr}}(G) \cdot \mathrm{poly}(\Delta \cdot \log n)\right)$, and $\mathsf{OPT}_{\mathsf{cnwrs}}(I) \leq O\left(\mathsf{OPT}_{\mathsf{cr}}(G) \cdot \mathrm{poly}(\Delta \cdot \log n)\right)$. Moreover, there is an efficient algorithm that, given any solution of value $X$ to instance $I$ of* MCNwRS*, computes a drawing of $G$ with the number of crossings bounded by $O\left((X + \mathsf{OPT}_{\mathsf{cr}}(G)) \cdot \mathrm{poly}(\Delta \cdot \log n)\right)$.*

Combining Theorem 1.1 and Theorem 1.2, we immediately obtain the following corollary, whose proof appears in Section A of Appendix.

**Corollary 1.3** *There is an efficient randomized algorithm, that, given a simple $n$-vertex graph $G$ with maximum vertex degree $\Delta$, computes a drawing of $G$, such that, w.h.p., the number of crossings in the drawing is at most $O\left(2^{O((\log n)^{7/8} \log\log n)} \cdot \mathrm{poly}(\Delta) \cdot \mathsf{OPT}_{\mathsf{cr}}(G)\right)$.*

### 1.2 Our Techniques

In this subsection we provide an overview of the techniques used in the proof of our main technical result, Theorem 1.1. For the sake of clarity of exposition, some of the discussion here is somewhat imprecise. Our algorithm relies on the divide-and-conquer technique. Given an instance $I = (G, \Sigma)$ of the MCNwRS problem, we compute a collection $\mathcal{I}$ of new instances, whose corresponding graphs are significantly smaller than $G$, and then solve each of the resulting new instances separately. Collection $\mathcal{I}$ of instances is called a *decomposition of $I$*. We require that the decomposition has several useful properties that will allow us to use it in order to obtain the guarantees from Theorem 1.1, by solving the instances in $\mathcal{I}$ recursively. Before we define the notion of decomposition of an instance, we need the notion of a *contracted graph*, that we use throughout the paper. Suppose $G$ is a graph, and let $\mathcal{R} = \{R_1, \ldots, R_q\}$ be a collection of disjoint subsets of vertices of $G$. The contracted graph of $G$ with respect to $\mathcal{R}$, that we denote by $G_{|\mathcal{R}}$, is a graph that is obtained from $G$, by contracting, for all $1 \leq i \leq q$, the vertices of $R_i$ into a supernode $u_i$. Note that every edge of the resulting graph $G_{|\mathcal{R}}$ corresponds to some edge of $G$, and we do not distinguish between them. The vertices in set $V(G_{|\mathcal{R}}) \setminus \{u_1, \ldots, u_q\}$ are called *regular vertices*. Each such vertex $v$ also lies in $G$, and moreover, $\delta_{G_{|\mathcal{R}}}(v) = \delta_G(v)$. Abusing the notation, given a collection $\mathcal{C} = \{C_1, \ldots, C_r\}$ of disjoint subgraphs of $G$, we denote by $G_{|\mathcal{C}}$ the contracted graph of $G$ with respect to the collection $\{V(C_1), \ldots, V(C_r)\}$ of subsets of vertices of $G$. Given a graph $G$ and its drawing $\varphi$, we denote by $\mathsf{cr}(\varphi)$ the number of crossings in $\varphi$.

**Decomposition of an Instance.** Given an instance $I = (G, \Sigma)$ of the MCNwRS problem, we will informally refer to $|E(G)|$ as the *size of the instance*. Assume that we are given an instance $I = (G, \Sigma)$ of MCNwRS with $|E(G)| = m$, and some parameter $\eta$ (we will generally use $\eta = 2^{O((\log m)^{3/4} \log\log m)}$). Assume further that we are given another collection $\mathcal{I}$ of instances of MCNwRS. We say that $\mathcal{I}$ is an *$\eta$-decomposition of $I$*, if $\sum_{I'=(G',\Sigma') \in \mathcal{I}} |E(G')| \leq m \,\mathrm{poly}\log m$, and $\sum_{I' \in \mathcal{I}} \mathsf{OPT}_{\mathsf{cnwrs}}(I') \leq$

$(\mathsf{OPT}_{\mathsf{cnwrs}}(I) + |E(G)|) \cdot \eta$. Additionally, we require that there is an efficient algorithm $\mathsf{Alg}(I)$, that, given a feasible solution $\varphi(I')$ to every instance $I' \in \mathcal{I}$, computes a feasible solution $\varphi$ for instance $I$, with at most $O\left(\sum_{I' \in \mathcal{I}} \mathsf{cr}(\varphi(I'))\right)$ crossings.

At a high level, our algorithm starts with the input instance $I^* = (G^*, \Sigma^*)$ of the MCNwRS problem. Throughout the algorithm, we denote $m^* = |E(G^*)|$, and we use a parameter $\mu = 2^{O((\log m^*)^{7/8} \log\log m^*)}$. Over the course of the algorithm, we consider various other instances $I$ of MCNwRS, but parameters $m^*$ and $\mu$ remain unchanged, and they are defined with respect to the original input instance $I^*$. The main subroutine of the algorithm, that we call AlgDecompose, receives as input an instance $I = (G, \Sigma)$ of MCNwRS, and computes an $\eta$-decomposition $\mathcal{I}$ of $I$, for $\eta = 2^{O((\log m)^{3/4} \log\log m)}$, where $m = |E(G)|$. The subroutine additionally ensures that every instance in the decomposition is sufficiently small compared to $I$, that is, for each instance $I' = (G', \Sigma') \in \mathcal{I}$, $|E(G')| \leq |E(G)|/\mu$. We note that this subroutine is in fact randomized, and, instead of ensuring that $\sum_{I' \in \mathcal{I}} \mathsf{OPT}_{\mathsf{cnwrs}}(I') \leq (\mathsf{OPT}_{\mathsf{cnwrs}}(I) + |E(G)|) \cdot \eta$, it only ensures this in expectation. We will ignore this minor technicality in this high-level exposition.

It is now easy to complete the proof of Theorem 1.1 using Algorithm AlgDecompose: we simply apply Algorithm AlgDecompose to the input instance $I^*$, obtaining a collection $\mathcal{I}$ of new instances. We recursively solve each instance in $\mathcal{I}$, and then combine the resulting solutions using Algorithm $\mathsf{Alg}(I^*)$, in order to obtain the final solution to instance $I^*$. Since the sizes of the instances decrease by the factor of at least $\mu$ with each application of the algorithm, the depth of the recursion is bounded by $O\left((\log m^*)^{1/8}\right)$. At each recursive level, the sum of the optimal solution costs and of the number of edges in all instances at that recursive level increases by at most factor $2^{O((\log m^*)^{3/4} \log\log m^*)}$, leading to the final bound of $2^{O((\log m^*)^{7/8} \log\log m^*)} \cdot (\mathsf{OPT}_{\mathsf{cnwrs}}(I^*) + m^*)$ on the solution cost.

From now on we focus on the description of Algorithm AlgDecompose. We start by describing several technical tools that this algorithm builds on. Throughout, given a graph $G$, we refer to connected vertex-induced subgraphs of $G$ as *clusters*. Given a collection $\mathcal{C}$ of disjoint clusters of $G$, we denote by $E_G^{\mathsf{out}}(\mathcal{C})$ the set of all edges $e \in E(G)$, such that the endpoints of $e$ do not lie in the same cluster of $\mathcal{C}$. We will also use the notion of *subinstances* that we define next.

**Subinstances.** Suppose we are given two instances $I = (G, \Sigma)$ and $I' = (G', \Sigma')$ of MCNwRS. We say that $I'$ is a *subinstance* of instance $I$, if the following hold. First, graph $G'$ must be a graph that is obtained from a subgraph of $G$ by contracting some subsets of its vertices into supernodes. Formally[1], there must be a graph $G'' \subseteq G$, and a collection $\mathcal{R} = \{R_1, \ldots, R_q\}$ of disjoint subsets of vertices of $G''$, such that $G' = G''_{|\mathcal{R}}$. For every regular vertex $v$ of $G'$, the rotation $\mathcal{O}_v \in \Sigma'$ must be consistent with the rotation $\mathcal{O}_v \in \Sigma$ (recall that $\delta_{G'}(v) \subseteq \delta_G(v)$). For every supernode $u_i$ of $G'$, its rotation $\mathcal{O}_{u_i} \in \Sigma'$ can be chosen arbitrarily. Note that the notion of subinstances is transitive: if $I'$ is a subinstance of $I$ and $I''$ is a subinstance of $I'$, then $I''$ is a subinstance of $I$.

The main tool that we use is *disengagement of clusters*. Intuitively, given an instance $I = (G, \Sigma)$ of MCNwRS, and a collection $\mathcal{C}$ of disjoint clusters of $G$, the goal is to compute an $\eta$-decomposition $\mathcal{I}$ of $I$, such that every instance $I' = (G', \Sigma') \in \mathcal{I}$ is a subinstance of $I$, and moreover, there is at most one cluster $C \in \mathcal{C}$ that is contained in $G'$, and all edges of $G'$ that do not lie in $C$ must belong to $E_G^{\mathsf{out}}(\mathcal{C})$. Assume for now that we can design an efficient algorithm for computing such a decomposition. In this case, the high-level plan for implementing Algorithm AlgDecompose would be as follows. First, we compute a collection $\mathcal{C}$ of disjoint clusters of graph $G$, such that, for each cluster $C \in \mathcal{C}$, $|E(C)| \leq |E(G)|/(2\mu)$, and $|E_G^{\mathsf{out}}(\mathcal{C})| \leq |E(G)|/(2\mu)$. Then we perform disengagement of clusters in $\mathcal{C}$, obtaining an $\eta$-decomposition of the input instance $I$. We are then guaranteed that

[1] We note that this definition closely resembles the notion of graph minors, but, in contrast to the definition of minors, we do not require that the induced subgraphs $\{G[R_i]\}_{1 \leq i \leq q}$ are connected.

each resulting instance in $\mathcal{I}$ is sufficiently small. We note that it is not immediately clear how to compute the desired collection $\mathcal{C}$ of disjoint clusters of $G$; we discuss this later. For now we focus on algorithms for computing disengagement of clusters. We do not currently have an algorithm to compute the disengagement of clusters in the most general setting described above. In this paper, we design a number of algorithms for computing disengagement of clusters, under some conditions. We start with the simplest algorithm that only works in some restricted settings, and then generalize it to more advanced algorithms that work in more and more general settings. In order to describe the disengagement algorithm for the most basic setting, we need the notion of congestion, and of internal and external routers, that we use throughout the paper, and describe next.

**Congestion, Internal Routers, and External Routers.** Given a graph $G$ and a set $\mathcal{P}$ of paths in $G$, the *congestion* that the set $\mathcal{P}$ of paths causes on an edge $e \in E(G)$, that we denote by $\mathrm{cong}_G(\mathcal{P}, e)$, is the number of paths in $\mathcal{P}$ containing $e$. The total congestion caused by the set $\mathcal{P}$ of paths in $G$ is $\mathrm{cong}_G(\mathcal{P}) = \max_{e \in E(G)} \{\mathrm{cong}_G(\mathcal{P}, e)\}$.

Consider now a graph $G$ and a cluster $C \subseteq G$. We denote by $\delta_G(C)$ the set of all edges $e \in E(G)$, such that exactly one endpoint of $e$ lies in $C$. An *internal $C$-router* is a collection $\mathcal{Q}(C) = \{Q(e) \mid e \in \delta_G(C)\}$ of paths, such that, for each edge $e \in \delta_G(C)$, path $Q(e)$ has $e$ as its first edge, and all its inner vertices lie in $C$. We additionally require that all paths in $\mathcal{Q}(C)$ terminate at a single vertex of $C$, that we call the *center vertex of the router*. Similarly, an *external $C$-router* is a collection $\mathcal{Q}'(C) = \{Q'(e) \mid e \in \delta_G(C)\}$ of paths, such that, for each edge $e \in \delta_G(C)$, path $Q'(e)$ has $e$ as its first edge, and all its inner vertices lie in $V(G) \setminus V(C)$. We additionally require that all paths in $\mathcal{Q}'(C)$ terminate at a single vertex of $V(G) \setminus V(C)$, that we call the *center vertex of the router*. For a cluster $C \subseteq G$, we denote by $\Lambda_G(C)$ and $\Lambda'_G(C)$ the sets of all internal and all external $C$-routers, respectively.

**Basic Cluster Disengagement.** In the most basic setting for cluster disengagement, we are given an instance $I = (G, \Sigma)$ of the MCNwRS problem, and a collection $\mathcal{C}$ of disjoint clusters of $G$. Additionally, for each cluster $C \in \mathcal{C}$, we are given an internal $C$-router $Q(C)$, whose center vertex we denote by $u(C)$, and an external $C$-router $Q'(C)$, whose center vertex we denote by $u'(C)$. The output of the disengagement procedure is a collection $\mathcal{I}$ of subinstances of $I$, that consists of a single global instance $\hat{I} = (\hat{G}, \hat{\Sigma})$, and, for every cluster $C \in \mathcal{C}$, an instance $I_C = (G_C, \Sigma_C)$ associated with it. Graph $\hat{G}$ is the contracted graph of $G$ with respect to $\mathcal{C}$; that is, it is obtained from $G$ by contracting every cluster $C \in \mathcal{C}$ into a supernode $v_C$. For each cluster $C \in \mathcal{C}$, graph $G_C$ is obtained from $G$ by contracting the vertices of $V(G) \setminus V(C)$ into a supernode $v^*_C$. For every cluster $C \in \mathcal{C}$, the rotation $\mathcal{O}_{v_C} \in \hat{\Sigma}$ of the supernode $v_C$ in instance $\hat{I}$ and the rotation $\mathcal{O}_{v^*_C} \in \Sigma_C$ of the supernode $v^*_C$ in instance $I_C$ need to be defined carefully, in order to ensure that the sum of the optimal solution costs of all resulting instances is low, and that we can combine the solutions to these instances to obtain a solution to $I$. Observe that the set of edges incident to vertex $v_C$ in $\hat{G}$ and the set of edges incident to vertex $v^*_C$ in $G_C$ are both equal to $\delta_G(C)$. We define a single ordering $\mathcal{O}^C$ of the edge set $\delta_G(C)$, that will serve both as the rotation $\mathcal{O}_{v_C} \in \hat{\Sigma}$, and as the rotation $\mathcal{O}_{v^*_C} \in \Sigma_C$. The ordering $\mathcal{O}^C$ is defined via the internal $C$-router $\mathcal{Q}(C)$, and the order in which the images of the paths of $\mathcal{Q}(C)$ enter the image of vertex $u(C)$. On the one hand, letting $\mathcal{O}_{v_C} = \mathcal{O}_{v^*_C}$ for every cluster $C \in \mathcal{C}$ allows us to easily combine solutions $\varphi(I')$ to instances $I' \in \mathcal{I}$, in order to obtain a solution to instance $I$, whose cost is at most $O\left(\sum_{I' \in \mathcal{I}'} \mathsf{cr}(\varphi(I'))\right)$. On the other hand, defining $\mathcal{O}^C$ via the set $\mathcal{Q}(C)$ of paths, for each cluster $C \in \mathcal{C}$, allows us to bound $\sum_{I' \in \mathcal{I}} \mathsf{OPT}_{\mathsf{cnwrs}}(I')$.

We now briefly describe how this latter bound is established, since it will motivate the remainder of the algorithm and clarify the bottlenecks of this approach. We consider an optimal solution $\varphi^*$ to instance $I$, and we use it in order to construct, for each instance $I' \in \mathcal{I}$, a solution $\psi(I')$, such that $\sum_{I' \in \mathcal{I}} \mathsf{cr}(\psi(I'))$ is relatively small compared to $\mathsf{cr}(\varphi^*) + |E(G)|$. In order to construct a solution $\psi(\hat{I})$ to the global instance $\hat{I}$, we start with solution $\varphi^*$ to instance $I$. We erase from this solution all edges and vertices that lie in the clusters of $\mathcal{C}$. For each cluster $C \in \mathcal{C}$, we let the image of the supernode

$v_C$ coincide with the original image of the vertex $u(C)$ – the center of the internal $C$-router $Q(C)$. In order to draw the edges that are incident to the supernode $v_C$ in $\hat{G}$ (that is, the edges of $\delta_G(C)$), we utilize the images of the paths of the internal $C$-router $\mathcal{Q}(C)$ in $\varphi^*$, that connect, for each edge $e \in \delta_G(C)$, the original image of edge $e$ to the original image of vertex $u(C)$.

Consider now some cluster $C \in \mathcal{C}$. In order to construct a solution $\psi(I_C)$ to instance $I_C$, we start again with the solution $\varphi^*$ to instance $I$. We erase from it all edges and vertices except for those lying in $C$. We let the image of the supernode $v_C^*$ be the original image of vertex $u'(C)$ – the center of the external $C$-router $\mathcal{Q}'(C)$. In order to draw the edges that are incident to the supernode $v_C^*$ in $G_C$ (that is, the edges of $\delta_G(C)$), we utilize the images of the paths of the external $C$-router $\mathcal{Q}'(C)$, that connect, for each edge $e \in \delta_G(C)$, the original image of edge $e$ to the original image of vertex $u'(C)$.

Observe that the only increase in $\sum_{I' \in \mathcal{I}} \mathsf{cr}(\psi(I'))$, relatively to $\mathsf{cr}(\varphi^*)$, is due to the crossings incurred by drawing the edges incident to the supernodes in $\{v_C\}_{C \in \mathcal{C}}$ in instance $\hat{I}$, and for each subinstance $I_C$, drawing the edges incident to supernode $v_C^*$. All such edges are drawn along the images of the paths in $\bigcup_{C \in \mathcal{C}} (\mathcal{Q}(C) \cup \mathcal{Q}'(C))$ in $\varphi^*$. However, an edge may belong to a number of such paths. With careful accounting we can bound this number of new crossings as follows. Assume that, for every cluster $C \in \mathcal{C}$, $\mathrm{cong}_G(\mathcal{Q}'(C)) \leq \beta$. Assume further that, for each cluster $C \in \mathcal{C}$, and for each edge $e \in E(C)$, $(\mathrm{cong}_G(\mathcal{Q}(C), e))^2 \leq \beta$. Then $\sum_{I' \in \mathcal{I}} \mathsf{cr}(\psi(I')) \leq O(\beta^2 \cdot (\mathsf{OPT}_{\mathsf{cnwrs}}(I) + |E(G)|))$. Therefore, in order to ensure that the collection $\mathcal{I}$ of subinstances of $I$ that we have obtained via the cluster disengagement procedure is an $\eta$-decomposition of $I$, we need to ensure that, for every cluster $C \in \mathcal{C}$, $\mathrm{cong}_G(\mathcal{Q}'(C)) \leq \beta$, and, for every edge $e \in E(C)$, $(\mathrm{cong}_G(\mathcal{Q}(C), e))^2 \leq \beta$, for $\beta = O(\eta^{1/2})$. This requirement seems impossible to achieve. For example, if maximum vertex degree in graph $G$ is small (say a constant), then some edges incident to the center vertices $\{u(C), u'(C)\}_{C \in \mathcal{C}}$ must incur very high congestion. In order to overcome this obstacle, we slightly weaken our requirements. Instead of providing, for every cluster $C \in \mathcal{C}$, a single internal $C$-router $Q(C)$, and a single external $C$-router $Q'(C)$, it is sufficient for us to obtain, for each cluster $C \in \mathcal{C}$, a *distribution* $\mathcal{D}(C)$ over the collection $\Lambda_G(C)$ of internal $C$-routers, such that, for every edge $e \in E(C)$, $\mathbf{E}_{\mathcal{Q}(C) \sim \mathcal{D}(C)} \left[(\mathrm{cong}_G(\mathcal{Q}(C), e))^2\right] \leq \beta$, and a distribution $\mathcal{D}'(C)$ over the collection $\Lambda'_G(C)$ of external $C$-routers, such that for every edge $e$, $\mathbf{E}_{\mathcal{Q}'(C) \sim \mathcal{D}'(C)} \left[\mathrm{cong}_G(\mathcal{Q}'(C), e)\right] \leq \beta$.

To recap, in order to use the Basic Cluster Disengagement procedure described above to compute an $\eta$-decomposition of the input instance $I$ of MCNwRS into sufficiently small instances, it is now enough to design a procedure that, given an instance $I = (G, \Sigma)$ of MCNwRS, computes a collection $\mathcal{C}$ of disjoint clusters of $G$, and, for every cluster $C \in \mathcal{C}$, a distribution $\mathcal{D}(C)$ over the collection $\Lambda_G(C)$ of internal $C$-routers, such that, for every edge $e \in E(C)$, $\mathbf{E}_{\mathcal{Q}(C) \sim \mathcal{D}(C)} \left[(\mathrm{cong}_G(\mathcal{Q}(C), e))^2\right] \leq \beta$, together with a distribution $\mathcal{D}'(C)$ over the collection $\Lambda'_G(C)$ of external $C$-routers, such that, for every edge $e$, $\mathbf{E}_{\mathcal{Q}'(C) \sim \mathcal{D}'(C)} \left[\mathrm{cong}_G(\mathcal{Q}'(C), e)\right] \leq \beta$, for $\beta = O(\sqrt{\eta})$. Additionally, we need to ensure that, for every cluster $C \in \mathcal{C}$, $|E(C)| \leq |E(G)|/(2\mu)$, and that $|E_G^{\mathsf{out}}(\mathcal{C})| \leq |E(G)|/(2\mu)$. While computing a collection $\mathcal{C}$ of clusters with the latter two properties seems possible (at least when the maximum vertex degree in $G$ is small), computing the distributions over the internal and the external routers for each cluster $C$ with the required properties seems quite challenging. As a first step towards this goal, we employ the standard notions of well-linkedness and bandwidth property of clusters as a proxy to constructing internal $C$-routers with the required properties. Before we turn to discuss these notions, we note that the Basic Cluster Disengagement procedure that we have just described can be easily generalized to the more general setting, where the set $\mathcal{C}$ of clusters is laminar (instead of only containing disjoint clusters). This generalization will be useful for us later.

Assume that we are given a laminar family $\mathcal{C}$ of clusters (that is, for every pair $C, C' \in \mathcal{C}$ of clusters, either $C \subseteq C'$, or $C' \subseteq C$, or $C \cap C' = \emptyset$ holds), with $G \in \mathcal{C}$. Assume further that we are given, for each cluster $C \in \mathcal{C}$, a distribution $\mathcal{D}(C)$ over the collection $\Lambda_G(C)$ of internal $C$-routers, in which, for every edge $e \in E(C)$, $\mathbf{E}_{\mathcal{Q}(C) \sim \mathcal{D}(C)} \left[(\mathrm{cong}_G(\mathcal{Q}(C), e))^2\right] \leq \beta$, together with a distribution $\mathcal{D}'(C)$ over the

collection $\Lambda'_G(C)$ of external $C$-routers, where for every edge $e$, $\mathbf{E}_{\mathcal{Q}'(C)\sim\mathcal{D}'(C)}\left[\mathrm{cong}_G(\mathcal{Q}'(C),e)\right]\leq\beta$, for some parameter $\beta$. The Basic Cluster Disengagement procedure, when applied to $\mathcal{C}$, produces a collection $\mathcal{I}=\{I_C=(G_C,\Sigma_C)\mid C\in\mathcal{C}\}$ of instances. For every cluster $C\in\mathcal{C}$, graph $G_C$ associated with instance $I_C$ is obtained from graph $G$, by first contracting all vertices of $V(G)\setminus V(C)$ into a supernode $v^*_C$, and then contracting, for each child-cluster $C'\in\mathcal{C}$ of $C$, the vertices of $V(C')$ into a supernode $v_{C'}$. We define, for every cluster $C$, an ordering of the set $\delta_G(C)$ of edges via an internal $C$-router that is selected from the distribution $\mathcal{D}(C)$, and we let the rotation $\mathcal{O}_{v^*_C}$ in the rotation system $\Sigma_C$, and the rotation $\mathcal{O}_{v_C}$ in the rotation system $\Sigma_{C'}$, where $C'$ is the parent-cluster of $C$, to be identical to this ordering. Using the same reasoning as in the case where $\mathcal{C}$ is a set of disjoint clusters, we show that $\mathbf{E}\left[\sum_{I'\in\mathcal{I}}\mathsf{OPT}_{\mathsf{cnwrs}}(I')\right]\leq O\left(\beta^2\cdot\mathsf{dep}(\mathcal{C})\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+|E(G)|)\right)$, where $\mathsf{dep}(\mathcal{C})$ is the depth of the laminar family $\mathcal{C}$ of clusters. We then show that $\mathcal{I}'$ is an $\eta'$-decomposition of instance $I$, where $\eta'=O(\beta^2\cdot\mathsf{dep}(\mathcal{C}))$.

As noted already, one of the difficulties in exploiting the Basic Cluster Disengagement procedure in order to compute an $\eta$-decomposition of the input instance $\mathcal{I}$ is the need to compute distributions over the sets of internal and the external $C$-routers for every cluster $C\in\mathcal{C}$, with the required properties. We turn instead to the notions of well-linkedness and bandwidth properties of clusters. These notions are extensively studied, and there are many known algorithms for computing a collection $\mathcal{C}$ of clusters that have bandwidth property in a graph. We will use this property as a proxy, that will eventually allow us to construct a distribution over the sets of internal $C$-routers for each cluster $C\in\mathcal{C}$, with the required properties.

**Well-Linkedness, Bandwidth Property, and Cluster Classification.** We use the standard graph-theoretic notion of well-linkedness. Let $G$ be a graph, let $T$ be a subset of the vertices of $G$, and let $0<\alpha<1$ be a parameter. We say that the set $T$ of vertices is *$\alpha$-well-linked* in $G$ if for every partition $(A,B)$ of vertices of $G$ into two subsets, $|E_G(A,B)|\geq\alpha\cdot\min\{|A\cap T|,|B\cap T|\}$.

We also use a closely related notion of *bandwidth property* of clusters. Suppose we are given a graph $G$ and a cluster $C\subseteq G$. Intuitively, cluster $C$ has the $\alpha$-bandwidth property (for a parameter $0<\alpha<1$), if the edges of $\delta_G(C)$ are $\alpha$-well-linked in $C$. More formally, we consider the augmentation $C^+$ of cluster $C$, that is defined as follows. We start with the graph $G$, and subdivide every edge $e\in\delta_G(C)$ with a vertex $t_e$, denoting by $T=\{t_e\mid e\in\delta_G(C)\}$ this new set of vertices. The augmentation $C^+$ of $C$ is the subgraph of the resulting graph induced by $V(C)\cup T$. We say that cluster $C$ has the $\alpha$-bandwidth property if set $T$ of vertices is $\alpha$-well-linked in $C^+$.

We note that, if a cluster $C$ has the $\alpha$-bandwidth property, then, using known techniques, we can efficiently construct a distribution $\mathcal{D}$ over the set $\Lambda_G(C)$ of internal $C$-routers, such that, for every edge $e\in E(C)$, $\mathbf{E}_{\mathcal{Q}(C)\sim\mathcal{D}(C)}\left[\mathrm{cong}(\mathcal{Q}(C),e)\right]\leq O(1/\alpha)$. However, in order to use the Basic Cluster Disengagement procedure, we need a stronger property: namely, for every edge $e\in E(C)$, we require that $\mathbf{E}_{\mathcal{Q}(C)\sim\mathcal{D}(C)}\left[(\mathrm{cong}(\mathcal{Q}(C),e))^2\right]\leq\beta$, for some parameter $\beta$. If we are given a distribution $\mathcal{D}(C)$ over the set $\Lambda_G(C)$ of internal $C$-routers with this latter property, then we say that cluster $C$ is *$\eta$-light* with respect to $\mathcal{D}(C)$. Computing a distribution $\mathcal{D}(C)$ for which cluster $C$ is $\eta$-light is a much more challenging task. We come close to achieving it in our Cluster Classification Theorem. Before we describe the theorem, we need one more definition. Let $C$ be a cluster of a graph $G$, and let $\eta'$ be some parameter. Assume that we are given some rotation system $\Sigma$ for graph $G$, and let $\Sigma^C$ be the rotation system for cluster $C$ that is induced by $\Sigma$. Let $I^C=(C,\Sigma^C)$ be the resulting instance of MCNwRS. We say that cluster $C$ is *$\eta'$-bad* if $\mathsf{OPT}_{\mathsf{cnwrs}}(I^C)\geq|\delta_G(C)|^2/\eta'$.

In the Cluster Classification Theorem, we provide an efficient algorithm, that, given an instance $I=(G,\Sigma)$ of MCNwRS with $|E(G)|=m$, and a cluster $C\subseteq G$ that has the $\alpha$-bandwidth property (where $\alpha=\Omega(1/\operatorname{poly}\log m)$), either correctly establishes that cluster $C$ is $\eta'$-bad, for $\eta'=2^{O((\log m)^{3/4}\log\log m)}$, or produces a distribution $\mathcal{D}(C)$ over the set $\Lambda_G(C)$ of internal $C$-routers, such that cluster $C$ is $\beta$-light with respect to $\mathcal{D}(C)$, for $\beta=2^{O(\sqrt{\log m}\cdot\log\log m)}$. In fact, the algorithm is randomized, and, with a

small probability, it may erroneously classify cluster $C$ as being $\eta'$-bad, when this is not the case. This small technicality is immaterial to this high-level exposition, and we will ignore it here. The proof of the Cluster Classification Theorem is long and technically involved, and is partially responsible for the high approximation factor that we eventually obtain. It is our hope that a simpler and a cleaner proof of the theorem with better parameters will be discovered in the future. We believe that the theorem is a graph-theoretic result that is interesting in its own right. We now provide a high-level summary of the main challenges in its proof.

At the heart of the proof is an algorithm that we called AlgFindGuiding. Suppose we are given an instance $I = (H, \Sigma)$ of MCNwRS, and a set $T$ of $k$ vertices of $H$ called terminals, that are $\alpha$-well-linked in $H$, for some parameter $0 < \alpha < 1$. Denote $C = H \setminus T$ and $|V(H)| = n$. The goal of the algorithm is to either establish that $\mathsf{OPT}_{\mathsf{cnwrs}}(H) + |E(H)| \geq k^2 \operatorname{poly}(\alpha/\log n)$; or to compute a distribution $\mathcal{D}(C)$ over internal $C$-routers, such that cluster $C$ is $\eta' = \operatorname{poly}(\log n/\alpha)$-light with respect to $\mathcal{D}(C)$.

Consider first a much simpler setting, where $H$ is a grid graph, and $T$ is the set of vertices on the first row of the grid. For this special case, the algorithm of [Sid10] (see also Lemma D.10 in the full version of [Chu11]) provides the construction of a distribution $\mathcal{D}(C)$ over internal $C$-routers with the required properties. This result can be easily generalized to the case where $H$ is a bounded-degree planar graph, since such a graph must contain a large grid minor. If $H$ is a planar graph, but its maximum vertex degree is no longer bounded, we can still compute a grid-like structure in it, and apply the same arguments as in [Sid10] in order to compute the desired distribution $\mathcal{D}(C)$. The difficulty in our case is that the graph $H$ may be far from being planar, and, even though, from the Excluded Grid theorem of Robertson and Seymour [RS86, RST94], it must contain a large grid-like structure, without having a drawing of $H$ in the plane with a small number of crossings, we do not know how to compute such a structure[2]. We provide an algorithm that either establishes that $\mathsf{OPT}_{\mathsf{cnwrs}}(H)$ is large compared to $k^2$, or computes a grid-like structure in graph $H$, even if it is not a planar graph. Unfortunately, this algorithm only works in the setting where $|E(H)|$ is not too large comparable to $k$. Specifically, if we ensure that $|E(H)| \leq k \cdot \hat{\eta}$ for some parameter $\hat{\eta}$, then the algorithm either computes a distribution $\mathcal{D}(C)$ over internal $C$-routers that is $\eta'$-light (with $\eta' = \operatorname{poly}(\log n/\alpha)$ as before), or it establishes that $\mathsf{OPT}_{\mathsf{cnwrs}}(H) + |E(H)| \geq k^2 \operatorname{poly}(\alpha/(\hat{\eta} \log n))$.

Typically, this algorithm would be used in the following setting: we are given a cluster $C$ of a graph $G$, that has the $\alpha$-bandwidth property. We then let $H = C^+$ be the augmentation of $C$, and we let $T$ be the set of vertices of $C^+$ corresponding to the edges of $\delta_H(C)$. In order for this result to be meaningful, we need to ensure that $|E(C)|$ is not too large compared to $|\delta_H(C)|$. Unfortunately, we may need to apply the classification theorem to clusters $C$ for which $|E(C)| \gg |\delta_H(C)|$ holds. In order to overcome this difficulty, given such a cluster $C$, we construct a recursive decomposition of $C$ into smaller and smaller clusters. Let $\mathcal{L}$ denote the resulting family of clusters, which is a laminar family of subgraphs of $C$. We ensure that every cluster $C' \in \mathcal{L}$ has $\alpha = \Omega(1/\operatorname{poly}\log m)$-bandwidth property, and, additionally, if we let $\hat{C}'$ be the graph obtained from $C'$ by contracting every child-cluster of $C'$ into a supernode, then the number of edges in $\hat{C}'$ is comparable to $|\delta_H(C')|$. We consider the clusters of $\mathcal{L}$ from smallest to largest. For each such cluster $C'$, we carefully apply Algorithm AlgFindGuiding to the corresponding contracted graph $\hat{C}'$, in order to either classify cluster $\hat{C}'$ as $\eta(C')$-bad, or to compute a distribution $\mathcal{D}(C')$ over internal $C'$-routers, such that $C'$ is $\beta(C')$-light with respect to $\mathcal{D}(C')$. Parameters $\eta(C')$ and $\beta(C')$ depend on the height of the cluster $C'$ in the decomposition tree that is associated with the laminar family $\mathcal{L}$ of clusters. This recursive algorithm is eventually used to either establish that cluster $C$ is $\eta(C)$-bad, or to compute a distribution $\mathcal{D}(C)$ over the set $\Lambda_G(C)$ of internal $C$-routers, such that cluster $C$ is $\beta(C)$-light with respect to $\mathcal{D}(C)$. The final parameters $\eta(C)$

[2]We note that we need the grid-like structure to have dimensions $(k' \times k')$, where $k'$ is almost linear in $k$. Therefore, we cannot use the known bounds for the Excluded Minor Theorem (e.g. from [CT19]) for general graphs, and instead we need to use an analogue of the stronger version of the theorem for planar graphs.

and $\beta(C)$ depend exponentially on the height of the decomposition tree associated with the laminar family $\mathcal{L}$. This strong dependence on $\mathsf{dep}(\mathcal{L})$ is one of the reasons for the high approximation factor that our algorithm eventually achieves.

**Obstacles to Using Basic Cluster Disengagement.** Let us now revisit the Basic Cluster Disengagement routine. We start with an instance $I = (G, \Sigma)$ of MCNwRS, and denote $|E(G)| = m$. Throughout, we use a parameter $\eta = 2^{O((\log m)^{3/4} \log\log m)}$, and $\beta = \eta^{1/8}$. Recall that the input to the procedure is a collection $\mathcal{C}$ of disjoint clusters of $G$. For every cluster $C \in \mathcal{C}$, we are also given a distribution $\mathcal{D}'(C)$ over the set of external $C$-routers, such that, for every edge $e$, $\mathbf{E}_{\mathcal{Q}'(C)\sim\mathcal{D}'(C)}\left[\mathrm{cong}_G(\mathcal{Q}'(C), e)\right] \leq \beta$, and a distribution $\mathcal{D}(C)$ over the set of internal $C$-routers, such that cluster $C$ is $\beta$-light with respect to $\mathcal{D}(C)$. We are then guaranteed that the collection $\mathcal{I}$ of subinstances of $I$ that is constructed via Basic Cluster Disengagement is an $\eta$-decomposition of $I$. We can slightly generalize this procedure to handle bad clusters as well. Specifically, suppose we are given a partition $(\mathcal{C}^{\mathrm{light}}, \mathcal{C}^{\mathrm{bad}})$ of the clusters in $\mathcal{C}$, and, for each cluster $C \in \mathcal{C}^{\mathrm{light}}$, a distribution $\mathcal{D}(C)$ over internal $C$-routers, such that cluster $C$ is $\beta$-light with respect to $\mathcal{D}(C)$. Assume further that each cluster $C \in \mathcal{C}^{\mathrm{bad}}$ is $\beta$-bad. Additionally, assume that we are given, for every cluster $C \in \mathcal{C}$, a distribution $\mathcal{D}'(C)$ over external $C$-routers, such that, for every edge $e$, $\mathbf{E}_{\mathcal{Q}'(C)\sim\mathcal{D}'(C)}\left[\mathrm{cong}_G(\mathcal{Q}'(C), e)\right] \leq \beta$, and that every cluster $C \in \mathcal{C}$ has the $\alpha$-bandwidth property, for some $\alpha = \Omega(1/\operatorname{poly}\log m)$. We can then generalize the Basic Cluster Disengagement procedure to provide the same guarantees as before in this setting, to obtain an $\eta$-decomposition of instance $I$.

Assume now that we are given an instance $I = (G, \Sigma)$ of MCNwRS, with $|E(G)| = m$. For simplicity, assume for now that the maximum vertex degree in $G$ is quite small (it is sufficient, for example, that it is significantly smaller than $m$.) Using known techniques, we can compute a collection $\mathcal{C}$ of disjoint clusters of $G$, such that, for every cluster $C \in \mathcal{C}$, $|E(C)| \leq m/(2\mu)$; $|E_G^{\mathsf{out}}(\mathcal{C})| \leq m/(2\mu)$; and every cluster $C \in \mathcal{C}$ has $\alpha$-bandwidth property. If we could, additionally, compute, for each cluster $C \in \mathcal{C}$, a distribution $\mathcal{D}'(C)$ over external $C$-routers, such that, for every edge $e$, $\mathbf{E}_{\mathcal{Q}'(C)\sim\mathcal{D}'(C)}\left[\mathrm{cong}_G(\mathcal{Q}'(C), e)\right] \leq \beta$, then we could use the Cluster Classification Theorem to partition the set $\mathcal{C}$ of clusters into subsets $\mathcal{C}^{\mathrm{light}}$ and $\mathcal{C}^{\mathrm{bad}}$, and to compute, for every cluster $C \in \mathcal{C}^{\mathrm{light}}$, a distribution $\mathcal{D}(C)$ over the set of its internal routers, such that every cluster in $\mathcal{C}^{\mathrm{bad}}$ is $\eta'$-bad, and every cluster $C \in \mathcal{C}^{\mathrm{light}}$ is $\eta'$-light with respect to $\mathcal{D}(C)$, for some parameter $\eta'$. We could then apply the Basic Cluster Disengagement procedure in order to compute the desired $\eta$-decomposition of the input instance $I$. Unfortunately, we currently do not have an algorithm that computes both the collection $\mathcal{C}$ of clusters of $G$ with the above properties, and the required distributions over the external $C$-routers for each such cluster $C$. In order to overcome this difficulty, we design Advanced Cluster Disengagement procedure, that generalizes Basic Cluster Disengagement, and no longer requires the distribution over external $C$-routers for each cluster $C \in \mathcal{C}$.

**Advanced Cluster Disengagement.** The input to the Advanced Cluster Disengagement procedure is an instance $I = (G, \Sigma)$ of MCNwRS, and a set $\mathcal{C}$ of disjoint clusters of $G$, that we refer to as *basic clusters*. Let $m = |E(G)|$, and $\eta = 2^{O((\log m)^{3/4} \log\log m)}$ as before. The output is an $\eta$-decomposition $\mathcal{I}$ of $I$, such that every instance $I' = (G', \Sigma') \in \mathcal{I}$ is a subinstance of $I$, and moreover, there is at most one basic cluster $C \in \mathcal{C}$ with $E(C) \subseteq E(G')$, with all other edges of $G'$ lying in $E_G^{\mathsf{out}}(\mathcal{C})$. The algorithm for the Advanced Cluster Disengagement and its analysis are significantly more involved than those of Basic Cluster Disengagement. We start with some intuition.

Consider the contracted graph $H = G_{|\mathcal{C}}$, and its Gomory-Hu tree $T$. This tree naturally defines a laminar family $\mathcal{L}$ of clusters of $H$: for every vertex $v \in V(H)$, we add to $\mathcal{L}$ the cluster $U_v$, that is the subgraph of $H$ induced by vertex set $V(T_v)$, where $T_v$ is the subtree of $T$ rooted at $v$. From the properties of Gomory-Hu trees, if $v'$ is the parent-vertex of vertex $v$ in $T$, there is an external $U_v$-router $\mathcal{Q}'(U_v)$ in graph $H$ with $\mathrm{cong}_H(\mathcal{Q}'(U_v)) = 1$. Laminar family $\mathcal{L}$ of clusters of $H$ naturally defines a laminar family $\mathcal{L}'$ of clusters of the original graph $G$, where for each cluster $U_v \in \mathcal{L}$, set $\mathcal{L}'$

contains a corresponding cluster $U'_v$, that is obtained from $U_v$, by un-contracting all supernodes that correspond to clusters of $\mathcal{C}$. For each such cluster $U'_v \in \mathcal{L}'$, we can use the external $U_v$-router $\mathcal{Q}'(U_v)$ in graph $H$ in order to construct a distribution $\mathcal{D}'(U'_v)$ over external $U'_v$-routers in graph $G$, where for every edge $e$, $\mathbf{E}_{\mathcal{Q}'(U'_v)\sim\mathcal{D}'(U'_v)}\left[\mathrm{cong}_G(\mathcal{Q}'(U'_v), e)\right] \leq O(1/\alpha)$. We can then apply the Basic Cluster Disengagement procedure to the laminar family $\mathcal{L}'$ and the distributions $\{\mathcal{D}'(U'_v)\}_{U'_v\in\mathcal{L}'}$ in order to compute an $\eta^*$-decomposition $\mathcal{I}$ of instance $I$, where every instance in $\mathcal{I}$ is a subinstance of $I$. Recall that the parameter $\eta^*$ depends on the depth of the laminar family $\mathcal{L}'$, which is equal to the depth of the laminar family $\mathcal{L}$. Therefore, if $\mathsf{dep}(\mathcal{L})$ is not too large (for example, it is at most $2^{O((\log m)^{3/4}\log\log m)}$), then we will obtain the desired $\eta$-decomposition of $I$. But unfortunately we have no control over the depth of the laminar family $\mathcal{L}$, and in particular the tools described so far do not work when the Gomory-Hu tree $T$ is a path.

Roughly speaking, we would like to design a different disengagement procedure for the case where the tree $T$ is a path, and then reduce the general problem (by exploiting Basic Cluster Disengagement) to this special case. In fact we follow a similar plan. We define a special type of instances (that we call *nice instances*), that resemble the case where the Gomory-Hu tree of the contracted graph $H = G_{|\mathcal{C}}$ is a path. While the motivation behind the definition of nice instances is indeed this special case, the specifics of the definition are somewhat different, in that it is more general in some of its aspects, and more restrictive and well-structured in others. We provide an algorithm for Cluster Disengagment of nice instances, that ensures that, for each resulting subinstance $I' = (G', \Sigma')$, there is at most one cluster $C \in \mathcal{C}$ with $C \subseteq G'$, and all other edges of $G'$ lie in $E^{\mathsf{out}}_G(\mathcal{C})$. We also provide another algorithm, that, given an instance $I = (G, \Sigma)$ of MCNwRS and a collection $\mathcal{C}$ of disjoint basic clusters of graph $G$, computes a decomposition $\mathcal{I}'$ of instance $I$, such that each resulting instance $I' = (G', \Sigma') \in \mathcal{I}'$ is a subinstance of $I$ and a nice instance, with respect to the subset $\mathcal{C}(I') \subseteq \mathcal{C}$ of clusters, that contains every cluster $C \in \mathcal{C}$ with $C \subseteq G'$. Combining these two algorithms allows us to compute Advanced Cluster Disengagement.

**Algorithm** AlgDecompose**.** Recall that Algorithm AlgDecompose, given an instance $I = (G, \Sigma)$ of MCNwRS, needs to compute an $\eta$-decomposition $\mathcal{I}$ of $I$, where $\eta = 2^{O((\log m)^{3/4}\log\log m)}$ and $m = |E(G)|$, such that, for each instance $I' = (G', \Sigma') \in \mathcal{I}$, $|E(G')| \leq |E(G)|/\mu$. We say that a vertex $v$ of $G$ is a *high-degree vertex* if $|\delta_G(v)| \geq m/\operatorname{poly}(\mu)$ (here, $\mu = 2^{O((\log m^*)^{7/8}\log\log m^*)}$, and $m^*$ is the number of edges in the original input instance $I^*$ of MCNwRS).

Consider first the special case where no vertex of $G$ is a high-degree vertex. For this case, it is not hard to generalize known well-linked decomposition techniques to obtain a collection $\mathcal{C}$ of disjoint clusters of $G$, such that each cluster $C \in \mathcal{C}$ has $\alpha = \Omega(1/\operatorname{poly}\log m)$-bandwidth property, with $|E(C)| < O(m/\mu)$, and, additionally, $|E^{\mathsf{out}}_G(\mathcal{C})| \leq O(m/\mu)$. We can now apply the Advanced Cluster Disengagement procedure to the set $\mathcal{C}$ of clusters, in order to obtain the desired $\eta$-decomposition of $I$. Recall that we are guaranteed that each resulting instance $I' = (G', \Sigma') \in \mathcal{I}$ is a subinstance of $I$, and there is at most one cluster $C \in \mathcal{C}$ with $C \subseteq G'$, with all other edges of $G'$ lying in $E^{\mathsf{out}}_G(\mathcal{C})$. This ensures that $|E(G')| \leq m/\mu$, as required.

In general, however, it is possible that the input instance $I = (G, \Sigma)$ contains high-degree vertices. We then consider two cases. We say that instance $I$ is *wide* if there is a vertex $v \in V(G)$, a partition $(E_1, E_2)$ of the edges of $\delta_G(v)$, such that the edges of $E_1$ appear consecutively in the rotation $\mathcal{O}_v \in \Sigma$, and so do the edges of $E_2$, and a collection $\mathcal{P}$ of at least $m/\operatorname{poly}(\mu)$ simple edge-disjoint cycles in $G$, such that every cycle $P \in \mathcal{P}$ contains one edge of $E_1$ and one edge of $E_2$. An instance that is not wide is called *narrow*. We provide separate algorithms for dealing with narrow and wide instances.

**Narrow Instances.** The algorithm for decomposing narrow instances relies on and generalizes the algorithm for the special case where $G$ has no high-degree vertices. As a first step, we compute a collection $\mathcal{C}$ of disjoint clusters of $G$, such that each cluster $C \in \mathcal{C}$ has $\alpha = \Omega(1/\operatorname{poly}\log m)$-bandwidth property, and $|E^{\mathsf{out}}_G(\mathcal{C})| < O(E(G)/\mu)$. The set $\mathcal{C}$ of clusters is partitioned into two subsets: set $\mathcal{C}^s$ of

small clusters, and set $\mathcal{C}^f$ of *flower clusters*. For each cluster $C \in \mathcal{C}^s$, $|E(C)| < O(|E(G)|/\mu)$ holds. If $C$ is a cluster of $\mathcal{C}^f$, then we no longer guarantee that $|E(C)|$ is small. Instead, we guarantee that cluster $C$ has a special structure. Specifically, $C$ must contain a single high-degree vertex $u(C)$, that we call the *flower center*, and all other vertices of $C$ must be low-degree vertices. Additionally, there must be a set $\mathcal{X}(C) = \{X_1, \ldots, X_k\}$ of subgraphs of $C$, that we call *petals*, such that, for all $1 \leq i < j \leq k$, $V(X_i) \cap V(X_j) = \{u(C)\}$. We also require that, for all $1 \leq i \leq k$, there is a set $E_i$ of $\Theta(m/\operatorname{poly}(\mu))$ edges of $\delta_G(u(C))$ that are contiguous in the rotation $\mathcal{O}_{u(C)} \in \Sigma$, and lie in $X_i$ (see Figure 1). Lastly, we require that, for all $1 \leq i \leq k$, there is a set $\mathcal{Q}_i$ of edge-disjoint paths, connecting every edge of $\delta_G(X_i) \setminus \delta_G(u(C))$ to vertex $u(C)$, with all inner vertices on the paths lying in $X_i$.

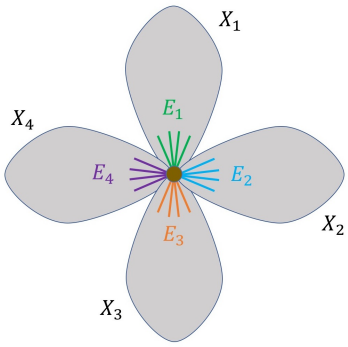



Figure 1: An illustration of a 4-petal flower cluster.

We apply Advanced Cluster Disengagement to the set $\mathcal{C}$ of clusters, in order to compute an initial decomposition $\mathcal{I}_1$ of the input instance $I$, such that every instance in $\mathcal{I}_1$ is a subinstance of $I$. Unfortunately, it is possible that, for some instances $I' = (G', \Sigma') \in \mathcal{I}_1$, $|E(G')| > m/\mu$. For each such instance $I'$, there must be some flower cluster $C \in \mathcal{C}^f$ that is contained in $G'$, and all other edges of $G'$ must lie in $E_G^{\mathsf{out}}(\mathcal{C})$.

We now consider each instance $I' = (G', \Sigma') \in \mathcal{I}_1$ with $|E(G')| > m/\mu$ one by one. Assume that $C \in \mathcal{C}^f$ is the flower cluster that is contained in $G'$, and $\mathcal{X}(C) = \{X_1, \ldots, X_k\}$ is the set of its petals. We further decompose instance $I'$ into a collection $\mathcal{I}(C)$ of subinstances, that consists of a single global instance $\hat{I}(C)$, and $k$ additional instances $I_1(C), \ldots, I_k(C)$. We ensure that the graph $\hat{G}(C)$ associated with the global instance $\hat{I}(C)$ only contains edges of $E_G^{\mathsf{out}}(\mathcal{C})$, so $|E(\hat{G}(C))| < m/\mu$ holds. For all $1 \leq j \leq k$, graph $G_j(C)$ associated with instance $I_j(C) \in \mathcal{I}(C)$ contains the petal $X_j$, and all other edges of $G_j(C)$ lie in $E_G^{\mathsf{out}}(\mathcal{C})$. We note that unfortunately it is still possible that, for some $1 \leq j \leq k$, graph $G_j(C)$ contains too many edges (this can only happen if $|E(X_j)|$ is large). However, our construction ensures that, for each such instance $I_j(C)$, no high-degree vertices lie in graph $G_j(C)$. We can then further decompose instance $I_j(C)$ into subinstances using the algorithm that we designed for the case where no vertex of the input graph is a high-degree vertex. After this final decomposition, we are guaranteed that each of the final subinstances of instance $I$ that we obtain contains fewer than $m/\mu$ edges, as required.

**Wide Instances.** Suppose we are given a wide instance $I = (G, \Sigma)$ of MCNwRS. In this case, we compute an $\eta$-decomposition $\mathcal{I}$ of instance $I$, such that, for each resulting instance $I' = (G', \Sigma') \in \mathcal{I}$, either $|E(G')| < m/\mu$ (in which case we say that $I'$ is a *small instance*), or $I'$ is a narrow instance. We will then further decompose each resulting narrow instance using the algorithm described above.

In order to obtain the decomposition $\mathcal{I}$ of $I$, we start with $\mathcal{I} = \{I\}$. As long as set $\mathcal{I}$ contains at least one wide instance $I' = (G', \Sigma')$ with $|E(G')| \geq m/\mu$, we perform a procedure that "splits" instance $I'$ into two smaller subinstances. We now turn to describe this procedure at a high level.

Let $I' = (G', \Sigma') \in \mathcal{I}$ be a wide instance with $|E(G')| \geq m/\mu$. Recall that from the definition of a wide

instance, there is a vertex $v \in V(G')$, a partition $(E_1, E_2)$ of the edges of $\delta_{G'}(v)$, such that the edges of $E_1$ appear consecutively in the rotation $\mathcal{O}_v \in \Sigma'$, and a collection $\mathcal{P}$ of at least $m/\operatorname{poly}(\mu)$ simple edge-disjoint cycles in $G'$, such that every cycle in $\mathcal{P}$ contains one edge of $E_1$ and one edge of $E_2$. Consider the experiment, in which we choose a cycle $W \in \mathcal{P}$ uniformly at random. Since $|\mathcal{P}|$ is very large, with reasonably high probability, the edges of the cycle $W$ participate in relatively few crossings in the optimal solution to instance $I'$ of MCNwRS. We show that with high enough probability, there is a near-optimal solution to $I'$, in which cycle $W$ is drawn in the natural way. We use the cycle $W$ in order to partition instance $I'$ into two subinstances $I_1, I_2$ (intuitively, one subinstance corresponds to edges and vertices of $G'$ that are drawn "inside" the cycle $W$ in the near-optimal solution to $I'$, and the other subinstance contains all edges and vertices that are drawn "outside" $W$). Each of the resulting two instances contains the cycle $W$, and, in order to be able to combine the solutions to the two subinstances to obtain a solution to $I'$, we contract all vertices and edges of $W$, in each of the two instances, into a supernode. Let $I'_1, I'_2$ denote these two resulting instances. The main difficulty in the analysis is to show that there is a solution to each resulting instance of MCNwRS, such that the sum of the costs of two solutions is close to $\mathsf{OPT}_{\mathsf{cnwrs}}(I')$. The difficulty arises from the fact that we do not know what the optimal solution to instance $I'$ looks like, and so our partition of $G'$ into two subgraphs that are drawn on different sides of the cycle $W$ in that solution may be imprecise. Instead, we need to "fix" the solutions to instances $I_1, I_2$ (that are induced by the optimal solution to $I'$) in order to move all edges and vertices of each subinstance to lie on one side of the cycle $W$. In fact we are unable to do so directly. Instead, we show that we can compute a relatively small collection $E'$ of edges, such that, if we remove the edges of $E'$ from the graphs corresponding to instances $I_1, I_2$, then each of the resulting subinstances has the desired structure: namely, it can be drawn completely inside or completely outside the cycle $W$ with relatively few crossings compared to $\mathsf{OPT}_{\mathsf{cnwrs}}(I')$. After we solve the two resulting subinstances recursively, we combine the resulting solutions, and add the images of the edges of $E'$ back, in order to obtain a solution to instance $I'$.

### 1.3 Organization

We start with preliminaries in Section 2. We then provide, in Section 3, the definitions of several main concepts that we use (such as wide and narrow instances), and state three main technical theorems that allow us to decompose wide and narrow instances. We then provide the proof of Theorem 1.1 using these three theorems. In Section 4 we provide additional definitions, notation and summary of known results that we use, together with some easy extensions. This section can be thought of as an expanded version of preliminaries. We then develop our main technical tools: Basic Cluster Disengagement in Section 5, Cluster Classification Theorem in Section 6 (with parts of the proof delayed to Section 11), and Advanced Cluster Disengagement in Section 7. In Sections 8, 9 and 10 we provide the proofs of the three main theorems. Sections 8 and 9 deal with decomposing wide instances, and Section 9 provides an algorithm for decomposing a narrow instance.

## 2 Preliminaries

By default, all logarithms in this paper are to the base of 2. All graphs are undirected and finite. Graphs may contain parallel edges but they may not contain self loops. Graphs without parallel edges are explicitly referred to as simple graphs.

## 2.1 Graph-Theoretic Notation

We follow standard graph-theoretic notation. Let $G = (V, E)$ be a graph. For a vertex $v \in V$, we denote by $\delta_G(v)$ the set of all edges of $G$ that are incident to $v$, and we denote $\deg_G(v) = |\delta_G(v)|$. For two disjoint subsets $A, B$ of vertices of $G$, we denote by $E_G(A, B)$ the set of all edges with one endpoint in $A$ and the other in $B$. For a subset $S \subseteq V$ of vertices, we denote by $G[S]$ the subgraph of $G$ induced by $S$, by $E_G(S)$ the set of all edges with both endpoints in $S$, and by $\delta_G(S)$ the set of all edges with exactly one endpoint in $S$. Abusing the notation, for a subgraph $C$ of $G$, we use $\delta_G(C)$ to denote $\delta_G(V(C))$.

**Definition 2.1 (Congestion)** *Let $G$ be a graph, let $e$ be an edge of $G$, and let $\mathcal{Q}$ be a set of paths in $G$. The* congestion that the set $\mathcal{Q}$ of paths causes on edge $e$*, denoted by* $\mathrm{cong}_G(\mathcal{Q}, e)$*, is the number of paths in $\mathcal{Q}$ that contain $e$. The* total congestion of $\mathcal{Q}$ in $G$ *is* $\mathrm{cong}_G(\mathcal{Q}) = \max_{e \in E(G)} \{\mathrm{cong}_G(\mathcal{Q}, e)\}$.

## 2.2 Curves in General Position, Graph Drawings, Faces, and Crossings

Let $\gamma$ be an open curve in the plane, and let $P$ be a set of points in the plane. We say that $\gamma$ is *internally disjoint* from $P$ if no inner point of $\gamma$ lies in $P$. In other words, $P \cap \gamma$ may only contain the endpoints of $\gamma$. Given a set $\Gamma$ of open curves in the plane, we say that the curves in $\Gamma$ are *internally disjoint* if, for every pair $\gamma, \gamma' \in \Gamma$ of distinct curves, every point $p \in \gamma \cap \gamma'$ is an endpoint of both curves. We use the following definition of curves in general position.

**Definition 2.2 (Curves in general position)** *Let $\Gamma$ be a finite set of open curves in the plane. We say that the curves of $\Gamma$ are* in general position*, if the following conditions hold:*

- *for every pair $\gamma, \gamma' \in \Gamma$ of distinct curves, there is a finite number of points $p$ with $p \in \gamma \cap \gamma'$;*
- *for every pair $\gamma, \gamma' \in \Gamma$ of distinct curves, an endpoint of $\gamma$ may not serve as an inner point of $\gamma'$ or of $\gamma$; and*
- *for every triple $\gamma, \gamma', \gamma'' \in \Gamma$ of distinct curves, if some point $p$ lies on all three curves, then it must be an endpoint of each of these three curves.*

Let $\Gamma$ be a set of curves in general position, and let $\gamma, \gamma' \in \Gamma$ be a pair of curves. Let $p$ be any point that lies on both $\gamma$ and $\gamma'$, but is not an endpoint of either curve. We then say that point $p$ is a *crossing* between $\gamma$ and $\gamma'$, or that curves $\gamma$ and $\gamma'$ *cross* at point $p$. We are now ready to formally define graph drawings.

**Definition 2.3 (Graph Drawings)** *A* drawing $\varphi$ *of a graph $G$ in the plane is a map $\varphi$, that maps every vertex $v$ of $G$ to a point $\varphi(v)$ in the plane (called the* image of $v$*), and every edge $e = (u, v)$ of $G$ to a simple curve $\varphi(e)$ in the plane whose endpoints are $\varphi(u)$ and $\varphi(v)$ (called the* image of $e$*), such that all points in set $\{\varphi(v) \mid v \in V(G)\}$ are distinct, and the set $\{\varphi(e) \mid e \in E(G)\}$ of curves is in general position. Additionally, for every vertex $v \in V(G)$ and edge $e \in E(G)$, $\varphi(v) \in \varphi(e)$ only if $v$ is an endpoint of $e$.*

Assume now that we are given some drawing $\varphi$ of graph $G$ in the plane, and assume that for some pair $e, e'$ of edges, their images $\varphi(e), \varphi(e')$ cross at point $p$. Then we say that $(e, e')_p$ is a *crossing* in the drawing $\varphi$ (we may sometimes omit the subscript $p$ if the images of the two edges only cross at one point). We also say that $p$ is a *crossing point* of drawing $\varphi$. We denote by $\mathsf{cr}(\varphi)$ the total number of crossings in the drawing $\varphi$.

Note that a drawing of a graph $G$ in the plane naturally defines a drawing of $G$ on the sphere and vice versa; we use both types of drawings.

For convenience, given a drawing $\varphi$ of a graph $G$, we sometimes will not distinguish between the edges of $G$ and their images. For example, we may say that edges $e, e'$ cross in drawing $\varphi$ to indicate that their images cross. Similarly, we may not distinguish between vertices and their images. For example, we may talk about the order in which edges of $\delta_G(v)$ enter vertex $v$ in drawing $\varphi$, to mean the order in which the images of the edges of $\delta_G(v)$ enter the image of $v$. We denote by $\varphi(G) = \left(\bigcup_{e\in E(G)} \varphi(e)\right) \cup \{\varphi(v) \mid v \in V(G)\}$.

**Images of Paths.** Assume that we are given a graph $G$, its drawing $\varphi$, and a path $P$ in $G$. The *image of path $P$ in $\varphi$*, denoted by $\varphi(P)$, is the curve that is obtained by concatenating the images of all edges $e \in E(P)$. Equivalently, $\varphi(P) = \bigcup_{e\in E(P)} \varphi(e)$. If $P = \{v\}$ for some vertex $v$, then $\varphi(P) = \varphi(v)$.

**Planar Graphs and Planar Drawings.** A graph $G$ is *planar* if there is a drawing of $G$ in the plane with no crossings. A drawing $\varphi$ of a graph $G$ in the plane with $\mathsf{cr}(\varphi) = 0$ is called a *planar drawing* of $G$. We use the following result by Hopcroft and Tarjan.

**Theorem 2.4 ([HT74])** *There is an algorithm, that, given a graph $G$, correctly establishes whether $G$ is planar, and if so, computes a planar drawing of $G$. The running time of the algorithm in $O(|V(G)|)$.*

**Faces of a Drawing.** Suppose we are given a graph $G$ and a drawing $\varphi$ of $G$ in the plane or on the sphere. The set of faces of $\varphi$ is the set of all connected regions of $\mathbb{R}^2 \setminus \varphi(G)$. If $G$ is drawn in the plane, then we designate a single face of $\varphi$ as the "outer", or the "infinite" face.

**Identical Drawings and Orientations.** Assume that we are given some planar drawing $\varphi$ of a graph $G$. We can associate, with every face $F$ of this drawing, a subgraph $\partial(F)$ of $G$, containing all vertices and edges of $G$ whose images are contained in the boundary of $F$. Drawing $\varphi$ of $G$ can be uniquely defined by the list $\mathcal{F}$ of all its faces, and, for every face $F \in \mathcal{F}$, the corresponding subgraph $\partial(F)$ of $G$. In particular, if $\varphi, \varphi'$ are two planar drawings of the graph $G$, and there is an one-to-one mapping between the set $\mathcal{F}$ of the faces of $\varphi$ and the set $\mathcal{F}'$ of the faces of $\varphi'$, and, for every face $F$, the graph $\partial(F)$ is identical in both drawings, then we say that drawings $\varphi$ and $\varphi'$ are *identical*.

Assume now that we are given a (possibly non-planar) drawing $\varphi$ of a graph $G$. Let $G'$ be the graph obtained from $G$ by placing a vertex on every crossing of $\varphi$. We then obtain a planar drawing $\psi$ of the resulting graph $G'$, where every vertex $v \in V(G') \setminus V(G)$ corresponds to a unique crossing point of $\varphi$. For every edge $e \in E(G)$, let $L(e)$ be the list of all vertices of $G'$ that correspond to crossings in which edge $e$ participates in $\varphi$, ordered in the order in which these crossings appear on the image of edge $e$ in $\varphi$, as we traverse it from one endpoint to another. Graph $G'$, its planar drawing $\psi$, and the lists $\{L(e)\}_{e\in E(G)}$ uniquely define the drawing $\varphi$ of $G$. In other words, if $\varphi, \varphi'$ are two drawings of the graph $G$, for which (i) the corresponding graphs $G'$ are the same (up to renaming the vertices of $V(G') \setminus V(G)$); (ii) the induced planar drawings $\psi$ of $G'$ are identical; and (iii) the vertex lists $\{L(e)\}_{e\in E(G)}$ are identical, then $\varphi$ and $\varphi'$ are *identical drawings* of $G$.

Assume now that $\varphi$ is a drawing of a graph $G$ in the plane, and let $\varphi'$ be the drawing of $G$ that is the mirror image of $\varphi$. We say that $\varphi$ and $\varphi'$ are *identical* drawings of $G$, and that their *orientations* are *different*, or *opposite*. We sometime say that $\varphi'$ is obtained by *flipping* the drawing $\varphi$.

We say that a graph $G$ is 3-connected, if for every pair $u, v \in V(G)$ of its vertices, $G \setminus \{u, v\}$ is a connected graph. We use the following well known result.

**Theorem 2.5 ([Whi92])** *Every 3-connected planar graph has a unique planar drawing.*

## 2.3 Grids and Their Standard Drawings

The $(r \times r)$-grid is a graph whose vertex set is $\{v_{i,j} \mid 1 \leq i, j \leq r\}$, and edge set is the union of the set $\{(v_{i,j}, v_{i,j+1}) \mid 1 \leq i \leq r, 1 \leq j < r\}$ of *horizontal edges*, and the set $\{(v_{i,j}, v_{i+1,j}) \mid 1 \leq i < r, 1 \leq j \leq r\}$ of *vertical edges*. For $1 \leq i \leq r$, the *ith row* of the grid is the subgraph of the grid graph induced by vertex set $\{v_{i,j} \mid 1 \leq j \leq r\}$. Similarly, for $1 \leq j \leq r$, the *jth column* of the grid is the subgraph of the grid graph induced by vertex set $\{v_{i,j} \mid 1 \leq i \leq r\}$. Given an $(r \times r)$-grid, we refer to vertices $v_{1,1}, v_{1,r}, v_{r,1}$, and $v_{r,r}$ as the *corners* of the grid. We also refer to the graph that is obtained from the union of row 1, row $r$, column 1, and column $r$, as the *boundary* of the grid.

It is not hard to see that the $(r \times r)$-grid has a unique planar drawing (this is since the $(1\times1)$-grid and the $(2\times2)$-grid have unique planar drawings, and for all $r \geq 3$, if we suppress the corner vertices of the grid, we obtain a planar 3-connected graph, that has a unique planar drawing). We refer to this unique planar drawing of the grid as its *standard drawing* (see Figure 2). For all $1 \leq i, j \leq r-1$, we let $\mathsf{Cell}_{i,j}$ be the face of the standard drawing, that contains the images of the vertices $v_{i,j}, v_{i,j+1}, v_{i+1,j}, v_{i+1,j+1}$ on its boundary.

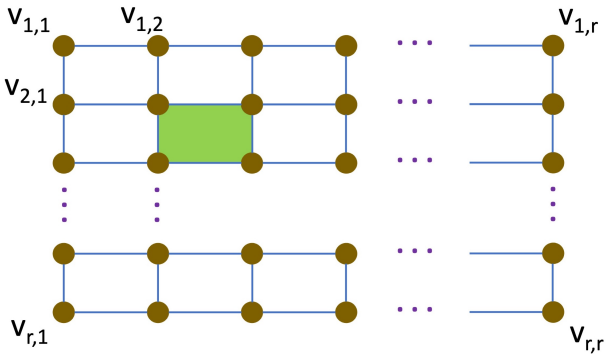



Figure 2: The standard drawing of the $(r \times r)$-grid with $\mathsf{Cell}_{2,2}$ shown in green.

## 2.4 Circular Orderings, Orientations, and Rotation Systems

Suppose we are given a collection $U = \{u_1, \ldots, u_r\}$ of elements. Let $D$ be any disc in the plane. Assume further that we are given, for every element $u_i \in U$, a point $p_i$ on the boundary of $D$, so that all resulting points in $\{p_1, \ldots, p_r\}$ are distinct. As we traverse the boundary of the disc $D$ in the clock-wise direction, the order in which we encounter the points $p_1, \ldots, p_r$ defines a *circular ordering* $\mathcal{O}$ *of the elements of* $U$. If we traverse the boundary of the disc $D$ in the counter-clock-wise direction, we obtain a circular ordering $\mathcal{O}'$ of the elements of $U$, which is the mirror image of the ordering $\mathcal{O}$. We say that the orderings $\mathcal{O}$ and $\mathcal{O}'$ are *identical* but their *orientations* are different, or opposite: $\mathcal{O}$ has a negative and $\mathcal{O}'$ has a positive orientation. Whenever we refer to an ordering $\mathcal{O}$ of elements, we view it as *unoriented* (that is, the orientation can be chosen arbitrarily). When the orientation of the ordering is fixed, we call it an *oriented ordering*, and denote it by $(\mathcal{O}, b)$, where $\mathcal{O}$ is the associated (unoriented) ordering of elements of $U$, and $b \in \{-1, 1\}$ is the orientation, with $b = -1$ indicating a negative (that is, clock-wise), orientation.

We will also consider graph drawings on the sphere. In this case, when we say we traverse the boundary of a disc $D$ in the clock-wise direction, we mean that we traverse the boundary of $D$ so that the interior of $D$ lies to our right. Similarly, we traverse the boundary of $D$ in the counter-clock-wise direction, if the interior of $D$ lies to our left. Circular orderings and orientations are then defined similarly.

Given a graph $G$ and a vertex $v \in V(G)$, a circular ordering $\mathcal{O}_v$ of the edges of $\delta_G(v)$ is called a *rotation*. A collection of circular orderings $\mathcal{O}_v$ for all vertices $v \in V(G)$ is called a *rotation system* for graph $G$.

### 2.5 Tiny $v$-Discs and Drawings that Obey Rotations

Given a graph $G$, its drawing $\varphi$, and a vertex $v \in V(G)$, we will sometimes utilize the notion of a *tiny $v$-disc*, that we define next.

**Definition 2.6 (Tiny $v$-Disc)** *Let $G$ be a graph and let $\varphi$ be a drawing of $G$ on the sphere or in the plane. For each vertex $v \in V(G)$, we denote by $D_\varphi(v)$ a very small disc containing the image of $v$ in its interior, and we refer to $D_\varphi(v)$ as* tiny $v$-disc*. Disc $D_\varphi(v)$ must be small enough to ensure that, for every edge $e \in \delta_G(v)$, the image $\varphi(e)$ of $e$ intersects the boundary of $D_\varphi(v)$ at a single point, and $\varphi(e) \cap D_\varphi(v)$ is a contiguous curve. Additionally, we require that for every vertex $u \in V(G) \setminus \{v\}$, $\varphi(u) \notin D_\varphi(v)$; for every edge $e' \in E(G) \setminus \delta_G(v)$, $\varphi(e') \cap D_\varphi(v) = \emptyset$; and that no crossing point of drawing $\varphi$ is contained in $D_\varphi(v)$. Lastly, we require that all discs in $\{D_\varphi(v) \mid v \in V(G)\}$ are mutually disjoint.*

Consider now a graph $G$, a vertex $v \in V(G)$, and a drawing $\varphi$ of $G$. Consider the tiny $v$-disc $D = D_\varphi(v)$. For every edge $e \in \delta_G(v)$, let $p_e$ be the (unique) intersection of the image $\varphi(e)$ of $e$ and the boundary of the disc $D$. Let $\mathcal{O}$ be the (unoriented) circular ordering in which the points of $\{p_e\}_{e \in \delta_G(v)}$ appear on the boundary of $D$. Then $\mathcal{O}$ naturally defines a circular ordering $\mathcal{O}_v^*$ of the edges of $\delta_G(v)$: ordering $\mathcal{O}_v^*$ is obtained from $\mathcal{O}$ by replacing, for each edge $e \in \delta_G(v)$, point $p_e$ with the edge $e$. We say that *the images of the edges of $\delta_G(v)$ enter the image of $v$ in the order $\mathcal{O}_v^*$* in the drawing $\varphi$. For brevity, we may sometimes say that the edges of $\delta_G(v)$ enter $v$ in the order $\mathcal{O}_v^*$ in $\varphi$. While we view the ordering $\mathcal{O}_v^*$ as unoriented, drawing $\varphi$ also defines an orientation for this ordering. If the points in set $\{p_e \mid e \in \delta_G(v)\}$ are encountered in the order $\mathcal{O}_v^*$ when traversing the boundary of $D$ in the counter-clock-wise direction, then the orientation is 1, and otherwise it is $-1$.

Assume now that we are given a graph $G$ and a rotation system $\Sigma$ for $G$. Let $\varphi$ be a drawing of $G$. Consider any vertex $v \in V(G)$, and its rotation $\mathcal{O}_v \in \Sigma$. We say that the drawing $\varphi$ *obeys the rotation* $\mathcal{O}_v \in \Sigma$, if the order in which the edges of $\delta_G(v)$ enter $v$ in $\varphi$ is precisely $\mathcal{O}_v$ (note that both orderings are unoriented). We say that the *orientation of $v$ is $-1$*, or *negative*, in the drawing $\varphi$ if the orientation of the ordering $\mathcal{O}_v$ of the edges of $\delta_G(v)$ as they enter $v$ is $-1$, and otherwise, the orientation of $v$ in $\varphi$ is 1, or positive. We say that drawing $\varphi$ of $G$ *obeys the rotation system* $\Sigma$, if it obeys the rotation $\mathcal{O}_v \in \Sigma$ for every vertex $v \in V(G)$.

Assume now that we are given a set $\Gamma$ of curves in general position, where each curve $\gamma \in \Gamma$ is an open curve. Let $p$ be any point that serves as an endpoint of at least one curve in $\Gamma$, and let $\Gamma' \subseteq \Gamma$ be the set of curves for which $p$ serves as an endpoint. We then define a *tiny $p$-disc $D(p)$* to be a small disc that contains the point $p$ in its interior; does not contain any other point that serves as an endpoint of a curve in $\Gamma$; and does not contain any crossing point of curves in $\Gamma$. Additionally, we ensure that, for every curve $\gamma \in \Gamma$, if $\gamma \in \Gamma'$, then $\gamma \cap D(p)$ is a simple curve, and otherwise $\gamma \cap D(p) = \emptyset$. For every curve $\gamma \in \Gamma'$, let $q(\gamma)$ be the unique point of $\gamma$ lying on the boundary of the disc $D(p)$. Note that all points in $\{q(\gamma) \mid \gamma \in \Gamma'\}$ are distinct. Let $\mathcal{O}$ be the circular order in which these points are encountered when we traverse the boundary of $D(p)$. As before, this ordering naturally defines a circular ordering $\mathcal{O}'$ of the curves in $\Gamma'$. We then say that the curves of $\Gamma'$ *enter the point $p$ in the order $\mathcal{O}'$*.

### 2.6 Problem Definitions and Trivial Algorithms

In the Minimum Crossing Number problem, the input is an $n$-vertex graph $G$, and the goal is to compute a drawing of $G$ in the plane with minimum number of crossings. The value of the optimal solution,

also called the *crossing number* of $G$, is denoted by $\mathsf{OPT}_{\mathsf{cr}}(G)$.

We also consider a closely related problem called Minimum Crossing Number with Rotation System (MCNwRS). In this problem, the input is a graph $G$, and a rotation system $\Sigma$ for $G$. Given an instance $I = (G, \Sigma)$ of the MCNwRS problem, we say that a drawing $\varphi$ of $G$ is a *feasible solution* for $I$ if $\varphi$ obeys the rotation system $\Sigma$. The *cost* of the solution is the number of crossings in $\varphi$. The goal in the MCNwRS problem is to compute a feasible solution to the given input instance $I$ of smallest possible cost. We denote the cost of the optimal solution of the MCNwRS instance $I$ by $\mathsf{OPT}_{\mathsf{cnwrs}}(I)$.

We use the following two simple theorems about the MCNwRS problem, whose proofs are deferred to Appendix B.1 and Appendix B.2, respectively.

**Theorem 2.7** *There is an efficient algorithm, that, given an instance $I = (G, \Sigma)$ of MCNwRS, correctly determines whether $\mathsf{OPT}_{\mathsf{cnwrs}}(I) = 0$, and, if so, computes a feasible solution to instance $I$ of cost 0.*

**Theorem 2.8** *There is an efficient algorithm, that given an instance $I = (G, \Sigma)$ of MCNwRS, computes a feasible solution to $I$, of cost at most $|E(G)|^2$.*

We refer to the solution computed by the algorithm from Theorem 2.8 as a *trivial solution*. We will also use the following lemma from [CMT20], that allows us to insert edges into a partial solution to MCNwRS problem instance.

**Lemma 2.9 (Lemma 9.2 of [CMT20])** *There is an efficient algorithm, that, given an instance $I = (G, \Sigma)$ of the MCNwRS problem, a subset $E' \subseteq E(G)$ of edges of $G$, and a drawing $\varphi$ of graph $G \setminus E'$ that obeys $\Sigma$, computes a solution $\varphi'$ to instance $I$, with $\mathsf{cr}(\varphi') \leq \mathsf{cr}(\varphi) + |E'| \cdot |E(G)|$.*

## 2.7 A $\nu$-Decomposition of an Instance

A central tool that we use in our divide-and-conquer algorithm is a $\nu$-decomposition of instances.

**Definition 2.10 ($\nu$-Decomposition of Instances)** *Let $I = (G, \Sigma)$ be an instance of MCNwRS with $|E(G)| = m$, and let $\nu \geq 1$ be a parameter. We say that a collection $\mathcal{I}$ of instances of MCNwRS is a $\nu$-decomposition of $I$, if the following hold:*

- *D1.* $\sum_{I'=(G',\Sigma') \in \mathcal{I}} |E(G')| \leq m \cdot (\log m)^{O(1)}$;
- *D2.* $\sum_{I' \in \mathcal{I}} \mathsf{OPT}_{\mathsf{cnwrs}}(I') \leq (\mathsf{OPT}_{\mathsf{cnwrs}}(I) + m) \cdot \nu$; *and*
- *D3. there is an efficient algorithm $\mathsf{Alg}(\mathcal{I})$, that, given, a feasible solution $\varphi(I')$ to every instance $I' \in \mathcal{I}$, computes a feasible solution $\varphi$ to instance $I$, of cost $\mathsf{cr}(\varphi) \leq O\left(\sum_{I' \in \mathcal{I}} \mathsf{cr}(\varphi(I'))\right)$.*

*We say that a randomized algorithm Alg is a* $\nu$-decomposition algorithm for a family $\mathcal{I}^*$ of instances of MCNwRS *if Alg is an efficient algorithm, that, given an instance $I = (G, \Sigma) \in \mathcal{I}^*$, produces a collection $\mathcal{I}$ of instances that has properties D1 and D3, and ensures the following additional property (that replaces Property D2):*

- *D'2.* $\mathbf{E}\left[\sum_{I' \in \mathcal{I}} \mathsf{OPT}_{\mathsf{cnwrs}}(I')\right] \leq (\mathsf{OPT}_{\mathsf{cnwrs}}(I) + |E(G)|) \cdot \nu$.

In the following claim, whose proof appears in Appendix B.3, we show that algorithms for computing $\nu$-decompositions can be naturally composed together.

**Claim 2.11** *Let* $\mathsf{Alg}_1$ *be a randomized* $\nu'$*-decomposition algorithm for some family* $\mathcal{I}^*$ *of instances of* MCNwRS*. Assume that, given an instance* $I \in \mathcal{I}^*$*, algorithm* $\mathsf{Alg}_1$ *produces a collection* $\mathcal{I}'$ *of instances, all of which belong to some family* $\mathcal{I}^{**}$ *of instances of* MCNwRS*. Let* $\mathsf{Alg}_2$ *be a randomized* $\nu''$*-decomposition algorithm for family* $\mathcal{I}^{**}$ *of instances of* MCNwRS*. Lastly, let* Alg *be a randomized algorithm, that, given an instance* $I \in \mathcal{I}^*$ *of* MCNwRS*, applies Algorithm* $\mathsf{Alg}_1$ *to* $I$*, to obtain a collection* $\mathcal{I}'$ *of instances, and then, for every instance* $I' \in \mathcal{I}'$*, applies Algorithm* $\mathsf{Alg}_2$ *to* $I'$*, obtaining a collection* $\mathcal{I}''(I')$ *of instances. The output of algorithm* Alg *is the collection* $\mathcal{I} = \bigcup_{I' \in \mathcal{I}'} \mathcal{I}''(I')$ *of instances of* MCNwRS*. Then* Alg *is a randomized* $\nu$*-decomposition algorithm for family* $\mathcal{I}^*$ *of instances of* MCNwRS*, for* $\nu = \nu'' \cdot \max\left\{2\nu', (\log m)^{O(1)}\right\}$*, where* $m$ *is the number of edges in instance* $I$*.*

### 2.8 Subinstances

We use the following definition of subinstances.

**Definition 2.12 (Subinstances)** *Let* $I = (G, \Sigma)$ *and* $I' = (G', \Sigma')$ *be two instances of* MCNwRS*. We say that instance* $I'$ *is a* subinstance *of instance* $I$*, if there is a subgraph* $\tilde{G} \subseteq G$*, and a collection* $S_1, \ldots, S_r$ *of mutually disjoint subsets of vertices of* $\tilde{G}$*, such that graph* $G'$ *can be obtained from* $\tilde{G}$ *by contracting, for all* $1 \leq i \leq r$*, every vertex set* $S_i$ *into a supernode* $u_i$*; we keep parallel edges but remove self-loops*[3]*. We do not distinguish between the edges incident to the supernodes in graph* $G'$ *and their counterparts in graph* $G$*. For every vertex* $v \in V(G') \cap V(G)$*, its rotation* $\mathcal{O}'_v$ *in* $\Sigma'$ *must be consistent with the rotation* $\mathcal{O}_v \in \Sigma$*, while for every supernode* $u_i$*, its rotation* $\mathcal{O}'_{u_i}$ *in* $\Sigma'$ *can be defined arbitrarily.*

Observe that, if instance $I' = (G', \Sigma')$ is a subinstance of $I = (G, \Sigma)$, then $|E(G')| \leq |E(G)|$. Also notice that the subinstance relation is transitive: if instance $I_1$ is a subinstance of instance $I_0$, and instance $I_2$ is a subinstance of $I_1$, then $I_2$ is a subinstance of $I_0$.

## 3 An Algorithm for MCNwRS– Proof of Theorem 1.1

In this section we provide the proof of Theorem 1.1, with some of the details deferred to subsequent sections. Throughout the paper, we denote by $I^* = (G^*, \Sigma^*)$ the input instance of the MCNwRS problem, and we denote $m^* = |E(G^*)|$. We also use the following parameter that is central to our algorithm: $\mu = 2^{c^*(\log m^*)^{7/8} \log\log m^*}$, where $c^*$ is a large enough constant.

As mentioned already, our algorithm for solving the MCNwRS problem is recursive, and, over the course of the recursion, we will consider various other instances $I$ of MCNwRS. Throughout the algorithm, parameters $m^*$ and $\mu$ remain unchanged, and are defined with respect to the original input instance $I^*$. The main technical ingredient of the proof is the following theorem.

**Theorem 3.1** *There is a constant* $c''$*, and an efficient randomized algorithm, that, given an instance* $I = (G, \Sigma)$ *of* MCNwRS *with* $m = |E(G)|$*, such that* $\mu^{c''} \leq m \leq m^*$*, either returns FAIL, or computes a collection* $\mathcal{I}$ *of instances of* MCNwRS *with the following properties:*

- *for every instance* $I' = (G', \Sigma') \in \mathcal{I}$*,* $|E(G')| \leq m/\mu$*;*
- $\sum_{I'=(G',\Sigma') \in \mathcal{I}} |E(G')| \leq m \cdot (\log m)^{O(1)}$*;*
- *there is an efficient algorithm called* AlgCombineDrawings*, that, given a solution* $\varphi(I')$ *to every instance* $I' \in \mathcal{I}$*, computes a solution* $\varphi$ *to instance* $I$*; and*

[3]Note that this definition is similar to the definition of a minor, except that we do not require that the induced subgraphs $G[S_i]$ of $G$ are connected.

- *if* $\mathsf{OPT}_{\mathsf{cnwrs}}(I) \leq |E(G)|^2/\mu^{c''}$*, then with probability at least* $15/16$*, all of the following hold:*
  - *the algorithm does not return FAIL;*
  - $\mathcal{I} \neq \emptyset$*;*
  - $\sum_{I' \in \mathcal{I}} \mathsf{OPT}_{\mathsf{cnwrs}}(I') \leq (\mathsf{OPT}_{\mathsf{cnwrs}}(I) + m) \cdot 2^{O((\log m)^{3/4} \log\log m)}$*; and*
  - *if algorithm* AlgCombineDrawings *is given as input a solution* $\varphi(I')$ *to every instance* $I' \in \mathcal{I}$*, then the resulting solution* $\varphi$ *to instance* $I$ *that it computes has cost at most:*

$$O\left(\sum_{I' \in \mathcal{I}} \mathsf{cr}(\varphi(I'))\right) + (\mathsf{OPT}_{\mathsf{cnwrs}}(I) + m) \cdot \mu^{O(1)}.$$

The remainder of this paper is dedicated to the proof of Theorem 3.1. In the following subsection, we complete the proof of Theorem 1.1 using Theorem 3.1.

### 3.1 Proof of Theorem 1.1

Throughout the proof, we assume that $m^*$ is larger than a sufficiently large constant, since otherwise we can return a trivial solution to instance $I^*$, from Theorem 2.8.

We let $c_g > 100$ be a large enough constant, so that, for example, when the algorithm from Theorem 3.1 is applied to an instance $I = (G, \Sigma)$ with $m = |E(G)|$, such that $\mu^{c''} \leq m \leq m^*$ holds, it is guaranteed to return a family $\mathcal{I}$ of instances of MCNwRS, with $\sum_{I'=(G',\Sigma') \in \mathcal{I}} |E(G')| \leq m \cdot (\log m)^{c_g}$. We say that the algorithm from Theorem 3.1 is *successful* if all of the following hold:

- the algorithm does not return FAIL;
- if $\mathcal{I}$ is the collection of instances returned by the algorithm, then $\mathcal{I} \neq \emptyset$;
- $\sum_{I' \in \mathcal{I}} \mathsf{OPT}_{\mathsf{cnwrs}}(I') \leq (\mathsf{OPT}_{\mathsf{cnwrs}}(I) + m) \cdot 2^{c_g((\log m)^{3/4} \log\log m)}$; and
- if algorithm AlgCombineDrawings is given a solution $\varphi(I')$ to every instance $I' \in \mathcal{I}$, then it computes a solution $\varphi$ to instance $I$, of cost at most $c_g \cdot (\sum_{I' \in \mathcal{I}} \mathsf{cr}(\varphi(I')) + (\mathsf{OPT}_{\mathsf{cnwrs}}(I) + m) \cdot \mu^{c_g}$.

By letting $c_g$ be a large enough constant, Theorem 3.1 guarantees that, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I) \leq |E(G)|^2/\mu^{c''}$, then with probability at least $15/16$ the algorithm is successful. We assume that the parameter $c^*$ in the definition of $\mu$ is sufficiently large, so that, e.g., $c^* > 2c_g$.

We use a simple recursive algorithm called AlgRecursiveCNwRS, that appears in Figure 3.

In order to analyze the algorithm, it is convenient to associate a *partitioning tree* $T$ with it, whose vertices correspond to all instances of MCNwRS considered over the course of the algorithm. Let $L = \lceil \log m^* \rceil$. We start with the tree $T$ containing a single root vertex $v(I^*)$, representing the input instance $I^*$. Consider now some vertex $v(I)$ of the tree, representing some instance $I = (G, \Sigma)$. When Algorithm AlgRecursiveCNwRS was applied to instance $I$, if it did not terminate after the first three steps, it constructed $L$ collections $\mathcal{I}_1(I), \ldots, \mathcal{I}_L(I)$ of instances (some of which may be empty, in case the algorithm from Theorem 3.1 returned FAIL in the corresponding iteration). For each such instance $I' \in \bigcup_{j=1}^{L} \mathcal{I}_j(I)$, we add a vertex $v(I')$ representing instance $I'$ to $T$, that becomes a child vertex of $v(I)$. This concludes the description of the partitioning tree $T$.

We denote by $\mathcal{I}^* = \{I \mid v(I) \in V(T)\}$ the set of all instances of MCNwRS, whose corresponding vertex appears in the tree $T$. For each such instance $I \in \mathcal{I}^*$, its *recursive level* is the distance from vertex $v(I)$ to the root vertex $v(I^*)$ in the tree $T$ (so the recursive level of $v(I^*)$ is 0). For $j \geq 0$, we denote by $\hat{\mathcal{I}}_j \subseteq \mathcal{I}^*$ the set of all instances $I \in \mathcal{I}^*$, whose recursive level is $j$. Lastly, the *depth* of the tree

AlgRecursiveCNwRS

**Input:** an instance $I = (G, \Sigma)$ of the MCNwRS problem, with $|E(G)| \leq m^*$.
**Output:** a feasible solution to instance $I$.

1. Use the algorithm from Theorem 2.7 to determine whether $\mathsf{OPT}_{\mathsf{cnwrs}}(I) = 0$. If so, use the algorithm from Theorem 2.7 to compute a solution to $I$ of cost 0. Return this solution, and terminate the algorithm.
2. Use the algorithm from Theorem 2.8 to compute a trivial solution $\varphi'$ to instance $I$.
3. If $|E(G)| \leq \mu^{c''}$, return the trivial solution $\varphi'$ and terminate the algorithm.
4. For $1 \leq j \leq \lceil \log m^* \rceil$:
   (a) Apply the algorithm from Theorem 3.1 to instance $I$.
   (b) If the algorithm returns FAIL, let $\varphi_j = \varphi'$ be the trivial solution to instance $I$, and set $\mathcal{I}_j(I) = \emptyset$.
   (c) Otherwise:
      i. Let $\mathcal{I}_j(I)$ be the collection of instances computed by the algorithm.
      ii. For every instance $I' \in \mathcal{I}_j(I)$, apply Algorithm AlgRecursiveCNwRS to instance $I'$, to obtain a solution $\varphi(I')$ to this instance.
      iii. Apply Algorithm AlgCombineDrawings from Theorem 3.1 to solutions $\{\varphi(I')\}_{I' \in \mathcal{I}_j(I)}$, to obtain a solution $\varphi_j$ to instance $I$.

   Return a solution to instance $I$ from among $\left\{\varphi', \varphi_1, \ldots, \varphi_{\lceil \log m^* \rceil}\right\}$ that has fewest crossings.

Figure 3: AlgRecursiveCNwRS

$T$, denoted by $\mathsf{dep}(T)$, is the largest recursive level of any instance in $\mathcal{I}^*$. In order to analyze the algorithm, we start with the following two simple observations.

**Observation 3.2** $\mathsf{dep}(T) \leq \frac{(\log m^*)^{1/8}}{c^* \log\log m^*}$.

**Proof:** Consider any non-root vertex $v(I)$ in the tree $T$, and let $v(I')$ be the parent-vertex of $v(I)$. Denote $I = (G, \Sigma)$ and $I' = (G', \Sigma')$. From the construction of tree $T$, instance $I$ belongs to some collection of instances obtained by applying the algorithm from Theorem 3.1 to instance $I'$. Therefore, from Theorem 3.1, $|E(G)| \leq |E(G')|/\mu$ must hold. Therefore, for all $j \geq 0$, for every instance $I = (G, \Sigma) \in \hat{\mathcal{I}}_j$, $|E(G)| \leq m^*/\mu^j$. Since $\mu = 2^{c^*(\log m^*)^{7/8} \log\log m^*}$, we get that $\mathsf{dep}(T) \leq \frac{(\log m^*)^{1/8}}{c^* \log\log m^*}$. □

**Observation 3.3** $\sum_{I=(G,\Sigma)\in\mathcal{I}^*} |E(G)| \leq m^* \cdot 2^{(\log m^*)^{1/8}}$.

**Proof:** Consider any non-leaf vertex $v(I)$ of the tree $T$, and denote $I = (G, \Sigma)$. Recall that, when Algorithm $\mathsf{AlgRecursiveCNwRS}$ is applied to instance $I$, it uses the algorithm from Theorem 3.1 to compute $L$ collections $\mathcal{I}_1(I), \ldots, \mathcal{I}_L(I)$ of instances, such that, if we denote $|E(G)| = m$, then, for all $1 \leq j \leq L$:

$$\sum_{I'=(G',\Sigma')\in\mathcal{I}_j(I)} |E(G')| \leq m \cdot (\log m)^{c_g} \leq m \cdot (\log m^*)^{c_g}$$

(since $m \leq m^*$ must hold). Since $L \leq 2\log m^*$, and $m^*$ is sufficiently large, we get that:

$$\sum_{j=1}^{L} \sum_{I'=(G',\Sigma')\in\mathcal{I}_j(I)} |E(G')| \leq m \cdot (\log m^*)^{c_g+2}.$$

For all $j \geq 0$, we denote by $N_j$ the total number of edges in all instances in set $\hat{\mathcal{I}}_j$, $N_j = \sum_{I=(G,\Sigma)\in\hat{\mathcal{I}}_j} |E(G)|$. Clearly, $N_0 = m^*$, and, from the above discussion, for all $j > 0$, $N_j \leq N_{j-1} \cdot (\log m^*)^{c_g+2}$.

Since $\mathsf{dep}(T) \leq \frac{(\log m^*)^{1/8}}{c^* \log\log m^*}$, we conclude that:

$$\sum_{I=(G,\Sigma)\in\mathcal{I}^*} |E(G)| \leq m^* \cdot (\log m^*)^{2c_g \cdot (\log m^*)^{1/8}/(c^* \log\log m^*)} \leq m^* \cdot 2^{(\log m^*)^{1/8}},$$

since $c_g \leq c^*/2$. □

We use the following corollary, that follows immediately from Observation 3.3.

**Corollary 3.4** *The number of instances $I = (G, \Sigma) \in \mathcal{I}^*$ with $|E(G)| \geq \mu^{c''}$ is at most $m^*$.*

We say that an instance $I \in \mathcal{I}^*$ is a *leaf instance*, if vertex $v(I)$ is a leaf vertex of the tree $T$, and we say that it is a non-leaf instance otherwise. Consider now a non-leaf instance $I = (G, \Sigma) \in \mathcal{I}^*$. We say that a bad event $\mathcal{E}(I)$ happens, if $0 < \mathsf{OPT}_{\mathsf{cnwrs}}(I) \leq |E(G)|^2/\mu^{c''}$, and, for all $1 \leq j \leq L$, the $j$th application of the algorithm from Theorem 3.1 to instance $I$ was unsuccessful. Clearly, from Theorem 3.1, $\mathbf{Pr}[\mathcal{E}(I)] \leq (1/16)^L \leq 1/(m^*)^4$. Let $\mathcal{E}$ be the bad event that event $\mathcal{E}(I)$ happened for any instance $I \in \mathcal{I}^*$. From the Union Bound and Corollary 3.4, we get that $\mathbf{Pr}[\mathcal{E}] \leq 1/(m^*)^2$. We use the following immediate observation.

**Observation 3.5** *If Event $\mathcal{E}$ does not happen, then for every leaf vertex $v(I)$ of $T$ with $I = (G, \Sigma)$, either $|E(G)| \leq \mu^{c''}$; or $\mathsf{OPT}_{\mathsf{cnwrs}}(I) = 0$; or $\mathsf{OPT}_{\mathsf{cnwrs}}(I) > |E(G)|^2/\mu^{c''}$.*

We use the following lemma to complete the proof of Theorem 1.1.

**Lemma 3.6** *If Event $\mathcal{E}$ does not happen, then Algorithm* AlgRecursiveCNwRS *computes a solution for instance $I^* = (G^*, \Sigma^*)$ of cost at most $2^{O((\log m^*)^{7/8} \log\log m^*)} \cdot (\mathsf{OPT}_{\mathsf{cnwrs}}(I^*) + |E(G^*)|)$.*

**Proof:** Consider a non-leaf instance $I = (G, \Sigma)$, and let $\mathcal{I}_1(I), \ldots, \mathcal{I}_L(I)$ be families of instances of MCNwRS that Algorithm AlgRecursiveCNwRS computed, when applied to instance $I$. Recall that, for each $1 \leq j \leq L$ with $\mathcal{I}_j(I) \neq \emptyset$, the algorithm computes a solution $\varphi_j$ to instance $I$, by first solving each of the instances in $\mathcal{I}_j(I)$ recursively, and then combining the resulting solutions using Algorithm AlgCombineDrawings. Eventually, the algorithm returns the best solution of $\{\varphi', \varphi_1, \ldots, \varphi_L\}$, where $\varphi'$ is the trivial solution, whose cost is at most $|E(G)|^2$. We fix an arbitrary index $1 \leq j \leq L$, such that the $j$th application of the algorithm from Theorem 3.1 to instance $I$ was successful. Note that the cost of the solution to instance $I$ that the algorithm returns is at most $\mathsf{cr}(\varphi_j)$. We then *mark* the vertices of $\{v(I') \mid I' \in \mathcal{I}_j(I)\}$ in the tree $T$. We also mark the root vertex of the tree.

Let $T^*$ be the subgraph of $T$ induced by all marked vertices. It is easy to verify that $T^*$ is a tree, and moreover, if $\mathcal{E}$ did not happen, every leaf vertex of $T^*$ is also a leaf vertex of $T$. For a vertex $v(I) \in V(T^*)$, we denote by $h(I)$ the length of the longest path in tree $T^*$, connecting vertex $v(I)$ to any of its descendants in the tree. We use the following claim, whose proof is straightforward conceptually but somewhat technical; we defer the proof to Appendix C.1.

**Claim 3.7** *Assume that Event $\mathcal{E}$ did not happen. Then there is a fixed constant $\tilde{c} \geq \max\{c'', c_g, c^*\}$, such that, for every vertex $v(I) \in V(T^*)$, whose corresponding instance is denoted by $I = (G, \Sigma)$, the cost of the solution that the algorithm computes for $I$ is at most:*

$$2^{\tilde{c} \cdot h(I) \cdot (\log m^*)^{3/4} \log\log m^*} \cdot \mu^{c'' \cdot c_g} \cdot \mathsf{OPT}_{\mathsf{cnwrs}}(I) + (\log m^*)^{4c_g h(I)} \mu^{2c'' \cdot \tilde{c}} \cdot |E(G)|.$$

We are now ready to complete the proof of Lemma 3.6. Recall that $h(I^*) = \mathsf{dep}(T^*) \leq \mathsf{dep}(T) \leq \frac{(\log m^*)^{1/8}}{c^* \log\log m^*}$ from Observation 3.2. Therefore, from Claim 3.7, the cost of the solution that the algorithm computes for instance $I^*$ is bounded by:

$$\begin{aligned}
&2^{O(\mathsf{dep}(T)) \cdot (\log m^*)^{3/4} \log\log m^*} \cdot \mu^{O(1)} \cdot \mathsf{OPT}_{\mathsf{cnwrs}}(I^*) + (\log m^*)^{O(\mathsf{dep}(T))} \cdot \mu^{O(1)} \cdot m^* \\
&\qquad \leq 2^{O((\log m^*)^{7/8})} \cdot \mu^{O(1)} \cdot \mathsf{OPT}_{\mathsf{cnwrs}}(I^*) + (\log m^*)^{O((\log m^*)^{1/8}/\log\log m^*)} \cdot \mu^{O(1)} \cdot m^* \\
&\qquad \leq 2^{O((\log m^*)^{7/8} \log\log m^*)} \cdot (\mathsf{OPT}_{\mathsf{cnwrs}}(I^*) + |E(G^*)|)\,,
\end{aligned}$$

since $\mu = 2^{O((\log m^*)^{7/8} \log\log m^*)}$. □

In order to complete the proof of Theorem 1.1, it is now enough to prove Theorem 3.1. The remainder of the paper is dedicated to the proof of Theorem 3.1.

## 3.2 Proof of Theorem 3.1 – Main Definitions and Theorems

We classify instances of MCNwRS into *wide* and *narrow*. Wide instances are, in turn, classified into *well-connected* and not well-connected instances. We then provide different algorithms for decomposing instances of each of the resulting three kinds. We use the following notion of a high-degree vertex.

**Definition 3.8 (High-degree vertex)** *Let $G$ be any graph. A vertex $v \in V(G)$ is a* high-degree *vertex, if $\deg_G(v) \geq |E(G)|/\mu^4$.*

We are now ready to define wide and narrow instances.

**Definition 3.9 (Wide and Narrow Instances)** *Let $I = (G, \Sigma)$ be an instance of* MCNwRS *with $|E(G)| = m$. We say that $I$ is a* wide *instance, if there is a high-degree vertex $v \in V(G)$, a partition*

$(E_1, E_2)$ *of the edges of* $\delta_G(v)$*, such that the edges of* $E_1$ *appear consecutively in the rotation* $\mathcal{O}_v \in \Sigma$*, and so do the edges of* $E_2$*, and there is a collection* $\mathcal{P}$ *of at least* $\lfloor m/\mu^{50} \rfloor$ *simple edge-disjoint cycles in* $G$*, such that every cycle* $P \in \mathcal{P}$ *contains one edge of* $E_1$ *and one edge of* $E_2$*. An instance that is not wide is called* narrow*.*

Note that there is an efficient algorithm to check whether a given instance $I$ of MCNwRS is wide, and, if so, to compute the corresponding set $\mathcal{P}$ of cycles, via standard algorithms for maximum flow. (For every vertex $v \in V(G)$, we try all possible partitions $(E_1, E_2)$ of $\delta_G(v)$ with the required properties, as the number of such partitions is bounded by $|\delta_G(v)|^2$.) We will use the following simple observation regarding narrow instances.

**Observation 3.10** *If an instance* $I = (G, \Sigma)$ *of* MCNwRS *is narrow, then for every pair* $u, v$ *of distinct high-degree vertices of* $G$*, and any set* $\mathcal{P}$ *of edge-disjoint paths connecting* $u$ *to* $v$ *in* $G$*,* $|\mathcal{P}| \leq 2\left\lceil |E(G)|/\mu^{50} \right\rceil$ *must hold.*

**Proof:** Assume for contradiction that $I = (G, \Sigma)$ is a narrow instance of MCNwRS, with $|E(G)| = m$, and that there are two high-degree vertices $u, v$ of $G$, and a set $\mathcal{P}$ of more than $2\left\lceil m/\mu^{50} \right\rceil$ edge-disjoint paths in $G$ connecting $u$ to $v$. We denote $|\mathcal{P}| = k$. Let $E' \subseteq \delta_G(v)$ be the set of all edges $e \in \delta_G(v)$, such that $e$ is the first edge on some path in $\mathcal{P}$. We denote $E' = \{e_1, \ldots, e_k\}$, where the edges are indexed according to their ordering in the rotation $\mathcal{O}_v \in \Sigma$. We also denote $\mathcal{P} = \{P(e_i) \mid 1 \leq i \leq k\}$, where path $P(e_i)$ contains the edge $e_i$ as its first edge. We can then compute a partition $(E_1, E_2)$ of $\delta_G(v)$, such that the edges of $E_1$ appear consecutively in the rotation $\mathcal{O}_v \in \Sigma$, and so do the edges of $E_2$. Additionally, we can ensure that $e_1, \ldots, e_{\lceil k/2 \rceil} \in E_1$, while the remaining edges of $E'$ lie in $E_2$. For each $1 \leq i \leq \left\lceil m/\mu^{50} \right\rceil$, we let $Q_i$ be the cycle obtained by concatenating the paths $P(e_i)$ and $P(e_{k-i+1})$. We turn $Q_i$ into a simple cycle, by removing from it all cycles that are disjoint from vertex $v$. It is then immediate to verify that the set $\left\{Q_i \mid 1 \leq i \leq \left\lceil m/\mu^{50} \right\rceil\right\}$ of cycles has all the required properties to establish that instance $I$ is wide, a contradiction. □

Next, we define well-connected wide instances.

**Definition 3.11 (Well-Connected Wide Instances)** *Let* $I = (G, \Sigma)$ *be a wide instance of* MCNwRS *with* $|E(G)| = m$*. We say that it is a* well-connected *instance iff for every pair* $u, v$ *of distinct vertices of* $G$ *with* $\deg_G(v), \deg_G(u) \geq m/\mu^5$*, there is a collection of at least* $\frac{8m}{\mu^{50}}$ *edge-disjoint paths connecting* $u$ *to* $v$ *in* $G$*.*

The proof of Theorem 3.1 relies on the following three theorems. The first theorem deals with wide instances that are not necessarily well-connected. Its proof is deferred to Section 8.

**Theorem 3.12** *There is an efficient randomized algorithm, whose input is a wide instance* $I = (G, \Sigma)$ *of* MCNwRS*, with* $m = |E(G)|$*, such that* $\mu^{20} \leq m \leq m^*$ *holds. The algorithm computes a* $\nu$*-decomposition* $\mathcal{I}$ *of* $I$*, for* $\nu = 2^{O((\log m)^{3/4} \log\log m)}$*, such that every instance* $I' = (G', \Sigma') \in \mathcal{I}$ *is a subinstance of* $I$*, and one of the following holds for it:*

- *either* $|E(G')| \leq m/\mu$*;*
- *or* $I'$ *is a narrow instance;*
- *or* $I'$ *is a wide and well-connected instance.*

The second theorem deals with wide well-connected instances. Its proof appears in Section 9.

**Theorem 3.13** *There is an efficient randomized algorithm, whose input is a wide and well-connected instance $I = (G, \Sigma)$ of* MCNwRS*, with $m = |E(G)|$, such that $\mu^{c'} \leq m \leq m^*$ holds, for some large enough constant $c'$. The algorithm either returns FAIL, or computes a non-empty collection $\mathcal{I}$ of instances of* MCNwRS*, such that the following hold:*

- $\sum_{I'=(G',\Sigma')\in\mathcal{I}} |E(G')| \leq 2|E(G)|$;
- *for every instance $I' = (G', \Sigma') \in \mathcal{I}$, either $|E(G')| \leq m/\mu$, or instance $I'$ narrow;*
- *there is an efficient algorithm called* AlgCombineDrawings$'$*, that, given a solution $\varphi(I')$ to every instance $I' \in \mathcal{I}$, computes a solution $\varphi$ to instance $I$; and*
- *if $\mathsf{OPT}_{\mathsf{cnwrs}}(I) \leq m^2/\mu^{c'}$ then, with probability at least $1 - 1/\mu^2$, all of the following hold:*
  - *the algorithm does not return FAIL;*
  - *$\sum_{I'\in\mathcal{I}} \mathsf{OPT}_{\mathsf{cnwrs}}(I') \leq \mathsf{OPT}_{\mathsf{cnwrs}}(I) \cdot (\log m)^{O(1)}$; and*
  - *if algorithm* AlgCombineDrawings$'$ *is given as input a solution $\varphi(I')$ to every instance $I' \in \mathcal{I}$, then the resulting solution $\varphi$ to instance $I$ that it computes has cost at most: $\mathsf{cr}(\varphi) \leq \sum_{I'\in\mathcal{I}} \mathsf{cr}(\varphi(I')) + \mathsf{OPT}_{\mathsf{cnwrs}}(I) \cdot \mu^{O(1)}$.*

The third theorem deals with narrow instances, and its proof appears in Section 10.

**Theorem 3.14** *There is an efficient randomized algorithm, whose input is a narrow instance $I = (G, \Sigma)$ of* MCNwRS*, with $m = |E(G)|$, such that $\mu^{50} \leq |E(G)| \leq 2m^*$. The algorithm either returns FAIL, or computes a $\nu$-decomposition $\mathcal{I}$ of $I$, for $\nu = 2^{O((\log m)^{3/4} \log\log m)}$, such that, for every instance $I' = (G', \Sigma') \in \mathcal{I}$, $|E(G')| \leq m/(2\mu)$. Moreover, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I) < m^2/\mu^{21}$, then the probability that the algorithm returns FAIL is at most $O(1/\mu^2)$.*

The majority of the remainder of this paper is dedicated to the proofs of the above three theorems. Before we provide these proofs, we develop central technical tools that they use, in Sections 5 – 7. In the remainder of this section, we complete the proof of Theorem 3.1 using Theorems 3.12, 3.13, and 3.14.

Recall that we are given an instance $I = (G, \Sigma)$ of MCNwRS, with $\mu^{c''} \leq |E(G)| \leq m^*$, for some large enough constant $c''$. We assume that $c'' > 100c'$, where $c'$ is the constant in Theorem 3.13. We use another large constant $c'_g$, and we assume that $c^* > c'_g > c''$, where $c^*$ is the constant in the definition of the parameter $\mu$. Throughout, we denote $m = |E(G)|$. We compute the desired collection $\mathcal{I}^*$ of instances in three steps.

### Step 1

Assume first that the input instance $I$ is a wide instance. We apply the algorithm from Theorem 3.12 to $I$. Let $\hat{\mathcal{I}}$ be the resulting collection of instances. We partition the set $\hat{\mathcal{I}}$ of instances into three subsets. The first set, denoted by $\hat{\mathcal{I}}_{\mathsf{small}}$, contains all instances $I' = (G', \Sigma') \in \hat{\mathcal{I}}$ with $|E(G')| \leq m/\mu$. The second set, denoted by $\hat{\mathcal{I}}^{(n)}_{\mathsf{large}}$, contains all narrow instances in $\hat{\mathcal{I}} \setminus \hat{\mathcal{I}}_{\mathsf{small}}$. The third set, denoted by $\hat{\mathcal{I}}^{(w)}_{\mathsf{large}}$, contains all remaining instances of $\hat{\mathcal{I}}$. From Theorem 3.12, every instance in $\hat{\mathcal{I}}^{(w)}_{\mathsf{large}}$ is wide and well-connected. Since every instance $I' = (G', \Sigma') \in \hat{\mathcal{I}}$ is a subinstance of $I$, $|E(G')| \leq |E(G)| \leq m^*$ must hold. Recall that, from Theorem 3.12, $\hat{\mathcal{I}}$ is a $\nu_1$-decomposition for $I$, for $\nu_1 = 2^{O((\log m)^{3/4} \log\log m)}$. Therefore:

$$\sum_{I'=(G',\Sigma')\in\hat{\mathcal{I}}} |E(G')| \leq m\cdot(\log m)^{c'_g}, \tag{1}$$

and

$$\mathbf{E}\left[\sum_{I'\in\hat{\mathcal{I}}} \mathsf{OPT}_{\mathsf{cnwrs}}(I')\right] \leq (\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\cdot\nu_1.$$

**Bad Event $\mathcal{E}_1$.** We say that a bad event $\mathcal{E}_1$ happens if $\sum_{I'\in\hat{\mathcal{I}}}\mathsf{OPT}_{\mathsf{cnwrs}}(I') > 100\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\cdot\nu_1$. From the Markov Bound, $\mathbf{Pr}[\mathcal{E}_1]\leq 1/100$. Note that, if event $\mathcal{E}_1$ did not happen, then for each instance $I'\in\hat{\mathcal{I}}$, $\mathsf{OPT}_{\mathsf{cnwrs}}(I')\leq 100\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\cdot\nu_1$. We need the following simple observation.

**Observation 3.15** *Assume that $\mathsf{OPT}_{\mathsf{cnwrs}}(I)\leq m^2/\mu^{c''}$, and that Event $\mathcal{E}_1$ did not happen. Then for every instance $I'=(G',\Sigma')\in\hat{\mathcal{I}}^{(n)}_{\mathit{large}}\cup\hat{\mathcal{I}}^{(w)}_{\mathit{large}}$, $\mathsf{OPT}_{\mathsf{cnwrs}}(I')\leq|E(G')|^2/\mu^{c'}$.*

**Proof:** If $\mathsf{OPT}_{\mathsf{cnwrs}}(I)\leq m^2/\mu^{c''}$, and Event $\mathcal{E}_1$ did not happen, then for every instance $I'\in\hat{\mathcal{I}}^{(n)}_{\mathsf{large}}\cup\hat{\mathcal{I}}^{(w)}_{\mathsf{large}}$:

$$\mathsf{OPT}_{\mathsf{cnwrs}}(I')\leq 100\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\cdot\nu_1\leq\mu\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\leq\mu\cdot\left(\frac{m^2}{\mu^{c''}}+m\right)\leq\frac{m^2}{\mu^{c'}}$$

(since $c''>100c'$ is a large enough constant and $m\geq\mu^{c''}$). □

Assume now that instance $I$ is a narrow instance. Then we simply set $\hat{\mathcal{I}}=\hat{\mathcal{I}}^{(n)}_{\mathsf{large}}=\{I\}$ and $\hat{\mathcal{I}}_{\mathsf{small}}=\hat{\mathcal{I}}^{(w)}_{\mathsf{large}}=\emptyset$. This completes the description of the first step.

**Step 2**

In the second step, we apply the algorithm from Theorem 3.13 to every instance $I'\in\hat{\mathcal{I}}^{(w)}_{\mathsf{large}}$. If the algorithm returns FAIL, then we terminate our algorithm and return FAIL as well. Assume now that the algorithm from Theorem 3.13, when applied to instance $I'$, did not return FAIL. We let $\tilde{\mathcal{I}}(I')$ be the collection of instances that the algorithm computes. Recall that we are guaranteed that, for each instance $\tilde{I}=(\tilde{G},\tilde{\Sigma})\in\tilde{\mathcal{I}}(I')$, either $\tilde{I}$ is a narrow instance, or $|E(\tilde{G})|\leq\frac{|E(G')|}{\mu}\leq\frac{m}{\mu}$ (we have used the fact that $|E(G')|\leq m$, since $I'=(G',\Sigma')$ is a subinstance of $I$). Additionally, we are guaranteed that:

$$\sum_{\tilde{I}=(\tilde{G},\tilde{\Sigma})\in\tilde{\mathcal{I}}(I')}|E(\tilde{G})|\leq 2|E(G')|. \tag{2}$$

In particular, for every instance $\tilde{I}=(\tilde{G},\tilde{\Sigma})\in\tilde{\mathcal{I}}(I')$, $|E(\tilde{G})|\leq 2|E(G')|\leq 2m\leq 2m^*$.

We say that the application of the algorithm from Theorem 3.13 to an instance $I'=(G',\Sigma')\in\hat{\mathcal{I}}^{(w)}_{\mathsf{large}}$ is *successful*, if (i) the algorithm does not return FAIL; (ii) $\sum_{\tilde{I}\in\tilde{\mathcal{I}}(I')}\mathsf{OPT}_{\mathsf{cnwrs}}(\tilde{I})\leq\mathsf{OPT}_{\mathsf{cnwrs}}(I')\cdot(\log m)^{c'_g}$; and (iii) there is an efficient algorithm $\mathsf{AlgCombineDrawings}'$, that, given a solution $\varphi(\tilde{I})$ to every instance $\tilde{I}\in\tilde{\mathcal{I}}(I')$, computes a solution $\varphi(I')$ to instance $I'$ with $\mathsf{cr}(\varphi(I'))\leq\sum_{\tilde{I}\in\tilde{\mathcal{I}}(I')}\mathsf{cr}(\varphi(\tilde{I}))+\mathsf{OPT}_{\mathsf{cnwrs}}(I')\cdot\mu^{c'_g}$.

**Bad Event $\mathcal{E}_2$.** For an instance $I' = (G', \Sigma') \in \hat{\mathcal{I}}^{(w)}_{\mathsf{large}}$, we say that a bad event $\mathcal{E}_2(I')$ happens if the algorithm from Theorem 3.13, when applied to instance $I'$, was not successful. From Theorem 3.13 and Observation 3.15, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I) \leq m^2/\mu^{c''}$, then $\mathbf{Pr}\left[\mathcal{E}_2(I') \mid \neg\mathcal{E}_1\right] \leq 1/\mu^2$ (since we can assume that $c'_g$ is a large enough constant).

We let $\mathcal{E}_2$ be the bad event that at least of the events in $\left\{\mathcal{E}_2(I') \mid I' \in \hat{\mathcal{I}}^{(w)}_{\mathsf{large}}\right\}$ happened.

Recall that, from the definition of the set $\mathcal{I}^{(w)}_{\mathsf{large}}$ of instances, for every instance $I' = (G', \Sigma') \in \hat{\mathcal{I}}^{(w)}_{\mathsf{large}}$, $|E(G')| \geq \frac{m}{\mu}$ holds. On the other hand, from Equation 1,

$$\sum_{I'=(G',\Sigma')\in\hat{\mathcal{I}}^{(w)}_{\mathsf{large}}} |E(G')| \leq \sum_{I'=(G',\Sigma')\in\hat{\mathcal{I}}} |E(G')| \leq m\cdot(\log m)^{c'_g}.$$

Therefore, $|\hat{\mathcal{I}}^{(w)}_{\mathsf{large}}| \leq \mu\cdot(\log m)^{c'_g}$. From the Union Bound, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I) \leq \frac{m^2}{\mu^{c''}}$, then $\mathbf{Pr}\left[\mathcal{E}_2 \mid \neg\mathcal{E}_1\right] \leq \frac{\mu\cdot(\log m)^{c'_g}}{\mu^2} \leq \frac{1}{100}$.

Let $\tilde{\mathcal{I}} = \bigcup_{I'\in\hat{\mathcal{I}}^{(w)}_{\mathsf{large}}} \tilde{\mathcal{I}}(I')$. Note that, from Inequalities 1 and 2, we get that:

$$\sum_{\tilde{I}=(\tilde{G},\tilde{\Sigma})\in\tilde{\mathcal{I}}} |E(\tilde{G})| \leq 2m\cdot(\log m)^{c'_g}. \tag{3}$$

We partition the instances in set $\tilde{\mathcal{I}}$ into two subsets: set $\tilde{\mathcal{I}}_{\mathsf{small}}$, containing all instances $\tilde{I} = (\tilde{G}, \tilde{\Sigma})$ in $\tilde{\mathcal{I}}$ with $|E(\tilde{G})| \leq m/\mu$, and set $\tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}$, containing all remaining instances. From Theorem 3.13, every instance $\tilde{I} \in \tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}$ is narrow. This completes the description of the second step.

### Step 3

We focus on four sets of instances that we have constructed so far: $\hat{\mathcal{I}}_{\mathsf{small}}, \hat{\mathcal{I}}^{(n)}_{\mathsf{large}}, \tilde{\mathcal{I}}_{\mathsf{small}}, \tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}$. Recall that, if instance $I' = (G', \Sigma')$ belongs to set $\hat{\mathcal{I}}_{\mathsf{small}} \cup \tilde{\mathcal{I}}_{\mathsf{small}}$, then $|E(G')| \leq m/\mu$. If instance $I' = (G', \Sigma')$ belongs to set $\hat{\mathcal{I}}^{(n)}_{\mathsf{large}}, \cup\tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}$, then $m/\mu < |E(G')| \leq 2m$, and instance $I'$ is narrow. We use the following simple observation.

**Observation 3.16** *If $\mathsf{OPT}_{\mathsf{cnwrs}}(I) \leq m^2/\mu^{c''}$, and neither of the events $\mathcal{E}_1, \mathcal{E}_2$ happened, then for every instance $I' = (G', \Sigma') \in \hat{\mathcal{I}}^{(n)}_{large} \cup \tilde{\mathcal{I}}^{(n)}_{large}$, $\mathsf{OPT}_{\mathsf{cnwrs}}(I') < |E(G')|^2/\mu^{21}$.*

**Proof:** From Observation 3.15, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I) \leq m^2/\mu^{c''}$, and the bad event $\mathcal{E}_1$ did not happen, then for every instance $I' = (G', \Sigma') \in \hat{\mathcal{I}}^{(n)}_{\mathsf{large}} \cup \hat{\mathcal{I}}^{(w)}_{\mathsf{large}}$, $\mathsf{OPT}_{\mathsf{cnwrs}}(I') \leq |E(G')|^2/\mu^{c'}$.

Consider now some instance $I' = (G', \Sigma') \in \hat{\mathcal{I}}^{(w)}_{\mathsf{large}}$. If, additionally, event $\mathcal{E}_2$ did not happen, then:

$$\sum_{\tilde{I}\in\tilde{\mathcal{I}}(I')} \mathsf{OPT}_{\mathsf{cnwrs}}(\tilde{I}) \leq \mathsf{OPT}_{\mathsf{cnwrs}}(I')\cdot(\log m)^{c'_g}.$$

Therefore, for every instance $\tilde{I} \in \tilde{\mathcal{I}}(I')$:

$$\mathsf{OPT}_{\mathsf{cnwrs}}(\tilde{I}) \leq \mathsf{OPT}_{\mathsf{cnwrs}}(I')\cdot(\log m)^{c'_g} \leq \frac{|E(G')|^2}{\mu^{c'}}\cdot(\log m)^{c'_g} \leq \frac{m^2}{\mu^{c'-1}}.$$

We conclude that for every instance $\tilde{I} = (\tilde{G}, \tilde{\Sigma}) \in \tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}$, $\mathsf{OPT}_{\mathsf{cnwrs}}(\tilde{I}) \leq \frac{m^2}{\mu^{c'-1}}$. Since, from the definition of the set $\tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}$ of instances, $|E(\tilde{G})| \geq \frac{m}{\mu}$, we get that:

$$\mathsf{OPT}_{\mathsf{cnwrs}}(\tilde{I}) \leq \frac{m^2}{\mu^{c'-1}} < \frac{|E(\tilde{G})|^2}{\mu^{21}},$$

assuming that $c'$ is a large enough constant. □

Next, we process every instance $I' \in \hat{\mathcal{I}}^{(n)}_{\mathsf{large}} \cup \tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}$ one by one. Notice that for each such instance $I' = (G', \Sigma')$, $|E(G')| \geq m/\mu \geq \mu^{50}$ must hold, since $m \geq \mu^{c''}$. Additionally, as observed already, $|E(G')| \leq 2m \leq 2m^*$. When instance $I' = (G', \Sigma')$ is processed, we apply the algorithm from Theorem 3.14 to it. If the algorithm returns FAIL, then we terminate the algorithm and return FAIL as well. Otherwise, we obtain a collection $\overline{\mathcal{I}}(I')$ of instances of MCNwRS. From Theorem 3.14, for every instance $I'' = (G'', \Sigma'') \in \overline{\mathcal{I}}(I')$, $|E(G'')| \leq \frac{|E(G')|}{2\mu} \leq \frac{m}{\mu}$. Moreover, from the definition of a $\nu$-decomposition of an instance, and from the fact that $|E(G')| \leq 2m$, we get that:

$$\sum_{I''=(G'',\Sigma'') \in \overline{\mathcal{I}}(I')} |E(G'')| \leq |E(G')| \cdot (\log m)^{c'_g}. \tag{4}$$

**Bad Events $\mathcal{E}_3$ and $\mathcal{E}$.** For an instance $I' = (G', \Sigma') \in \hat{\mathcal{I}}^{(n)}_{\mathsf{large}} \cup \tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}$, we say that the bad event $\mathcal{E}_3(I')$ happens if the algorithm from Theorem 3.14, when applied to instance $I'$, returns FAIL. From Theorem 3.14, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I') < |E(G')|^2/\mu^{21}$, then the probability that the algorithm returns FAIL is at most $O(1/\mu^2)$. Therefore, from Observation 3.16, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I) \leq m^2/\mu^{c''}$, then $\mathbf{Pr}\left[\mathcal{E}_3(I') \mid \neg\mathcal{E}_1 \wedge \neg\mathcal{E}_2\right] \leq O(1/\mu^2)$. We let $\mathcal{E}_3$ to be the bad event that $\mathcal{E}_3(I')$ happened for any instance $I' \in \hat{\mathcal{I}}^{(n)}_{\mathsf{large}} \cup \tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}$. Recall that, for every instance $I' = (G', \Sigma') \in \hat{\mathcal{I}}^{(n)}_{\mathsf{large}} \cup \tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}$, $|E(G')| \geq \frac{m}{\mu}$. On the other hand, from Inequality 1, $\sum_{I'=(G',\Sigma') \in \hat{\mathcal{I}}^{(n)}_{\mathsf{large}}} |E(G')| \leq m \cdot (\log m)^{c'_g}$, and from Inequality 3, $\sum_{I'=(G',\Sigma') \in \tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}} |E(G')| \leq 2m \cdot (\log m)^{c'_g}$. Therefore, $|\hat{\mathcal{I}}^{(n)}_{\mathsf{large}} \cup \tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}| \leq 3\mu \cdot (\log m)^{c'_g}$. From the Union Bound, assuming that the constant $c^*$ in the definition of the parameter $\mu$ is large enough, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I) \leq m^2/\mu^{c''}$, then $\mathbf{Pr}\left[\mathcal{E}_3 \mid \neg\mathcal{E}_1 \wedge \neg\mathcal{E}_2\right] \leq O\left(\frac{\mu \cdot (\log m)^{c'_g}}{\mu^2}\right) \leq \frac{1}{100}$. Lastly, we define bad event $\mathcal{E}$ to be the event that at least one of the events $\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3$ happened. Note that $\mathbf{Pr}\left[\mathcal{E}\right] \leq \mathbf{Pr}\left[\mathcal{E}_1\right] + \mathbf{Pr}\left[\mathcal{E}_2 \mid \neg\mathcal{E}_1\right] + \mathbf{Pr}\left[\mathcal{E}_3 \mid \neg\mathcal{E}_1 \wedge \neg\mathcal{E}_2\right]$. Therefore, altogether, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I) \leq m^2/\mu^{c''}$, then $\mathbf{Pr}\left[\mathcal{E}\right] \leq \frac{3}{100} \leq \frac{1}{30}$. Note that, if bad event $\mathcal{E}$ does not happen, then the algorithm does not return FAIL.

If the third step of the algorithm did not terminate with a FAIL, we let $\overline{\mathcal{I}}_{\mathsf{small}} = \bigcup_{I' \in \hat{\mathcal{I}}^{(n)}_{\mathsf{large}} \cup \tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}} \overline{\mathcal{I}}(I')$. By combining Equations 1, 3 and 4, we get that:

$$\sum_{I''=(G'',\Sigma'') \in \overline{\mathcal{I}}_{\mathsf{small}}} |E(G'')| \leq 3m \cdot (\log m)^{2c'_g}. \tag{5}$$

The output of the algorithm is the collection $\mathcal{I}^* = \hat{\mathcal{I}}_{\mathsf{small}} \cup \tilde{\mathcal{I}}_{\mathsf{small}} \cup \overline{\mathcal{I}}_{\mathsf{small}}$ of instances of MCNwRS. From the above discussion, for every instance $I'' = (G'', \Sigma'') \in \mathcal{I}^*$, $|E(G'')| \leq m/\mu$. As discussed already, if bad event $\mathcal{E}$ does not happen, then the algorithm does not return FAIL.

From now on we assume that the algorithm did not return FAIL. From Inequalities 1, 3 and 5, we get that:

$$\sum_{I''=(G'',\Sigma'')\in\mathcal{I}^*} |E(G'')| \leq 6m\cdot(\log m)^{2c'_g}.$$

Next, we provide Algorithm $\mathsf{AlgCombineDrawings}$ in the following claim, whose proof is conceptually straightforward but somewhat technical, and is deferred to Section C.2 of Appendix.

**Claim 3.17** *There is an efficient algorithm, called* $\mathsf{AlgCombineDrawings}$*, that, given a solution* $\varphi(I'')$ *to every instance* $I''\in\mathcal{I}^*$*, computes a solution* $\varphi(I)$ *to instance* $I$*. Moreover, if* $\mathsf{OPT}_{\mathsf{cnwrs}}(I)\leq m^2/\mu^{c''}$*, and event* $\mathcal{E}$ *did not happen, then* $\mathsf{cr}(\varphi(I))\leq O(\sum_{I''\in\mathcal{I}^*}\mathsf{cr}(\varphi(I'')))+(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\cdot\mu^{O(1)}$*.*

The following observation, whose proof is deferred to Section C.3 of Appendix, will complete the proof of Theorem 3.1.

**Observation 3.18** *If* $\mathsf{OPT}_{\mathsf{cnwrs}}(I)\leq|E(G)|^2/\mu^{c'}$ *and bad event* $\mathcal{E}$ *did not happen, then for some constant* $c$*, with probability at least* $99/100$*:*

$$\sum_{I''\in\mathcal{I}^*}\mathsf{OPT}_{\mathsf{cnwrs}}(I'')\leq(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\cdot 2^{c(\log m)^{3/4}\log\log m}.$$

Let $\mathcal{E}'$ be the bad event that $\sum_{I''\in\mathcal{I}^*}\mathsf{OPT}_{\mathsf{cnwrs}}(I'')>(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\cdot 2^{c(\log m)^{3/4}\log\log m}$. Clearly, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I)\leq m^2/\mu^{c''}$, then the probability that either of the events $\mathcal{E}$ or $\mathcal{E}'$ happens is at most $\mathbf{Pr}[\mathcal{E}]+\mathbf{Pr}[\mathcal{E}'\mid\neg\mathcal{E}]\leq 1/16$. Therefore, we conclude that, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I)\leq m^2/\mu^{c''}$, then with probability at least $15/16$, all of the following hold: (i) the algorithm does not return FAIL; (ii) $\mathcal{I}^*\neq\emptyset$; (iii) $\sum_{I''\in\mathcal{I}^*}\mathsf{OPT}_{\mathsf{cnwrs}}(I'')\leq(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\cdot 2^{O((\log m)^{3/4}\log\log m)}$; and (iv) if algorithm $\mathsf{AlgCombineDrawings}$ is given as input a solution $\varphi(I'')$ to every instance $I''\in\mathcal{I}^*$, then the resulting solution $\varphi$ to instance $I$ that it computes has cost at most: $O\left(\sum_{I''\in\mathcal{I}^*}\mathsf{cr}(\varphi(I''))\right)+(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\cdot\mu^{O(1)}$. This concludes the proof of Theorem 3.1 from Theorems 3.12, 3.13, and 3.14.

# 4 Definitions, Notation, Known Results, and their Easy Extensions

In this section we provide additional definitions and notation, together with known results and their easy extensions that we use throughout the paper.

## 4.1 Clusters, Paths, Flows, and Routers

### 4.1.1 Clusters and Augmentations of Clusters

Let $G$ be a graph. A *cluster* of $G$ is a vertex-induced connected subgraph of $G$. For a set $\mathcal{C}$ of mutually disjoint clusters of $G$, we denote by $E^{\mathsf{out}}_G(\mathcal{C})$ the set of all edges $e=(u,v)$ of $G$, with endpoints $u$ and $v$ lying in distinct clusters of $\mathcal{C}$. We sometimes omit the subscript $G$ when clear from the context.

Next, we define the notion of augmentation of a cluster.

**Definition 4.1 (Augmentation of Clusters)** *Let* $C$ *be a cluster of a graph* $G$*. The* augmentation *of cluster* $C$*, denoted by* $C^+$*, is a graph that is obtained from* $G$ *as follows. First, we subdivide every edge* $e\in\delta_G(C)$ *with a vertex* $t_e$*, and let* $T(C)=\{t_e\mid e\in\delta_G(C)\}$ *be the resulting set of newly added vertices. We then let* $C^+$ *be the subgraph of the resulting graph induced by the set* $V(C)\cup T(C)$ *of vertices.*

### 4.1.2 Paths and Flows

As mentioned already, all graphs that we consider in this paper are undirected. However, sometimes it will be convenient for us to assign direction to paths in such graphs. We do so by designating one endpoint of the path as its first endpoint, and another endpoint as its last endpoint. We will then view the path as being directed from its first endpoint towards its last endpoint. We will sometimes refer to a path with an assigned direction as a directed path, even though the underlying graph is an undirected graph.

Let $G$ be a graph, and let $\mathcal{P}$ be a collection of paths in $G$. We say that the paths of $\mathcal{P}$ are *edge-disjoint* if every edge of $G$ belongs to at most one path of $\mathcal{P}$. We say that the paths in $\mathcal{P}$ are *vertex-disjoint* if every vertex of $G$ belongs to at most one path of $\mathcal{P}$. We say that the paths in $\mathcal{P}$ are *internally disjoint* if every vertex $v \in V(G)$ that serves as an inner vertex of some path in $\mathcal{P}$ only belongs to one path of $\mathcal{P}$. Given a subset $S$ of vertices of $G$, we say that the paths in $\mathcal{P}$ are *internally disjoint from* $S$ if no vertex of $S$ serves as an inner vertex of any path in $\mathcal{P}$. Abusing the notation, for a subgraph $C$ of $G$, we sometimes say that a set $\mathcal{P}$ of paths is internally disjoint from $C$ to indicate that it is internally disjoint from $V(C)$.

**Flows.** Let $G$ be a graph, and let $\mathcal{P}$ be a collection of directed paths in graph $G$. A *flow* over the set $\mathcal{P}$ of paths is an assignment of non-negative values $f(P) \geq 0$, called *flow-values*, to every path $P \in \mathcal{P}$. We sometimes refer to paths in $\mathcal{P}$ as *flow-paths for flow* $f$. For each edge $e \in E(G)$, let $\mathcal{P}(e) \subseteq \mathcal{P}$ be the set of all paths whose first edge is $e$, and let $\mathcal{P}'(e) \subseteq \mathcal{P}$ be the set of all paths whose last edge is $e$. We say that edge $e$ *sends* $z$ *flow units* in $f$ if $\sum_{P \in \mathcal{P}(e)} f(e) = z$, and we say that edge $e$ *receives* $z$ *flow units* in $f$ if $\sum_{P \in \mathcal{P}'(e)} f(P) = z$. Similarly, for a vertex $v \in V(G)$, we say that $v$ sends $z$ flow units in $f$ if the sum of flow-values of all paths $P \in \mathcal{P}$ that originate at $v$ is $z$. We say that $v$ receives $z$ flow units in $f$ if the sum of the flow-values of all paths $P \in \mathcal{P}$ that terminate at $v$ is $z$. The *congestion* that flow $f$ causes on an edge $e$ is $\sum_{\substack{P \in \mathcal{P}: \\ e \in E(P)}} f(P)$, and the *total congestion* of the flow $f$ is the maximum congestion that it causes on any edge $e \in E(G)$.

An *$s$-$t$ flow network* consists of a graph $G$, non-negative capacities $c(e) \geq 0$ for each edge $e \in E(G)$, and two special vertices: source $s$ and destination $t$. Let $\mathcal{P}$ be the set of all paths in graph $G$ originating at $s$ and terminating at $t$. An *$s$-$t$ flow* in $G$ is a flow $f$ that is defined over the set $\mathcal{P}$ of paths, such that for every edge $e \in E(G)$, the congestion that $f$ causes on edge $e$ is at most $c(e)$. The *value* of the flow is $\sum_{P \in \mathcal{P}} f(P)$. Maximum *$s$-$t$ flow* is an *$s$-$t$ flow* of largest possible value. We say that a flow $f$ is *integral* if, for every path $P$, value $f(P)$ is an integer. It is a well known fact (called *integrality of flow*) that, if all edge capacities in a flow network are integral, then there is a maximum *$s$-$t$ flow* that is integral, and such a flow can be found efficiently. In case where the capacity of every edge is unit, such a flow defines a maximum-cardinality collection of edge-disjoint *$s$-$t$* paths.

**Congestion Reduction.** We repeatedly use the following simple claim, whose proof follows from integrality of flow, and appears in Appendix D.1.

**Claim 4.2** *Let $G$ be a graph and let $\mathcal{P}$ be a set of directed paths in $G$. For each vertex $v \in V(G)$, let $n_S(v)$ and $n_T(v)$ denote the numbers of paths in $\mathcal{P}$ originating and terminating at $v$, respectively. Then there is a set $\mathcal{P}'$ of at least $|\mathcal{P}|/\operatorname{cong}_G(\mathcal{P})$ edge-disjoint directed paths in $G$, such that, for every vertex $v$, at most $n_S(v)$ paths of $\mathcal{P}'$ originate at $v$, and at most $n_T(v)$ paths of $\mathcal{P}'$ terminate at $v$. Moreover, there is an efficient algorithm, that, given $G$ and $\mathcal{P}$, computes a set $\mathcal{P}'$ of paths with these properties.*

### 4.1.3 Routing Paths, Internal Routers and External Routers

**Routing Paths.** Suppose we are given a graph $G$, two sets $S, T \subseteq V(G)$ of its vertices, and a set $\mathcal{Q}$ of paths. We say that $\mathcal{Q}$ is a *routing of vertices of $S$ to vertices of $T$*, or that $\mathcal{Q}$ *routes vertices of $S$ to vertices of $T$* if $\mathcal{Q} = \{Q_v \mid v \in S\}$, and, for every vertex $v \in S$, path $Q_v$ originates at $v$ and terminates at a vertex of $T$. If, additionally, for every vertex $t \in T$, exactly one path in $\mathcal{Q}$ terminates at $t$, then we say that $\mathcal{Q}$ is a *one-to-one routing* of vertices of $S$ to vertices of $T$.

Similarly, given two sets $E_1, E_2$ of edges of $G$, we say that a set $\mathcal{Q} = \{Q_e \mid e \in E_1\}$ of paths is a *routing of edges of $E_1$ to edges of $E_2$*, or that $\mathcal{Q}$ *routes edges of $E_1$ to edges of $E_2$*, if, for every edge $e \in E_1$, path $Q_e$ has $e$ as its first edge, and some edge of $E_2$ as its last edge. If, additionally, every edge of $E_2$ serves as the last edge of exactly one path in $\mathcal{Q}$, then we say that $\mathcal{Q}$ is a *one-to-one routing of edges of $E_1$ to edges of $E_2$*.

Next, we define the notions of internal and external routers for clusters, which are central notions that are used throughout our algorithms.

**Definition 4.3 (Internal and External Routers for Clusters)** *Let $G$ be a graph, let $C$ be a cluster of $G$, and let $\mathcal{Q}(C)$ be a set of paths in $G$. We say that $\mathcal{Q}(C)$ is an* internal router *for $C$, or an* internal $C$-router*, if there is some vertex $u \in V(C)$, such that $\mathcal{Q}(C) = \{Q_e \mid e \in \delta_G(C)\}$, and, for each edge $e \in \delta_G(C)$, path $Q_e$ has $e$ as its first edge, $u$ as its last vertex, and all edges of $E(Q_e) \setminus \{e\}$ lie in $C$. We refer to vertex $u$ as the* center of the router*. Similarly, we say that a set $\mathcal{Q}'(C)$ of paths in $G$ is an* external router *for $C$, or an* external $C$-router*, if there is some vertex $u' \in V(G) \setminus V(C)$, such that $\mathcal{Q}'(C) = \{Q'_e \mid e \in \delta_G(C)\}$, and, for each edge $e \in \delta_G(C)$, path $Q_e$ has $e$ as its first edge, $u'$ as its last vertex, is internally disjoint from $C$. We refer to $u'$ as the* center of the router*. We denote by $\Lambda_G(C)$ the set of all internal $C$-routers, and by $\Lambda'_G(C)$ the set of all external $C$-routers in $G$. We may omit the subscript $G$ when clear from the context.*

Throughout the paper, we will be working with distributions over the set $\Lambda_G(C)$ of internal $C$-routers and distributions over the set $\Lambda'_G(C)$ of external $C$-routers for various clusters $C$ of a given graph $G$. We say that a distribution $\mathcal{D}$ over a set $U$ of elements is given *explicitly*, if we are given a list $U' \subseteq U$ of elements, whose probability in $\mathcal{D}$ is non-zero, together with their associated probability values. We say that distribution $\mathcal{D}$ is given *implicitly* if we are given an efficient randomized algorithm that draws an element from $U$ according to the distribution. When the distribution $\mathcal{D}$ is over a set of routers in a graph $G$, the running time of the algorithm should be bounded by $\mathrm{poly}(|E(G)|)$.

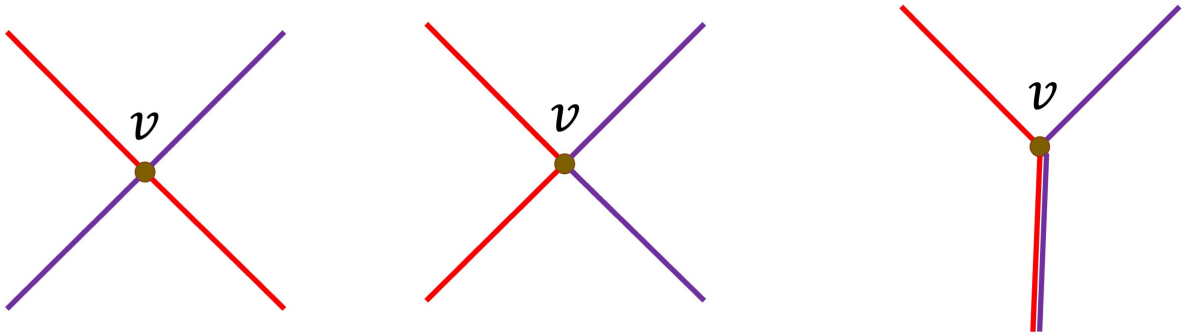
### 4.1.4 Non-Transversal Paths and Path Splicing

We start by defining the notions of transversal and non-transversal intersections of paths and cycles, which we then use to define non-transversal paths.

**Definition 4.4 (Non-transversal Intersection of Paths and Cycles)** *Let $I = (G, \Sigma)$ be an intance of* MCNwRS*, let $P_1, P_2$ be two simple paths in $G$, and let $u$ be a vertex in $V(P_1) \cap V(P_2)$. Denote by $E_1$ the set of (one or two) edges of $P_1$ that are incident to $u$, and similarly denote by $E_2$ the set of (one or two) edges of $P_2$ that are incident to $u$. We say that the intersection of the paths $P_1, P_2$ at vertex $u$ is* non-transversal with respect to $\Sigma$ *if one of the following holds:*

- *either the set $E_1 \cup E_2$ contains fewer than 4 distinct edges; or*
- *$E_1 = \{e_1, e'_1\}$ and $E_2 = \{e_2, e'_2\}$, all edges in set $\{e_1, e'_1, e_2, e'_2\}$ are distinct, and they appear in the ordering $\mathcal{O}_u \in \Sigma$ in one of the following circular orders: $(e_1, e'_1, e_2, e'_2)$, or $(e_1, e'_1, e'_2, e_2)$ (recall that the orderings are unoriented, so the reversals of the above two orderings are also included in this definition).*

*Otherwise, we say that the intersection of the paths $P_1, P_2$ at vertex $u$ is* transversal *(see Figure 4). If $R_1, R_2$ are simple cycles in $G$, and $u$ is a vertex in $V(R_1) \cap V(R_2)$, then we classify the intersection of $R_1$ and $R_2$ and $u$ as transversal or non-transversal with respect to $\Sigma$ similarly.*




(a) The intersection of paths $P_1$ (red) and $P_2$ (purple) is transversal at $v$.

(b) The intersection of path $P_1$ (red) and path $P_2$ (purple) is non-transversal at $v$.

Figure 4: Transversal and non-transversal intersections of paths.

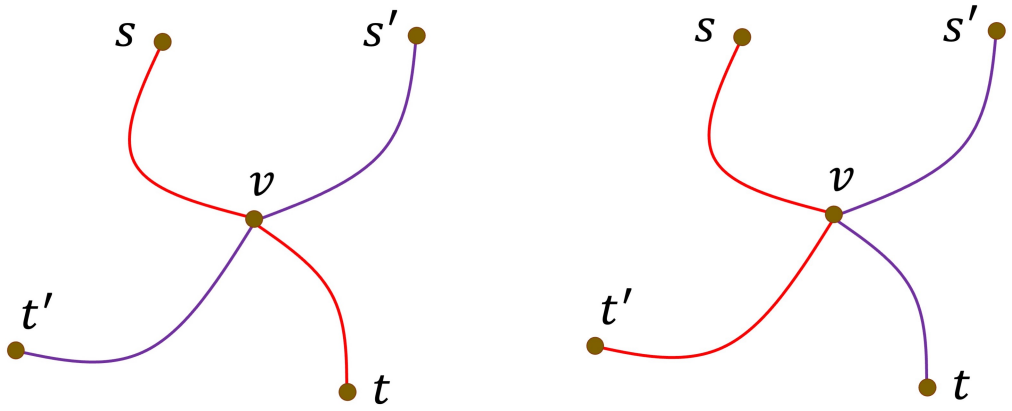
**Definition 4.5 (Non-transversal Set of Paths)** *Let $I = (G, \Sigma)$ be an intance of* MCNwRS*, and let $\mathcal{P}$ be a collection of simple paths in $G$. We say that the set $\mathcal{P}$ of paths is* non-transversal with respect to $\Sigma$ *if, for every pair $P_1, P_2 \in \mathcal{P}$ of paths, for every vertex $u \in V(P_1) \cap V(P_2)$, the intersection of $P_1$ and $P_2$ at $u$ is non-transversal with respect to $\Sigma$.*

Assume now that we are given some instance $I = (G, \Sigma)$ of MCNwRS, and a collection $\mathcal{Q}$ of simple paths in $G$. We let $\Pi^T(\mathcal{Q})$ denote the set of all triples $(Q, Q', v)$, such that $Q, Q' \in \mathcal{Q}$, $v$ is an inner vertex of both $Q$ and $Q'$, and the intersection of $Q$ and $Q'$ at $v$ is transversal with respect to $\Sigma$.

We need to design a subroutine, that, given a set $\mathcal{Q}$ of simple directed paths in a graph $G$, transforms it into a set $\mathcal{Q}'$ of paths that is non-transversal with respect to the given rotation system $\Sigma$ for $G$. We need to ensure that the multisets containing the first vertex of every path in $\mathcal{Q}$ and in $\mathcal{Q}'$, respectively, remain unchanged, and the same holds for multisets containing the last vertex of every path in both path sets. We also need to ensure that for each edge $e \in E(G)$, $\mathrm{cong}_G(\mathcal{Q}') \leq \mathrm{cong}_G(\mathcal{Q})$. Below we provide a procedure for performing such a transformation. The procedure uses a simple subroutine that we call *path splicing* and describe next.

**Path Splicing.** Suppose we are given an instance $I = (G, \Sigma)$ of MCNwRS, two simple paths $P, P'$ in $G$, and a vertex $v$, that serves as an inner vertex of both $P$ and $P'$, such that the intersection of $P$ and $P'$ at vertex $v$ is transversal with respect to $\Sigma$. We assume that each of the paths $P, P'$ is assigned a direction, and we denote by $s$ and $t$ the first and the last endpoints of $P$, respectively, and by $s'$ and $t'$ the first and the last endpoints of $P'$, respectively. The *splicing* of $P$ and $P'$ at vertex $v$ produces two new paths: path $\tilde{P}$, that is a concatenation of the subpath of $P$ from $s$ to $v$, and the subpath of $P'$ from $v$ to $t'$; and path $\tilde{P}'$, that is a concatenation of the subpath of $P'$ from $s'$ to $v$, and the subpath of $P$ from $v$ to $t$. See Figure 5 for an illustration.

For a set $\mathcal{P}$ of directed paths in a graph $G$, we denote by $S(\mathcal{P})$ and $T(\mathcal{P})$ the multisets containing the first vertex on every path in $\mathcal{P}$, and the last vertex on every path in $\mathcal{P}$, respectively. We use the following simple observation regarding the splicing procedure, whose proof is deferred to Section D.2 of Appendix.




(a) Before: Path $P$ is shown in red and path $P'$ is shown in purple.

(b) After: Path $\tilde{P}$ is shown in red and path $\tilde{P}'$ is shown in purple.

Figure 5: An illustration of path splicing at vertex $v$.

**Observation 4.6** *Let $I = (G, \Sigma)$ be an instance of* MCNwRS*, let $\mathcal{P}$ be a set of simple directed paths in $G$, and let $(P, P', v)$ be a triple in $\Pi^T(\mathcal{P})$. Let $\tilde{P}, \tilde{P}'$ be the pair of paths obtained by splicing $P$ and $P'$ at $v$, and let $\mathcal{P}' = \left(\mathcal{P} \setminus \{P, P'\}\right) \cup \left\{\tilde{P}, \tilde{P}'\right\}$. Then $S(\mathcal{P}') = S(\mathcal{P})$ and $T(\mathcal{P}') = T(\mathcal{P})$. Additionally, either (i) at least one of the paths $\tilde{P}, \tilde{P}'$ is a non-simple path; or (ii) $|\Pi^T(\mathcal{P}')| < |\Pi^T(\mathcal{P})|$.*

Using Observation 4.6, we can prove the following lemma that allows us to transform an arbitrary set $\mathcal{R}$ of paths into a set $\mathcal{R}'$ of non-transversal paths, while preserving the multisets containing the first endpoint and the last endpoint of every path, and without increasing the congestion on any edge. The proof of the lemma below is similar to the proof of Lemma 9.5 in [CMT20], and is provided in Appendix D.3 for completeness.

**Lemma 4.7** *There is an efficient algorithm, that, given an instance $(G, \Sigma)$ of* MCNwRS *and a set $\mathcal{R}$ of directed paths in $G$, computes another set $\mathcal{R}'$ of simple directed paths in $G$, such that $S(\mathcal{R}') = S(\mathcal{R})$, $T(\mathcal{R}') = T(\mathcal{R})$, and the paths in $\mathcal{R}'$ are non-transversal with respect to $\Sigma$. Moreover, for every edge $e \in E(G)$, $\mathrm{cong}_G(\mathcal{R}', e) \leq \mathrm{cong}_G(\mathcal{R}, e)$.*

## 4.2 Cuts, Well-Linkedness, and Related Notions

### 4.2.1 Minimum Cuts

A *cut* in a graph $G$ is a bipartition $(A, B)$ of its vertices into non-empty subsets. The *value* of the cut is $|E(A, B)|$. We sometimes consider cuts in edge-capacitated graphs. Given a graph $G$ with capacities $c(e) \geq 0$ on edges $e \in E(G)$ and a cut $(A, B)$ in $G$, the value of the cut is $\sum_{e \in E_G(A,B)} c(e)$. When edge capacities are not specified, we assume that they are unit.

Given two disjoint subsets $S, T$ of vertices of $G$, an $S$-$T$ cut, or a *cut separating $S$ from $T$* is a cut $(A, B)$ with $S \subseteq A$, $T \subseteq B$. A *minimum $S$-$T$ cut* is an $S$-$T$ cut $(A, B)$ of minimum value. When $S = \{s\}$ and $T = \{t\}$, we refer to $S$-$T$ cuts as $s$-$t$ cuts. We will use the following lemma, whose proof is provided in Section D.4 of Appendix.

**Lemma 4.8** *There is an efficient algorithm, that, given a graph $G$ and a collection $S = \{s_1, \ldots, s_k\}$ of its vertices, computes, for all $1 \leq i \leq k$, a set $A_i$ of vertices of $G$, and a collection $\mathcal{Q}_i$ of paths in $G$, such that the following hold:*

- *for all* $1 \leq i \leq k$, $S \cap A_i = \{s_i\}$, *and moreover,* $(A_i, V(G) \setminus A_i)$ *is a minimum cut separating* $s_i$ *from the vertices of* $S \setminus \{s_i\}$ *in* $G$*;*
- *for all* $1 \leq i < i' \leq k$, $A_i \cap A_{i'} = \emptyset$*; and*
- *for all* $1 \leq i \leq k$, $\mathcal{Q}_i = \{Q_i(e) \mid e \in \delta_G(A_i)\}$, *where for each* $e \in \delta_G(A_i)$, *path* $Q_i(e)$ *has* $e$ *as its first edge,* $s_i$ *as its last vertex, and all internal vertices of* $Q_i(e)$ *lie in* $A_i$*. Moreover, the paths in set* $\mathcal{Q}_i$ *are edge-disjoint.*

### 4.2.2 Gomory-Hu Trees

Gomory-Hu tree is a convenient structure that represents all minimum $s$-$t$ cuts in a given graph $G$. We summarize its properties in the following theorem.

**Theorem 4.9 ([GH61])** *There is an efficient algorithm, that, given a graph* $G = (V, E)$ *with capacities* $c(e) \geq 0$ *on its edges* $e \in E$, *computes a tree* $\tau = (V, E')$ *with capacities* $c'(e) \geq 0$ *on its edges* $e \in E'$, *such that the following hold:*

- *for every pair* $s, t$ *of distinct vertices of* $V$, *the value of the minimum* $s$-$t$ *cut in* $G$ *is equal to* $\min_{e \in E(P_{s,t})} \{c'(e)\}$, *where* $P_{s,t}$ *is the unique path connecting* $s$ *to* $t$ *in* $\tau$*; and*
- *for every pair* $s, t$ *of distinct vertices of* $V$, *if* $(A, B)$ *is a minimum* $s$-$t$ *cut in graph* $G$, *then* $(A, B)$ *is a minimum* $s$-$t$ *cut in graph* $\tau$, *and vice versa.*

We obtain the following immediate corollary of Theorem 4.9.

**Corollary 4.10** *Let* $G$ *be an edge-capacitated graph, and let* $\tau$ *be a Gomory-Hu tree of graph* $G$*. Then for every edge* $e = (u, u') \in E(\tau)$, *if we denote by* $U, U'$ *the vertex sets of the two connected components of* $\tau \setminus \{e\}$, *with* $u \in U$, *then* $(U, U')$ *is a minimum* $u$-$u'$ *cut in graph* $G$*.*

### 4.2.3 Balanced Cut and Sparsest Cut

Suppose we are given a graph $G = (V, E)$, and a subset $T \subseteq V$ of its vertices. We say that a cut $(X, Y)$ in $G$ is a valid $T$-cut iff $X \cap T, Y \cap T \neq \emptyset$. The *sparsity* of a valid $T$-cut $(X, Y)$ with respect to $T$ is $\frac{|E(X,Y)|}{\min\{|X \cap T|, |Y \cap T|\}}$. In the Sparsest Cut problem, given a graph $G$ and a subset $T$ of its vertices, the goal is to compute a valid $T$-cut of minimum sparsity. Arora, Rao and Vazirani [ARV09] designed an $O(\sqrt{\log n})$-approximation algorithm for the sparsest cut problem[4], where $n = |V(G)|$. We denote this algorithm by $\mathcal{A}_{\text{ARV}}$, and its approximation factor by $\beta_{\text{ARV}}(n) = O(\sqrt{\log n})$.

We say that a cut $(A, B)$ in a graph $G$ is *$\eta$-edge-balanced*, or just *$\eta$-balanced*, for a parameter $0 < \eta < 1$, if $|E(A)|, |E(B)| \leq \eta \cdot |E(G)|$. We say that a cut $(A, B)$ is a *minimum $\eta$-balanced cut in $G$* if $(A, B)$ is an $\eta$-balanced cut of minimum value $|E(A, B)|$. We will use the following theorem that follows from the work of [ARV09]. The proof is provided in Section D.5 of Appendix.

**Theorem 4.11** *For every constant* $1/2 < \hat{\eta} < 1$, *there is another constant* $\hat{\eta} < \hat{\eta}' < 1$ *and an efficient algorithm, that, given a connected (not necessarily simple) graph* $G$ *with* $m$ *edges, computes a* $\hat{\eta}'$*-balanced cut* $(A, B)$ *in* $G$, *whose value is at most* $O(\beta_{\text{ARV}}(m))$ *times the value of the minimum* $\hat{\eta}$*-balanced cut in* $G$*.*

[4]The algorithm was originally designed for simple graphs, but it can be easily generalized to graphs with parallel edges by exploiting edge capacities.

The following lemma is a simple consequence of the Planar Separator Theorem of Lipton and Tarjan [LT79]. A version of the lemma for vertex-balanced cuts was proved in [PSS96]. For completeness, we provide the proof of the lemma in Section D.6 of Appendix.

**Lemma 4.12** *Let $G$ be a connected (not necessarily simple) graph with $m$ edges and maximum vertex degree $\Delta \leq m/2^{40}$. If $\mathsf{OPT}_{\mathsf{cr}}(G) \leq m^2/2^{40}$, then the value of a minimum $(3/4)$-edge-balanced cut in $G$ is at most $O(\sqrt{\mathsf{OPT}_{\mathsf{cr}}(G) + \Delta \cdot m})$.*

### 4.2.4 Well-Linkedness, Bandwidth Property, and Routing Well-Linked Vertex Sets

The notion of well-linkedness plays a central role in graph theory and graph algorithms (see e.g. [Rac02, CKS04, And10, CL12, Chu12, CC16, Chu16, CT19]). We use the following standard definitions, which are equivalent to those used in much of previous work.

**Definition 4.13 (Well-Linkedness)** *We say that a set $T$ of vertices in a graph $G$ is* $\alpha$-well-linked*, for a parameter $0 < \alpha < 1$, if the sparsity of every valid $T$-cut in graph $G$ is at least $\alpha$. Equivalently, for every partition $(A, B)$ of $V(G)$ with $A \cap T, B \cap T \neq \emptyset$, $|E_G(A, B)| \geq \alpha \cdot \min\{|A \cap T|, |B \cap T|\}$ must hold.*

The next simple observation, that has been used extensively in previous work, shows that the set of vertices lying on the first row of the $(r \times r)$-grid is 1-well-linked. For completeness, we provide its proof in Section D.7 of Appendix.

**Observation 4.14** *Let $r \geq 1$ be an integer, and let $H$ be the $(r \times r)$-grid graph. Let $S$ be the set of vertices lying on the first row of the grid. Then vertex set $S$ is 1-well-linked in $H$.*

Next, we define the notion of bandwidth property, that was also used extensively in graph algorithms.

**Definition 4.15 ($\alpha$-Bandwidth Property)** *We say that a cluster $C$ of a graph $G$ has the* $\alpha$-bandwidth property *in $G$, for some parameter $0 < \alpha < 1$, if, for every partition $(A, B)$ of vertices of $C$, $|E_G(A, B)| \geq \alpha \cdot \min\{|\delta_G(A) \cap \delta_G(C)|, |\delta_G(B) \cap \delta_G(C)|\}$.*

The following immediate observation provides an equivalent definition of the bandwidth property that is helpful to keep in mind. Recall that, for a cluster $C$ of a graph $G$, its augmentation $C^+$ is a graph that is obtained from graph $G$ as follows. We subdivide every edge $e \in \delta_G(C)$ with a vertex $t_e$, and let $T(C) = \{t_e \mid e \in \delta_G(C)\}$ be the resulting set of newly added vertices. We then let $C^+$ be the subgraph of the resulting graph induced by vertex set $V(C) \cup T(C)$.

**Observation 4.16** *Let $G$ be a graph, let $C \subseteq G$ a cluster of $G$, and let $0 < \alpha < 1$ be a parameter. Cluster $C$ has the $\alpha$-bandwidth property iff the set $T(C) = \{t_e \mid e \in \delta_G(C)\}$ of vertices is $\alpha$-well-linked in graph $C^+$, which is the augmentation of cluster $C$ in $G$.*

One useful property of well-linked sets of vertices is that routing is easy between vertices of such sets. We summarize this property, that has been used extensively in past work, in the following theorem, and we provide its proof in Appendix D.8 for completeness. The theorem uses the notion of one-to-one routing that was defined in Section 4.1.3.

**Theorem 4.17** *There is an efficient algorithm, that, given a graph $G$, a set $T$ of vertices of $G$ that is $\alpha$-well-linked, and a pair $T_1, T_2$ of disjoint equal-cardinality subsets of $T$, computes a one-to-one routing $\mathcal{Q}$ of vertices of $T_1$ to vertices of $T_2$, with $\mathrm{cong}_G(\mathcal{Q}) \leq \lceil 1/\alpha \rceil$.*

The next corollary follows immediately from Observation 4.16 and Theorem 4.17.

**Corollary 4.18** *There is an efficient algorithm, that, given a graph $G$, a cluster $S$ of $G$ that has the $\alpha$-bandwidth property for some $0 < \alpha < 1$, and a pair $E_1, E_2$ of disjoint equal-cardinality subsets of the edge set $\delta_G(S)$, computes a one-to-one routing $\mathcal{Q}$ of edges of $E_1$ to edges of $E_2$, with $\mathrm{cong}_G(\mathcal{Q}) \leq \lceil 1/\alpha \rceil$, such that, for every path $Q \in \mathcal{Q}$, all inner vertices of $Q$ lie in $S$.*

### 4.2.5 Basic Well-Linked Decomposition

Typically, in a well-linked decomposition, we are given a graph $G$ together with a cluster $S$ of $G$, and our goal is to compute a partition of $S$ into clusters, each of which has the $\alpha$-bandwidth property in graph $G$, for some given parameter $0 < \alpha < 1$. Algorithms for computing well-linked decompositions were used extensively in prior work on graph-based problems (see e.g. [Rac02, CKS04, And10, CL12, Chu12, CC16, Chu16, CT19]). We use a variation of this technique, that, in addition to ensuring that each cluster $R$ in the decomposition has the $\alpha$-bandwidth property, provides a collection $\mathcal{P}(R)$ of paths routing the edges of $\delta_G(R)$ to edges of $\delta_G(S)$, such that the paths in $\mathcal{P}(R)$ are internally disjoint from $R$ and cause low congestion. The proof uses standard techniques and is deferred to Section D.9 of Appendix.

**Theorem 4.19** *There is an efficient algorithm, whose input is a graph $G$, a connected cluster $S$ of $G$, and parameters $m$ and $\alpha$, for which $|E(G)| \leq m$ and $0 < \alpha < \min\left\{\frac{1}{64\beta_{\mathrm{ARV}}(m)\cdot\log m}, \frac{1}{48\log^2 m}\right\}$ hold. The algorithm computes a collection $\mathcal{R}$ of vertex-disjoint clusters of $S$, such that:*

- $\bigcup_{R\in\mathcal{R}} V(R) = V(S)$;
- *for every cluster $R \in \mathcal{R}$, $|\delta_G(R)| \leq |\delta_G(S)|$;*
- *every cluster $R \in \mathcal{R}$ has the $\alpha$-bandwidth property in graph $G$; and*
- $\sum_{R\in\mathcal{R}} |\delta_G(R)| \leq |\delta_G(S)| \cdot \left(1 + O(\alpha \cdot \log^{1.5} m)\right)$.

*Additionally, the algorithm computes, for every cluster $R \in \mathcal{R}$, a set $\mathcal{P}(R) = \{P(e) \mid e \in \delta_G(R)\}$ of paths in graph $G$ with $\mathrm{cong}_G(\mathcal{P}(R)) \leq 100$, such that, for every edge $e \in \delta_G(R)$, path $P(e)$ has $e$ as its first edge and some edge of $\delta_G(S)$ as its last edge, and all inner vertices of $P(e)$ lie in $V(S) \setminus V(R)$.*

We note that, while the above theorem requires that cluster $S$ is connected, it can also be used when this is not the case, by simply applying the algorithm to every connected component of $S$ and then taking the union of all resulting sets of clusters; all properties that the theorem guarantees will continue to hold.

### 4.2.6 Layered Well-Linked Decomposition

To the best of our knowledge, layered well-linked decomposition was first introduced by Andrews [And10]. It is similar to the basic well-linked decomposition, except that it has some additional useful properties. We start by defining a layered well-linked decomposition formally. Our definition is very similar to that of [And10], except that we require some additional properties.

Let $H$ be a graph with $|E(H)| = m$ and $C \subseteq H$ a cluster of $H$. Let $\mathcal{W}$ be a collection of disjoint clusters of $H \setminus C$ with $\bigcup_{W\in\mathcal{W}} V(W) = V(H \setminus C)$, and let $(\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_r)$ be a partition of $\mathcal{W}$ into subsets that we call *layers*. We denote $\mathcal{L}_0 = \{C\}$, and, for all $1 \leq i \leq r$, for every cluster $W \in \mathcal{L}_i$, we partition the set $\delta_H(W)$ of edges into two subsets: set $\delta^{\mathrm{down}}(W)$ containing all edges $(u, v)$ with $u \in V(W)$ and $v$ lying in a cluster of $\mathcal{L}_0 \cup \cdots \cup \mathcal{L}_{i-1}$, and set $\delta^{\mathrm{up}}(W)$ containing all remaining edges of $\delta(W)$, namely: all edges $(u, v)$ with $u \in V(W)$ and $v$ lying in a cluster of $\mathcal{L}_i \cup \cdots \cup \mathcal{L}_r$ (see Figure 6).

We say that the collection $\mathcal{W}$ of clusters, together with its partition $(\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_r)$ into layers is a *valid layered $\alpha$-well-linked decomposition of $H$ with respect to $C$*, for some parameter $0 < \alpha < 1$, iff the following conditions hold:

- L1. For every pair $W, W'$ of distinct clusters in $\mathcal{W}$, $V(W) \cap V(W') = \emptyset$, and $\bigcup_{W \in \mathcal{W}} V(W) = V(H) \setminus V(C)$;
- L2. each cluster $W \in \mathcal{W}$ has the $\alpha$-bandwidth property in $H$;
- L3. for every cluster $W \in \mathcal{W}$, $|\delta_H(W)| \leq |\delta_H(C)|$, and $|E_H(W)| \geq |\delta_H(W)|/(64 \log m)$;
- L4. for every cluster $W \in \mathcal{W}$, $|\delta^{\mathrm{up}}(W)| < |\delta^{\mathrm{down}}(W)|/\log m$;
- L5. $\sum_{W \in \mathcal{W}} |\delta_H(W)| \leq 4|\delta_H(C)|$; and
- L6. for every cluster $W \in \mathcal{W}$, there is a collection $\mathcal{P}(W) = \{P(e) \mid e \in \delta_H(W)\}$ of paths in $H$, that cause congestion at most $200/\alpha$, and for all $e \in \delta_H(W)$, path $P(e)$ contains $e$ as its first edge, some edge $e' \in \delta_H(C)$ as its last edge, and all inner vertices of $P(e)$ are disjoint from $W$.

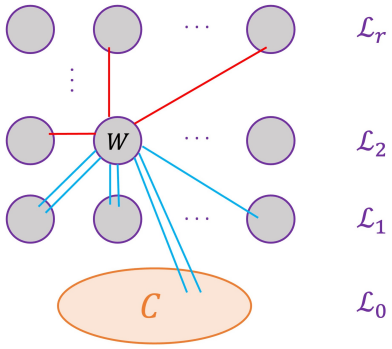



Figure 6: An illustration of a layered well-linked decomposition of $H$ with respect to $C$. For cluster $W \in \mathcal{L}_2$, the edges of $\delta^{\mathrm{up}}(W)$ are shown in red, and the edges of $\delta^{\mathrm{down}}(W)$ are shown in blue.

Recall that, given a graph $H$ and two sets $E', E''$ of its edges, we say that a set $\mathcal{P}$ of paths in $H$ routes edges of $E'$ to edges of $E''$ if $\mathcal{P} = \{P(e) \mid e \in E'\}$, and, for each edge $e \in E'$, path $P(e)$ has $e$ as its first edge and some edge of $E''$ as its last edge. Given a cluster $W$ of $H$, we say that the set $\mathcal{P}$ of paths *avoids* $W$ if, for every path $P \in \mathcal{P}$, no inner vertex of $P$ lies in $W$. Therefore, Condition L6 equivalently requires that for every cluster $W \in \mathcal{W}$, there is a collection $\mathcal{P}(W)$ of paths in $H$ routing the edges of $\delta_H(W)$ to the edges of $\delta_H(C)$, such that the paths in $\mathcal{P}(W)$ avoid $W$. This property is the main difference between our definition of a layered well-linked decomposition and that of [And10], which did not require this property.

The following theorem allows us to compute a layered well-linked decomposition in any graph. Its proof is practically identical to the algorithm of [And10]. The main difference is that we need to prove that the resulting decomposition has property L6. The proof of the theorem is deferred to Section D.10 of Appendix.

**Theorem 4.20** *There is a large enough constant $c$, and an efficient algorithm, that given a connected graph $H$ with $|E(H)| = m \geq c$ and a cluster $C$ of $H$, computes a valid layered $\alpha$-well-linked decomposition $(\mathcal{W}, (\mathcal{L}_1, \ldots, \mathcal{L}_r))$ of $H$ with respect to $C$, for $\alpha = \frac{1}{c \log^{2.5} m}$. The number of layers in the decomposition is $r \leq \log m$.*

### 4.3 Expanders, Graph Embeddings, and Routing Well-Linked Sets

We will use the notion of expanders, that we define next.

**Definition 4.21 (Expanders)** *We say that a graph $W$ is an $\alpha$-expander, for some $0 < \alpha < 1$, if, for every partition $(A, B)$ of $V(W)$ into non-empty subsets, $|E_W(A, B)| \geq \alpha \cdot \min\{|A|, |B|\}$; equivalently, the set $V(W)$ of vertices is $\alpha$-well-linked in $W$.*

We will also use a standard notion of graph embeddings.

**Definition 4.22 (Embedding of Graphs)** *Let $H$, $G$ be a pair of graphs with $V(H) \subseteq V(G)$. An* embedding *of $H$ into $G$ is a collection $\mathcal{P} = \{P(e) \mid e \in E(H)\}$ of paths in graph $G$, where for each edge $e = (u, v) \in E(H)$, path $P(e)$ has endpoints $u$ and $v$. The* congestion of the embedding *is $\mathrm{cong}_G(\mathcal{P})$.*

The following well known claim shows a connection between well-linked sets of vertices and embeddings of expanders. The proof is standard and deferred to Section D.11 of Appendix.

**Claim 4.23** *There is a universal constant $c_{\mathrm{CMG}}$, and an efficient randomized algorithm that, given a graph $G$ together with a subset $T$ of its vertices of cardinality $k$, such that $T$ is $\alpha$-well-linked in $G$, for some $0 < \alpha < 1$, constructs another graph $W$ with $V(W) = T$ and maximum vertex degree at most $c_{\mathrm{CMG}} \log^2 k$, together with an embedding $\mathcal{P}$ of $W$ into $G$ with congestion at most $\frac{c_{\mathrm{CMG}} \log^2 k}{\alpha}$, such that with high probability graph $W$ is an $(1/4)$-expander.*

We show in the following observation that, if $W$ is the outcome expander of the algorithm from Claim 4.23, then it has a high crossing number. The proof is provided in Section D.12 of Appendix.

**Observation 4.24** *There is some constant $c$, such that, if $W$ is an $(1/4)$-expander, with $|V(W)| = k > c$ and maximum vertex degree $O(\log^2 k)$, then $\mathsf{OPT}_{\mathsf{cr}}(W) \geq k^2/(c \log^8 k)$.*

We obtain the following useful corollary of Claim 4.23, that allows us to route specific pairs of vertices of a well-linked vertex set $T$. We provide its proof in Appendix D.13.

**Corollary 4.25** *There is an efficient randomized algorithm that, given a graph $G$, a subset $T$ of its vertices of cardinality $k$, that is $\alpha$-well-linked in $G$, for some $0 < \alpha < 1$, together with a partial matching $M$ over the vertices of $T$, computes a set $\mathcal{R}(M) = \{R(u, v) \mid (u, v) \in M\}$ of paths in graph $G$, such that for every pair $(u, v) \in M$ of vertices, path $R(u, v)$ connects $u$ to $v$. Moreover, with high probability, the congestion caused by the paths in $\mathcal{R}(M)$ in $G$ is $O((\log^4 k)/\alpha)$.*

Let $K_z$ be a complete graph, whose vertex set has cardinality $z$. We obtain the following immediate corollary of Corollary 4.25, whose proof appears in Section D.14 of Appendix.

**Corollary 4.26** *There is an efficient randomized algorithm that, given a graph $G$ and a subset $T$ of its vertices of cardinality $z$, such that $T$ is $\alpha$-well-linked in $G$, for some $0 < \alpha < 1$, computes an embedding $\tilde{\mathcal{P}}$ of the complete graph $K_z$ with $V(K_z) = T$ into $G$, such that, with high probability, the congestion of the embedding is $O((z \log^4 z)/\alpha)$.*

#### 4.3.1 Constructing Internal Routers

We now provide an efficient algorithm, that, given a graph $G$ and a cluster $C$ of $G$ that has the $\alpha$-bandwidth property, constructs a distribution $\mathcal{D}(C)$ over the internal $C$-routers, such that the expected congestion on every edge of $C$ is small. We start with the following lemma, that provides a similar result for a graph $G$ and a set $T$ of vertices of $G$ that is well-linked.

**Lemma 4.27** *There is an efficient randomized algorithm, whose input is a graph $G$ and set $T$ of its vertices called terminals, such that $|T| = z$, and $T$ is $\alpha$-well-linked in $G$, for some $0 < \alpha < 1$. The algorithm computes, for every terminal $t \in T$, a set $\mathcal{Q}_t = \{Q_t(t') \mid t' \in T \setminus \{t\}\}$ of paths, where, for all $t' \in T \setminus \{t\}$, path $Q_t(t')$ connects $t'$ to $t$. Moreover, if we select a vertex $t \in T$ uniformly at random, then, for every edge $e \in E(G)$, $\mathbf{E}[\mathrm{cong}(\mathcal{Q}_t, e)] \leq O(\log^4 z/\alpha)$.*

**Proof:** We use the algorithm from Corollary 4.26, in order to compute an embedding $\tilde{\mathcal{P}}$ the complete graph $K_z$ with $V(K_z) = T$ into $G$. Recall that the algorithm ensures that, with high probability, the congestion of the embedding is at most $(cz \log^4 z)/\alpha$, for some constant $c$. If the congestion caused by the paths in $\tilde{\mathcal{P}}$ is greater than this bound, then we say that the algorithm from Corollary 4.26 failed. We repeat the algorithm from Corollary 4.26 $O(\log |E(G)|)$ times. Let $\mathcal{E}_1$ be the event that the algorithm failed in each of these applications. Then $\mathbf{Pr}[\mathcal{E}_1] \leq 1/\operatorname{poly}(z)$. In this case, for every terminal $t \in T$, we return a set $\mathcal{Q}_t = \{Q_t(t') \mid t' \in T \setminus \{t\}\}$ of paths, where for every terminal $t' \in T \setminus \{t\}$, $Q_t(t')$ is an arbitrary path connecting $t$ to $t'$ in $G$. Clearly, for all $t \in T$, for every edge $e \in E(G)$, $\mathrm{cong}_G(\mathcal{Q}_t, e) \leq z$.

We assume from now on that, in some application of the algorithm from Corollary 4.26, it returned a set $\tilde{\mathcal{P}}$ of paths with $\mathrm{cong}_G(\tilde{\mathcal{P}}) \leq O((z \log^4 z)/\alpha)$.

We now fix a terminal $t \in T$, and define the corresponding set $\mathcal{Q}_t = \{Q_t(t') \mid t' \in T \setminus \{t\}\}$ of paths. For every terminal $t' \in T \setminus \{t\}$, we let $Q_t(t')$ be the unique path in set $\tilde{\mathcal{P}}$ that serves as the embedding of the edge $(t, t') \in E(K_z)$. Clearly, path $Q_t(t')$ connects $t'$ to $t$ as required.

Consider now an edge $e$, and let $\eta_e = \mathrm{cong}_G(\tilde{\mathcal{P}}, e) \leq O((z \log^4 z)/\alpha)$. Since every path of $\tilde{\mathcal{P}}$ may lie in at most two path sets of $\{\mathcal{Q}_t\}_{t \in T}$, we get that $\sum_{t \in T} \mathrm{cong}_G(\mathcal{Q}_t, e) \leq 2\eta_e$. Therefore, if Event $\mathcal{E}_1$ did not happen, and a terminal $t \in T$ is selected uniformly at random, then $\mathbf{E}[\mathrm{cong}(\mathcal{Q}_t, e)] \leq 2\eta_e/z \leq O(\log^4 z/\alpha)$. Overall, for every edge $e \in E(G)$, $\mathbf{E}[\mathrm{cong}(\mathcal{Q}_t, e)] \leq \mathbf{E}[\mathrm{cong}(\mathcal{Q}_t, e) \mid \neg\mathcal{E}_1] + \mathbf{E}[\mathrm{cong}(\mathcal{Q}_t, e) \mid \mathcal{E}_1] \cdot \mathbf{Pr}[\mathcal{E}_1] \leq O(\log^4 z/\alpha) + O(1/z) \leq O(\log^4 z/\alpha)$. □

The following corollary allows us to compute a distribution over internal $C$-routers for a cluster $C$ of a graph $G$, such that the expected congestion on every edge of $C$ is small. The corollary follows immediately by applying the algorithm from Lemma 4.27 to the augmentation $C^+$ of the cluster $C$ in graph $G$. The proof of the corollary is omitted.

**Corollary 4.28** *There is an efficient randomized algorithm, whose input is a graph $G$ and a cluster $C$ of $G$ that has the $\alpha$-bandwidth property for some $0 < \alpha < 1$. The algorithm returns (explicitly) a distribution $\mathcal{D}$ over the set $\Lambda(C)$ of internal $C$-routers, such that, for every edge $e \in E(C)$, $\mathbf{E}_{\mathcal{Q} \sim \mathcal{D}}[\mathrm{cong}(\mathcal{Q}, e)] \leq O((\log |\delta_G(C)|)^4/\alpha)$.*

## 4.4 Curves in the Plane or on a Sphere

### 4.4.1 Reordering Curves

Assume that we are given two oriented orderings $(\mathcal{O}, b), (\mathcal{O}', b')$ on a set $U = \{u_1, \ldots, u_r\}$ of elements. Assume for simplicity that $b = b' = 1$ (otherwise the corresponding ordering can be flipped). Consider a disc $D$, with a collection $\{p_1, \ldots, p_r\}$ of distinct points appearing on the boundary of $D$ (we will view each point $p_i$ as representing element $u_i$ of $U$), such that the order in which these points are encountered, as we traverse the boundary of $D$ in the counter-clock-wise direction, is precisely $\mathcal{O}$. Let $D' \subseteq D$ be another disc that is contained in $D$, whose boundary is disjont from the boundary of $D$. Assume that a collection $\{p'_1, \ldots, p'_r\}$ of points appear on the boundary of $D'$, and that the order in which these points are encountered as we traverse the boundary of $D'$ in the counter-clock-wise direction is precisely $\mathcal{O}'$. As before, for each $1 \leq i \leq r$, we view point $p'_i$ as representing element $u_i \in U$. We now define reordering curves between the oriented orderings $(\mathcal{O}, b)$ and $(\mathcal{O}', b')$, which are then used in order to define the distance between the two orderings.

**Definition 4.29 (Reordering curves)** *We say that a collection $\Gamma = \{\gamma_1, \ldots, \gamma_r\}$ of curves is a* set of reordering curves for oriented orderings $(\mathcal{O}, b)$ and $(\mathcal{O}', b')$ *iff (i) the curves in $\Gamma$ are in general position; (ii) each curve $\gamma_i \in \Gamma$ is simple and its interior is contained in $D \setminus D'$; and (iii) for all $1 \leq i \leq r$, curve $\gamma_i$ has $p_i, p_i'$ as its endpoints. The* cost *of the collection $\Gamma$ is the total number of crossings between its curves.*

**Definition 4.30 (Distance between orderings)** *Let $(\mathcal{O}, b)$ and $(\mathcal{O}', b')$ be two oriented orderings on a set $U$ of elements. The* distance *between the two oriented orderings, denoted by $\mathsf{dist}((\mathcal{O}, b), (\mathcal{O}', b'))$, is the smallest cost of any collection $\Gamma$ of reordering curves for $(\mathcal{O}, b)$ and $(\mathcal{O}, b')$. For two unoriented orderings $\mathcal{O}, \mathcal{O}'$ on $U$, we define $\mathsf{dist}(\mathcal{O}, \mathcal{O}') = \min_{b,b' \in \{-1,1\}} \{\mathsf{dist}((\mathcal{O}, b), (\mathcal{O}', b'))\}$.*

The following lemma, that follows from Section 4 of [PSŠ09] and Section 5.2 of [PSŠ11], provides an efficient algorithm to compute a collection of reordering curves of near-optimal cost for a given pair of oriented orderings. The proof is deferred to Section D.15 of Appendix.

**Lemma 4.31** *There is an efficient algorithm, that, given a pair $(\mathcal{O}, b)$, $(\mathcal{O}', b')$ of oriented orderings on a set $U$ of elements, computes a collection $\Gamma$ of reordering curves for $(\mathcal{O}, b)$ and $(\mathcal{O}', b')$, of cost at most $2 \cdot \mathsf{dist}((\mathcal{O}, b), (\mathcal{O}', b'))$.*

We will use the following simple corollary of the lemma, whose proof is provided Appendix D.16.

**Corollary 4.32** *There is an efficient algorithm, whose input is a graph $G$, a drawing $\varphi$ of $G$ in the plane, a vertex $v \in V(G)$, and a circular ordering $\mathcal{O}_v$ of the edges of $\delta_G(v)$. Let $\mathcal{O}_v'$ be the circular order in which the edges of $\delta_G(v)$ enter the image of $v$ in $\varphi$, and let $D = D_\varphi(v)$ be a tiny $v$-disc. The algorithm produces a new drawing $\varphi'$ of $G$, with $\mathsf{cr}(\varphi') \leq \mathsf{cr}(\varphi) + 2 \cdot \mathsf{dist}(\mathcal{O}_v, \mathcal{O}_v')$, such that the following hold:*

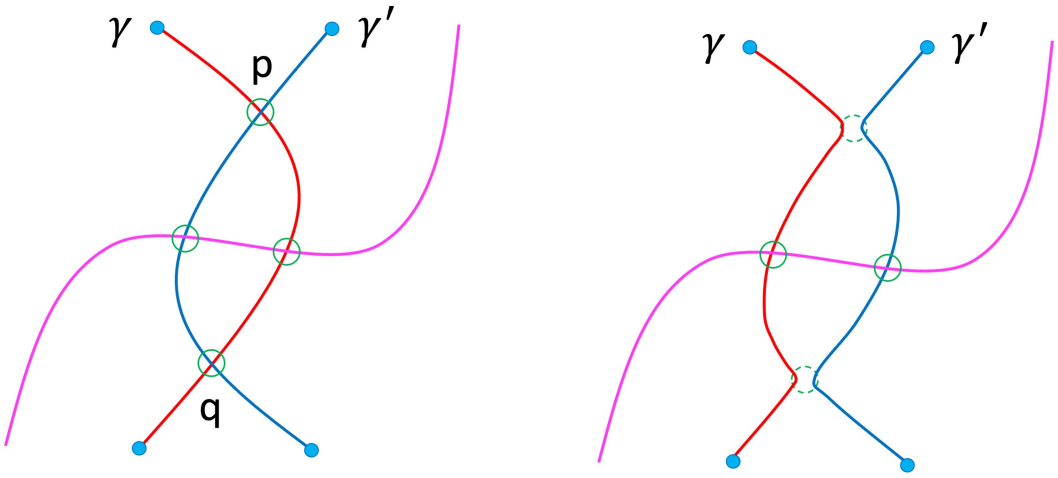
- *the images of the edges of $\delta_G(v)$ enter the image of $v$ in the order $\mathcal{O}_v$ in $\varphi'$; and*
- *the drawings $\varphi$ and $\varphi'$ are identical, except that, for each edge $e \in \delta_G(v)$, the segment of the image of $e$ lying inside the disc $D$ may be different in the two drawings.*

### 4.4.2 Type-1 Uncrossing of Curves

In this subsection we consider a set $\Gamma$ of curves in the plane (or on a sphere) that are in general position, and provide a simple operation, called *type*-1 *uncrossing*, whose goal is to "simplify" this collection of curves by eliminating some of the crossings between them. Specifically, we modify the curves in $\Gamma$, without changing their endpoints, to ensure that every pair of curves cross at most once. We now describe the type-1 uncrossing operation formally.

Let $\Gamma$ be a set of simple curves in the plane that are in general position. For a pair $\Gamma_1, \Gamma_2$ of disjoint subsets of $\Gamma$, we denote by $\chi(\Gamma_1, \Gamma_2)$ the total number of crossings between the curves in $\Gamma_1$ and the curves in $\Gamma_2$. In other words, $\chi(\Gamma_1, \Gamma_2)$ is the number of points $p$, such that $p$ lies on a curve in $\Gamma_1$ and on a curve in $\Gamma_2$, and $p$ is not an endpoint of these curves. If $\Gamma_1 = \{\gamma\}$, then we use the shorthand $\chi(\gamma, \Gamma_2)$ instead of $\chi(\{\gamma\}, \Gamma_2)$.

The type-1 uncrossing operation iteratively considers pairs $\gamma, \gamma' \in \Gamma$ of distinct curves that cross more than once, and then locally modifies them, as shown in Figure 7, to eliminate two crossings. This operation ensures that no new crossings are created, and preserves the endpoints of both curves. The following theorem summarizes this operation. The proof of the theorem is standard and is deferred to Section D.17 of Appendix for completeness.




(a) Before: Curves $\gamma$ and $\gamma'$ cross twice, at points $p$ and $q$. The crossing points of both curves with the third curve are circled as well.

(b) After: Each of the new curves $\gamma$ and $\gamma'$ has same endpoints as before. The two curves no longer cross each other, and the pink curve still participates in two crossings with $\gamma$ and $\gamma'$.

Figure 7: Type-1 uncrossing operation.

**Theorem 4.33 (Type-1 Uncrossing)** *There is an algorithm, that, given a set $\Gamma$ of simple curves in general position, that are partitioned into two disjoint subsets $\Gamma_1, \Gamma_2$, computes, for each curve $\gamma \in \Gamma_1$, a simple curve $\gamma'$ that has same endpoints as $\gamma$, such that, if we denote by $\Gamma_1' = \{\gamma' \mid \gamma \in \Gamma_1\}$, then the following hold:*

- *the curves in set $\Gamma_1' \cup \Gamma_2$ are in general position;*
- *every pair of distinct curves in $\Gamma_1'$ cross at most once;*
- *for every curve $\gamma \in \Gamma_2$, $\chi(\gamma, \Gamma_1') \leq \chi(\gamma, \Gamma_1)$; and*
- *the total number of crossings between the curves of $\Gamma_1' \cup \Gamma_2$ is bounded by the total number of crossings between the curves of $\Gamma$.*

*The running time of the algorithm is bounded by $\mathrm{poly}(n \cdot N)$, where $n$ is the number of bits in the representation of the set $\Gamma$ of curves, and $N$ is the number of crossing points between the curves of $\Gamma$.*

### 4.4.3 Curves in a Disc and Nudging of Curves

Suppose we are given a disc $D$, and a collection $\{s_1, t_1, \ldots, s_k, t_k\}$ of distinct points on its boundary. For all $1 \leq i < j \leq k$, we say that the two pairs $(s_i, t_i), (s_j, t_j)$ of points *cross* iff the unoriented circular ordering of the points $s_i, s_j, t_i, t_j$ on the boundary of $D$ is $(s_i, s_j, t_i, t_j)$. We use the following simple claim, whose proof is deferred to Appendix D.18.

**Claim 4.34** *There is an efficient algorithm that, given a disc $D$, and a collection $\{s_1, t_1, \ldots, s_k, t_k\}$ of distinct points on the boundary of $D$, computes a collection $\Gamma = \{\gamma_1, \ldots, \gamma_k\}$ of curves, such that, for all $1 \leq i \leq k$, curve $\gamma_i$ has $s_i$ and $t_i$ as its endpoints, and its interior is contained in the interior of $D$. Moreover, for every pair $1 \leq i < j \leq k$ of indices, if the two pairs $(s_i, t_i), (s_j, t_j)$ of points cross then curves $\gamma_i, \gamma_j$ intersect at exactly one point; otherwise, curves $\gamma_i, \gamma_j$ do not intersect. Lastly, every point in the interior of $D$ may be contained in at most two curves of $\Gamma$.*

**Nudging Procedure.** In a nudging procedure, we are given an instance $I = (G, \Sigma)$ of MCNwRS, a subset $U$ of vertices of $G$, and a collection $\mathcal{P}$ of edge-disjoint paths, such that, for every path $P \in \mathcal{P}$, all inner vertices of $P$ lie in $U$, and the endpoints of $P$ do not lie in $U$. Additionally, we are given some solution $\varphi$ to instance $I$. For every path $P \in \mathcal{P}$, we denote by $\gamma(P)$ the image of path $P$ in $\varphi$, that is, $\gamma(P)$ is the concatenation of the images of all edges of $P$. Notice that the resulting collection $\Gamma = \{\gamma(P) \mid P \in \mathcal{P}\}$ may not be in general position. This is since some vertices $u \in U$ may lie on more than 2 paths in $\mathcal{P}$, and in such a case more than 2 curves in $\Gamma$ contain the point $\varphi(u)$. The purpose of the nudging procedure is to slightly modify the curves in $\Gamma$ in the viccinity of the images of such vertices to ensure that the resulting collection of curves $\Gamma' = \{\gamma'(P) \mid P \in \mathcal{P}\}$ is in general position, while introducing relatively few crossings. Additionally, the procedure ensures that, for every path $P \in \mathcal{P}$, the endpoints of the new curve $\gamma'(P)$ are identical to those of the original curve $\gamma(P)$.

We start by letting, for every path $P$, curve $\gamma'(P)$ be the original curve $\gamma(P)$, and we set $\Gamma' = \{\gamma'(P) \mid P \in \mathcal{P}\}$. We then process every vertex $u \in U$ one by one. Consider an iteration when any such vertex $u$ is processed. Let $\mathcal{P}(u) \subseteq \mathcal{P}$ be a set of all paths $P \in \mathcal{P}$ with $u \in V(P)$. We denote $\mathcal{P}(u) = \{P_1, \ldots, P_k\}$. Consider the tiny $u$-disc $D(u) = D_\varphi(u)$ in the drawing $\varphi$ of graph $G$. For all $1 \leq i \leq k$, we let $s_i, t_i$ be the two points at which curve $\gamma'(P_i)$ intersects the boundary of the disc $D(u)$. Note that all points $s_1, t_1, \ldots, s_k, t_k$ must be distinct, as the paths in $\mathcal{P}$ are edge-disjoint. We use the algorithm from Claim 4.34 in order to construct a collection $\{\sigma_1, \ldots, \sigma_k\}$ of curves, such that, for all $1 \leq i \leq k$, curve $\sigma_i$ has $s_i$ and $t_i$ as its endpoints, and is completely contained in $D(u)$. Recall that the claim ensures that, for every pair $1 \leq i < j \leq k$ of indices, if the two pairs $(s_i, t_i), (s_j, t_j)$ of points cross, then curves $\sigma_i, \sigma_j$ intersect at exactly one point; otherwise, curves $\sigma_i, \sigma_j$ do not intersect. The former may only happen if paths $P_i, P_j$ have a transversal intersection at vertex $u$. For all $1 \leq i \leq k$, we modify the curve $\gamma'(P_i)$ as follows: we replace the segment of the curve between points $s_i, t_i$ with the curve $\sigma_i$. Once every vertex of $U$ is processed, we obtain the final collection $\Gamma' = \{\gamma'(P) \mid P \in \mathcal{P}\}$ of curves. From the above discussion, we get the following observation.

**Observation 4.35** *The set $\Gamma' = \{\gamma'(P) \mid P \in \mathcal{P}\}$ of curves is in general position, and, for every path $P \in \mathcal{P}$, the endpoints of curve $\gamma'(P)$ are identical to the endpoints of curve $\gamma(P)$. Moreover, if $\chi$ denotes the set of all crossings $(e, e')_p$ in $\varphi$, where $e$ and $e'$ are edges of $\bigcup_{P \in \mathcal{P}} E(P)$, then the number of crossings between the curves of $\Gamma'$ is bounded by $|\chi| + |\Pi^T(\mathcal{P})|$. Lastly, if the paths in $\mathcal{P}$ are non-transversal with respect to $\Sigma$, then for every path $P \in \mathcal{P}$, the number of crossings between $\gamma'(P)$ and $\Gamma' \setminus \{\gamma'(P)\}$ is bounded by the number of crossings $(e, e')_p$ in $\varphi$ where exactly one of the edges $e, e'$ belongs to $P$.*

### 4.4.4 Type-2 Uncrossing of Curves

In this subsection we provide another subroutine, called *type-2 uncrossing of curves*, that allows us to simplify a given set $\Gamma$ of curves by removing some of the crossings between them. Unlike the type-1 uncrossing operation, we no longer preserve the endpoints of every curve, but we ensure that the multisets containing the endpoints of the curves are preserved under this operation.

It will sometimes be useful for us to assign a *direction* to a curve $\gamma$, by designating one of its endpoints, that we denote by $s(\gamma)$, as its first endpoint, and the other endpoint, denoted by $t(\gamma)$, as its last endpoint. If $\Gamma$ is a collection of curves, and each curve in $\Gamma$ is assigned a direction, then we say that $\Gamma$ is a collection of *directed curves*. In such a case, we let $S(\Gamma)$ be the multiset of points containing the first endpoint of every curve in $\Gamma$, and we let $T(\Gamma)$ be the multiset of points containing the last endpoint of every curve in $\Gamma$.

For the type-2 uncrossing operation, we will consider curves that arise from some drawing $\varphi$ of a graph $G$. We first need to define curves that are aligned with a graph drawing. For intuition, consider first some planar graph $G$, and its planar drawing $\varphi$. In this case, curve $\gamma$ is aligned with the drawing $\varphi$

of $G$, if there is some path $P$ in $G$, such that $\gamma$ can be obtained by first concatenating the images of all edges if $P$, and then possibly modifying the resulting curve within tiny discs $D_\varphi(v)$ for vertices $v \in V(P)$ (typically via a nudging operation). If $\varphi$ is a non-planar drawing of some graph $G$, then the definition of a curve $\gamma$ being aligned with the drawing is similar, but now we allow the curve $\gamma$ to "switch" from the image of one edge to another, at a crossing point between the two edges. Therefore, we can define a sequence $e_1, e_2, \ldots, e_{r-1}$ of edges of $G$, such that the curve "follows" segments of these edges. The curve $\gamma$ itself can then be partitioned into segments $\sigma_1, \sigma'_1, \sigma_2, \sigma'_2, \ldots, \sigma'_{r-1}, \sigma_r$, where for all $1 \leq i \leq r-1$, $\sigma'_i$ is a contiguous segment of the image of edge $e_i$. For a pair $\sigma'_i, \sigma'_{i+1}$ of such segments, either the last endpoint of $\sigma'_i$ and the first endpoint of $\sigma'_{i+1}$ are identical (and it is a crossing point between the images of $e_i$ and $e_{i+1}$); or segment $\sigma_{i+1}$ is contained in disc $D_\varphi(v_{i+1})$, where $v_{i+1}$ is a common endpoint of $e_i$ and $e_{i+1}$. With this intuition in mind, we now define the notion of alignment of a curve with a drawing of a graph.

**Definition 4.36 (Curve aligned with a drawing of a graph)** *Let $G$ be a graph and $\varphi$ a drawing of $G$ in the plane. We say that a curve $\gamma$ is* aligned *with the drawing $\varphi$ of $G$ if there is a sequence $(e_1, e_2, \ldots, e_{r-1})$ of edges of $G$, and a partition $(\sigma_1, \sigma'_1, \sigma_2, \sigma'_2, \ldots, \sigma'_{r-1}, \sigma_r)$ of $\gamma$ into consecutive segments, such that, if we denote, for all $1 \leq i < r$, $e_i = (v_i, v_{i+1})$, then the following hold:*

- *for all $1 \leq i \leq r-1$, $\sigma'_i$ is a contiguous segment of non-zero length of $\varphi(e_i)$, and it is disjoint from all discs in $\{D_\varphi(u)\}_{u \in V(G)}$, except that its first endpoint may lie on the boundary of $D_\varphi(v_i)$, and its last endpoint may lie on the boundary of $D_\varphi(v_{i+1})$;*
- *for all $1 \leq i \leq r$, segment $\sigma_i$ is either contained in disc $D_\varphi(v_i)$, or it is contained in a tiny p-disc $D(p)$, where $p$ is a crossing point of $\varphi(e_{i-1})$ and $\varphi(e_i)$;*
- *$\sigma_1 = \varphi(e_1) \cap D_\varphi(v_1)$; and*
- *$\sigma_r = \varphi(e_{r-1}) \cap D_\varphi(v_r)$.*

In order to perform a type-2 uncrossing operation, we consider a graph $G$, a drawing $\varphi$ of $G$, and a set $\mathcal{Q}$ of simple directed paths in $G$. We assume that no vertex of $G$ may serve simultaneously as an endpoint of a path of $\mathcal{Q}$ and an inner vertex of some other path of $\mathcal{Q}$. We can then define a set $\Gamma = \{\gamma(Q) \mid Q \in \mathcal{Q}\}$ of curves, where, for every path $Q \in \mathcal{Q}$, curve $\gamma(Q)$ is obtained by concatenating the images of the edges of $Q$. Note however that the curves in the resulting set $\Gamma$ are not necessarily in general position. Type-2 uncrossing allows us to fix this, and moreover to eliminate all crossings between the resulting set $\Gamma'$ of curves. Unlike type-1 uncrossing, we only guarantee that the multisets containing the first and last endpoints of the curves in $\Gamma'$ remain identical to those corresponding to $\Gamma$, but we no longer guarantee that they are matched in the same way to each other. For technical reasons, we need to consider two different settings for the type-2 uncrossing: one where the paths in set $\mathcal{Q}$ are edge-disjoint, in which case we can provide somewhat stronger guarantees, and another where this is not the case. These two settings for type-2 uncrossing are provided in the following theorem and its corollary, whose proofs are simple and are deferred to Sections D.19 and D.20 of Appendix, respectively. We start with the setting where the paths in set $\mathcal{Q}$ are edge-disjoint.

**Theorem 4.37** *There is an efficient algorithm, whose input consists of a graph $G$, a drawing $\varphi$ of $G$ on the sphere, and a collection $\mathcal{Q}$ of edge-disjoint paths in $G$, such that no vertex of $G$ may simultaneously serve as an endpoint of some path in $\mathcal{Q}$ and an inner vertex of some path in $\mathcal{Q}$. Additionally, for each path $Q \in \mathcal{Q}$, one of its endpoints is designated as its first endpoint and is denoted by $s(Q)$, and the other endpoint is designated as its last endpoint and denoted by $t(Q)$. The algorithm computes a set $\Gamma = \{\gamma(Q) \mid Q \in \mathcal{Q}\}$ of directed simple curves on the sphere with the following properties:*

- *every curve $\gamma(Q) \in \Gamma$ is aligned with the drawing of the graph $\bigcup_{Q' \in \mathcal{Q}} Q'$ induced by $\varphi$;*

- *for each path $Q \in \mathcal{Q}$, $s(\gamma(Q)) = \varphi(s(Q))$; moreover, if $e_1(Q)$ is the first edge of $Q$, then curve $\gamma(Q)$ contains the segment $\varphi(e_1(Q)) \cap D_\varphi(s(Q))$;*
- *the multiset $T(\Gamma)$, containing the last endpoint of every curve in $\Gamma$, is precisely the multiset $\{\varphi(t(Q)) \mid Q \in \mathcal{Q}\}$, containing the image of the last vertex on every path of $\mathcal{Q}$ in $\varphi$; and*
- *the curves in $\Gamma$ do not cross each other.*

We emphasize that the curves in $\Gamma$ may match the mutisets $\{\varphi(s(Q)) \mid Q \in \mathcal{Q}\}$ and $\{\varphi(t(Q)) \mid Q \in \mathcal{Q}\}$ differently from the paths in $\mathcal{Q}$.

We will sometimes use Theorem 4.37 in a setting where we are additionally given a subgraph $C \subseteq G$, and the paths of $\mathcal{Q}$ are internally disjoint from $C$. In such a case, from the definition of aligned curves, and from the fact that the curves of $\Gamma$ do not cross each other, for every edge $e \in E(C)$, the number of crossings between $\varphi(e)$ and the curves in $\Gamma$ is bounded by the number of crossings between $\varphi(e)$ and the curves of $\left\{\varphi(e') \mid e' \in \bigcup_{Q \in \mathcal{Q}} E(Q)\right\}$.

We use the following corollary of Theorem 4.37, that deals with the setting where paths in set $\mathcal{Q}$ may share edges. The proof is deferred to Section D.20 of Appendix.

**Corollary 4.38** *There is an efficient algorithm, whose input consists of a graph $G$, a drawing $\varphi$ of $G$ on the sphere, a subgraph $C$ of $G$, and a collection $\mathcal{Q}$ of paths in $G$, that are internally disjoint from $C$, such that no vertex of $G$ may simultaneously serve as an endpoint of some path in $\mathcal{Q}$ and an inner vertex of some path in $\mathcal{Q}$. Additionally, for each path $Q \in \mathcal{Q}$, one of its endpoints is designated as its first endpoint and is denoted by $s(Q)$, and the other endpoint is designated as its last endpoint and is denoted by $t(Q)$. The algorithm computes a set $\Gamma = \{\gamma(Q) \mid Q \in \mathcal{Q}\}$ of directed simple curves on the sphere with the following properties:*

- *for every path $Q \in \mathcal{Q}$, $s(\gamma(Q)) = \varphi(s(Q))$;*
- *the multiset $T(\Gamma)$, containing the last endpoint of every curve in $\Gamma$, is precisely the multiset $\{\varphi(t(Q)) \mid Q \in \mathcal{Q}\}$, containing the image of the last vertex on every path of $\mathcal{Q}$;*
- *the curves in $\Gamma$ do not cross each other; and*
- *for each edge $e \in E(C)$, the number of crossings between $\varphi(e)$ and the curves in $\Gamma$ is bounded by $\sum_{e' \in E(G) \setminus E(C)} \chi(e, e') \cdot \mathrm{cong}_G(\mathcal{Q}, e')$, where $\chi(e, e')$ is the number of crossings between $\varphi(e)$ and $\varphi(e')$.*

## 4.5 Contracted Graphs

Let $G$ be a graph and let $\mathcal{C}$ be a collection of disjoint clusters of $G$. We define the *contracted graph* $G_{|\mathcal{C}}$ to be the graph obtained from $G$ by contracting each cluster $C \in \mathcal{C}$ into a supernode $v_C$; we remove self-loops but keep parallel edges. Note that every edge of graph $G_{|\mathcal{C}}$ corresponds to some edge of graph $G$. We do not distinguish between such edges, so $E(G_{|\mathcal{C}}) \subseteq E(G)$. We refer to vertices of $G_{|\mathcal{C}}$ that are not supernodes as *regular* vertices. In the following claim, we derive well-linkedness properties of a set $T$ of vertices in a graph $G$ from well-linkedness of $T$ in a contracted graph $G_{|\mathcal{C}}$ and bandwidth properties of the clusters of $\mathcal{C}$ in $G$. The proof is deferred to Section D.21 of Appendix.

**Claim 4.39** *Let $G = (V, E)$ be a graph, $T \subseteq V$ a subset of its vertices, and $\mathcal{C}$ a collection of disjoint clusters of $G$, such that $T \cap (\bigcup_{C \in \mathcal{C}} V(C)) = \emptyset$. Assume that each cluster $C \in \mathcal{C}$ has the $\alpha_1$-bandwidth property in $G$, and that the set $T$ of vertices is $\alpha_2$-well-linked in the contracted graph $G_{|\mathcal{C}}$, for some parameters $0 < \alpha_1, \alpha_2 < 1$. Then $T$ is $(\alpha_1 \cdot \alpha_2)$-well-linked in $G$.*

The following corollary of Claim 4.39 essentially replaces the well-linkedness property of the set $T$ of vertices with the equivalent bandwidth property of a cluster of a given graph $G$. The proof is deferred to Section D.22 of Appendix.

**Corollary 4.40** *Let $G$ be a graph, and let $R$ be a cluster of $G$. Let $\mathcal{C}$ be a collection of disjoint clusters of $R$, such that every cluster $C \in \mathcal{C}$ has the $\alpha_1$-bandwidth property in graph $G$, for some parameter $0 < \alpha_1 < 1$. Denote $\hat{R} = R_{|\mathcal{C}}$ and $\hat{G} = G_{|\mathcal{C}}$, and assume further that $\hat{R}$ has the $\alpha_2$-bandwidth property in graph $\hat{G}$, for some $0 < \alpha_2 < 1$. Then cluster $R$ has the $(\alpha_1 \cdot \alpha_2)$-bandwidth property in graph $G$.*

The following simple claim allows us to transform a routing in a contracted graph $G_{|\mathcal{C}}$ into a routing in the original graph $G$. The proof appears in Section D.23 of Appendix.

**Claim 4.41** *There is an efficient algorithm, that takes as input a graph $G$, a set $\mathcal{C}$ of disjoint clusters of $G$, such that each cluster $C \in \mathcal{C}$ has the $\alpha$-bandwidth property in $G$ for some $0 < \alpha < 1$, and a collection $\mathcal{P}$ of edge-disjoint paths in the contracted graph $G_{|\mathcal{C}}$, routing some set $T \subseteq V(G) \cap V(G_{|\mathcal{C}})$ of vertices to some vertex $x \in V(G) \cap V(G_{|\mathcal{C}})$. The algorithm produces a collection $\mathcal{P}'$ of paths in graph $G$, routing the vertices of $T$ to vertex $x$, such that, for each edge $e \in E(G) \setminus \left(\bigcup_{C \in \mathcal{C}} E(C)\right)$, $\mathrm{cong}_G(\mathcal{P}', e) \leq 1$, and for each edge $e \in \bigcup_{C \in \mathcal{C}} E(C)$, $\mathrm{cong}_G(\mathcal{P}', e) \leq \lceil 1/\alpha \rceil$. Additionally, the algorithm produces another set $\mathcal{P}''$ of edge-disjoint paths in graph $G$, of cardinality at least $\alpha \cdot |T|/2$, routing a subset $T' \subseteq T$ of vertices to $x$.*

The following claim allows us to bound the crossing number of a contracted graph. The proof is provided in Section D.24 of the Appendix.

**Claim 4.42** *Let $I = (G, \Sigma)$ be an instance of the* MCNwRS *problem, and let $\mathcal{C}$ be a collection of disjoint clusters of $G$, such that each cluster in $\mathcal{C}$ has the $\alpha$-bandwidth property, for some $0 < \alpha < 1$. Then there is a drawing $\varphi$ of the contracted graph $G_{|\mathcal{C}}$, with $\mathsf{cr}(\varphi) \leq O(\mathsf{OPT}_{\mathsf{cnwrs}}(I) \cdot \log^8 m/\alpha^2)$, where $m = |E(G)|$. Moreover, for every regular vertex $x \in V(G_{|\mathcal{C}}) \cap V(G)$, the ordering of the edges of $\delta_G(x)$ as they enter $x$ in $\varphi$ is consistent with the rotation $\mathcal{O}_x \in \Sigma$.*

# 5 First Set of Tools: Light Clusters, Bad Clusters, Path-Guided Orderings, and Basic Cluster Disengagement

The main goal of this section is to define and analyze the Basic Cluster Disengagement procedure. Along the way we will define several other central tools that we use throughout the paper, such as light clusters, bad clusters, and path-guided orderings.

We start by defining and analyzing laminar family-based disengagement procedure, which will serve as the basis of the basic disengagement procedure.

## 5.1 Laminar Family-Based Disengagment

We start by defining a laminar family of clusters and its associated partitioning tree.

### 5.1.1 Laminar Family of Clusters and Partitioning Tree

**Definition 5.1 (Laminar family of clusters)** *Let $G$ be a graph, and let $\mathcal{L}$ be a family of clusters of $G$. We say that $\mathcal{L}$ is a* laminar family*, if $G \in \mathcal{L}$, and additionally, for all $S, S' \in \mathcal{L}$, either $S \cap S' = \emptyset$, or $S \subseteq S'$, or $S' \subseteq S$ holds.*

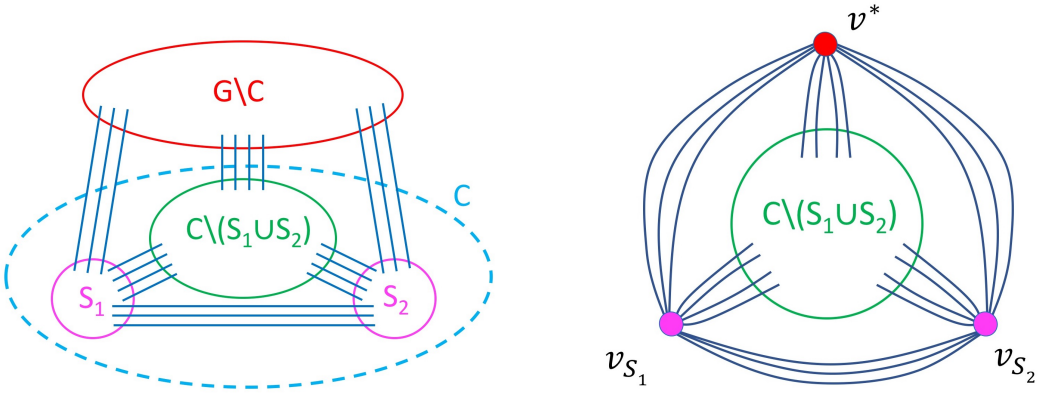
Given a laminar family $\mathcal{L}$ of clusters of $G$, we associate a *paritioning tree* $\tau(\mathcal{L})$ with it, that is defined as follows. The vertex set of the tree is $\{v(S) \mid S \in \mathcal{L}\}$; for every cluster $S \in \mathcal{L}$, we view vertex $v(S)$ as representing the cluster $S$. The root of the tree is $v(G)$ – the vertex associated with the graph $G$ itself. In order to define the edge set, consider a pair $S, S' \in \mathcal{L}$ of clusters. If $S \subsetneq S'$, and there is no other cluster $S'' \in \mathcal{L}$ with $S \subsetneq S'' \subsetneq S'$, then we add an edge $(v(S), v(S'))$ to the tree $\tau(\mathcal{L})$; vertex $v(S)$ becomes a child vertex of $v(S')$ in the tree. We also say that cluster $S$ is a *child cluster* of cluster $S'$, and cluster $S'$ is the *parent cluster* of $S$. Similarly, we define an ancestor-descendant relation between clusters in a natural way: cluster $S \in \mathcal{L}$ is a descendant-cluster of a cluster $S' \in \mathcal{L}$ if vertex $v(S')$ lies on the unique path connecting $v(S)$ to $v(G)$ in the tree $\tau(\mathcal{L})$. If $S$ is a descendant-cluster of $S'$, then $S'$ is an ancestor-cluster of $S$. Notice that every cluster is its own ancestor and its own descendant.

The *depth* of the laminar family $\mathcal{L}$ of clusters, denoted by $\mathsf{dep}(\mathcal{L})$, is the length of the longest root-to-leaf path in tree $\tau(\mathcal{L})$. We also say that cluster $S$ *lies at level* $i$ *of the laminar family* $\mathcal{L}$ iff the distance from $v(S)$ to the root of the tree $\tau(\mathcal{L})$ is exactly $i$.

### 5.1.2 Definition of Laminar Family-Based Disengagement

The input to the Laminar Family-Based Disengegement is an instance $I = (G, \Sigma)$ of MCNwRS, a laminar family $\mathcal{L}$ of clusters of $G$, and, for every cluster $C \in \mathcal{L}$, a circular ordering $\mathcal{O}(C)$ of the edges of $\delta_G(C)$ (for $C = G$, $\delta_G(C) = \emptyset$ and the ordering $\mathcal{O}(C)$ is trivial). The output of the procedure is a collection $\mathcal{I} = \{I_C = (G_C, \Sigma_C) \mid C \in \mathcal{L}\}$ of subinstances of $I$, that are defined as follows.

Consider a cluster $C \in \mathcal{L}$, and denote by $\mathcal{W}(C) \subseteq \mathcal{L}$ the set of all child-clusters of $C$. In order to construct the graph $G_C$, we start with $G_C = G$. For every cluster $C' \in \mathcal{W}(C)$, we contract the vertices of $C'$ into a supernode $v_{C'}$. Additionally, if $C \neq G$, then we contract all vertices of $V(G) \setminus V(C)$ into a supernode $v^*$. This completes the definition of the graph $G_C$ (see Figure 8). We now define the rotation system $\Sigma_C$ for $G_C$. If $C \neq G$, then the set of edges incident to $v^*$ in $G_C$ is exactly $\delta_G(C)$. We set the rotation $\mathcal{O}_{v^*} \in \Sigma_C$ to be $\mathcal{O}(C)$. For every cluster $C' \in \mathcal{W}(C)$, the set of edges incident to $v_{C'}$ in $G_C$ is $\delta_G(C')$. We set the rotation $\mathcal{O}_{v_{C'}} \in \Sigma_C$ to be $\mathcal{O}(C')$. For every regular vertex $x \in V(G_C) \cap V(G)$, $\delta_{G_C}(v) = \delta_G(v)$ holds, and its rotation $\mathcal{O}_v \in \Sigma_C$ remains the same as in $\Sigma$.




(a) Layout of graph $G$ with respect to $C$.

(b) Graph $G_C$.

Figure 8: Construction of graph $G_C$, where $C \in \mathcal{L}$ is a cluster with two child-clusters $S_1, S_2$.

We refer to the resulting collection $\mathcal{I}$ of clusters as *disengagement of instance* $I$ *via the laminar family* $\mathcal{L}$ *and the collection* $\{\mathcal{O}(C)\}_{C \in \mathcal{L}}$ *of orderings.*

### 5.1.3 Analysis

We start by showing that the total number of edges in all instances that we obtain via the laminar family-based disengagement procedure is small compared to $|E(G)|$.

**Lemma 5.2** *Let $I = (G, \Sigma)$ be an instance of* MCNwRS*, let $\mathcal{L}$ be a laminar family of clusters of $G$, and let $\{\mathcal{O}(C)\}_{C\in\mathcal{L}}$ be a collection of orderings of the edges of $\delta_G(C)$, for every cluster $C \in \mathcal{L}$. Consider the collection $\mathcal{I} = \{I_C = (G_C, \Sigma_C) \mid C \in \mathcal{L}\}$ of subinstances of $I$ obtained by applying the laminar family-based decomposition to instance $I$ via the laminar family $\mathcal{L}$ and the orderings in $\{\mathcal{O}(C)\}_{C\in\mathcal{L}}$. Then $\sum_{C\in\mathcal{L}} |E(G_C)| \leq O(\mathsf{dep}(\mathcal{L}) \cdot |E(G)|)$.*

**Proof:** Fix an integer $1 \leq i \leq \mathsf{dep}(\mathcal{L})$ and denote by $\mathcal{L}_i \subseteq \mathcal{L}$ the set of all clusters of $\mathcal{L}$ that lie at level $i$ of the partitioning tree. From the definition of the laminar family $\mathcal{L}$ and the partitioning tree, all clusters in set $\mathcal{L}_i$ are mutually disjoint. Consider now some cluster $C \in \mathcal{L}_i$, and its corresponding graph $G_C$. Note that every edge of $G_C$ corresponds to some distinct edge of cluster $C$, except for the edges incident to the supernode $v^*$. However, the number of edges incident to $v^*$ is at most $|\delta_G(C)|$. Therefore, overall, $|E(G_C)| \leq |E_G(C)| + |\delta_G(C)|$. Since all clusters in $\mathcal{L}_i$ are mutually disjoint, we get that:

$$\sum_{C\in\mathcal{L}_i} |E(G_C)| \leq \sum_{C\in\mathcal{L}_i} (|E_G(C)| + |\delta_G(C)|) \leq O(|E(G)|).$$

Summing over all indices $1 \leq i \leq \mathsf{dep}(\mathcal{L})$, we get that $\sum_{C\in\mathcal{L}} |E(G_C)| = O(\mathsf{dep}(\mathcal{L}) \cdot |E(G)|)$. □

Next, we show that solutions to the instances in $\mathcal{I}$ can be efficiently combined in order to obtain a solution to instance $I$ of relatively low cost. The proof is conceptually simple but somewhat technical, and is deferred to Section E.1 of Appendix.

**Lemma 5.3** *There is an efficient algorithm, that takes as input an instance $I = (G, \Sigma)$ of* MCNwRS*, a laminar family $\mathcal{L}$ of clusters of $G$, a collection $\{\mathcal{O}(C)\}_{C\in\mathcal{L}}$ containing an ordering of the edges of $\delta_G(C)$ for every cluster $C \in \mathcal{L}$, and, for every cluster $C \in \mathcal{L}$, a solution $\varphi(I_C)$ to the instance $I_C \in \mathcal{I}$ of* MCNwRS*, where $\mathcal{I} = \{I_C \mid C \in \mathcal{L}\}$ is the collection of subinstances of $I$ obtained via laminar family-based disengagement of $I$ via $(\mathcal{L}, \{\mathcal{O}(C)\}_{C\in\mathcal{L}})$. The output of the algorithm is a solution to instance $I$ with cost at most $\sum_{I_C\in\mathcal{I}} \mathsf{cr}(\varphi(I_C))$.*

So far we have shown that, if $\mathcal{I}$ is the collection of instances that is constructed via a laminar family-based disengagement of instance $I$, then the total number of edges in all resulting instances is at most $O(|E(G)| \cdot \mathsf{dep}(\mathcal{L}))$, and that there is an efficient algorithm for combining the solutions to the resulting instances, in order to obtain a low-cost solution to the original instance $I$. Another highly desirable property of the family $\mathcal{L}$ of clusters would be for $\sum_{I'\in\mathcal{I}} \mathsf{OPT}_{\mathsf{cnwrs}}(I')$ to be small compared to $\mathsf{OPT}_{\mathsf{cnwrs}}(I) + |E(G)|$. Unfortunately, we cannot show that this property holds, except for some special cases. The Basic Cluster Disengagement procedure essentially considers one such special case, in which $\sum_{I'\in\mathcal{I}} \mathsf{OPT}_{\mathsf{cnwrs}}(I')$ can be appropriately bounded. In order to define this procedure formally, we first need to define several central notions that are used throughout our algorithm, namely: light clusters, bad clusters, and path-guided orderings.

## 5.2 Light Clusters, Bad Clusters, and Path-Guided Orderings

**Definition 5.4 (Light Cluster)** *Let $I = (G, \Sigma)$ be an instance of* MCNwRS*, and let $C$ be a cluster of $G$. Assume further that we are given a distribution $\mathcal{D}(C)$ over the set $\Lambda(C)$ of internal $C$-routers. We say that cluster $C$ is* $\beta$-light with respect to $\mathcal{D}(C)$ *if, for every edge $e \in E(C)$:*

$$\mathbf{E}_{\mathcal{Q}\sim\mathcal{D}(C)} \left[(\mathrm{cong}_G(\mathcal{Q}, e))^2\right] \leq \beta.$$

**Definition 5.5 (Bad Cluster)** *Let $I = (G, \Sigma)$ be an instance of* MCNwRS*, let $C$ be a cluster of $G$, and let $\Sigma(C)$ be the rotation system for $C$ induced by $\Sigma$. We say that $C$ is a $\beta$-*bad *cluster, if:*

$$\mathsf{OPT}_{\mathsf{cnwrs}}(C, \Sigma(C)) + |E(C)| \geq \frac{|\delta_G(C)|^2}{\beta}.$$

**Path-Guided Orderings.** Let $I = (G, \Sigma)$ be an instance of MCNwRS, and let $C$ be a cluster of $G$. Consider an internal $C$-router $\mathcal{Q} = \{Q(e) \mid e \in \delta_G(C)\}$. Recall that there is some vertex $u \in V(C)$ (the center of the router), such that, for all $e \in \delta_G(C)$, path $Q(e)$ has edge $e$ as its first edge, vertex $u$ as its last vertex, and all inner vertices of $Q(e)$ lie in $C$.

We will use the internal $C$-router $\mathcal{Q}$, and the rotation system $\Sigma$ for $G$, in order to define a circular ordering $\mathcal{O}$ of the edges of $\delta_G(C)$. We refer to the ordering $\mathcal{O}$ as *an ordering guided by $\mathcal{Q}$ and $\Sigma$*. Ordering $\mathcal{O}$ of the edges of $\delta_G(C)$ is constructed as follows. Denote $\delta_G(u) = \{a_1, \ldots, a_r\}$, where the edges are indexed according to their circular ordering $\mathcal{O}_u \in \Sigma$. For all $1 \leq i \leq r$, let $\mathcal{Q}_i \subseteq \mathcal{Q}$ be the set of paths whose last edge is $a_i$. We first define an ordering $\hat{\mathcal{O}}$ of the paths in $\mathcal{Q}$, where the paths in sets $\mathcal{Q}_1, \ldots, \mathcal{Q}_r$ appear in the natural order of their indices, and for all $1 \leq i \leq r$, the ordering of the paths in $\mathcal{Q}_i$ is arbitrary. Ordering $\hat{\mathcal{O}}$ of the paths in $\mathcal{Q}$ naturally defines the ordering $\mathcal{O}$ of the edges of $\delta_G(C)$: we obtain the ordering $\mathcal{O}$ from $\hat{\mathcal{O}}$ by replacing, for every path $Q(e) \in \mathcal{Q}$, the path $Q(e)$ in $\hat{\mathcal{O}}$ with the edge $e$ (the first edge of $Q(e)$). We refer to $\mathcal{O}$ as the ordering of the edges of $\delta_G(C)$ that is guided by $\mathcal{Q}$ and $\Sigma$, and we denote it by $\mathcal{O}^{\text{guided}}(\mathcal{Q}, \Sigma)$. A convenient way to think of the ordering $\mathcal{O}^{\text{guided}}(\mathcal{Q}, \Sigma)$ of the edges of $\delta_G(C)$ is that this order is determined by the order in which the paths of $\mathcal{Q}$ enter the vertex $u$, which in turn is determined by the rotation $\mathcal{O}_u \in \Sigma$ (as the last edge on each path in $\mathcal{Q}$ lies in $\delta_G(u)$).

## 5.3 Basic Cluster Disengagement

The input to the Basic Cluster Disengagement procedure consists of an instance $I = (G, \Sigma)$ of MCNwRS and a laminar family $\mathcal{L}$ of clusters of $G$. Recall that, by the definition, $G \in \mathcal{L}$ must hold. We further assume that we are given a partition $(\mathcal{L}^{\text{light}}, \mathcal{L}^{\text{bad}})$ of the clusters of $\mathcal{L} \setminus \{G\}$, and, for every cluster $C \in \mathcal{L}^{\text{light}}$, a distribution $\mathcal{D}(C)$ over its internal $C$-routers (that may be given implicitly). The output of the procedure is a collection $\mathcal{I} = \{I_C \mid C \in \mathcal{L}\}$ of subinstances of $I$. In order to define the instances of $\mathcal{I}$, we will define, for every cluster $C \in \mathcal{L}$, an ordering $\mathcal{O}(C)$ of the edges of $\delta_G(C)$. Family $\mathcal{I}$ of subinstances of $I$ is then constructed by disengaging instance $I$ via the laminar family $\mathcal{L}$ and the collection $\{\mathcal{O}(C)\}_{C \in \mathcal{L}}$ of orderings.

In order to describe the algorithm for computing the collection $\mathcal{I}$ of subinstances of $I$, it is now enough to describe an algorithm that computes, for every cluster $C \in \mathcal{L}$, an ordering $\mathcal{O}(C)$ of the edges of $\delta_G(C)$. Consider any such cluster $C \in \mathcal{L}$. If $C = G$, then $\delta_G(C) = \emptyset$, and the ordering $\mathcal{O}(C)$ is trivial. If $C \in \mathcal{L}^{\text{bad}}$, then we let $\mathcal{O}(C)$ be an arbitrary ordering of the edges of $\delta_G(C)$. Lastly, consider a cluster $C \in \mathcal{L}^{\text{light}}$. We select an internal router $\mathcal{Q}(C) \in \Lambda_G(C)$ from the given distribution $\mathcal{D}(C)$ at random. We view the paths in $\mathcal{Q}(C)$ as being directed towards the center vertex $u(C)$ of the router. We use the algorithm from Lemma 4.7 to compute a non-transversal path set $\tilde{\mathcal{Q}}(C)$, routing all edges of $\delta_G(C)$ to vertex $u(C)$, so $\tilde{\mathcal{Q}}(C)$ is also an internal $C$-router. The algorithm ensures that, for every edge $e \in E(G)$, $\text{cong}_G(\tilde{\mathcal{Q}}(C), e) \leq \text{cong}_G(\mathcal{Q}(C), e)$. We then let the ordering $\mathcal{O}(C)$ of the edges of $\delta_G(C)$ be an ordering guided by the set $\tilde{\mathcal{Q}}(C)$ of paths in graph $G$, and the rotation system $\Sigma$, so $\mathcal{O}(C) = \mathcal{O}^{\text{guided}}(\tilde{\mathcal{Q}}(C), \Sigma)$.

This completes the description of the algorithm for selecting an ordering $\mathcal{O}(C)$ of the edges in $\delta_G(C)$ for each cluster $C \in \mathcal{L}$; note that this algorithm is randomized. This also completes the description of the algorithm for performing Basic Cluster Disengagement, that we refer to as AlgBasicDisengagement in the remainder of the paper. Since this algorithm essentially performs disengagement via a laminar

family of clusters of $G$, Lemma 5.2 and Lemma 5.3 continue to hold for the resulting collection $\mathcal{I}$ of instances. But we can now show that, under some conditions, we can bound the expected value of $\sum_{I' \in \mathcal{I}} \mathsf{OPT}_{\mathsf{cnwrs}}(I')$.

When using the algorithm AlgBasicDisengagement for performing Basic Cluster Disengagement of an instance $I$ of MCNwRS via a laminar family $\mathcal{L}$, we will typically require that the following properties hold, for some parameter $\beta$:

P1. every cluster $C \in \mathcal{L}^{\text{bad}}$ is $\beta$-bad, and has the $\alpha_0$-bandwidth property in $G$, for some $\alpha_0 \geq \Omega(1/\log^{12} m)$;

P2. every cluster $C \in \mathcal{L}^{\text{light}}$ is $\beta$-light with respect to the given distribution $\mathcal{D}(C)$ over the set $\Lambda(C)$ of its internal routers; and

P3. for every cluster $C \in \mathcal{L}$, there is a distribution $\mathcal{D}'(C)$ over the set $\Lambda'(C)$ of external $C$-routers, such that for every edge $e \in E(G \setminus C)$, $\mathbf{E}_{\mathcal{Q}'(C) \sim \mathcal{D}'(C)} \left[\text{cong}_G(\mathcal{Q}'(C), e)\right] \leq \beta$.

Observe that the algorithm for computing the family $\mathcal{I}$ of clusters is randomized. We show in the following lemma that, if all the above conditions hold, then the expected value of $\sum_{I' \in \mathcal{I}} \mathsf{OPT}_{\mathsf{cnwrs}}(I')$ is suitably bounded. The proof is somewhat technical, and is deferred to Section E.2 of Appendix.

**Lemma 5.6** *Let $I = (G, \Sigma)$ be an instance of the* MCNwRS *problem, $\mathcal{L}$ a laminar family of clusters of $G$, $(\mathcal{L}^{\text{light}}, \mathcal{L}^{\text{bad}})$ a partition of cluster set $\mathcal{L} \setminus \{G\}$, and, for every cluster $C \in \mathcal{L}^{\text{light}}$, $\mathcal{D}(C)$ a distribution over internal $C$-routers. Let $\mathcal{I}$ be the collection of subinstances of $I$ obtained by applying Algorithm* AlgBasicDisengagement *to instance $I$, with laminar family $\mathcal{L}$, cluster sets $\mathcal{L}^{\text{light}}, \mathcal{L}^{\text{bad}}$, and distributions $\{\mathcal{D}(C)\}_{C \in \mathcal{L}^{\text{light}}}$. Assume further that Properties P1 – P3 hold for some parameter $\beta \geq c(\log |E(G)|)^{18}$, where $c$ is a large enough constant. Then:*

$$\mathbf{E}\left[\sum_{I' \in \mathcal{I}} \mathsf{OPT}_{\mathsf{cnwrs}}(I')\right] \leq O(\mathsf{dep}(\mathcal{L}) \cdot \beta^2 \cdot (\mathsf{OPT}_{\mathsf{cnwrs}}(I) + |E(G)|)).$$

We note that the distributions $\{\mathcal{D}'(C)\}_{C \in \mathcal{L}}$ over external routers of the clusters play no role in constructing the collection $\mathcal{I}$ of subinstances of $I$, but they are essential in order to ensure that the expectation of $\sum_{I' \in \mathcal{I}} \mathsf{OPT}(I')$ is suitably bounded. Since this property is essential to us, we will only use Basic Cluster Disengagement when such distributions are given. Therefore, abusing the notation, we refer to the family $\mathcal{I}$ of subinstances of $I$ computed via Basic Cluster Disengagement of instance $I$ as described above, as a Basic Cluster Disengagement of $I$ via the tuple $(\mathcal{L}, \mathcal{L}^{\text{bad}}, \mathcal{L}^{\text{light}}, \{\mathcal{D}'(C)\}_{C \in \mathcal{L}}, \{\mathcal{D}(C)\}_{C \in \mathcal{L}^{\text{light}}})$.

# 6 Second Main Tool: Cluster Classification

In this section we introduce our second main tool, the algorithm AlgClassifyCluster, that is summarized in the following theorem.

**Theorem 6.1** *There is a randomized algorithm, that, given an instance $I = (G, \Sigma)$ of* MCNwRS *problem with $|E(G)| = m$, a cluster $J \subseteq G$ that has the $\alpha_0$-bandwidth property in $G$, for $\alpha_0 = 1/\log^{50} m$, and a parameter $0 < p < 1$, either returns FAIL, or computes a distribution $\mathcal{D}(J)$ over the set $\Lambda(J)$ of internal $J$-routers, such that cluster $J$ is $\beta^*$-light with respect to $\mathcal{D}(J)$, where $\beta^* = 2^{O(\sqrt{\log m} \cdot \log\log m)}$. Moreover, if cluster $J$ is not $\eta^*$-bad, for $\eta^* = 2^{O((\log m)^{3/4} \log\log m)}$, then the probability that the algorithm returns FAIL is at most $p$. The running time of the algorithm is* $\text{poly}(m \cdot \log(1/p))$.

We will sometimes say that the algorithm AlgClassifyCluster *errs* if it returns FAIL and yet cluster $J$ is not $\eta^*$-bad. Clearly, the probability that the algorithm errs is at most $p$. We note that the distribution $\mathcal{D}(J)$ over the set $\Lambda(J)$ of internal $J$-routers that the algorithm computes may be returned by the algorithm implicitly, by providing another efficient algorithm to draw a router from the distribution.

In order to prove Theorem 6.1, it is sufficient to prove the following theorem.

**Theorem 6.2** *There is an efficient randomized algorithm, that, given an instance $I = (G, \Sigma)$ of* MCNwRS *with $|E(G)| = m$, a cluster $J \subseteq G$ that has the $\alpha_0$-bandwidth property in $G$, for $\alpha_0 = \Omega(1/\log^{50} m)$, either returns FAIL, or computes a distribution $\mathcal{D}(J)$ over the set of internal $J$-routers, such that cluster $J$ is $\beta^*$-light with respect to $\mathcal{D}(J)$, where $\beta^* = 2^{O(\sqrt{\log m}\cdot\log\log m)}$. Moreover, if cluster $J$ is not $\eta^*$-bad, for $\eta^* = 2^{O((\log m)^{3/4}\log\log m)}$, then the probability that the algorithm returns FAIL is at most $1/2$.*

Indeed, given a graph $G$, a cluster $J$ of $G$ and a parameter $0 < p < 1$, as in the statement of Theorem 6.1, we simply run the algorithm from Theorem 6.2 $\lceil \log(1/p) \rceil$ times on the input instance $(G, \Sigma)$ and cluster $J$ of $G$. If the algorithm returns FAIL in every of these iterations, then we also return FAIL. Otherwise, in at least one of the iterations, the algorithm from Theorem 6.2 returns a distribution $\mathcal{D}(J)$ over the set $\Lambda(J)$ of internal $J$-routers, such that cluster $J$ is $\beta^*$-light with respect to $\mathcal{D}(J)$. We then return the distribution $\mathcal{D}(J)$ as the algorithm's outcome. It is immediate to verify that the probability that the algorithm errs is at most $p$, and that its running time is $\mathrm{poly}(m \cdot \log(1/p))$, as required.

In the remainder of this section, we focus on the proof of Theorem 6.2. It will be convenient for us to consider the augmentation $J^+$ of cluster $J$. Recall that this is the graph that is obtained from $G$ by subdiving every edge $e \in \delta_G(J)$ with a vertex $t_e$, letting $T = \{t_e \mid e \in \delta_G(J)\}$ be the set of the new vertices, and then letting $J^+$ be the subgraph of the resulting graph induced by $T \cup V(J)$. We refer to vertices of $T$ as *terminals*, and we denote $|T| = k$. Recall that, from the $\alpha_0$-bandwidth property of cluster $J$, the set $T$ of terminals is $\alpha_0$-well-linked in $J^+$. Since the degree of every terminal in $J^+$ is 1, the rotation system $\Sigma$ for graph $G$ naturally defines a unique rotation system $\Sigma(J^+)$ for $J^+$. Moreover, cluster $J$ is $\eta^*$-bad iff $\mathsf{OPT}_{\mathsf{cnwrs}}(J^+, \Sigma(J^+)) + |E(J^+ \setminus T)| \geq k^2/\eta^*$.

Let $\Lambda(J^+, T)$ denote the collection of all sets $\mathcal{Q}$ of paths, such that paths in $\mathcal{Q}$ route all vertices of $T$ to some vertex $x \in V(J^+) \setminus T$, in graph $J^+$. We sometimes also call $\mathcal{Q}$ a router, and refer to $x$ as the *center vertex* of the router. Notice that, if we are given a distribution $\mathcal{D}$ over sets of paths in $\Lambda(J^+, T)$, such that, for every edge $e \in E(J^+)$, $\mathbf{E}_{\mathcal{Q}\sim\mathcal{D}}\left[(\mathrm{cong}_{J^+}(\mathcal{Q}, e))^2\right] \leq \beta^*$, then we can immediately obtain a distribution $\mathcal{D}(J)$ over the set $\Lambda(J)$ of internal $J$-routers, such that cluster $J$ is $\beta^*$-light with respect to $\mathcal{D}(J)$.

From now on we will only focus on graph $J^+$ and the corresponding rotation system $\Sigma(J^+)$, so it will be convenient for us to denote graph $J^+$ by $G$ and $\Sigma(J^+)$ by $\Sigma$. We denote by $I = (G, \Sigma)$ the resulting instance of MCNwRS. From now on our goal is to design a randomized algorithm, that either computes a distribution $\mathcal{D}$ over the set $\Lambda(G, T)$ of internal routers, such that, for every edge $e \in E(G)$, $\mathbf{E}_{\mathcal{Q}\sim\mathcal{D}}\left[(\mathrm{cong}_G(\mathcal{Q}, e))^2\right] \leq \beta^*$, or returns FAIL. We need to ensure that, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I) + |E(G \setminus T)| < k^2/\eta^*$, then the probability that the algorithm returns FAIL is at most $1/2$.

We now provide some intuition. We first show below an algorithm called AlgFindGuiding that "almost" provides the required guarantees. Specifically, if we are guaranteed that $|E(G)| \leq k \cdot \eta$ for some small parameter $\eta$, then the algorithm either computes a distribution $\mathcal{D}$ over sets of paths in $\Lambda(G, T)$, such that, for every edge $e \in E(G)$, $\mathbf{E}_{\mathcal{Q}\sim\mathcal{D}}\left[(\mathrm{cong}_G(\mathcal{Q}, e))^2\right] \leq \mathrm{poly}(\log m)/\mathrm{poly}(\alpha_0)$, or it returns FAIL, with the guarantee that, if $\mathsf{OPT}_{\mathsf{cnwrs}}(G, \Sigma) + |E(G \setminus T)| < \frac{k^2\,\mathrm{poly}(\alpha_0)}{\mathrm{poly}(\eta\log m)}$, then the probability that the algorithm returns FAIL is at most $1/2$. We could use this theorem directly if $|E(G)| \leq k \cdot \eta$ holds for some $\eta \leq (\eta^*)^\epsilon$ where $\epsilon$ is a constant, but unfortunately this is not guaranteed in the statement of

Theorem 6.1, and $|E(G)|$ may be arbitrarily large compared to $k$. In order to overcome this difficulty, we use another algorithm, that, given a graph $G$ and a set $T$ of its terminals as above, computes a collection $\mathcal{C}$ of disjoint clusters of $G \setminus T$, such that, for each cluster $C \in \mathcal{C}$, either (i) there is an internal $C$-router $\mathcal{Q}(C) \in \Lambda(C)$ such that the paths in $\mathcal{Q}(C)$ are edge-disjoint; or (ii) $C$ is $\eta$-bad for some parameter $\eta \ll \eta^*$; or (iii) $|E(C)| \leq |\delta_G(C)| \cdot O(\mathrm{poly}(\eta \log m))$. In the latter case, we say that $C$ is a *concise* cluster. We then apply the algorithm AlgFindGuiding to each concise cluster. As a result, for each such concise cluster $C \in \mathcal{C}$, we will either establish, with high probability, that it is a $\eta'$-bad cluster, for some parameter $\eta'$, or we will compute a distribution $\mathcal{D}(C)$ over the set $\Lambda(C)$ of internal $C$-routers, such that cluster $C$ is $\beta'$-light with respect to $\mathcal{D}(C)$, for some parameter $\beta'$. The algorithm for computing the collection $\mathcal{C}$ of clusters of $G$ also guarantees that each cluster $C \in \mathcal{C}$ has the $\alpha'$-bandwidth property, for $\alpha' = \Omega(1/\log^{1.5} m)$, and that the corresponding contracted graph $G_{|\mathcal{C}}$ contains significantly fewer edges: $|E(G_{|\mathcal{C}})| \leq |E(G)|/\eta$. Intuitively, we would then like to continue with the contracted graph $G_{|\mathcal{C}}$, applying exactly the same algorithm to this graph. We could continue this process, obtaining a clustering $\mathcal{C}'$ of this new contracted graph, and so on, until we reach a final contracted graph $\hat{G}$, with $|E(\hat{G})| \leq O(k\eta)$. At this point we can apply the algorithm AlgFindGuiding to graph $\hat{G}$ directly, and as a result, we either obtain the desired distribution $\mathcal{D}$ over path sets in $\Lambda(G,T)$, or establish, with high probability, that $\mathsf{OPT}_{\mathsf{cnwrs}}(G,\Sigma) + |E(G \setminus T)|$ is sufficiently high. A problem with this approach is that the algorithm AlgFindGuiding requires a rotation system $\Sigma'$ for its input graph $H$. Recall that the algorithm guarantees that, if $\mathsf{OPT}_{\mathsf{cnwrs}}(H,\Sigma') + |E(H)|$ is sufficiently low, then it only returns FAIL with probability at most $1/2$. The difficulty is that, if $H$ is the contracted graph $G_{|\mathcal{C}}$, then it is not immediately clear how to define the rotation system $\Sigma'$ for $H$, such that $\mathsf{OPT}_{\mathsf{cnwrs}}(H,\Sigma')$ is not much higher than $\mathsf{OPT}_{\mathsf{cnwrs}}(G,\Sigma)$.

In order to overcome this difficulty, we design the algorithm AlgFindGuiding for a more general setting. In this setting, the input is a graph $H$, a rotation system $\Sigma'$ for $H$, and a set $T'$ of terminals of $H$, such that the terminals of $T'$ are $\alpha$-well-linked in $H$. Additionally, we are given some collection $\mathcal{C}'$ of disjoint $\eta'$-bad clusters in $H$. We also require that $|E(H_{|\mathcal{C}'})| \leq |T'|\eta'$, for some parameter $\eta'$. The algorithm either returns FAIL, or computes a distribution $\mathcal{D}'$ over the set $\Lambda(H,T')$ of routers, such that, for every edge $e \in E(H)$, $\mathbf{E}_{\mathcal{Q}\sim\mathcal{D}'}\left[(\mathrm{cong}(\mathcal{Q},e))^2\right]$ is sufficiently low. We are also guaranteed that, if $|\mathsf{OPT}_{\mathsf{cnwrs}}(H,T')| + |E(H \setminus T')|$ is sufficiently small compared to $|T'|^2$, then the probability that the algorithm returns FAIL is at most $1/2$. This stronger version of algorithm AlgFindGuiding will allow us to carry out the algorithm outlined above.

We now provide formal descriptions of the two main tools that our algorithm uses. The first tool allows us to compute a decomposition of an input graph $G$ into a collection of clusters, each of which is either light, bad, or concise. The proof of the theorem uses rather standard techniques and is deferred to Section F.1 of Appendix.

**Theorem 6.3** *There is an efficient algorithm, that we refer to as* AlgInitPartition*, whose input consists of a connected $m$-edge graph $G$, a set $T \subseteq V(G)$ of $k$ vertices called* terminals*, such that each vertex of $T$ has degree 1 in $G$, and a parameter $\eta > \log m$, such that $k \leq \frac{m}{16\eta \log m}$. The algorithm computes a collection $\mathcal{C}$ of vertex-disjoint clusters of $G \setminus T$, a partition $(\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3)$ of $\mathcal{C}$ into three subsets, and, for every cluster $C \in \mathcal{C}_3$, an internal $C$-router $\mathcal{Q}(C) \in \Lambda(C)$, where the paths of $\mathcal{Q}(C)$ are edge-disjoint, such that the following additional properties hold:*

- *every cluster $C \in \mathcal{C}$ has the $\alpha'$-bandwidth property, where $\alpha' = \frac{1}{16\beta_{\mathrm{ARV}}(m)\cdot\log m} = \Omega\left(\frac{1}{\log^{1.5} m}\right)$;*
- *for every cluster $C \in \mathcal{C}_1$, $|E(C)| \leq O(\eta^4 \log^8 m) \cdot |\delta_G(C)|$;*
- *for every cluster $C \in \mathcal{C}_2$, $\mathsf{OPT}_{\mathsf{cr}}(C) \geq \Omega(|E(C)|^2/(\eta^2 \,\mathrm{poly}\log m))$, and $|E(C)| \geq \Omega(\eta^4 |\delta_G(C)| \log^8 m)$;*
- *$\bigcup_{C\in\mathcal{C}} V(C) = V(G) \setminus T$; and*

- $|E(G_{|\mathcal{C}})| \leq |E(G)|/\eta$.

Note that, from the theorem statement, for every cluster $C \in \mathcal{C}_2$, $\mathsf{OPT}_{\mathsf{cr}}(C) \geq \Omega(|\delta_G(C)|^2\eta^6/\operatorname{poly}\log m)$. We will informally refer to clusters in $\mathcal{C}_1$ as concise clusters.

Let $H$ be a graph and let $T$ be a set of vertices of $H$ called terminals. We say that a set $\mathcal{Q} = \{Q(t) \mid t \in T\}$ of paths in graph $H$ is a *router for $H$ and $T$* if there is a vertex $x \in V(H)$, such that, for every terminal $t \in T$, path $Q(t)$ originates at $t$ and temrinates at $x$. We denote by $\Lambda(H,T)$ the set of all routers for $H$ and $T$. Our second tool is algorithm AlgFindGuiding, summarized in the following theorem.

**Theorem 6.4** *There are universal constants $c_0$ and $c^*$, and an efficient randomized algorithm, called* AlgFindGuiding*, that receives as input an instance $I = (H, \Sigma)$ of* MCNwRS*, where $|E(H)| = m$, a set $T \subseteq V(H)$ of $k$ vertices of $H$ called terminals, and a collection $\mathcal{C}$ of disjoint clusters of $H \setminus T$. Additionally, the algorithm receives as input parameters $0 \leq \alpha, \alpha' \leq 1$ and $\eta, \eta' \geq 1$, such that the following conditions hold:*

- *$\eta \geq \frac{c^* \log^{46} m}{\alpha^{10}(\alpha')^2}$ and $\eta' \geq \eta^{13}$;*
- *$k \geq |E(H_{|\mathcal{C}})|/\eta$;*
- *every terminal $t \in T$ has degree 1 in $H$;*
- *the set $T$ of terminals is $\alpha$-well-linked in the contracted graph $H_{|\mathcal{C}}$; and*
- *every cluster $C \in \mathcal{C}$ is $\eta'$-bad and has the $\alpha'$-bandwidth property in $H$.*

*The algorithm either returns FAIL or (explicitly) returns a distribution $\mathcal{D}$ over the routers in $\Lambda(H,T)$, such that, for every edge $e \in E(H)$, $\mathbf{E}_{\mathcal{Q}\sim\mathcal{D}}\left[(\operatorname{cong}(\mathcal{Q},e))^2\right] \leq O\left(\frac{\log^{32} m}{\alpha^{12}(\alpha')^8}\right)$. Moreover, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I) + |E(H \setminus T)| \leq \frac{(k\alpha^4\alpha')^2}{c_0\eta' \log^{50} m}$, then the probability that the algorithm returns FAIL is at most 1/2.*

The proof of Theorem 6.4 is quite technical, and is deferred to Section 11. We note that the algorithm returns the distribution $\mathcal{D}$ explicitly, that is, it lists all routers $\mathcal{Q} \in \Lambda(H,T)$ that have a non-zero probability, together with their probability values in $\mathcal{D}$. In the remainder of this section, we complete the proof of Theorem 6.1, using the algorithms AlgInitPartition and AlgFindGuiding.

Note that we can assume, throughout the proof, that $m$ is sufficiently large (larger than some large enough constant). Otherwise, since the vertices of $T$ are $\alpha$-well-linked in $G$, we get that $k \leq m \leq O(1)$. We can then let $\mathcal{D}$ be a distribution that gives a probability 1 to an internal router $\mathcal{Q}$ with target vertex $u$, where $u$ is an arbitrary vertex of $V(G) \setminus T$, and $\mathcal{Q}$ is an arbitrary collection of simple paths routing the vertices of $T$ to $u$. Clearly, for every edge $e \in E(G)$, $(\operatorname{cong}_G(\mathcal{Q},e))^2 \leq k^2 \leq O(1)$.

## 6.1 Main Parameters

We now introduce some parameters that our algorithm uses. The main parameter is $\eta = 2^{(\log m)^{3/4}}$. Our algorithm will consist of $(\ell-1)$ phases, for $\ell \leq O\left(\frac{\log m}{\log \eta}\right) = O((\log m)^{1/4})$. At the beginning of the $i$-th phase, we will be given a collection $\mathcal{C}_i$ of disjoint clusters of $G \setminus T$.

**Parameters for bandwidth property.** We will use the following parameters for bandwidth property of the clusters. Recall that $\alpha_0 = \frac{1}{(\log m)^{50}}$ is the parameter from the statement of Theorem 6.1. For $1 \leq i \leq \ell$, $\alpha_i = (\alpha_0)^i = \frac{1}{(\log m)^{50\cdot i}}$. We will ensure that for all $1 \leq i \leq \ell$, every cluster in $\mathcal{C}_i$ has the $\alpha_i$-bandwidth property. Note that $\alpha_\ell = 1/(\log m)^{50\cdot\ell} = 1/2^{O((\log m)^{1/4}\log\log m)}$.

**Parameters for light clusters.** We set $\beta_0 = 1$, and for $1 \leq i \leq \ell$, $\beta_i = \frac{(\log m)^{56}}{(\alpha_0)^{12} \cdot (\alpha_i)^8} \cdot \beta_{i-1}$. It is easy to verify that $\beta_\ell \leq (\log m)^{O(\ell^2)} \leq 2^{O((\log m)^{1/2} \log\log m)} \leq \beta^*$.

**Parameters for bad clusters.** We define the following parameters: $\eta_0 = \eta^4 \cdot \log^9 m = 2^{O((\log m)^{3/4})}$; $\eta_1 = \max\left\{(2\eta_0)^{13}, \eta_0 \cdot (\log m)^{80}\right\} = 2^{O((\log m)^{3/4})}$, and for each $1 \leq i \leq \ell$, $\eta_i = \eta_{i-1} \cdot \max\left\{(\log m)^{80+50 \cdot i}, \beta_{i-1}^3\right\}$. Clearly, $\eta_i \leq (\beta_\ell)^3 \cdot \eta_{i-1}$, and $\eta_\ell \leq \eta_1 \cdot (\beta_\ell)^{3\ell} \leq 2^{O((\log m)^{3/4} \log\log m)} \leq \eta^*$.

### 6.2 Algorithm Execution

As already mentioned, the algorithm consists of $(\ell - 1)$ phases, where $\ell = O((\log m)^{1/4})$. At the beginning of the $i$th phase, we will be given a collection $\mathcal{C}_i$ of disjoint clusters of $G \setminus T$, which is partitioned into two subsets: $\mathcal{C}_i^{\text{light}}$ and $\mathcal{C}_i^{\text{bad}}$. For each cluster $C \in \mathcal{C}_i^{\text{light}}$, we will also be given a distribution $\mathcal{D}(C)$ over the set $\Lambda(C)$ of internal $C$-routers. For all $1 \leq i \leq \ell$, we will also define a bad event $\mathcal{E}_i$, and we will ensure that it happens with probability at most $i/m^{10}$. We will ensure that the following properties hold for all $1 \leq i \leq \ell$:

R1. each cluster $C \in \mathcal{C}_i$ has the $\alpha_i$-bandwidth property;

R2. the number of edges in the contracted graph $G_{|\mathcal{C}_i}$ is at most $m/\eta^{i-1}$;

R3. each cluster $C \in \mathcal{C}_i^{\text{light}}$ is $\beta_i$-light with respect to the distribution $\mathcal{D}(C)$; and

R4. if the bad event $\mathcal{E}_i$ does not happen, then each cluster $C \in \mathcal{C}_i^{\text{bad}}$ is $\eta_i$-bad.

The input to the first phase, $\mathcal{C}_1 = \emptyset$. Clearly, all properties R1–R4 hold for this set of clusters. We now describe the execution of the $i$th phase. We assume that we are given as input a collection $\mathcal{C}_i$ of disjoint clusters of $G \setminus T$, which is partitioned into two subsets, $\mathcal{C}_i^{\text{light}}$ and $\mathcal{C}_i^{\text{bad}}$. We are also given, for each cluster $C \in \mathcal{C}_i^{\text{light}}$, a distribution $\mathcal{D}(C)$ over the set $\Lambda(C)$ of internal $C$-routers, and we are guaranteed that Properties R1–R4 hold.

We consider the contracted graph $G' = G_{|\mathcal{C}_i}$. The execution of the $i$th phase consists of two steps: in the first step, we apply the algorithm AlgInitPartition to the contracted graph $G'$, obtaining a collection $\mathcal{C}$ of clusters of $G'$, which we then convert into clusters of $G$. The set $\mathcal{C}$ of clusters is partitioned into three subsets. Informally, the clusters in the first subset are concise, the clusters in the second subset are $\eta_{i+1}$-bad if event $\mathcal{E}_i$ did not happen, and the clusters in the third set are $\beta_{i+1}$-light with respect to a distribution over the internal routers that we construct. In the second step we further process each concise cluster, using the algorithm AlgFindGuiding, in order to determine whether it is a $\beta_{i+1}$-light or an $\eta_{i+1}$-bad cluster, and in the former case, to compute the corresponding distribution $\mathcal{D}(C)$ over the set $\Lambda(C)$ of internal $C$-routers.

### 6.3 Step 1: Partition

We assume first that $|E(G')| > 16\eta k \log m$, where $\eta = 2^{(\log m)^{3/4}}$ is the parameter that we have defined above. If the inequality does not hold, then the current phase is the last phase of the algorithm, and we show how to execute this phase at the end of this subsection.

We apply the algorithm AlgInitPartition from Theorem 6.3 to graph $G'$, the set $T$ of terminals, and the parameter $\eta$ that we have defined. Notice that, since $\eta = 2^{(\log m)^{3/4}}$, and since we have assumed that $m$ is greater than some large enough constant, $\eta > \log m \geq \log(|E(G')|)$ must hold. We now consider the output of the algorithm, that consists of a collection $\mathcal{C}$ of disjoint clusters of $G' \setminus T$, a

partition $(\mathcal{C}'_1,\mathcal{C}'_2,\mathcal{C}'_3)$ of $\mathcal{C}$ into three subsets, and, for every cluster $C\in\mathcal{C}'_3$, a vertex $u(C)\in V(C)$, and an internal $C$-router $\mathcal{Q}(C)$, consisting of edge-disjoint paths routing the edges of $\delta_G(C)$ to $u(C)$. Recall that we are also guaranteed that every cluster $C\in\mathcal{C}$ has the $\alpha'$-bandwidth property, where $\alpha'=\frac{1}{16\beta_{\mathrm{ARV}}(m)\cdot\log m}\geq\alpha_0$.

Consider any cluster $C\in\mathcal{C}$. Recall that $C$ is a cluster of the contracted graph $G'$, and it has the $\alpha_0$-bandwidth property. Let $\mathcal{W}(C)$ be the set of all clusters $W\in\mathcal{C}_i$, whose corresponding supernode $v_W\in V(C)$. Recall that every cluster of $\mathcal{C}_i$ has the $\alpha_i$-bandwidth property from Property R1. Let $U_C$ be the set of vertices of $G$, that contains every regular (non-supernode) vertex of $C$, and every vertex lying in clusters of $\mathcal{W}(C)$. In other words, $U_C=(V(G)\cap V(C))\cup\left(\bigcup_{W\in\mathcal{W}(C)}V(W)\right)$. We then let $\tilde{C}=G[U_C]$. Since cluster $C$ has the $\alpha_0$-bandwidth property, and every cluster in $\mathcal{W}(C)$ has the $\alpha_i$-bandwidth property, from Claim 4.39 and Observation 4.16, cluster $\tilde{C}$ has the $\alpha_i\cdot\alpha_0=\alpha_{i+1}$-bandwidth property. We let $\mathcal{C}_{i+1}=\left\{\tilde{C}\mid C\in\mathcal{C}\right\}$. Notice that we have just established Property R1 for clusters in $\mathcal{C}_{i+1}$. It is immediate to verify that $G_{|\mathcal{C}_{i+1}}=G'_{|\mathcal{C}}$. Since Theorem 6.3 guarantees that $|E(G'_{|\mathcal{C}})|\leq|E(G')|/\eta$, and, from Property R2, $|E(G')|=|E(G_{|\mathcal{C}_i})|\leq m/\eta^{i-1}$, we get that $|E(G_{|\mathcal{C}_{i+1}})|\leq|E(G')|/\eta\leq m/\eta^i$, establishing Property R2 for the set $\mathcal{C}_{i+1}$ of clusters.

We now construct the partition $(\mathcal{C}^{\mathrm{bad}}_{i+1},\mathcal{C}^{\mathrm{light}}_{i+1})$ of the set $\mathcal{C}_{i+1}$ of clusters. We start by letting $\mathcal{C}^{\mathrm{bad}}_{i+1}=\left\{\tilde{C}\mid C\in\mathcal{C}'_2\right\}$ and $\mathcal{C}^{\mathrm{light}}_{i+1}=\emptyset$. We then consider every cluster $C\in\mathcal{C}'_3$ one by one. Recall that for each such cluster $C$, the algorithm from Theorem 6.3 provides an internal router $\mathcal{Q}(C)$, routing the edges of $\delta_{G'}(C)$ to $u(C)$, such that the paths in $\mathcal{Q}(C)$ are edge-disjoint. If vertex $u(C)$ is a supernode, whose corresponding cluster $W\in\mathcal{C}_i$ lies in set $\mathcal{C}^{\mathrm{bad}}_i$, then we add $\tilde{C}$ to $\mathcal{C}^{\mathrm{bad}}_{i+1}$; otherwise, we add $\tilde{C}$ to $\mathcal{C}^{\mathrm{light}}_{i+1}$. Lastly, we set $\mathcal{C}^{\mathrm{concise}}_{i+1}=\left\{\tilde{C}\mid C\in\mathcal{C}'_1\right\}$, and we refer to clusters in $\mathcal{C}^{\mathrm{concise}}_{i+1}$ as *concise clusters*. In Step 2, we will further process clusters in $\mathcal{C}^{\mathrm{concise}}_{i+1}$, and we will eventually add each such cluster to either $\mathcal{C}^{\mathrm{bad}}_{i+1}$ or to $\mathcal{C}^{\mathrm{light}}_{i+1}$. Before we do so, we establish Property R4 for clusters that are currently in $\mathcal{C}^{\mathrm{bad}}_{i+1}$, and we define a distribution $\mathcal{D}(\tilde{C})$ over the set $\Lambda_G(\tilde{C})$ of internal $\tilde{C}$-routers for every cluster $\tilde{C}$ that is currently in $\mathcal{C}^{\mathrm{light}}_{i+1}$, such that $\tilde{C}$ is $\beta_{i+1}$-light with respect to $\mathcal{D}(\tilde{C})$ (that is, we establish Property R3 for clusters that are currently in $\mathcal{C}^{\mathrm{light}}_{i+1}$).

**Bad Clusters.** Recall that Theorem 6.3 guaranteed that, for every cluster $C\in\mathcal{C}'_2$, $\mathsf{OPT}_{\mathsf{cr}}(C)\geq\Omega(|E(C)|^2/(\eta^2\operatorname{poly}\log m))$, and $|E(C)|>\Omega(\eta^4|\delta_G(C)|\log^8 m)$. Therefore:

$$\mathsf{OPT}_{\mathsf{cr}}(C)\geq\Omega\left(\frac{|E(C)|^2}{\eta^2\operatorname{poly}\log m}\right)\geq\Omega\left(\frac{|\delta_G(C)|^2\eta^6}{\operatorname{poly}\log m}\right)\geq|\delta_G(C)|^2.$$

From the definition of cluster $\tilde{C}$, graph $C$ is a contracted graph of $\tilde{C}$ with respect to clusters in $\mathcal{W}(C)$, that is, $C=\tilde{C}_{|\mathcal{W}(C)}$. As each cluster in $\mathcal{W}(C)\subseteq\mathcal{C}_i$ has the $\alpha_i$-bandwidth property (from Property R1), from Claim 4.42, there is a drawing of $C$ containing at most $O(\mathsf{OPT}_{\mathsf{cnwrs}}(\tilde{C},\Sigma_{\tilde{C}})\cdot\log^8 m/(\alpha_i)^2)$ crossings, where $\Sigma_{\tilde{C}}$ is the rotation system for $\tilde{C}$ induced by $\Sigma$. Since we have established that $\mathsf{OPT}_{\mathsf{cr}}(C)\geq|\delta_G(C)|^2$, we get that:

$$\mathsf{OPT}_{\mathsf{cnwrs}}(\tilde{C},\Sigma_{\tilde{C}})\geq\Omega\left(\frac{|\delta_G(C)|^2\cdot(\alpha_i)^2}{\log^8 m}\right)\geq\frac{|\delta_G(C)|^2}{\eta_{i+1}},$$

since, by the definition, $\eta_{i+1}>\eta\geq 2^{(\log m)^{3/4}}$, while $\alpha_i\geq\alpha_\ell\geq 1/2^{O((\log m)^{1/4}\log\log m)}$, and since we have assume that $m$ is large enough. We conclude that every cluster in $\left\{\tilde{C}\mid C\in\mathcal{C}'_2\right\}$ is $\eta_{i+1}$-bad.

Consider now some cluster $C\in\mathcal{C}'_3$, such that vertex $u(C)$ that serves as the center of the router $\mathcal{Q}(C)$ provided by the algorithm from Theorem 6.3 is a supernode, whose corresponding cluster

$W \in \mathcal{C}_i^{\text{bad}}$. From Property R4, if Event $\mathcal{E}_i$ did not happen, cluster $W$ is an $\eta_i$-bad cluster, that is, $\mathsf{OPT}_{\mathsf{cnwrs}}(W, \Sigma_W) + |E(W)| \geq \frac{|\delta_G(W)|^2}{\eta_i}$, where $\Sigma_W$ is the rotation system for $W$ induced by $\Sigma$. Since $W \subseteq \tilde{C}$, we get that $\mathsf{OPT}_{\mathsf{cnwrs}}(\tilde{C}, \Sigma_{\tilde{C}}) + |E(\tilde{C})| \geq \frac{|\delta_G(W)|^2}{\eta_i}$. Lastly, since there is a set $\mathcal{Q}(C)$ of edge-disjoint paths routing the edges of $\delta_{G'}(C)$ to vertex $u(C)$ inside $C$, we conclude that $|\delta_G(\tilde{C})| \leq |\delta_G(W)|$. Altogether, from the fact that $\eta_{i+1} > \eta_i$, we get that $\mathsf{OPT}_{\mathsf{cnwrs}}(\tilde{C}, \Sigma_{\tilde{C}}) + |E(\tilde{C})| \geq \frac{|\delta_G(\tilde{C})|^2}{\eta_{i+1}}$.

We conclude that, if event $\mathcal{E}_i$ did not happen, then every cluster that we have added to set $\mathcal{C}_{i+1}^{\text{bad}}$ so far is an $\eta_{i+1}$-bad cluster.

**Light Clusters.** Consider now some cluster $\tilde{C} \in \mathcal{C}_{i+1}^{\text{light}}$, and let $C \in \mathcal{C}$ be its corresponding cluster in graph $G'$. Recall that the algorithm from Theorem 6.3 provides a collection $\mathcal{Q}(C)$ of edge-disjoint paths routing the edges of $\delta_{G'}(C)$ to $u(C)$, such that, for every path in $\mathcal{Q}(C)$, all inner vertices of the path lie in $C$. We will now define a distribution $\mathcal{D}(\tilde{C})$ over the set $\Lambda_G(\tilde{C})$ of internal $\tilde{C}$-routers, so that $\tilde{C}$ is $\beta_{i+1}$-light with respect to $\mathcal{D}(\tilde{C})$.

Assume first that vertex $u(C)$ is a regular vertex in cluster $C$, that is, it is not a supernode. Since every cluster $W \in \mathcal{W}(C)$ has the $\alpha_i$-bandwidth property, we can use the algorithm from Claim 4.41 to compute a collection $\mathcal{Q}(\tilde{C})$ of paths, routing the edges of $\delta_G(\tilde{C})$ to $u(C)$, such that, for every path of $\mathcal{Q}(\tilde{C})$, all its inner vertices lie in $\tilde{C}$, and the largest congestion on an edge of $\tilde{C}$ is bounded by $\lceil 1/\alpha_i \rceil$. The resulting distribution $\mathcal{D}(\tilde{C})$ then consists of a single internal $\tilde{C}$-router $\mathcal{Q}(\tilde{C})$, that is chosen with probability 1. Clearly, for every edge $e \in E(\tilde{C})$:

$$\mathbf{E}_{\mathcal{Q}(\tilde{C}) \sim \mathcal{D}(\tilde{C})} \left[ (\text{cong}_G(\mathcal{Q}(\tilde{C}), e))^2 \right] \leq \lceil 1/\alpha_i \rceil^2 \leq \beta_{i+1},$$

since by definition $\beta_{i+1} = \frac{(\log m)^{56}}{(\alpha_0)^{12} \cdot (\alpha_{i+1})^8} \cdot \beta_i$.

Assume now that $u(C)$ is a supernode, corresponding to some cluster $W^* \in \mathcal{W}(C)$, such that $W^* \in \mathcal{C}_i^{\text{light}}$. Let $\tilde{C}'$ be the cluster obtained from $\tilde{C}$, after we contract cluster $W^*$ into a supernode $v_{W^*} = u(C)$. Using the same reasoning as in the previous case, we can compute a set $\mathcal{Q}(\tilde{C}')$ of paths in graph $\tilde{C}'$, routing the edges of $\delta_G(\tilde{C})$ to $u(C)$, such that, for every path in $\mathcal{Q}(\tilde{C}')$, every inner vertex on the path lies in $\tilde{C}'$, and the largest congestion on an edge of $\tilde{C}'$ is bounded by $\lceil 1/\alpha_i \rceil$. Moreover, the algorithm from Claim 4.41 guarantees that every edge in $\delta_{\tilde{C}'}(u(C))$ belongs to at most one path in $\mathcal{Q}(\tilde{C}')$ (and it is the last edge on that path).

Recall that cluster $W^*$ is $\beta_i$-light with respect to the distribution $\mathcal{D}(W^*)$ over the set $\Lambda_G(W^*)$ of internal $W^*$-routers. We choose an internal $W^*$-router $\mathcal{Q}(W^*) \in \Lambda(W^*)$ from the distribution $\mathcal{D}(W^*)$, routing the edges of $\delta_G(W^*)$ to a vertex $u(W^*)$ of $W^*$. We now consider every path $Q \in \mathcal{Q}(\tilde{C}')$ one by one. Let $e \in \delta_G(\tilde{C})$ be the first edge on $Q$, and let $e' \in \delta_G(W^*)$ be the last edge on $Q$. Let $Q^*$ be the unique path in $\mathcal{Q}(W^*)$ whose first edge is $e'$, and let $Q'$ be obtained by first deleting the edge $e'$ from $Q$, and then concatenating the resulting path with path $Q^*$. Notice that path $Q'$ connects the edge $e$ to the vertex $u(W^*)$, in graph $\tilde{C} \cup \delta_G(\tilde{C})$. We then set $\mathcal{Q}(\tilde{C}) = \left\{ Q' \mid Q \in \mathcal{Q}(\tilde{C}') \right\}$, so that $\mathcal{Q}(\tilde{C})$ is an internal $\tilde{C}$-router in $\Lambda_G(\tilde{C})$. This finishes the definition of the distribution $\mathcal{D}(\tilde{C})$ over the set $\Lambda(\tilde{C})$ of internal $\tilde{C}$-routers. Note that the distribution is given implicitly, that is, we provide an efficient algorithm to draw a router from the distribution.

Consider now some edge $e \in E(\tilde{C})$. If $e \notin E(W^*)$, then with probability 1 (over the choices of internal $\tilde{C}$-routers $\mathcal{Q}(\tilde{C})$ from $\mathcal{D}(\tilde{C})$), $\text{cong}_G(\mathcal{Q}(\tilde{C}), e) \leq \lceil 1/\alpha_i \rceil^2$, and so $(\text{cong}_G(\mathcal{Q}(\tilde{C}), e))^2 \leq \beta_{i+1}$ as argued before. If $e \in E(W^*)$, then $\text{cong}_G(\mathcal{Q}(\tilde{C}), e) = \text{cong}_G(\mathcal{Q}(W^*), e)$, and so:

$$\mathbf{E}_{\mathcal{Q}(\tilde{C}) \sim \mathcal{D}(\tilde{C})} \left[ (\text{cong}_G(\mathcal{Q}(\tilde{C}), e)^2 \right] = \mathbf{E}_{\mathcal{Q}(W^*) \sim \mathcal{D}(W^*)} \left[ (\text{cong}_G(\mathcal{Q}(W^*), e)^2 \right] \leq \beta_i \leq \beta_{i+1}.$$

We conclude that cluster $\tilde{C}$ is $\beta_{i+1}$-light with respect to the distribution $\mathcal{D}(\tilde{C})$ over the set $\Lambda_G(\tilde{C})$ of intenral $\tilde{C}$-routers that we have computed.

Recall that so far we assumed that $|E(G')| > 16\eta k \log m$ held, where $G' = G_{|\mathcal{C}_i}$. Assume now that $|E(G')| \leq 16\eta k \log m$. In this case, we let $\mathcal{C}^{\text{bad}}_{i+1} = \mathcal{C}^{\text{light}}_{i+1} = \emptyset$ and we let $\mathcal{C}^{\text{concise}}_{i+1}$ contain a single cluster, $\tilde{C} = G \setminus T$. We also let $\mathcal{C}'_1$ contain a single cluster, $C = G' \setminus T$, and we set $\mathcal{C}'_2 = \mathcal{C}'_3 = \emptyset$. We denote $\mathcal{W}(C) = \mathcal{C}_i$. The current phase will become the final phase of the algorithm.

### 6.4 Step 2: Concise Clusters

Observe that every cluster $C \in \mathcal{C}'_1$ has the $\alpha_0$-bandwidth property in $G'$, and $|E(C)| \leq \eta_0 \cdot |\delta_{G'}(C)|$ holds. Indeed, if $|E(G')| \leq 16\eta k \log m$ holds, then set $\mathcal{C}'_1$ contains a single cluster $C = G' \setminus T$, and $|E(C)| \leq 16\eta k \log m \leq \eta_0 |\delta_{G'}(C)|$ holds. Since the set $T$ of terminals is $\alpha_0$-well-linked in $G$ (from the statement of Theorem 6.1), cluster $\tilde{C} = G \setminus T$ has the $\alpha_0$-bandwidth property in $G$, and cluster $C$ has the $\alpha_0$-bandwidth property in $G'$.

Otherwise, Theorem 6.3 guarantees that every cluster $C \in \mathcal{C}'_1$ has the $\alpha_0$-bandwidth property, and moreover, $|E(C)| \leq O(\eta^4 \log^8 m) \cdot |\delta_{G'}(C)| \leq \eta_0 \cdot |\delta_G(C)|$ (since $\eta_0 = \eta^4 \cdot \log^9 m$, and we have assumed that $m$ is large enough). Recall that for every cluster $C \in \mathcal{C}'_1$, we have defined a collection $\mathcal{W}(C) \subseteq \mathcal{C}_i$ of clusters, such that for each cluster $W \in \mathcal{W}(C)$, its corresponding supernode $v_W$ lies in $C$. We have also defined a cluster $\tilde{C} \in \mathcal{C}^{\text{concise}}_{i+1}$, that is a subgraph of $G$ correpsonding to $C$. In other words, we can think of $\tilde{C}$ as being obtained from $C$ by un-contracting every cluster $W \in \mathcal{W}(C)$.

We would now like to apply the algorithm $\mathsf{AlgFindGuiding}$ to each such cluster $C \in \mathcal{C}'_1$, in order to classify the corresponding cluster $\tilde{C}$ of $G$ as either an $\eta_{i+1}$-bad or a $\beta_{i+1}$-light cluster. Notice however that the algorithm requires that we define a rotation system for $C$, and, if the algorithm classifies $C$ as an $\eta_{i+1}$-bad cluster (by returning "FAIL"), then we are only guaranteed that it is likely that the value of the optimal solution of the resulting instance is high. Therefore, ideally we would like to define a rotation system $\hat{\Sigma}(C)$ for cluster $C$ of $G'$, such that $\mathsf{OPT}_{\mathsf{cnwrs}}(C, \hat{\Sigma}(C))$ is not much higher than $\mathsf{OPT}_{\mathsf{cnwrs}}(\tilde{C}, \Sigma_{\tilde{C}})$. Unfortunately, it is not immediately clear how to define such a rotation system, mainly because it is unclear how to define the orderings on edges incident to supernodes. In order to overcome this difficulty, we consider a different graph, that can be thought of as an intermediate graph between $C$ and $\tilde{C}$. This new graph, that we denote by $H(\tilde{C})$, is obtained as follows. We start from graph $\tilde{C}^+$. Recall that $\tilde{C}^+$ is obtained from graph $G$ by first subdividing every edge $e \in \delta_G(\tilde{C})$ with a vertex $t_e$, and then letting $\tilde{C}^+$ be the subgraph of the resulting graph induced by $V(\tilde{C}) \cup \left\{t_e \mid e \in \delta_G(\tilde{C})\right\}$. We denote $T' = \left\{t_e \mid e \in \delta_G(\tilde{C})\right\}$. We partition the set $\mathcal{W}(C)$ of clusters into two subsets: set $\mathcal{W}^{\text{light}}(C) = \mathcal{W}(C) \cap \mathcal{C}^{\text{light}}_i$ and $\mathcal{W}^{\text{bad}}(C) = \mathcal{W}(C) \cap \mathcal{C}^{\text{bad}}_i$. Graph $H(\tilde{C})$ is then obtained from graph $\tilde{C}^+$, by contracting every cluster $W \in \mathcal{W}^{\text{light}}(C)$ into a supernode $v_W$. Additionally, we denote by $H'(\tilde{C})$ the graph obtained from $\tilde{C}^+$ by contracting every cluster $W \in \mathcal{W}(C)$ into a supernode $v_W$. Notice that graph $H'(\tilde{C})$ is precisely the augmentation $C^+$ of the cluster $C$ in $G'$.

In the remainder of this step, we focus on one specific cluster $C \in \mathcal{C}'_1$, so for convenience, we will denote $H(\tilde{C})$ by $H$, $H'(\tilde{C})$ by $H'$, $\mathcal{W}(C)$ by $\mathcal{W}$, and $\mathcal{W}^{\text{light}}(C), \mathcal{W}^{\text{bad}}(C)$ by $\mathcal{W}^{\text{light}}$ and $\mathcal{W}^{\text{bad}}$, respectively. From our construction, $H' = H_{|\mathcal{W}^{\text{bad}}}$.

Recall that we have already established that cluster $C$ has the $\alpha_0$-bandwidth property in $G'$. Therefore, the set $T'$ of vertices is $\alpha_0$-well-linked in graph $H'$. Additionally, from Property R1, every cluster $W \in \mathcal{W}^{\text{bad}}$ has the $\alpha_i$-bandwidth property, and, if event $\mathcal{E}_i$ did not happen, each such cluster is $\eta_i$-bad. Recall also that we are guaranteed that $|E(C)| \leq \eta_0 \cdot |\delta_{G'}(C)| = \eta_0 \cdot |T'|$. Therefore, $|E(H')| = |E(C)| + |T'| \leq 2\eta_0 |T'|$.

Intuitively, we would now like to apply the algorithm $\mathsf{AlgFindGuiding}$ from Theorem 6.4 to graph $H$, the set $T'$ of terminals, and the correpsonding collection $\mathcal{W}^{\text{bad}}$ of clusters. In order to do so,

we need to define a rotation system $\hat{\Sigma}$ for graph $H$. We do so using a randomized algorithm that exploits the distributions $\mathcal{D}(W)$ over the set $\Lambda_G(W)$ of internal $W$-routers for clusters $W \in \mathcal{W}^{\text{light}}$. We would like to use the algorithm $\mathsf{AlgFindGuiding}$ in order to decide whether to add cluster $\tilde{C}$ to the set $\mathcal{C}^{\text{light}}_{i+1}$ of light clusters or to the set $\mathcal{C}^{\text{bad}}_{i+1}$ of bad clusters. Specifically, if the algorithm returns FAIL, we would like to add it to $\mathcal{C}^{\text{bad}}_{i+1}$, and otherwise we would like to add it to set $\mathcal{C}^{\text{light}}_{i+1}$, together with the distribution $\mathcal{D}(\tilde{C})$ over the set $\Lambda_G(\tilde{C})$ of internal $\tilde{C}$-routers that we can compute using the distribution over internal routers in $\Lambda(H, T')$ that the algorithm $\mathsf{AlgFindGuiding}$ computes. Notice however, that, even if $\mathsf{OPT}_{\mathsf{cnwrs}}(H, \hat{\Sigma})$ is small, the algorithm may return FAIL with a constant probability. Additionally, the random choices that we make in defining the rotation system $\hat{\Sigma}$ for graph $H$ may also result in an instance whose solution value is too high (though this can only happen with relatively small probability). In order to ensure that our algorithm classifies cluster $\tilde{C}$ as a light or a bad cluster correctly with high probability, we will perform $m$ identical iterations (but in each iteration we construct the rotation system $\hat{\Sigma}$ for $H$ from scratch). We now describe a single iteration.

**Execution of a single iteration.** In order to perform a single iteration of the algorithm, we construct a rotation system $\hat{\Sigma}$ for graph $H$, as follows. Consider any vertex $v \in V(H)$. If $v \in T'$, then the degree of $v$ in $H$ is 1, and the corresponding ordering of its incident edges is trivial. Assume now that $v \in V(H) \setminus T'$, and that $v$ is not a supernode. In this case, there is a one-to-one correpsondence between the edges in set $\delta_H(v)$ and the edges in set $\delta_G(v)$. We use the ordering $\mathcal{O}_v \in \Sigma$ of the edges in $\delta_G(v)$ in order to define an ordering of the edges in $\delta_H(v)$, for the rotation system $\hat{\Sigma}$. Lastly, we assume that vertex $v \in V(H) \setminus T'$ is a supernode. In other words, $v = v_W$, where $W \in \mathcal{W}^{\text{light}}$ is a light cluster. Recall that we are given a distribution $\mathcal{D}(W)$ over the set $\Lambda_G(W)$ of internal $W$-routers, such that $W$ is $\beta_i$-light with respect to this distribution. We randomly select an internal $W$-router $\mathcal{Q}(W) \in \Lambda_G(W)$ from the distribution $\mathcal{D}(W)$. Let $u(W)$ be the center vertex of $\mathcal{Q}(W)$, so that $\mathcal{Q}(W)$ is a set of paths routing the edges of $\delta_G(W)$ to vertex $u(W)$. Also recall that, from the definition of light clusters, for every edge $e \in E(W)$, $\mathbf{E}_{\mathcal{Q}(W)\sim\mathcal{D}(W)}\left[(\text{cong}_G(\mathcal{Q}(W), e))^2\right] \leq \beta_i$.

Observe that the edges of $\delta_H(v_W)$ are precisely the edges of $\delta_G(W)$. Next, we transform the set $\mathcal{Q}(W)$ of paths into a set of non-transversal paths, by applying the algorithm from Lemma 4.7 to the set $\mathcal{Q}(W)$ of paths. We denote the resulting set of paths by $\hat{\mathcal{Q}}(W)$; note that $\hat{\mathcal{Q}}(W)$ is an internal $W$-router. Recall that we have defined an ordering of edges of $\delta_G(W)$ guided by the internal $W$-router $\hat{\mathcal{Q}}(W)$ (see Section 5.2). We let the ordering $\hat{\mathcal{O}}_{v_W} \in \hat{\Sigma}$ be the ordering of the edges of $\delta_G(W)$ guided by the paths in $\hat{\mathcal{Q}}(W)$. We need the following observation, whose proof follows arguments that are similar to those used in the proof of Lemma 5.6, and is deferred to Section F.2 of Appendix.

**Observation 6.5** $\mathbf{E}\left[\mathsf{OPT}_{\mathsf{cnwrs}}(H, \hat{\Sigma})\right] \leq O\left(\beta_i \cdot \left(\mathsf{OPT}_{\mathsf{cnwrs}}(\tilde{C}, \Sigma_{\tilde{C}}) + |E(\tilde{C})|\right)\right)$.

A single iteration of our algorithm consists of computing a rotation system $\hat{\Sigma}$ for graph $H$ from scratch, and then applying the algorithm $\mathsf{AlgFindGuiding}$ from Theorem 6.4 to instance $I = (H, \hat{\Sigma})$ of $\mathsf{MCNwRS}$, with the set $T'$ of terminals, and the collection $\mathcal{C} = \mathcal{W}^{\text{bad}}$ of clusters. We set the parameters for the algorithm from Theorem 6.4 as follows: $\alpha = \alpha_0$, $\alpha' = \alpha_i$, $\eta = 2\eta_0$, and $\eta' = \eta_i$. Recall that the set $T'$ of terminals is $\alpha_0$-well-linked in the contracted graph $H' = H_{|\mathcal{W}^{\text{bad}}}$, and $|T'| \geq |E(H'(\tilde{C}))|/(2\eta_0)$. Moreover, each cluster in $\mathcal{W}^{\text{bad}}(C)$ has the $\alpha_i$-bandwidth property, and, if event $\mathcal{E}_i$ did not happen, then each such cluster is $\eta_i$-bad (note that, if $i = 1$ then $\mathcal{W}^{\text{bad}}(C) = \emptyset$). Notice that $\eta' \geq \eta_1 \geq (2\eta_0)^{13} \geq \eta^{13}$, from the definition of the parameter $\eta_i$. It remains to verify that $\eta \geq \frac{c^* \log^{46} m}{\alpha^{10}(\alpha')^2}$, or, equivalently, that $\eta_0 \geq \frac{c^* \log^{46} m}{2\alpha_0^{10}(\alpha_i)^2}$. Recall that we set $\eta_0 = \eta^4 \cdot \log^9 m = 2^{O((\log m)^{3/4})}$, and we ensure that $\alpha_i, \alpha_0 \geq \alpha_\ell = 1/(\log m)^{50\cdot\ell} = 1/2^{O((\log m)^{1/4}\log\log m)}$. Since we assume that $m$ is large enough, the inequality clearly holds. Therefore, all conditions of Theorem 6.4 hold, and we can

apply the algorithm AlgFindGuiding to instance $I = (H, \hat{\Sigma})$ of MCNwRS, with the set $T'$ of terminals, the collection $\mathcal{C} = \mathcal{W}^{\text{bad}}$ of clusters, and the parameters defined above.

Recall that we perform $m$ such iterations. If, in every iteration, algorithm AlgFindGuiding returns FAIL, then we add cluster $\tilde{C}$ to set $\mathcal{C}^{\text{bad}}_{i+1}$. We next show that, if Event $\mathcal{E}_i$ did not happen, and $\tilde{C}$ is not an $\eta_{i+1}$-bad cluster in $G$, then the probability that $\tilde{C}$ is added to $\mathcal{C}^{\text{bad}}_{i+1}$ is small.

**Claim 6.6** *Let $\mathcal{E}_{i+1}(\tilde{C})$ denote the bad event that $\tilde{C}$ is not an $\eta_{i+1}$-bad cluster, but our algorithm adds $\tilde{C}$ to $\mathcal{C}^{\text{bad}}_{i+1}$. Then* $\mathbf{Pr}\left[\mathcal{E}_{i+1}(\tilde{C}) \mid \neg\mathcal{E}_i\right] \leq (3/4)^m$.

**Proof:** Consider a single iteration of the algorithm. Recall that, from Observation 6.5,

$$\mathbf{E}\left[\mathsf{OPT}_{\mathsf{cnwrs}}(H, \hat{\Sigma})\right] \leq O\left(\beta_i^2 \cdot \left(\mathsf{OPT}_{\mathsf{cnwrs}}(\tilde{C}, \Sigma_{\tilde{C}}) + |E(\tilde{C})|\right)\right).$$

If cluster $\tilde{C}$ is not $\eta_{i+1}$-bad, then $|E(\tilde{C})| + \mathsf{OPT}_{\mathsf{cnwrs}}(\tilde{C}, \Sigma_{\tilde{C}}) < |T'|^2/\eta_{i+1}$. So for some constant $c'$:

$$\mathbf{E}\left[|E(H \setminus T')| + \mathsf{OPT}_{\mathsf{cnwrs}}(H(\tilde{C}), \hat{\Sigma})\right] \leq c'\beta_i^2 \cdot |T'|^2/\eta_{i+1}.$$

Let $\mathcal{E}'$ denote the event that $|E(H \setminus T')| + \mathsf{OPT}_{\mathsf{cnwrs}}(H, \hat{\Sigma}) > 4c'\beta_i^2 \cdot |T'|^2/\eta_{i+1}$. From Markov's bound, $\mathbf{Pr}[\mathcal{E}'] \leq 1/4$. Denote $k = |T'|$, and recall that we have set $\eta' = \eta_i$, $\alpha = \alpha_0$, and $\alpha' = \alpha_i$. Since $\eta_{i+1} \geq \eta_i \cdot \beta_i^3$, and $\beta_i = \frac{(\log m)^{56}}{(\alpha_0)^{12} \cdot (\alpha_i)^8} \cdot \beta_{i-1}$, we get that:

$$\frac{4c'\beta_i^2}{\eta_{i+1}} \leq \frac{4c'}{\beta_i \eta_i} \leq \frac{(\alpha_0)^{12} \cdot (\alpha_i)^8}{c_0 \eta_i \log^{50} m}.$$

We conclude that, if event $\mathcal{E}'$ did not happen, and $\tilde{C}$ is not an $\eta_{i+1}$-bad cluster, then $|E(H \setminus T')| + \mathsf{OPT}_{\mathsf{cnwrs}}(H, \hat{\Sigma}) \leq \frac{(k\alpha^4\alpha')^2}{c_0\eta' \log^{50} m}$. Let $\mathcal{E}''$ be the bad event that the algorithm AlgFindGuiding returned FAIL. From Theorem 6.4, if cluster $\tilde{C}$ is not $\eta_{i+1}$-bad, then $\mathbf{Pr}[\mathcal{E}'' \mid \neg\mathcal{E}' \wedge \neg\mathcal{E}_i] \leq 1/2$.

Overall, assuming that the event $\mathcal{E}_i$ did not happen and cluster $\tilde{C}$ is not $\eta_i$-bad, then the algorithm AlgFindGuiding may only return FAIL if either $\mathcal{E}'$ or $\mathcal{E}''$ happen, which, from the above discussion, happens with probability at most $(3/4)$. Overall, since we repeat the above algorithm $m$ times, the probability that in every iteration the algorithm AlgFindGuiding returns FAIL is at most $(3/4)^m$. □

Assume now that, in any one of the iterations, the algorithm AlgFindGuiding did not return FAIL, and instead returned a distribution $\mathcal{D}$ over the routers of $\Lambda(H, T')$, such that for every edge $e \in E(H)$, $\mathbf{E}_{\mathcal{Q}\sim\mathcal{D}}\left[(\text{cong}(\mathcal{Q}, e))^2\right] \leq O\left(\frac{\log^{32} m}{\alpha_0^{12}\alpha_i^8}\right)$. We now provide a distribution $\mathcal{D}(\tilde{C})$ over the set $\Lambda_G(\tilde{C})$ of internal $\tilde{C}$-routers, such that $\tilde{C}$ is $\beta_{i+1}$-light with respect to $\mathcal{D}(\tilde{C})$. The distribution is provided implicitly: that is, we provide an efficient algorithm for drawing an internal $\tilde{C}$-router from the distribution.

In order to draw an internal $\tilde{C}$-router from distribution $\mathcal{D}(\tilde{C})$, we start by choosing a router $\mathcal{Q} \in \Lambda(H, T')$ from the distribution $\mathcal{D}$. Let $x$ be the center vertex of $\mathcal{Q}$, so $x$ is a vertex of $V(H \setminus T')$, and $\mathcal{Q}$ is a collection of paths in $H$, routing all terminals in $T'$ to $x$. Equivalently, we can view $\mathcal{Q}$ as a collection of paths that route the edges of $\delta_G(\tilde{C})$ to the vertex $x$, in the contracted graph $\tilde{C}_{|\mathcal{W}^{\text{light}}(C)} \cup \delta_G(\tilde{C})$. Additionally, for every cluster $W \in \mathcal{W}^{\text{light}}(C)$, we select an internal $W$-router $\mathcal{Q}(W) \in \Lambda_G(W)$ from the distribution $\mathcal{D}(W)$, and we denote by $u(W) \in V(W)$ its center vertex.

Assume first that $x$ is a regular vertex in graph $H$, that is, it is not a supernode representing a cluster of $\mathcal{W}^{\text{light}}(C)$. In this case, we set $u(\tilde{C}) = x$, and we will use $u(\tilde{C})$ as the center vertex for internal router $\mathcal{Q}(\tilde{C}) \in \Lambda_G(\tilde{C})$ that we construct. Otherwise, if $x = v_W$ for some cluster $W \in \mathcal{W}^{\text{light}}(C)$, then we set $u(\tilde{C}) = u(W)$, where $u(W)$ is the center vertex of the internal router $\mathcal{Q}(W) \in \Lambda_G(W)$ that we have selected for cluster $W$.

Next, we consider every path $Q \in \mathcal{Q}$ one by one. Let $Q$ be any such path, and assume that the first edge on $Q$ is $e \in \delta_G(\tilde{C})$. We transform $Q$ into a path $Q'$ connecting $e$ to $u(\tilde{C})$ in $G$, as follows. We consider supernodes $v_{W'}$ that lie on $Q$ one by one. For any such supernode $v_{W'}$ that is an inner vertex of $Q$, we let $e', e''$ be the two edges that appear immediately before and immediately after $v_{W'}$ on $Q$. Observe that $e', e'' \in \delta_G(W')$. Therefore, there is a path $P(e') \in \mathcal{Q}(W')$ connecting $e'$ to $u(W')$, whose inner vertices lie in $W'$, and a path $P(e'') \in \mathcal{Q}(W')$ connecting $e''$ to $u(W')$, whose inner vertices lie in $W'$. By concatenating these two paths, we obtain a path $P^*(Q, W')$, whose first edge is $e'$, last edge is $e''$, and all remaining edges lie in $W'$. We then replace the segment of path $Q$ consisting of the edges $e', e''$ with the path $P^*(Q, W')$. Lastly, if $v_W$ is the last vertex on path $Q$ (in which case $x = v_W$), then we let $e'$ be the last edge on $Q$. Notice that $e' \in \delta_G(W)$ must hold. Then there must be a path $P(e') \in \mathcal{Q}(W)$, whose first edge is $e'$ and last vertex is $u(W) = u(\tilde{C})$. We then replace the edge $e'$ on path $Q$ with the path $P(e')$. Let $Q'$ be the final path that is otbained from $Q$ after this transformation. Then $Q'$ is a path in graph $G$, whose first edge is $e$, last vertex is $u(\tilde{C})$, and all inner edges and vertices are contained in $\tilde{C}$.

Lastly, we let $\mathcal{Q}(\tilde{C}) = \{Q' \mid Q \in \mathcal{Q}\}$ be the resulting router in $\Lambda_G(\tilde{C})$. This finishes the definition of the distribution $\mathcal{D}(\tilde{C})$ over the set $\Lambda(\tilde{C})$ of internal $\tilde{C}$-routers. Notice that we do not provide the distribution explicitly, and instead we have described an algorithm that, given access to distribution $\mathcal{D}$ computed by the algorithm AlgFindGuiding, and distributions $\{\mathcal{D}(W)\}$ for clusters $W \in \mathcal{W}^{\text{light}}(C)$ (that may also be given implicitly), samples an internal $\tilde{C}$-router from the distribution $\mathcal{D}(\tilde{C})$. We add cluster $\tilde{C}$ to set $\mathcal{C}^{\text{light}}_{i+1}$, together with the distribution $\mathcal{D}(\tilde{C})$. It now remains to show that cluster $\tilde{C}$ is $\beta_{i+1}$-light with respect to the distribution $\mathcal{D}(\tilde{C})$, which we do in the following claim.

**Claim 6.7** *Cluster $\tilde{C}$ is $\beta_{i+1}$-light with respect to the distribution $\mathcal{D}(\tilde{C})$.*

**Proof:** Consider some edge $e \in E(\tilde{C})$. Assume first that edge $e$ does not lie in any cluster $W \in \mathcal{W}^{\text{light}}(C)$. In this case:

$$\mathbf{E}_{\mathcal{Q}(\tilde{C})\sim\mathcal{D}(\tilde{C})}\left[(\text{cong}_G(\mathcal{Q}(\tilde{C}), e))^2\right] = \mathbf{E}_{\mathcal{Q}\sim\mathcal{D}}\left[(\text{cong}_H(\mathcal{Q}, e))^2\right] \leq O\left(\frac{\log^{32} m}{\alpha_0^{12}(\alpha_i)^8}\right),$$

from Theorem 6.4. Since $\beta_{i+1} = \frac{(\log m)^{56}}{(\alpha_0)^{12}\cdot(\alpha_{i+1})^8} \cdot \beta_i$, and $\alpha_{i+1} \leq \alpha_i$, we get that:

$$\mathbf{E}_{\mathcal{Q}(\tilde{C})\sim\mathcal{D}(\tilde{C})}\left[(\text{cong}_G(\mathcal{Q}(\tilde{C}), e))^2\right] \leq \beta_{i+1}.$$

Next, we assume that $e$ lies in some cluster $W \in \mathcal{W}^{\text{light}}(C)$. In order to analyze $\mathbf{E}\left[(\text{cong}_G(\mathcal{Q}(\tilde{C}), e))^2\right]$, we consider the following two-step process. In the first step, we select an internal $W$-router $\mathcal{Q}(W) \in \Lambda(W)$ from the distribution $\mathcal{D}(W)$, and denote its center vertex by $u(W)$. Then, in the second step, we select a router $\mathcal{Q} \in \Lambda(H, T')$ from distribution $\mathcal{D}$. Lastly, composing the paths in $\mathcal{Q}$ with the paths in $\mathcal{Q}(W)$, similarly to our construction of the final set $\mathcal{Q}(\tilde{C})$ of paths, will establish the final congestion on edge $e$.

Let $\mathcal{Q}(W) \in \Lambda(W)$ be the internal $W$-router that was chosen from distribution $\mathcal{D}(W)$, and assume that the paths in $\mathcal{Q}(W)$ cause congestion $z$ on edge $e$. We denote $\mathcal{Q}(W) = \{Q(e') \mid e' \in \delta_G(W)\}$, where for every edge $e' \in \delta_G(W)$, path $Q(e')$ originates at edge $e'$ and terminates at vertex $u(W)$. Let $E' \subseteq \delta_G(W)$ be the set of edges $e'$ whose corresponding path $Q(e')$ contains the edge $e$, so $|E'| = z$.

Denoting $E'=\{e_1,\ldots,e_z\}$, and assuming that the set $\mathcal{Q}(W)$ of paths is fixed, we can now write:

$$\begin{aligned}\mathbf{E}_{\mathcal{Q}\sim\mathcal{D}}\left[(\mathrm{cong}_G(\mathcal{Q}(\tilde{C}),e))^2\right]&=\mathbf{E}_{\mathcal{Q}\sim\mathcal{D}}\left[\left(\sum_{i=1}^{z}\mathrm{cong}_H(\mathcal{Q},e_i)\right)^2\right]\\&\leq\mathbf{E}_{\mathcal{Q}\sim\mathcal{D}}\left[\sum_{i=1}^{z}2z\cdot(\mathrm{cong}_H(\mathcal{Q},e_i))^2\right]\\&\leq z^2\cdot O\left(\frac{\log^{32}m}{\alpha_0^{12}\alpha_i^8}\right).\end{aligned}$$

Recall that $z$ is the congestion caused by the set $\mathcal{Q}(W)$ of paths on edge $e$. Therefore, overall:

$$\begin{aligned}\mathbf{E}_{\mathcal{Q}(\tilde{C})\sim\mathcal{D}(\tilde{C})}\left[(\mathrm{cong}_G(\mathcal{Q}(\tilde{C}),e))^2\right]&\leq\mathbf{E}_{\mathcal{Q}(W)\sim\mathcal{D}(W)}\left[(\mathrm{cong}_G(\mathcal{Q}(W),e))^2\right]\cdot O\left(\frac{\log^{32}m}{\alpha_0^{12}\alpha_i^8}\right)\\&\leq\beta_i\cdot O\left(\frac{\log^{32}m}{\alpha_0^{12}\alpha_i^8}\right)\leq\beta_{i+1}.\end{aligned}$$

(here we have used the fact that cluster $W$ is $\beta_i$-light with respect to distribution $\mathcal{D}(W)$, that $\beta_{i+1}=\frac{(\log m)^{56}}{(\alpha_0)^{12}\cdot(\alpha_{i+1})^8}\cdot\beta_i$), and that $\alpha_{i+1}\leq\alpha_i$. □

We now summarize the second step of the algorithm. First, from Claim 6.7, every cluster $\tilde{C}$ that we have added to set $\mathcal{C}_{i+1}^{\mathrm{light}}$ over the course of Step 2 is a $\beta_{i+1}$-light cluster with respect to the distribution $\mathcal{D}(\tilde{C})$ over the set $\Lambda_G(\tilde{C})$ of internal $\tilde{C}$-routers that we have defined. Let $\mathcal{E}_{i+1}$ be the bad event that any cluster $\tilde{C}$ that was added to set $\mathcal{C}_{i+1}^{\mathrm{bad}}$ over the course of the current phase is not $\eta_{i+1}$-bad. From the discussion above, event $\mathcal{E}_{i+1}$ may only happen if either event $\mathcal{E}_i$ happened, or there is some cluster $C\in\mathcal{C}_1'$, for which event $\mathcal{E}_{i+1}(\tilde{C})$ happened. From Claim 6.6, and Property R4, the probability of $\mathcal{E}_{i+1}$ is bounded by $\mathbf{Pr}\left[\mathcal{E}_i\right]+m\cdot(3/4)^m\leq i/m^{10}+m\cdot(3/4)^m\leq(i+1)/m^{10}$, since we have assumed that $m$ is large enough.

Lastly, if the current phase is the last phase, that is, $|E(G')|\leq 16\eta k\log m$ holds, then set $\mathcal{C}_1'$ contains a single cluster $C=G'\setminus T$. If our algorithm added the corresponding cluster $\tilde{C}=G\setminus T$ to set $\mathcal{C}_{i+1}^{\mathrm{bad}}$, then we return FAIL. Notice that, if $\mathsf{OPT}_{\mathsf{cnwrs}}(G,\Sigma)+|E(G)\setminus T|<k^2/\eta^*\leq k^2/\eta_\ell\leq k^2/\eta_{i+1}$, then the algorithm may only return FAIL if event $\mathcal{E}_{i+1}$ happened, with may only happen with probability at most $\frac{i+1}{m^{10}}<\frac{1}{2}$. Otherwise, our algorithm added cluster $\tilde{C}$ to set $\mathcal{C}_{i+1}^{\mathrm{light}}$, and constructed a distribution $\mathcal{D}(\tilde{C})$ over the set $\Lambda(\tilde{C})$ of internal $\tilde{C}$-routers, such that cluster $\tilde{C}$ is $\beta_{i+1}$-light with respect to $\mathcal{D}(\tilde{C})$. Notice that $\mathcal{D}(\tilde{C})$ can also be viewed as a distribution over the routers of $\Lambda(G,T)$, and we are guaranteed that, for every edge $e\in E(G)$, $\mathbf{E}_{\mathcal{Q}\sim\mathcal{D}(\tilde{C})}\left[(\mathrm{cong}_G(\mathcal{Q},e))^2\right]\leq\beta_\ell\leq\beta^*$. From Property R2, after $(\ell-1)$ phases the algorithm terminates, for $\ell=O\left(\frac{\log m}{\log\eta}\right)$.

# 7 Third Main Tool - Advanced Disengagement

The goal of this section is to prove the following theorem that allows us to perform disengagement in a more general setting than that from basic disengagement.

**Theorem 7.1** *There is an efficient randomized algorithm, called* AlgAdvancedDisengagement*, whose input consists of an instance $I=(G,\Sigma)$ of* MCNwRS*, parameters $m$ and $\mu\geq 2^{c^*(\log m)^{7/8}\log\log m}$ for some large enough constant $c^*$, and a collection $\mathcal{C}$ of disjoint clusters of $G$, for which the following hold:*

- *$|V(G)|,|E(G)|\leq m$, and $m$ is greater than a sufficiently large constant;*

- *every cluster $C \in \mathcal{C}$ has the $\alpha_0$-bandwidth property, for $\alpha_0 = 1/\log^3 m$;*
- $\bigcup_{C \in \mathcal{C}} V(C) = V(G)$*; and*
- $\sum_{C \in \mathcal{C}} |\delta_G(C)| \leq |E(G)|/\mu^{0.1}$.

*The algorithm computes a $2^{O((\log m)^{3/4} \log\log m)}$-decomposition $\mathcal{I}$ of instance $I$, such that every instance $I' \in \mathcal{I}$ is a subinstance of $I$. Moreover, for each resulting instance $I' = (G', \Sigma') \in \mathcal{I}$, there is at most one cluster $C \in \mathcal{C}$ with $E(C) \subseteq E(G')$. If such a cluster exists, then $E(G') \subseteq E(C) \cup E^{\mathsf{out}}(\mathcal{C})$, and otherwise $E(G') \subseteq E^{\mathsf{out}}(\mathcal{C})$.*

The remainder of this section is dedicated to the proof of Theorem 7.1.

Over the course of the proof, we will consider subinstances of the input instance $I$. Recall that an instance $I' = (G', \Sigma')$ of MCNwRS is a *subinstance* of instance $I = (G, \Sigma)$ (see Definition 2.12), if there is a subgraph $\tilde{G} \subseteq G$, and a collection $\mathcal{R}$ of mutually disjoint subsets of vertices of $\tilde{G}$, such that graph $G'$ can be obtained from $\tilde{G}$ by contracting, for all $R \in \mathcal{R}$, vertex set $R$ into a supernode $v_R$; we keep parallel edges but remove self-loops. We do not distiguish between edges of $G'$ incident to supernodes and their corresponding edges in the original graph $G$. We call the non-supernode vertices of $G'$ *regular vertices*. We also require that, for every regular vertex $v \in V(G') \cap V(G)$, its rotation $\mathcal{O}'_v$ in $\Sigma'$ is the same as the rotation $\mathcal{O}_v \in \Sigma$. For each supernode $v_R$, its rotation $\mathcal{O}'_{v_R}$ can be defined arbitrarily. We will consider special types of subinstances of a given instance, that we call *canonical* subinstances.

**Definition 7.2 (Canonical Subinstances)** *Let $I' = (G', \Sigma')$ be an instance of* MCNwRS*, and let $\mathcal{C}'$ be a collection of disjoint clusters $G'$. We say that instance $I'' = (G'', \Sigma'')$ is a* canonical *subinstance of $I'$ with respect to $\mathcal{C}'$ if $I''$ is a subinstance of $I'$, and moreover, if $\tilde{G} \subseteq G'$, and $\mathcal{R}$ is a collection of disjoint subsets of vertices of $\tilde{G}$, such that $G''$ is obtained from $\tilde{G}$ by contracting every vertex set $R \in \mathcal{R}$ into a supernode $v_R$, then the following holds: For every cluster $C \in \mathcal{C}'$, either (i) there is some vertex set $R \in \mathcal{R}$ with $V(C) \subseteq R$ (in which case we say that $C$ is* contracted *in graph $G''$); or (ii) $C \subseteq \tilde{G}$, and for every vertex set $R \in \mathcal{R}$, $R \cap V(C) = \emptyset$ (in which case we say that $C$ is* not contracted *in $G''$); or (iii) $V(C) \cap V(\tilde{G}) = \emptyset$.*

We will consider canonical subinstances of instance $I$ with additional useful properties. We call such subinstances *nice subinstances*. For each such subinstance, we will use a specific *witness structure* to certify that it is indeed a nice subinstance. We will provide two theorems: the first theorem will be used to decompose the input instance $I$ into a collection of nice subinstances, and the second theorem will further decompose each resulting nice subinstance into a collection of subinstances that have the properties required by Theorem 7.1. In the next subsection, we define the witness structure and the nice subinstances, and then provide the statements of the two theorems that will allow us to complete the proof of Theorem 7.1.

## 7.1 Nice Witness Structure, Nice Subinstances, and Statements of Main Theorems

Let $G'$ be a graph, let $m > 0$ be an integer with $|E(G')| \leq m$, and let $\mathcal{C}'$ be a collection of disjoint clusters of $G'$. A *nice witness structure* for $G'$ with respect to $\mathcal{C}'$ consists of the following three main ingredients (see Figure 9):

1. The first ingredient is a sequence $\tilde{\mathcal{S}} = \left\{\tilde{S}_1, \ldots, \tilde{S}_r\right\}$ of disjoint vertex-induced subgraphs of $G'$, such that $\bigcup_{i=1}^{r} V(\tilde{S}_i) = V(G')$, and, for all $1 \leq i \leq r$, a cluster $\tilde{S}'_i \subseteq \tilde{S}_i$ that has the

$\alpha^* = \Omega(1/\log^{12} m)$-bandwidth property in $G'$. We require that, for all $1 \leq i \leq r$, there is at most one cluster $C \in \mathcal{C}'$ with $C \subseteq \tilde{S}'_i$. Moreover, if such cluster $C$ exists then $E(\tilde{S}_i) \subseteq E(C) \cup E(G'_{|\mathcal{C}'})$ must hold, and otherwise $E(\tilde{S}_i) \subseteq E(G'_{|\mathcal{C}'})$. We also require that for each cluster $C \in \mathcal{C}'$, there is an index $1 \leq i \leq r$, such that $C \subseteq \tilde{S}'_i$. We refer to the sequence $\tilde{\mathcal{S}} = \left\{\tilde{S}_1, \ldots, \tilde{S}_r\right\}$ as *the backbone* of the nice witness structure, and to the clusters in $\tilde{\mathcal{S}}' = \left\{\tilde{S}'_1, \ldots, \tilde{S}'_r\right\}$ as its *verterbrae*.

2. The second ingredient is a partition of the edges of $E(G')$ into two disjoint subsets, $\tilde{E}'$ and $\tilde{E}''$. Set $\tilde{E}'$ contains all edges of $\bigcup_{i=1}^{r} E(\tilde{S}'_i)$, and, additionally, for all $1 \leq i < r$, it contains every edge $e = (u, v)$ with $u \in \tilde{S}'_i$, $v \in \tilde{S}'_{i+1}$. Set $\tilde{E}''$ contains all remaining edges of $E(G')$. Additionally, we let $\hat{E} \subseteq \tilde{E}''$ be the set of all edges $(u, v) \in \tilde{E}''$, where $u$ and $v$ lie in different clusters of $\left\{\tilde{S}_1, \ldots, \tilde{S}_r\right\}$.

3. The third ingerdient is a set $\hat{\mathcal{P}} = \left\{P(e) \mid e \in \hat{E}\right\}$ of paths, that cause congestion at most $O(\log^{18} m)$ in $G'$ that we call *nice guiding paths*. For each edge $e = (v, u) \in \hat{E}$, if we assume that $v \in \tilde{S}_i$, $u \in \tilde{S}_j$, and $i < j$, then path $P(e)$ connects vertex $v$ to vertex $u$, does not contain the edge $e$, and consists of three subpaths $P^1(e)$, $P^2(e)$ and $P^3(e)$, that have the following properties:

   - There is an index $i' \leq i$, such that path $P^1(e)$ originates at vertex $v$ and terminates at some vertex $v' \in \tilde{S}'_{i'}$. Path $P^1(e)$ must be simple, and no vertex of $P^1(e) \setminus \{v'\}$ may lie in $\bigcup_{z=1}^{r} V(\tilde{S}'_z)$. Moreover, if we denote the sequence of vertices on path $P^1(e)$ by $v = v_0, v_1, \ldots, v_q = v'$, and, for all $0 \leq z \leq q$, we assume that $v_z \in \tilde{S}_{i_z}$, then $i = i_0 \geq i_1 \geq \cdots \geq i_q = i'$. In other words, the path visits the sets $\tilde{S}_a$ in the non-increasing order of index $a$, possibly skipping over some of the indices.
   - Similarly, there is an index $j' \geq j$, such that path $P^3(e)$ originates at vertex $u$ and terminates at some vertex $u' \in \tilde{S}'_{j'}$. Path $P^3(e)$ must be simple, and no vertex of $P^3(e) \setminus \{u'\}$ may lie in $\bigcup_{z=1}^{r} V(\tilde{S}'_z)$. Moreover, if we denote the sequence of vertices on path $P^3(e)$ by $u = u_0, u_1, \ldots, u_{q'} = u'$, and, for all $0 \leq z' \leq q'$, we assume that $u_{z'} \in \tilde{S}_{j_{z'}}$, then $j = j_0 \leq j_1 \leq \cdots \leq j_{q'} = j'$. In other words, the path visits the sets $\tilde{S}_a$ in the non-decreasing order of index $a$, possibly skipping over some of the indices.
   - Lastly, path $P^2(e)$ connects $v'$ to $u'$. It may only use edges of $\tilde{E}'$, and it can be partitioned into disjoint subpaths $Q_{i'}(e), Q_{i'+1}(e), \ldots, Q_{j'}(e)$, where for all $i' \leq x \leq j'$, $Q_x(e) \subseteq \tilde{S}'_x$, and $\bigcup_{i' \leq x \leq j'} V(Q_x(e)) = V(P^2(e))$.

Note that, by definition, for every edge $e \in \hat{E}$, the paths $P^1(e)$ and $P^3(e)$ only use edges of $\tilde{E}''$, while path $P^2(e)$ only uses edges of $\tilde{E}'$. If $e = (v, u) \in \hat{E}$ is an edge for which $v \in \tilde{S}'_i$ holds, then $i' = i$ and $P^1(e) = \{v\}$ must hold. Similarly, if $u \in \tilde{S}'_j$, then $j' = j$ and $P^2(e) = \{u\}$.

Clearly, the edge sets $\tilde{E}', \tilde{E}'', \hat{E}$ in the nice witness structure are completely determined by the sequences $\tilde{\mathcal{S}} = \left\{\tilde{S}_1, \ldots, \tilde{S}_r\right\}$ and $\tilde{\mathcal{S}}' = \left\{\tilde{S}'_1, \ldots, \tilde{S}'_r\right\}$ of clusters. Therefore, the nice witness structure is completely determined by $\tilde{\mathcal{S}}, \tilde{\mathcal{S}}'$, and the set $\hat{\mathcal{P}} = \left\{P(e) \mid e \in \hat{E}\right\}$ of nice guiding paths. We will use the shorthand $(\tilde{\mathcal{S}}, \tilde{\mathcal{S}}', \hat{\mathcal{P}})$ for a nice witness structure. For a path $P(e) \in \hat{\mathcal{P}}$, we sometimes refer to $P^1(e), P^3(e)$ and $P^2(e)$ as the *prefix*, the *suffix*, and the *mid-part* of path $P(e)$, respectively. This completes the definition of a nice witness structure. Next, we define nice subinstances of instance $I$.

Consider a subinsance $I' = (G', \Sigma')$ of the input instance $I$, and assume that $I'$ is a canonical subinstance of $I$ with respect to the set $\mathcal{C}$ of clusters. Recall that, from the definition of canonical subinstances, we are guaranteed that for every cluster $C \in \mathcal{C}$, either $C \subseteq G'$, or $V(C) \cap V(G') = \emptyset$.

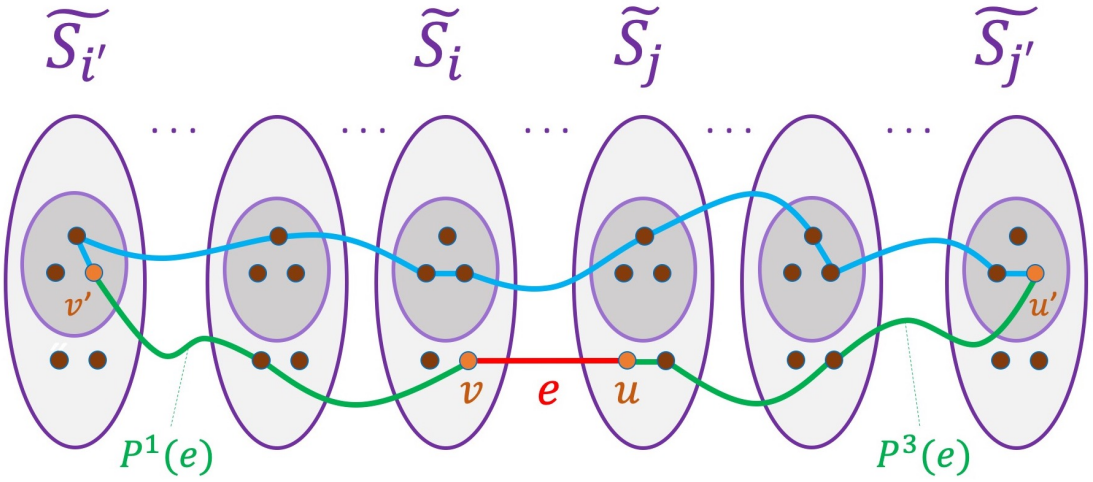



Figure 9: An illustration of a nice witness structure and a nice guiding path. An edge $e \in \hat{E}$ is shown in red. The prefix $P^1(e)$ and the suffix $P^3(e)$ of the nice guiding path $P(e)$ are shown in green, and the mid-part $P^2(e)$ is shown in blue.

We denote by $\mathcal{C}(G')$ the set of all clusters $C \in \mathcal{C}$ with $C \subseteq G'$. Lastly, we say that a subinstance $I' = (G', \Sigma')$ of $I$ is a *nice subinstance* of $I$ with respect to $\mathcal{C}$, if it is a canonical subinstance with respect to $\mathcal{C}$, and there is a nice witness structure for graph $G'$ with respect to the set $\mathcal{C}' = \mathcal{C}(G')$ of its clusters. The remainder of the proof of Theorem 7.1 uses the following two theorems. The first theorem allows us to decompose a given instance $I$ into a collection of nice subinstances.

**Theorem 7.3** *There is an efficient randomized algorithm, whose input consists of an instance $I = (G, \Sigma)$ of* MCNwRS*, parameters $m$ and $\mu \geq 2^{c^*(\log m)^{7/8} \log\log m}$ for some large enough constant $c^*$, and a collection $\mathcal{C}$ of disjoint clusters of $G$, for which the following hold:*

- *$|V(G)|, |E(G)| \leq m$, and $m$ is greater than a sufficiently large constant;*
- *every cluster $C \in \mathcal{C}$ has the $\alpha_0$-bandwidth property, for $\alpha_0 = 1/\log^3 m$;*
- *$\bigcup_{C \in \mathcal{C}} V(C) = V(G)$; and*
- *$\sum_{C \in \mathcal{C}} |\delta_G(C)| \leq |E(G)|/\mu^{0.1}$.*

*The algorithm either returns FAIL, or it computes a $2^{O((\log m)^{3/4} \log\log m)}$-decomposition $\mathcal{I}_1$ of instance $I$, such that each resulting instance $I' = (G', \Sigma') \in \mathcal{I}_1$ is a nice subinstance of $I$ with respect to $\mathcal{C}$. In the latter case, the algorithm also computes, for each instance $(G', \Sigma') \in \mathcal{I}_1$, a nice witness structure for graph $G'$ with respect to the set $\mathcal{C}(G')$ of clusters. The probability that the algorithm returns FAIL is at most $1/m^6$.*

The second theorem allows us to further decompose nice subinstances of instance $I$ into subinstances that have the desired properties.

**Theorem 7.4** *There is an efficient randomized algorithm, whose input consists of:*

- *an instance $I' = (G', \Sigma')$ of* MCNwRS*;*
- *a parameter $m$, such that $|E(G')| \leq m$;*

- *a collection $\mathcal{C}'$ of disjont clusters of $G'$; and*
- *a nice witness structure $(\tilde{\mathcal{S}}, \tilde{\mathcal{S}}', \hat{\mathcal{P}})$ for graph $G'$ with respect to the set $\mathcal{C}'$ of clusters.*

*The algorithm either returns FAIL, or computes a $2^{O((\log m)^{3/4}\log\log m)}$-decomposition $\mathcal{I}_2(I')$ of instance $I'$, such that each resulting instance $I''=(G'',\Sigma'')\in\mathcal{I}_2(I')$ is a subinstance of $I'$, and moreover, there is at most one cluster $C\in\mathcal{C}'$ with $E(C)\subseteq E(G'')$; if such a cluster exists then $E(G'')\subseteq E(C)\cup E(G'_{|\mathcal{C}'})$ holds, and otherwise $E(G'')\subseteq E(G'_{|\mathcal{C}'})$. The probability that the algorithm returns FAIL is $1/m^6$.*

Note that Theorem 7.1 immediately follows from Theorem 7.3 and Theorem 7.4. Indeed, we start by applying the algorithm from Theorem 7.3 to the input instance $I$ and the collection $\mathcal{C}$ of its clusters. Assume for now that the algorithm did not return FAIL. Then we obtain a collection $\mathcal{I}_1$ of nice subinstances of $I$, and, for each instance $I'=(G',\Sigma')\in\mathcal{I}_1$, a nice witness structure for $G'$ with respect to cluster set $\mathcal{C}(G')$. From the definition of a nice subinstance, for every cluster $C\in\mathcal{C}(G')$, $C\subseteq G'$, and for every cluster $C\in\mathcal{C}\setminus\mathcal{C}(G')$, $V(C)\cap V(G')=\emptyset$, so $E(G')\subseteq\left(\bigcup_{C\in\mathcal{C}(G')}E(C)\right)\cup E^{\mathsf{out}}(\mathcal{C})$. We then apply the algorithm from Theorem 7.4 to each such instance $I'=(G',\Sigma')\in\mathcal{I}_1$ and the corresponding nice witness structure. Assume for now that this algorithm did not return FAIL. Then we obtain a collection $\mathcal{I}_2(I')$ of subinstances of $I'$. We are guaranteed that, for each resulting instance $I''=(G'',\Sigma'')\in\mathcal{I}_2(I')$, there is at most one cluster $C\in\mathcal{C}(G')$ with $E(C)\subseteq E(G'')$. If such a cluster exists, then $E(G'')\subseteq E(C)\cup E(G'_{|\mathcal{C}(G')})\subseteq E(C)\cup E^{\mathsf{out}}(\mathcal{C})$ holds, and otherwise $E(G'')\subseteq E(G'_{|\mathcal{C}(G')})\subseteq E^{\mathsf{out}}(\mathcal{C}')$, since $E(G')\subseteq\left(\bigcup_{C'\in\mathcal{C}(G')}E(C')\right)\cup E^{\mathsf{out}}(\mathcal{C})$. If the algorithm from Theorem 7.3 did not return FAIL, and neither application of the algorithm from Theorem 7.4 returned FAIL, then we return the collection of instances $\mathcal{I}=\bigcup_{I'\in\mathcal{I}_1}\mathcal{I}_2(I')$. From Claim 2.11, we obtain a randomized algorithm that computes a $2^{O((\log m)^{3/4}\log\log m)}$-decomposition $\mathcal{I}$ of the input instance $I$.

It now remains to consider a case where the algorithm from Theorem 7.3 or any of the applications of the algorithm from Theorem 7.4 returned FAIL (which may only happen with probability at most $1/m^4$). In this case, we construct the collection $\mathcal{I}$ of subinstances of $I$ directly, as follows. For every cluster $C\in\mathcal{C}$, we let $\mathcal{O}(C)$ be an arbitrary circular ordering of the edges of $\delta_G(C)$. Set $\mathcal{I}$ will contain one global instance $\hat{I}=(\hat{G},\hat{\Sigma})$, and, for each cluster $C\in\mathcal{C}$, a cluster-based instance $I_C=(G_C,\Sigma_C)$. Consider first a cluster $C\in\mathcal{C}$. We let $G_C$ be the graph obtained from $G$ by contracting all vertices of $V(G)\setminus V(C)$ into a supernode $u_C$. We define the rotation system $\Sigma_C$ for graph $G_C$ as follows: for every vertex $v\in V(C)$, its rotation $\mathcal{O}_v$ in $\Sigma_C$ remains the same as that in $\Sigma$. Observe that $\delta_{G_C}(u_C)=\delta_G(C)$. The rotation $\mathcal{O}_{u_C}$ of vertex $u_C$ in $\Sigma_C$ is defined to be $\mathcal{O}(C)$. This completes the definition of the cluster-based instance $I_C=(G_C,\Sigma_C)$. We now define the global instance $\hat{I}=(\hat{G},\hat{\Sigma})$. Graph $\hat{G}$ is obtained from graph $G$ by contracting, for every cluster $C\in\mathcal{C}$, the set $V(C)$ of vertices into a supernode $u'_C$. Notice that the set of edges incident to $u'_C$ in $\hat{G}$ is precisely $\delta_G(C)$. We then define a rotation of $u'_C$ in $\hat{\Sigma}$ to be $\mathcal{O}(C)$. This completes the definition of the global instance $\hat{I}$. Consider now the resulting collection $\mathcal{I}$ of subinstances of $I$. It is immediate to verify that $\sum_{(G',\Sigma')\in\mathcal{I}}|E(G')|\leq O(|E(G)|)$. Assume now that we are given, for each instance $I'\in\mathcal{I}$, a feasible solution $\varphi(I')$. We can combine these solutions together to obtain a solution $\varphi$ to instance $I$, of cost at most $O\left(\sum_{I'\in\mathcal{I}}\mathsf{cr}(\varphi(I'))\right)$, by employing an algorithm similar to that from Lemma 5.3 (the algorithm that was used for basic disengagement). Lastly, from Theorem 2.8, it is easy to verify that $\sum_{I'\in\mathcal{I}}\mathsf{OPT}_{\mathsf{cnwrs}}(I')\leq O(m^2)$. Since the probability that the algorithm from Theorem 7.3, or any of the applications of the algorithm from Theorem 7.4 return FAIL is at most $1/m^4$, overall we have obtained a randomized algorithm that computes a $2^{O((\log m)^{3/4}\log\log m)}$-decomposition $\mathcal{I}$ of the input instance $I$ with required properties.

In order to complete the proof of Theorem 7.1, it is now enough to prove Theorem 7.3 and Theorem 7.4, which we do in Sections 7.2 and 7.3, respectively.

## 7.2 Decomposition into Nice Instances – Proof of Theorem 7.3

This subsection is dedicated to the proof of Theorem 7.3. The main idea of the proof is to carefully construct a laminar family $\mathcal{L}$ of clusters of $G$, whose depth is $2^{O((\log m)^{3/4}\log\log m)}$, and then apply Algorithm AlgBasicDisengagement from Section 5.3, to compute a $2^{O((\log m)^{3/4}\log\log m)}$-decomposition $\mathcal{I}_1$ of instance $I$ via basic disengagement, using the laminar family $\mathcal{L}$. The main challenge is to construct the laminar family $\mathcal{L}$ in such a way that each resulting instance $I'=(G',\Sigma')\in\mathcal{I}_1$ is a nice subinstance of $I$ with respect to $\mathcal{C}$, and to compute a nice witness structure for each such graph $G'$. We will construct the laminar family gradually, in the top-bottom fashion, using the notion of *legal clustering*.

In order to define the legal clustering, we consider a graph $G'$, together with a special vertex $v^*\in V(G)$. Intuitively, graph $G'$ represents some cluster $S\in\mathcal{L}$ that we have constructed already, and it is a graph that is obtained from $G$ by contracting all vertices of $G\setminus S$ into the special vertex $v^*$. We will also consider the subset $\mathcal{C}'\subseteq\mathcal{C}$ of all clusters $C\in\mathcal{C}$ with $C\subseteq S$. Intuitively, our goal is to construct a collection $\mathcal{R}$ of disjoint clusters of $G'$, each of which must be a subgraph of $S$, that will then be added to $\mathcal{L}$. Recall that, if $\mathcal{L}$ is a laminar family of clusters of graph $G$, and $\mathcal{I}_1$ is a collection of subinstances of $I$ obtained by decomposing $I$ via basic disengagement, then every cluster $S\in\mathcal{L}$ has a subinstance $I(S)=(G(S),\Sigma(S))\in\mathcal{I}_1$ associated with it. Graph $G(S)$ is obtained from graph $G$ as follows. First, we contract the vertices of $V(G)\setminus V(S)$ into a supernode $v^*$, obtaining graph $G'$. Next, for every child-cluster $R\in\mathcal{L}$ of $S$, we contract $R$ into a supernode $v_R$. Therefore, if $\mathcal{R}$ is the set of child-clusters of $S$, then $G(S)=G'_{|\mathcal{R}}$. Recall that we need to ensure that instance $I(S)$ is a nice instance. Given a graph $G'$, a special vertex $v^*$ in $G'$, and a collection $\mathcal{C}'$ of disjoint basic clusters of $G'$, the notion of legal clustering of $G'$ with respect to $v^*$ and $\mathcal{C}'$ is designed to ensure that every instance in our final decomposition $\mathcal{I}_1$ of $I$, created via the process described above, is a nice instance.

Consider a graph $G'$ with a special vertex $v^*\in G'$. We will consider clusters $R\subseteq G'\setminus\{u^*\}$. Recall we have defined a collection $\Lambda'_{G'}(R)$ of external routers for $R$, where each router $\mathcal{Q}'(R)\in\Lambda'_{G'}(R)$ is a collection of paths routing all edges of $\delta_{G'}(R)$ to a single vertex of $G'\setminus R$, such that all paths in $\mathcal{Q}'(R)$ are internally disjoint from $R$. We start by defining the notion of *helpful clustering*, which will be used in the definition of legal clustering. We fix two parameters that will be used throughout this section: $\alpha_1=(\alpha_0)^2=1/\log^6 m$ and $\beta=\log^{18} m$.

**Definition 7.5 (Helpful Clustering)** *Let $G'$ be a graph with a special vertex $v^*\in V(G')$, and let $\mathcal{C}'$ be a collection of disjoint vertex-induced subgraphs of $G'\setminus\{v^*\}$, that we call* basic clusters. *Let $\mathcal{R}$ be another collection of disjoint clusters of $G'$, and assume that for every cluster $R\in\mathcal{R}$, we are given a distribution $\mathcal{D}'(R)$ over the external routers in $\Lambda'_{G'}(R)$. We say that $(\mathcal{R},\{\mathcal{D}'(R)\}_{R\in\mathcal{R}})$ is a* helpful clustering *of $G'$ with respect to $v^*$ and $\mathcal{C}'$, iff the following conditions hold:*

- *vertex $v^*$ does not belong to any of the clusters in $\mathcal{R}$;*
- *for every basic cluster $C'\in\mathcal{C}$, and for every cluster $R\in\mathcal{R}$, either $C'\subseteq R$, or $V(C')\cap V(R)=\emptyset$;*
- *every cluster $R\in\mathcal{R}$ has the $\alpha_1$-bandwidth property in $G'$; and*
- *for every cluster $R\in\mathcal{R}$, for every edge $e\in E(G')\setminus E(R)$, $\mathbf{E}_{\mathcal{Q}'(R)\sim\mathcal{D}'(R)}\left[\mathrm{cong}_{G'}(\mathcal{Q}'(R),e)\right]\leq\beta$.*

Consider again a graph $G'$ with a special vertex $v^*\in V(G')$, and some cluster $R\subseteq G'$. We say that an external $R$-router $\mathcal{Q}'(R)\in\Lambda'_{G'}(R)$ is *careful* with respect to the special vertex $v^*$, if each edge of $\delta_{G'}(v^*)$ belongs to at most one path in $\mathcal{Q}'(R)$ (note that in general paths in $\mathcal{Q}'(R)$ may cause an arbitrarily large congestion in $G'$). We denote by $\Lambda''_{G'}(R)\subseteq\Lambda'_{G'}(R)$ the collection of all external $R$-routers $\mathcal{Q}'(R)$ that are careful with respect to $v^*$. We say that a distribution $\mathcal{D}'(R)$ over the collection

$\Lambda'_{G'}(R)$ of external $R$-routers is *careful* with respect to $v^*$, if every router $\mathcal{Q}'(R) \in \Lambda'_{G'}(R)$ to which $\mathcal{D}'(R)$ assigns a non-zero probability lies in $\Lambda''_{G'}(R)$.

We will consider two different types of legal clustering. We start by defining the first, and the simpler type of legal clusterings.

**Definition 7.6 (Type-1 Legal Clustering)** *Let $G'$ be a graph with a special vertex $v^* \in V(G')$, and let $\mathcal{C}'$ be a collection of disjoint vertex-induced subgraphs of $G' \setminus \{v^*\}$, that we call basic clusters. Let $\mathcal{R}$ be another collection of disjoint clusters of $G'$, and assume that for every cluster $R \in \mathcal{R}$, we are given a distribution $\mathcal{D}'(R)$ over the external routers in $\Lambda'_{G'}(R)$. We say that $(\mathcal{R}, \{\mathcal{D}'(R)\}_{R\in\mathcal{R}})$ is a* type-1 legal clustering *of $G'$ with respect to $v^*$ and $\mathcal{C}'$, if the following conditions hold:*

- *$(\mathcal{R}, \{\mathcal{D}'(R)\}_{R\in\mathcal{R}})$ is a helpful clustering of $G'$ with respect to $v^*$ and $\mathcal{C}'$;*
- *there is at most one cluster $C \in \mathcal{C}'$, that is contained in $G' \setminus \left(\bigcup_{R\in\mathcal{R}} R\right)$; and*
- *for every cluster $R \in \mathcal{R}$, distribution $\mathcal{D}'(R)$ over external routers is careful with respect to $v^*$.*

While type-1 legal clustering would be ideal in order to construct the laminar family $\mathcal{L}$ and to perform a basic disengagement of instance $I$ via $\mathcal{L}$, we may not always succeed in computing a type-1 legal clustering of a given graph $G'$, and we may need to employ type-2 legal clustering, that is defined below, instead. Before we define the type-2 legal clustering formally, we provide some intuition. Type-2 legal clustering is defined somewhat similarly to type-1 legal clustering, except that we no longer require that, for every cluster $R \in \mathcal{R}$, the distribution $\mathcal{D}'(R)$ is careful with respect to $v^*$. We also no longer require that at most one cluster of $\mathcal{C}'$ is contained in $G' \setminus \bigcup_{R\in\mathcal{R}} R$. However, we require that, additionally, the decomposition provides a nice witness structure for the graph $G'_{|\mathcal{R}}$ with respect to the set $\mathcal{C}'(G'_{|\mathcal{R}})$ of clusters (all clusers of $\mathcal{C}'$ that are contained in graph $G'_{|\mathcal{R}}$). Unfortunately, the relaxation of the requirement that the distributions $\mathcal{D}'(R)$ for clusters $R \in \mathcal{R}$ is careful with respect to $v^*$ creates some major difficulties. For intuition, recall that we will construct the laminar family $\mathcal{L}$ of clusters of $G$ gradually, in the top-bottom fashion. Assume that $S$ is some cluster of the current laminar family $\mathcal{L}$, such that no cluster of $\mathcal{L}$ is strictly contained in $S$. Let $G'$ be the graph obtained from $G$ by contracting all vertices of $V(G) \setminus V(S)$ into the special vertex $v^*$, and let $\mathcal{C}'$ be the set of all clusters of $\mathcal{C}$ that are contained in $S$. The idea of our algorithm is to compute a type-1 or a type-2 legal clustering $\mathcal{R}$ in graph $G'$; assume that we compute a type-2 legal clustering. We then add the clusters of $\mathcal{R}$ to the laminar family $\mathcal{L}$, and continue to the next iteration. From the discussion so far, for each such cluster $R \in \mathcal{R}$, the type-2 legal clustering provides a distribution $\mathcal{D}'(R)$ over the collection $\Lambda'_{G'}(R)$ of external routers in graph $G'$. However, in order to execute the basic disengagement via the laminar family $\mathcal{L}$ (see Section 5.3), we need the distribution $\mathcal{D}'(R)$ to be supported over the collection $\Lambda'_G(R)$ of external routers in graph $G$. In other words, the problem is that paths in sets $\mathcal{Q}'(R) \in \Lambda'_{G'}(R)$ that are assigned non-zero probability by $\mathcal{D}'(R)$ may contain the special vertex $v^*$, which is not a vertex of $G$. Recall however that special vertex $v^*$ represents the cluster $G \setminus S$, and so edges incident to $v^*$ in $G'$ are precisely the edges of $\delta_G(S)$. Therefore, we could exploit the distribution $\mathcal{D}'(S)$ over the external routers for cluster $S$ in $G$, in order to get rid of the special vertex $v^*$ on the paths of $\mathcal{Q}'(R)$, where $\mathcal{Q}'(R) \in \Lambda'_{G'}(R)$. In other words, by composing the distributions $\mathcal{D}'(R)$ and $\mathcal{D}'(S)$, we could obtain the desired distribution over the set $\Lambda'_G(R)$ of external routers for cluster $R$ in the original graph $G$. Unfortunately, this kind of recursive composition of distributions may lead to an explosion in the congestion of the resulting sets of paths. Even if the depth of the laminar family $\mathcal{L}$ is quite modest (say $O(\log m)$), we may obtain distributions $\mathcal{D}''(R)$ over the routers in $\Lambda'_G(R)$, for which the maximum expected congestion on an edge of $G$ may be as large as $|\delta_G(R)|$, which is unacceptable. If we could ensure that the distributions $\mathcal{D}'(R)$ obtained in type-2 legal clustering are careful with respect to $v^*$, then this accumulation of congestion could be avoided, but unfortunately we do not

know how to ensure that. In order to overcome this difficulty, we will carefully alternate between type-1 and type-2 legal clusterings. Specifically, we will require that a type-2 legal clustering contains a single distinguished cluster $R^*$, whose corresponding distribution $\mathcal{D}'(R^*)$ is careful with respect to $v^*$, and that $R^*$ contains a very large fraction of clusters of $\mathcal{C}'$. We will also require that a type-1 legal clustering $\mathcal{R}'$ of the graph associated with cluster $R^*$ is provided, and that for each cluster $R' \in \mathcal{R}'$, the number of clusters of $\mathcal{C}'$ contained in $R'$ is relatively small. This carefull alternation between type-1 and type-2 legal clusterings will allow us to compute distributions $\mathcal{D}'(S)$ over the routers of $\Lambda'_G(S)$ for each cluster $S \in \mathcal{L}$ of the laminar family that we construct, such that the expected congestion on every edge of $G$ due to the router drawn from the distribution is not too large. We now formally define a type-2 legal clustering.

**Definition 7.7 (Type-2 Legal Clustering)** *Let $G'$ be a graph with a special vertex $v^* \in V(G')$, and let $\mathcal{C}'$ be a collection of disjoint vertex-induced subgraphs of $G' \setminus \{v^*\}$, that we call basic clusters. A* type-2 legal clustering *of $G'$ with respect to $v^*$ and $\mathcal{C}'$ consists of the following four ingredients:*

1. *a helpful clustering $(\mathcal{R}, \{\mathcal{D}'(R)\}_{R\in\mathcal{R}})$ of $G'$ with respect to $v^*$ and $\mathcal{C}'$;*
2. *a nice witness structure for the graph $G'_{|\mathcal{R}}$ with respect to the set $\mathcal{C}''$ of clusters, where $\mathcal{C}''$ contains every cluster $C \in \mathcal{C}'$ with $C \subseteq G' \setminus \left(\bigcup_{R\in\mathcal{R}} V(R)\right)$;*
3. *a distinguished cluster $R^* \in \mathcal{R}$, that contains at least $\left\lfloor \left(1 - 1/2^{(\log m)^{3/4}}\right) |\mathcal{C}'| \right\rfloor$ clusters of $\mathcal{C}'$, such that the distribution $\mathcal{D}'(R^*)$ is careful with respect to $v^*$; and*
4. *a type-1 legal clustering $(\mathcal{R}', \{\mathcal{D}'(R)\}_{R\in\mathcal{R}'})$ of graph $G^*$, with respect to special vertex $v^{**}$, and cluster set $\mathcal{C}^*$, where $G^*$ is the graph that is obtained from graph $G'$ by contracting all vertices of $G' \setminus R^*$ into the special vertex $v^{**}$, and $\mathcal{C}^*$ contains all clusters $C \in \mathcal{C}'$ with $C \subseteq R^*$. We also require that every cluster $R' \in \mathcal{R}'$ contain at most $\left\lfloor \left(1 - 1/2^{(\log m)^{3/4}}\right) |\mathcal{C}'| \right\rfloor$ clusters of $\mathcal{C}'$.*

The key ingerdient of the proof of Theorem 7.3 is the following theorem, that will allow us to gradually construct the desired laminar family $\mathcal{L}$ of clusters.

**Theorem 7.8** *There is an efficient randomized algorithm, whose input consists of:*

- *a graph $G'$, and a parameter $m$ that is greater than a sufficiently large constant, such that $|V(G')|, |E(G')| \leq m$;*
- *a special vertex $v^* \in V(G')$, such that the cluster $G' \setminus \{v^*\}$ has the $\alpha_1$-bandwidth property in $G'$; and*
- *a collection $\mathcal{C}'$ of disjoint vertex-induced subgraphs of $G' \setminus \{v^*\}$ called* basic clusters*, such that every cluster $C \in \mathcal{C}'$ has the $\alpha_0$-bandwidth property, and $|\mathcal{C}'| \geq 2$.*

*The algorithm either returns FAIL, or computes a type-1 or a type-2 legal clustering of $G'$ with respect to $v^*$ and $\mathcal{C}'$. The probability that the algorithm returns FAIL is $1/m^8$. Moreover, if the algorithm computes a type-1 legal clustering $(\mathcal{R}, \{\mathcal{D}'(R)\}_{R\in\mathcal{R}})$, then every cluster $R \in \mathcal{R}$ contains at most $\left\lfloor \left(1 - 1/2^{(\log m)^{3/4}}\right) |\mathcal{C}'| \right\rfloor$ clusters of $\mathcal{C}'$.*

We prove Theorem 7.8 in the remainder of this subsection, after we complete the proof of Theorem 7.3 using it. We will construct a laminar family $\mathcal{L}$ of clusters of graph $G$, in a top-down manner. For every cluster $R \in \mathcal{L}$, we will define a distribution $\mathcal{D}''(R)$ over external routers in $\Lambda'_G$, such that, for

every edge $e \in E(G) \setminus E(R)$, $\mathbf{E}_{\mathcal{Q}'(R)\sim\mathcal{D}''(R)}\left[\mathrm{cong}_G(\mathcal{Q}'(R),e)\right] \leq \beta^{O((\log m)^{3/4})}$. We will also define a partition $(\mathcal{L}^{\mathrm{light}},\mathcal{L}^{\mathrm{light}})$ of the clusters of $\mathcal{L}$, and we will define, for each cluster $R \in \mathcal{L}^{\mathrm{light}}$ a distribution $\mathcal{D}(R)$ over the internal routers in $\Lambda_G(R)$, such that cluster $R$ is $\hat{\beta}$-light with respect to $\mathcal{D}(R)$, for $\hat{\beta} = 2^{O((\log m)^{3/4}\log\log m)}$. We will ensure that, with high probability, every cluser in $\mathcal{L}^{\mathrm{bad}}$ is $\hat{\beta}$-bad. Once we complete consructing the laminar family $\mathcal{L}$, we will apply algorithm AlgBasicDisengagement from Section 5.3 to the resulting tuple $(\mathcal{L},\mathcal{L}^{\mathrm{bad}},\mathcal{L}^{\mathrm{light}},\{\mathcal{D}''(R)\}_{R\in\mathcal{L}},\{\mathcal{D}(R)\}_{R\in\mathcal{L}^{\mathrm{light}}})$ to obtain the final collection $\mathcal{I}_1$ of instances. We will also provide a nice witness structure for each such resulting instance. We now proceed to describe tha algorithm for constructing the laminar family $\mathcal{L}$ of clusters.

Initially, we start with the laminar family $\mathcal{L}$ containing a single cluster – graph $G$. Since $\Lambda'_G(G) = \emptyset$ (as $\delta_G(G) = \emptyset$), the distribution $\mathcal{D}'(G)$ is defined in a trivial way (e.g. it selects $\emptyset$ with probability 1). We also let $\mathcal{L}^{\mathrm{light}}$ contain a single cluster – the cluster $G$, whose distribution $\mathcal{D}(G)$ over internal routers is defined in a similar trivial way. Lastly, we set $\mathcal{L}^{\mathrm{bad}} = \emptyset$.

We simultaneously consider the partitioning tree $\tau(\mathcal{L})$ associated with the laminar family $\mathcal{L}$ (see Section 5.1.1 for a definition). Initially, tree $\tau(\mathcal{L})$ consists of a single vertex $v(G)$, associated with the cluster $G$. The algorithm then performs iterations, as long as there is some cluster $R \in \mathcal{L}$, whose corresponding vertex $v(R)$ is a leaf vertex in the tree $\tau(\mathcal{L})$, and there are at least two clusters of $\mathcal{C}$ that are contained in $R$. We will ensure that every cluster $R' \in \mathcal{L}$ has the $\alpha_1$-bandwidth property in $G$. Notice that this trivially holds for the initial cluster $G$.

We now describe an interation for processing a cluster $R \in \mathcal{L}$. We assume that $v(R)$ is a leaf vertex in the current partitioning tree $\tau(\mathcal{L})$, and that there are at least two clusters of $\mathcal{C}$ that are contained in $R$.

In order to process cluster $R$, we construct a graph $G'$, with a special vertex $v^*$, as follows. If $R = G$, then we let $G'$ be a graph that is obtained from $G$, by adding a new special vertex $v^*$ to it, that connects with an edge to an arbitrary fixed vertex $v_0 \in V(G)$. Otherwise, if $R \subsetneq G$, then we let $G'$ be the graph that is obtained from $G$ by contracting all vertices of $V(G) \setminus V(R)$ into the special vertex $v^*$. Note that, since we are guaranteed that cluster $R$ has the $\alpha_1$-bandwidth property in $G$, cluster $G' \setminus \{v^*\}$ of $G'$ must have the $\alpha_1$-bandwidth property in $G'$ (in case where $R = G$ this property holds trivially, as $\delta'_G(G)$ contains a single edge). We let $\mathcal{C}' \subseteq \mathcal{C}$ be the set of all basic clusters $C \in \mathcal{C}$ with $C \subseteq R$. We then apply the algorithm from Theorem 7.8 to graph $G'$, special vertex $v^*$, and set $\mathcal{C}'$ of clusters; parameter $m$ remains the same as in the input to Theorem 7.3. If the algorithm returns FAIL, then we terminate our algorithm with a FAIL. Otherwise, consider the legal clustering that the algorithm produces (which may be a type-1 or a type-2 legal clustering), and let $\mathcal{R}$ be the resulting set of clusters.

We add every cluster $R' \in \mathcal{R}$ to the laminar family $\mathcal{L}$, where it becomes a child cluster of cluster $R$. Recall that, from the properties of a helpful clustering, vertex $v^*$ may not lie in $R'$, so $R' \subseteq R$ must hold. Moreover, $R'$ must have the $\alpha_1$-bandwidth property in $G'$, and hence in $G$. Recall that we also obtain a distribution $\mathcal{D}'(R')$ over external routers in $\Lambda'_{G'}(R')$, such that, for every edge $e \in E(G') \setminus E(R')$, $\mathbf{E}_{\mathcal{Q}'(R')\sim\mathcal{D}'(R')}\left[\mathrm{cong}_{G'}(\mathcal{Q}'(R'),e)\right] \leq \beta$. Unfortunately, this distribution is not sufficiently good for us, since we need the distribution $\mathcal{D}'(R')$ to be over the collection $\Lambda'_G(R')$ of external routers in graph $G$, and not in graph $G'$. We show how to modify this distribution later.

Next, we process each cluster $R' \in \mathcal{R}$ one by one. Consider any such cluster $R'$. We apply Algorithm AlgClassifyCluster from Theorem 6.1 to instance $I = (G,\Sigma)$ of MCNwRS, and cluster $R'$, that has the $\alpha_1$-bandwidth property in $G$, together with parameter $p = 1/m^{10}$. Recall that the running time of the algorithm is $O(\mathrm{poly}(m\log m))$. If the algorithm returns FAIL, then we add $R'$ to the set $\mathcal{L}^{\mathrm{bad}}$ of clusters. Otherwise, the algorithm computes a distribution $\mathcal{D}(R')$ over internal routers in $\Lambda_G(R')$, such that cluster $R'$ is $\beta^*$-light with respect to $\mathcal{D}(R')$, where $\beta^* = 2^{O(\sqrt{\log m}\cdot\log\log m)}$. We then add cluster $R'$ to set $\mathcal{L}^{\mathrm{light}}$. Recall that, if cluster $R'$ is not $\eta^*$-bad, for $\eta^* = 2^{O((\log m)^{3/4}\log\log m)}$, then the

probability that the algorithm returns FAIL (that is, the algorithm *errs*), is at most $1/m^{10}$. If the clustering $\mathcal{R}$ is a type-1 legal clustering, then we also *mark* the vertex $v(R')$ in the decomposition tree $\tau(\mathcal{L})$, to indicate that the distribution $\mathcal{D}'(R')$ is careful with respect to $v^*$. Otherwise, $\mathcal{R}'$ is a type-2 legal clustering, and we only mark vertex $v(R')$ if $R' = R^*$, where $R^*$ is the distinguished cluster. Recall that in this case, the distribution $\mathcal{D}'(R^*)$ over external routers of $R^*$ is also careful with respect to $v^*$.

If the algorithm from Theorem 7.8 returned a type-2 legal clustering, then we also consider the type-1 legal clustering $\mathcal{R}'$ of $R^*$, that is given as part of the type-2 legal clustering of $R$. We process every cluster $R'' \in \mathcal{R}'$ one by one. When cluster $R''$ is processed, we add it to the laminar family $\mathcal{L}$ and we add vertex $v(R'')$ to the partitioning tree $\tau(\mathcal{L})$ as a child of vertex $v(R^*)$; we also mark vertex $v(R'')$ in the tree, to indicate that the distribution $\mathcal{D}'(R'')$ over the external routers of $R''$ is careful with respect to $v^{**}$. As before, we apply the Algorithm AlgClassifyCluster from Theorem 6.1 to instance $I = (G, \Sigma)$ of MCNwRS, and cluster $R''$, that has the $\alpha_1$-bandwidth property in $G$, together with parameter $p = 1/m^{10}$, As before, if the algorithm returns FAIL, then we add $R''$ to $\mathcal{L}^{\text{bad}}$, and otherwise we add it to $\mathcal{L}^{\text{light}}$, together with the distribution $\mathcal{D}(R'')$ over internal routers in $\Lambda_G(R'')$, such that cluster $R''$ is $\beta^*$-light with respect to $\mathcal{D}(R'')$. This completes the description of the algorithm for constructing the laminar family $\mathcal{L}$ of clusters. We now establish some of its useful properties. The following claim, whose proof appears in Appendix G.1 will be used to bound the height of the tree $\tau$, and the number of marked vertices on any root-to-leaf path.

**Claim 7.9** *Consider any root-to-leaf path $P$ in the decomposition tree $\tau(\mathcal{L})$. Then $P$ contains at most $2^{O((\log m)^{3/4})}$ marked vertices, and at most $O(\log^{3/4} m)$ unmarked vertices. In particular, the depth of the tree $\tau(\mathcal{L})$ is at most $2^{O((\log m)^{3/4})}$.*

Next, we provide an algorithm for computing, for each cluster $R \in \mathcal{L}$, the desired distribution $\mathcal{D}''(R)$ over the external routers in $\Lambda'_G(R)$. The proof of the following claim is somewhat technical, and is deferred to Appendix G.2.

**Claim 7.10** *There is an efficient algorithm that, given a cluster $R \in \mathcal{L}$, computes a distribution $\mathcal{D}''(R)$ over the external $R$-routers in $\Lambda'_G(R)$, such that, for every edge $e \in E(G) \setminus E(R)$*

$$\mathbf{E}_{\mathcal{Q}'(R) \sim \mathcal{D}''(R)} \left[ \text{cong}_G(\mathcal{Q}'(R), e) \right] \leq \beta^{i+1},$$

*where $i$ is the total number of unmarked vertices on the unique path in tree $\tau(\mathcal{L})$, connecting $v(R)$ to the root of the tree.*

To summarize, if our algorithm did not return FAIL, we have now obtained a laminar family $\mathcal{L}$ of clusters of graph $G$, with $G \in \mathcal{L}$, so that the depth of the family $\mathcal{L}$ is at most $2^{O((\log m)^{3/4})}$. We have also computed a partition $(\mathcal{L}^{\text{light}}, \mathcal{L}^{\text{bad}})$ of clusters in $\mathcal{L}$, and, for each cluster $R \in \mathcal{L}^{\text{light}}$, a distribution $\mathcal{D}(R)$ over internal routers in $\Lambda_G(R)$, such that cluster $R$ is $\beta^*$-light with respect to $\mathcal{D}(R)$, for $\beta^* = 2^{O(\sqrt{\log m} \cdot \log\log m)}$. We are also guaranteed that, with probability at least $1 - 1/m^9$, every cluster $R \in \mathcal{L}^{\text{bad}}$ is $\eta^*$-bad, for $\eta^* = 2^{O((\log m)^{3/4} \log\log m)}$. Lastly, we have computed, for every cluster $R \in \mathcal{L}$, a distribution $\mathcal{D}''(R)$ over external routers in $\Lambda'_G(R)$, such that, for every edge $e \in E(G) \setminus E(R)$, $\mathbf{E}_{\mathcal{Q}'(R) \sim \mathcal{D}''(R)} \left[ \text{cong}_G(\mathcal{Q}'(R), e) \right] \leq \beta^{i+1}$, where $i$ is the total number of unmarked vertices on the unique path in tree $\tau(\mathcal{L})$, connecting $v(R)$ to the root of the tree. Since, from Claim 7.9, the number of such vertices is bounded by $O(\log^{3/4} m)$, we get that, for every edge $e \in E(G) \setminus E(R)$, $\mathbf{E}_{\mathcal{Q}'(R) \sim \mathcal{D}''(R)} \left[ \text{cong}_G(\mathcal{Q}'(R), e) \right] \leq \beta^{O((\log m)^{3/4})} \leq 2^{O((\log m)^{3/4} \log\log m)}$, since $\beta = \log^{18} m$.

We apply algorithm AlgBasicDisengagement from Section 5.3 to the resulting tuple $(\mathcal{L} = \mathcal{L}^{\text{bad}} \cup \mathcal{L}^{\text{light}}, \{\mathcal{D}''(R)\}_{R \in \mathcal{L}}, \{\mathcal{D}(R)\}_{R \in \mathcal{L}^{\text{light}}})$, to obtain the final collection $\mathcal{I}_1$ of subinstances of $I$. Let $\hat{\beta} =$

$2^{c(\log m)^{3/4}\log\log m}$ for some large enough constant $c$. We are then guaranteed that every cluster $R \in \mathcal{L}^{\text{light}}$ is $\hat{\beta}$-light with respect to the distribution $\mathcal{D}(R)$, and that, for every cluster $R \in \mathcal{L}$ and every edge $e \in E(G) \setminus E(R)$, $\mathbf{E}_{\mathcal{Q}'(R)\sim\mathcal{D}''(R)}\left[\text{cong}_G(\mathcal{Q}'(R), e)\right] \leq \hat{\beta}$. We can set $c$ to be large enough so that $\eta^* \leq \hat{\beta}$ holds. We say that a bad event $\mathcal{E}$ happens if some cluster $R \in \mathcal{L}^{\text{bad}}$ is not $\hat{\beta}$-bad. From the above discussion, the probability of $\mathcal{E}$ happening is at most $1/m^9$. If Event $\mathcal{E}$ does not happen, then, from Lemma 5.6, $\mathbf{E}\left[\sum_{I'\in\mathcal{I}_1}\mathsf{OPT}_{\mathsf{cnwrs}}(I')\right] \leq O(\mathsf{dep}(\mathcal{L})\cdot\hat{\beta}^2\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+|E(G)|)) \leq 2^{O((\log m)^{3/4}\log\log m)}\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+|E(G)|)$. If Event $\mathcal{E}$ happens (which happens with probability at most $1/m^9$), then clearly $\mathbf{E}\left[\sum_{I'\in\mathcal{I}_1}\mathsf{OPT}_{\mathsf{cnwrs}}(I')\right] \leq \sum_{I'=(G'',\Sigma'')\in\mathcal{I}_1}|E(G'')|^2 \leq m^3$. Therefore, overall, $\mathbf{E}\left[\sum_{I'\in\mathcal{I}_1}\mathsf{OPT}_{\mathsf{cnwrs}}(I')\right] \leq 2^{O((\log m)^{3/4}\log\log m)}\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+|E(G)|))$

Additionally, from Lemma 5.3, there is an efficient algorithm, that, given, for each instance $I' \in \mathcal{I}$, a solution $\varphi(I')$, computes a solution for instance $I$ of value at most $\sum_{I'\in\mathcal{I}}\mathsf{cr}(\varphi(I'))$. In order to prove that the algorithm computes a valid $2^{O((\log m)^{3/4}\log\log m)}$-decomposition of instance $I$, it is now sufficient to prove that $\sum_{I'=(G'',\Sigma'')\in\mathcal{I}_1}|E(G'')| \leq O(|E(G)|)$. We do so in the next claim, whose proof is deferred to Appendix G.3.

**Claim 7.11** $\sum_{I'=(G'',\Sigma'')\in\mathcal{I}_1}|E(G'')| \leq O(|E(G)|)$.

From the above discussion, if our algorithm did not return FAIL, it computed a valid $2^{O((\log m)^{3/4}\log\log m)}$-decomposition of instance $I$. Consider now any resulting instance $\tilde{I} = (\tilde{G}, \tilde{\Sigma}) \in \mathcal{I}_1$. From the definition of basic disengagment, this instance must correspond to some cluster $R \in \mathcal{L}$. Assume first that $R \neq G$, and let $\mathcal{R}$ be the set of all child clusters of $R$ in $\mathcal{L}$ (if $R$ corresponds to a leaf vertex of $\tau(\mathcal{L})$, then $\mathcal{R} = \emptyset$). Recall that graph $\tilde{G}$ is obtained from graph $G$ by first contracting all vertices of $G \setminus R$ into a supernode $v^*$; denote the resulting graph by $G'$. We then contract every cluster $R' \in \mathcal{R}$ into a supernode, obtaining graph $\tilde{G}$. In other words, $\tilde{G} = G'_{|\mathcal{R}}$.

We now consider three cases. The first case is when $\mathcal{R}$ is a type-1 legal clustering for cluster $R$. In this case, there is at most one cluster $C \in \mathcal{C}$ that is contained in $G' \setminus \bigcup_{R'\in\mathcal{R}} R'$, from the definition of type-1 legal clustering. Therefore, at most one cluster $C$ of $\mathcal{C}$ is contained in $\tilde{G}$. We define the nice witness structure for graph $\tilde{G}$, with respect to the set $\mathcal{C}'$ of basic clusters, where $\mathcal{C}' = \{C\}$ if there is a cluster $C \in \mathcal{C}$ that is contained in $\tilde{G}$, and $\mathcal{C}' = \emptyset$ otherwise. We let $\tilde{\mathcal{S}} = \left\{\tilde{S}_1\right\}$, where $\tilde{S}_1 = \hat{G}$, and we let $\tilde{\mathcal{S}}' = \left\{\tilde{S}'_1\right\}$, where $\tilde{S}'_1 = C$ if $\mathcal{C}' = \{C\}$, and $\tilde{S}'_1 = \{v\}$, where $v$ is an arbitrary vertex of $\tilde{G}$ otherwise. Note that, under these definitions, $\hat{E} = \emptyset$, and so we can set $\hat{\mathcal{P}} = \emptyset$. It is easy to verify that $(\tilde{\mathcal{S}}, \tilde{\mathcal{S}}', \hat{\mathcal{P}})$ is a nice witness structure for $\tilde{G}$.

The second case is when cluster $R$ corresponds to a leaf vertex of tree $\tau(\mathcal{L})$. From our algorithm, this means that there is at most one cluster $C \in \mathcal{C}$ with $C \subseteq R$. In this case, we define the nice witness structure for graph $\tilde{G}$ similarly to the first case.

Lastly, in the third case, when the algorithm from Theorem 7.8 was applied to cluster $R$, it returned a type-2 legal clustering, with the corresponding cluster set $\mathcal{R}$. In this case, the algorithm also must produce a nice witness structure for the graph $G'_{|\mathcal{R}} = \tilde{G}$, with respect to the set $\mathcal{C}''$ of clusters, that contains every cluster $C \in \mathcal{C}$ with $C \subseteq G' \setminus \left(\bigcup_{R'\in\mathcal{R}} V(R')\right)$. In other words, $\mathcal{C}'' = \mathcal{C}(\tilde{G})$.

It remains to consider the case where $R = G$. As before, we let $\mathcal{R}$ denote the set of all child-clusters of cluster $R$. Recall that in this case, graph $\tilde{G}$ is obtained from graph $G$ by contracting every cluster $R' \in \mathcal{R}$ into a supernode. Graph $G'$ was obtained from graph $G$ by adding a special vertex $v^*$ that connects to some vertex $v_0 \in V(G)$ with an edge. Therefore, graph $G'_{|\mathcal{R}}$ is a graph that is obtained from $\tilde{G}$ by adding a special vertex $v^*$ to it and connecting it to some vertex of $\tilde{G}$. Recall that vertex $v^*$ may not lie in any cluster of $\mathcal{R}$. We start by defining a nice structure for graph $G'_{|\mathcal{R}}$, with respect to the collection $\mathcal{C}''$ of clusters, that contains every cluster $C \in \mathcal{C}$ with $C \subseteq G'_{|\mathcal{R}}$ exactly as before (when

we assumed that $R \neq G$). Since vertex $v^*$ has degree 1, we can assume that no path in $\hat{\mathcal{P}}$ contains $v^*$. Moreover, if $v^* \in \tilde{S}'_i$, for some $\tilde{S}'_i \in \tilde{\mathcal{S}}'_i$, then cluster $\tilde{S}'_i \setminus \{v^*\}$ still has the $\alpha^*$-bandwidth property. By deleting vertex $v^*$ from the cluster of $\tilde{\mathcal{S}}$ to which it belongs, and also from a cluster of $\tilde{\mathcal{S}}'$ to which it belongs (if such a cluster exists), we obtain a nice witness structure for graph $\tilde{G}$, with respect to cluster set $\mathcal{C}''$, as required. The algorithm may return FAIL if any application of the algorithm from Theorem 7.8 returned FAIL. Since $|\mathcal{L}| \leq m$, and the probability that a singel application of the algorithm from Theorem 7.8 returns FAIL is at most $1/m^8$, overall, the probability that the algorithm returns FAIL is at most $1/m^6$.

In order to complete the proof of Theorem 7.3 it is now enough to prove Theorem 7.8, which we do next.

### 7.2.1 Proof of Theorem 7.8

Throughout the proof, we will consider various graphs, sets of disjoint clusters in these graphs, and the corresponding contracted graphs. Let $H$ be any graph, and let $\mathcal{R}$ be any set of disjoint vertex-induced subgraphs (clusters) of graph $H$. Let $\hat{H} = H_{|\mathcal{R}}$ be the contracted graph corresponding to $H$ and $\mathcal{R}$, that is obtained from $H$ by contracting every cluster $R \in \mathcal{R}$ into a supernode $v_R$. Observe that every subset $\hat{U} \subseteq V(\hat{H})$ of vertices of $\hat{H}$ naturally defines a vertex-induced subgraph of $H$, which is a subgraph of $H$ induced by vertex set $U = \left(\bigcup_{v_R \in \hat{U}} V(R)\right) \cup (V(H) \cap \hat{U})$; in other words, $U$ contains all regular vertices of $\hat{U}$, and the vertices of every cluster $R \in \mathcal{R}$ with $v_R \in \hat{U}$. We will refer to $H[U]$ as the *subgraph of $H$ (or cluster of $H$) defined by the set $\hat{U}$ of vertices of $\hat{H}$*. Similarly, if $S$ is a cluster of $\hat{H}$ induced by vertex set $\hat{U}$, we will refer to $H[U]$ as the cluster of $H$ defined by $S$.

Assume now that we are given any graph $H$, a special vertex $v^*$ in $H$, and a collection $\mathcal{R}$ of disjoint clusters of $H$, such that vertex $v^*$ does not lie in any cluster of $\mathcal{R}$, and every cluster $R \in \mathcal{R}$ has $\alpha$-bandwidth property, for some parameter $0 < \alpha < 1$. As before, we denote $\hat{H} = H_{|\mathcal{R}}$. Next, we consider a Gomory-Hu tree $\tau$ of the graph $\hat{H}$ (see Section 4.2.2 for a definition). We root the tree $\tau$ at the special vertex $v^*$. For every vertex $u \in V(\tau)$, we let $\tau_u$ be the subtree of $\tau$ rooted at $u$.

We will use the following useful observation multiple times. The proof is deferred to Appendix G.4.

**Observation 7.12** *Let $u \in V(\tau) \setminus \{v^*\}$ be any non-root vertex of the tree $\tau$, and let $S$ be the cluster of $H$ that is defined by the set $V(\tau_u)$ of vertices of $\hat{H}$. Then cluster $S$ has the $\alpha$-bandwidth property in $H$. Moreover, there is an efficient algorithm to compute a distribution $\mathcal{D}'(S)$ over the external routers in $\Lambda'_H(S)$, such that distribution $\mathcal{D}'(S)$ is careful with respect to $v^*$, and, for every edge $e \in E(H) \setminus E(S)$, $\mathbf{E}_{\mathcal{Q}'(S) \sim \mathcal{D}'(S)}\left[\mathrm{cong}_H(\mathcal{Q}'(S), e)\right] \leq O(\log^4 m/\alpha)$.*

For convenience, in the remainder of the proof, we denote graph $G'$ by $G$, and the set $\mathcal{C}'$ of clusters by $\mathcal{C}$. We start with the graph $G$ and the set $\mathcal{C}$ of basic clusters, and we let $H = G_{|\mathcal{C}}$ be the corresponding contracted graph. We consider the Gomory-Hu tree $\tau$ of the graph $H$. We root the tree $\tau$ at the special vertex $v^*$. For every vertex $u \in V(\tau)$, we let $\tau_u$ be the subtree of $\tau$ rooted at $u$, and we let the weight $w(u)$ be the number of supernodes (vertices corresponding to clusters in $\mathcal{C}$) in the tree $\tau_u$. Let $u^*$ be the vertex of $\tau$ that is furthest from the root $v^*$, such that $w(u^*) \geq \left\lfloor \left(1 - 1/2^{(\log m)^{3/4}}\right) |\mathcal{C}| \right\rfloor$. We now consider two cases.

The first case happens if $u^* = v^*$. In this case, we will compute a type-1 legal clustering of $G$. Let $u_1, \ldots, u_q$ denote all child vertices of $v^*$. For all $1 \leq i \leq q$, let $R_i$ be the cluster of the graph $G$ defined by the vertex set $V(\tau_{u_i})$. Denote $\mathcal{R} = \{R_1, \ldots, R_q\}$. Since every cluster $C \in \mathcal{C}$ has the $\alpha_0$-bandwidth property, and $H = G_{|\mathcal{C}}$, from Observation 7.12, each cluster $R_i \in \mathcal{R}$ has the $\alpha_0 \geq \alpha_1$-bandwidth property. From the construction, vertex $v^*$ may not lie in any of the clusters of $\mathcal{R}$, and, for each cluster $R \in \mathcal{R}$, and for every basic cluster $C \in \mathcal{C}$, either $C \subseteq R$ or $V(C) \cap V(R) = \emptyset$. We use the

algorithm from Observation 7.12 to construct, for every cluster $R \in \mathcal{R}$, a distribution $\mathcal{D}'(R)$ over the external $S$-routers in $\Lambda'_G(S)$, such that the distribution is careful with respect to $v^*$, and, for every edge $e \in E(G) \setminus E(R)$, $\mathbf{E}_{\mathcal{Q}'(R)\sim\mathcal{D}'(R)}\left[\mathrm{cong}_G(\mathcal{Q}'(R), e)\right] \leq O(\log^4 m/\alpha_0) \leq \beta$.

Note that $G \setminus \bigcup_{R\in\mathcal{R}} R$ consists of only one vertex – vertex $v^*$. Therefore, $\mathcal{R}$ is a legal type-1 clustering of graph $G$. We terminate the algorithm, and return this clustering.

We assume from now on that $u^* \neq v^*$. We will provide an algorithm for computing a type-2 legal clustering of $G$. Let $R^*$ be the subgraph of $G$ defined by vertex set $V(\tau_{u^*})$ of graph $H$. As before, from Observation 7.12, cluster $R^*$ has the $\alpha_0 \geq \alpha_1$-bandwidth property, it does not contain the verex $v^*$, and, for every basic cluster $C \in \mathcal{C}$, either $C \subseteq R^*$ or $V(C) \cap V(R^*) = \emptyset$. From the definition of vertex $u^*$, the total number of basic clusters of $\mathcal{C}$ that are contained in $R^*$ is at least $\left\lfloor \left(1 - 1/2^{(\log m)^{3/4}}\right)|\mathcal{C}| \right\rfloor$. We also use the algorithm from Observation 7.12 to construct a distribution $\mathcal{D}'(R^*)$ over the external $R^*$-routers in $\Lambda'_G(S)$, such that the distribution is careful with respect to $v^*$, and, for every edge $e \in E(G) \setminus E(R^*)$, $\mathbf{E}_{\mathcal{Q}'(R^*)\sim\mathcal{D}'(R^*)}\left[\mathrm{cong}_G(\mathcal{Q}'(R^*), e)\right] \leq O(\log^4 m/\alpha_0) \leq \beta$. In the final type-2 legal clustering $\mathcal{R}$ for graph $G$ that our algorithm will return, cluster $R^*$ will play the role of the distinguished cluster, and the distribution $\mathcal{D}'(R^*)$ over the set of its external routers will remain unchanged. Let $G^*$ be the graph associated with the cluster $R^*$: that is, graph $G^*$ is obtained from graph $G$ by contracting all vertices of $G \setminus R^*$ into a special vertex, that we denote by $v^{**}$. We also denote by $\mathcal{C}^* \subseteq \mathcal{C}$ the set of all basic clusters $C \in \mathcal{C}$ with $C \subseteq R^*$. We now construct a type-1 legal clustering $\mathcal{R}'$ of $G^*$, which is required as part of definition of type-2 legal clustering of $G$. Denote the child vertices of vertex $u^*$ in the tree $\tau$ by $u_1, \ldots, u_q$. For all $1 \leq i \leq q$, let $R_i$ be the cluster of the graph $G$ defined by the vertex set $V(\tau_{u_i})$. Denote $\mathcal{R}' = \{R_1, \ldots, R_q\}$. Since every cluster $C \in \mathcal{C}$ has the $\alpha_0$-bandwidth property, and $H = G_{|\mathcal{C}}$, from Observation 7.12, each cluster $R_i \in \mathcal{R}$ has the $\alpha_0 \geq \alpha_1$-bandwidth property. From the construction, vertex $v^{**}$ may not lie in any of the clusters of $\mathcal{R}'$, and, for each cluster $R \in \mathcal{R}'$, and for every basic cluster $C \in \mathcal{C}^*$, either $C \subseteq R$ or $V(C) \cap V(R) = \emptyset$ holds. Consider now some cluster $R_i \in \mathcal{R}'$, and denote $\delta_G(R_i) = E_i$. From the properties of the Gomory-Hu tree (see Theorem 4.9), there is a collection $\mathcal{Q}'_i$ of edge-disjoint paths in graph $H$, routing the edges of $E_i$ to vertex $u^*$, that are internally disjoint from $V(\tau_{u_i})$. Let $H^*$ be the graph obtained from $H$, by contracting all vertices of $V(H) \setminus V(\tau_{u^*})$ into a supernode $\hat{v}^*$. A simple transformation of the paths in $\mathcal{Q}'_i$ shows that there is a collection $\mathcal{Q}''_i$ of edge-disjoint paths in graph $H^*$, routing the edges of $E_i$ to $u^*$. Observe that graph $H^*$ is precisely the contracted graph of $G^*$ with respect to the set $\mathcal{C}^*$ of clusters, that is, $H^* = G^*_{|\mathcal{C}^*}$, and recall that each cluster $C \in \mathcal{C}^*$ has the $\alpha_0$-bandwidth property.

If vertex $u^*$ is not a supernode, then we apply the algorithm from Claim 4.41 to graph $H^*$, the set $\mathcal{C}^*$ of clusters, and the set $\mathcal{Q}''_i$ of paths, to obtain a set $\mathcal{Q}^*_i$ of paths in graph $G^*$, routing the edges of $E_i$ to vertex $u^*$, such that every path in $\mathcal{Q}^*_i$ is internally disjoint from $R_i$. Moreover, for every edge $e \in \bigcup_{C\in\mathcal{C}^*} E(C)$, the paths of $\mathcal{Q}^*_i$ cause congestion at most $\lceil 1/\alpha_0 \rceil$, while for every edge $e \in E(G^*) \setminus \left(\bigcup_{C\in\mathcal{C}^*} E(C)\right)$, the paths of $\mathcal{Q}^*_i$ cause congestion at most 1. In particular, the set $\mathcal{Q}^*_i$ of paths is careful with respect to vertex $v^{**}$. We then define a distribution $\mathcal{D}'(R_i)$ over the set $\Lambda'_{G^*}(R_i)$ of external $R_i$-routers to choose the set $\mathcal{Q}^*_i$ of paths with probability 1.

Assume now that vertex $u^*$ is a supernode, and that it represents some cluster $C \in \mathcal{C}^*$. We apply the algorithm from Claim 4.41 to graph $H^*$, the set $\mathcal{C}^* \setminus \{C\}$ of clusters, and the set $\mathcal{Q}''_i$ of paths, to obtain a set $\mathcal{Q}^*_i$ of paths in graph $G^*$, routing the edges of $E_i$ to edges of $\delta_{G^*}(C)$, such that every path in $\mathcal{Q}^*_i$ is internally disjoint from $R_i$. As before, for every edge $e \in \bigcup_{C'\in\mathcal{C}^*} E(C')$, the paths of $\mathcal{Q}^*_i$ cause congestion at most $\lceil 1/\alpha_0 \rceil$, while for every edge $e \in E(G^*) \setminus \left(\bigcup_{C'\in\mathcal{C}^*} E(C')\right)$, the paths of $\mathcal{Q}^*_i$ cause congestion at most 1. As before, the set $\mathcal{Q}^*_i$ of paths is careful with respect to vertex $v^{**}$. We use the algorithm from Lemma 4.27 to compute a distribution $\mathcal{D}(C)$ over internal $C$-routers in $\Lambda_{G^*}(C)$, such that, for every edge $e \in E(C)$, $\mathbf{E}_{\mathcal{Q}(C)\sim\mathcal{D}(C)}\left[\mathrm{cong}(\mathcal{Q}(C), e)\right] \leq \log^4 m/\alpha_0$. We now define a distribution $\mathcal{D}'(R_i)$ over the set $\Lambda'_{G^*}(R_i)$ of external $R_i$-routers. In order to draw a router from the distribution, we first choose an internal $C$-router $\mathcal{Q}(C)$ from the distribution $\mathcal{D}(C)$. Let $x$ be the vertex that serves

the center of the router. For every edge $e \in E_i$, we let $\tilde{Q}(e)$ be the path obtained as follows. First, we let $Q^*(e)$ be the unique path of $\mathcal{Q}_i^*$ that originates from edge $e$. We let $e'$ be the last edge on path $\mathcal{Q}_i^*$, that must belong to $\delta_{G^*}(C)$. We then let $\tilde{Q}(e)$ be the path obtained by concatenating path $Q^*(e)$ with the unique path of $\mathcal{Q}(C)$ that originates at edge $e$. We let $\mathcal{Q}'(R_i) = \left\{\tilde{Q}(e) \mid e \in E_i\right\}$ be the resulting external $R_i$-router, that routes the edges of $E_i$ to $x$. Since every edge of $\delta_{G^*}(C)$ may lie on at most one path of $\mathcal{Q}_i^*$, it is immediate to verify that, for every edge $e \in E(C)$, $\mathrm{cong}(\mathcal{Q}'(R_i), e) \leq \mathrm{cong}(\mathcal{Q}(C), e)$, and so overall, for every edge $e'$, $\mathbf{E}_{\mathcal{Q}'(R_i) \sim \mathcal{D}'(R_i)}\left[\mathrm{cong}_{G^*}(\mathcal{Q}'(R_i), e')\right] \leq \frac{\log^4 m}{\alpha_0} \leq \beta$, since $\alpha_0 = 1/\log^3 m$ and $\beta = \log^{18} m$. As before, distribution $\mathcal{D}'(R_i)$ is careful with respect to $v^{**}$.

Lastly, observe that at most one cluster $C \in \mathcal{C}$ may be contained in graph $R^* \setminus \bigcup_{R \in \mathcal{R}'} R$ – the cluster associated with vertex $u^*$, if $u^*$ is a supernode. Therefore, $(\mathcal{R}', \{\mathcal{D}'(R)\}_{R \in \mathcal{R}'})$ is a type-1 legal clustering of graph $G^*$, with cluster set $\mathcal{C}^*$ and special vertex $v^{**}$. Moreover, from the choice of vertex $u^*$, we are guaranteed that every cluster $R' \in \mathcal{R}'$ contain at most $\left\lfloor \left(1 - 1/2^{(\log m)^{3/4}}\right) |\mathcal{C}'| \right\rfloor$ clusters of $\mathcal{C}$.

The remainder of the algorithm is iterative. We start with a helpful clustering $(\mathcal{R} = \{R^*\}, \{\mathcal{D}'(R^*)\})$ of $G$, and we view $R^*$ as the distinguished cluster of $\mathcal{R}$. We then iterate. In every iteration, we either establish that the current helpful clustering $(\mathcal{R}, \{\mathcal{D}'(R)\}_{R \in \mathcal{R}})$ is a type-2 legal clustering, by computing a nice witness structure for graph $G_{|\mathcal{R}}$, with respect to the set $\mathcal{C}''$ of clusters, containing every cluster $C \in \mathcal{C}$ with $C \subseteq G \setminus \left(\bigcup_{R \in \mathcal{R}} V(R)\right)$; or we will compute another helpful clustering of $G$ that is "better" in some sense, and use it to replace the current helpful clustering $(\mathcal{R}, \{\mathcal{D}'(R)\}_{R \in \mathcal{R}})$. We will ensure that the helpful clustering $\mathcal{R}$ that the algorithm maintains always contains the cluster $R^*$ that we defined above, which will always remain the distinguished cluster of $\mathcal{R}$. The distribution $\mathcal{D}'(R^*)$ over the external $R^*$-routers in $\Lambda'_G(R^*)$, and the type-1 legal clustering $(\mathcal{R}', \{\mathcal{D}'(R)\}_{R \in \mathcal{R}'})$ of the graph $G^*$ associated with cluster $R^*$ will remain unchanged throughout the algorithm. We will use the following definition in order to compare different helpful clusterings of $G$.

**Definition 7.13 (Comparing clusterings)** *Let $\mathcal{R}_1$, $\mathcal{R}_2$ be two helpful clusterings of graph $G$, with respect to special vertex $v^*$ and set $\mathcal{C}$ of basic clusters, such that $R^* \in \mathcal{R}_1 \cap \mathcal{R}_2$. Denote by $\mathcal{C}_1 \subseteq \mathcal{C}$ the set of all clusters $C \in \mathcal{C}$ with $C \subseteq G \setminus \left(\bigcup_{R \in \mathcal{R}_1} R\right)$, and define a subset $\mathcal{C}_2 \subseteq \mathcal{C}$ of basic clusters for $\mathcal{R}_2$ similarly. We say that clustering $\mathcal{R}_2$ is* better *than clustering $\mathcal{R}_1$ if one of the following hold:*

- *either $|\mathcal{C}_2| < |\mathcal{C}_1|$; or*
- *$|\mathcal{C}_1| = |\mathcal{C}_2|$, and $|E(G_{|\mathcal{R}_2})| < |E(G_{|\mathcal{R}_1})|$.*

The following lemma is key in the proof of Theorem 7.8.

**Lemma 7.14** *There is an efficient randomized algorithm, that, given a helpful clustering $(\mathcal{R}, \{\mathcal{D}'(R)\}_{R \in \mathcal{R}})$ of graph $G$ with respect to vertex $v^*$ and set $\mathcal{C}$ of basic clusters, such that $R^* \in \mathcal{R}$, either (i) establishes that $\mathcal{R}$ is a type-2 legal clustering by providing a nice witness structure for graph $G_{|\mathcal{R}}$, with respect to the set $\mathcal{C}''$ of clusters, containing every cluster $C \in \mathcal{C}$ with $C \subseteq G \setminus \left(\bigcup_{R \in \mathcal{R}} V(R)\right)$, or (ii) computes another helpful clustering $(\tilde{\mathcal{R}}, \{\mathcal{D}'(R)\}_{R \in \tilde{\mathcal{R}}})$ of graph $G$ with respect to $v^*$ and $\mathcal{C}$ with $R^* \in \mathcal{R}$, such that $\tilde{\mathcal{R}}$ is a better clustering than $\mathcal{R}$; or (iii) returns FAIL. The latter may only happen with probability at most $1/m^{10}$.*

It is immediate to complete the proof of Theorem 7.8. using Lemma 7.14. Our algorithm starts with the helpful custering $(\mathcal{R} = \{R^*\}, \{\mathcal{D}'(R^*)\})$ of $G$, where $\mathcal{D}'(R^*)$ is the distribution over external $R^*$-routers that we have computed above, and then iterates. In every iteration, we apply the algorithm from Lemma 7.14 to the current helpful clustering $(\mathcal{R}, \{\mathcal{D}'(R)\}_{R \in \mathcal{R}})$. If the algorithm establishes that $\mathcal{R}$ is a type-2 legal clustering by providing a nice witness structure for graph for graph $G_{|\mathcal{R}}$, with respect to the set $\mathcal{C}''$ of clusters, containing every cluster $C \in \mathcal{C}$ with $C \subseteq G \setminus \left(\bigcup_{R \in \mathcal{R}} V(R)\right)$, then

we terminate the algorithm with the resulting type-2 legal clustering $(\mathcal{R}, \{\mathcal{D}'(R)\}_{R\in\mathcal{R}})$; we view $R^*$ as the distinguished cluster of $\mathcal{R}$, and the type-1 legal clustering $(\mathcal{R}', \{\mathcal{D}'(R)\}_{R\in\mathcal{R}'})$ of the graph $G^*$ corresponding to $R^*$ remains unchanged. Otherwise, if the algorithm returns another helpful clustering $(\tilde{\mathcal{R}}, \{\mathcal{D}'(R)\}_{R\in\tilde{\mathcal{R}}})$, then we replace $(\mathcal{R}, \{\mathcal{D}'(R)\}_{R\in\mathcal{R}})$ with $(\tilde{\mathcal{R}}, \{\mathcal{D}'(R)\}_{R\in\tilde{\mathcal{R}}})$ and continue to the next iteration. Lastly, if the algorithm from Lemma 7.14 returns FAIL, then we terminate the algorithm and return FAIL as well. Let $\mathcal{C}'$ be the set of all clusters $C \in \mathcal{C}$ with $C \subseteq G \setminus \left(\bigcup_{R\in\mathcal{R}} V(R)\right)$, where $\mathcal{R}$ is the current helpful clustering. Since, in every iteration, either $|\mathcal{C}'|$ decreases, or $|\mathcal{C}'|$ remains the same but the number of edges in graph $G_{|\mathcal{R}}$ decreases, the algorithm is guaranteed to terminate after at most $m^2$ iterations. Since the probability of the algorithm to return FAIL in each iteration is at most $1/m^{10}$, the total probability that the algorithm returns FAIL is at most $1/m^8$. If the algorithm does not return FAIL, then it returns a type-2 clustering of $G$ as required. In order to complete the proof of Theorem 7.8, it is now enough to prove Lemma 7.14, which we do next.

### 7.2.2 Proof of Lemma 7.14

Recall that we are given a graph $G$ and a special vertex $v^*$ of $G$. We are also given a collection $\mathcal{C}$ of disjoint vertex-induced subgraphs of $G \setminus \{v^*\}$ called basic clusters, such that every basic cluster $C \in \mathcal{C}$ has the $\alpha_0$-bandwidth property. Lastly, we are given a helpful clustering $(\mathcal{R}, \{\mathcal{D}'(R)\}_{R\in\mathcal{R}})$ of $G$ with respect to $v^*$ and $\mathcal{C}$. Recall that vertex $v^*$ may not lie in any cluster of $\mathcal{R}$, and every cluster $R \in \mathcal{R}$ has the $\alpha_1$-bandwidth property. For every cluster $R \in \mathcal{R}$, $\mathcal{D}'(R)$ is a distribution over the external $R$-routers in $\Lambda'_G(R)$, and, for every edge $e \in E(G) \setminus E(R)$, $\mathbf{E}_{\mathcal{Q}'(R)\sim\mathcal{D}'(R)}\left[\mathrm{cong}_{G'}(\mathcal{Q}'(R), e)\right] \leq \beta$. Additionally, there is a distingiushed cluster $R^* \in \mathcal{R}$, whose corresponding distribution $\mathcal{D}'(R^*)$ is careful with respect to $v^*$, and $R^*$ contains at least $\left\lfloor \left(1 - 1/2^{(\log m)^{3/4}}\right)|\mathcal{C}| \right\rfloor$ clusters of $\mathcal{C}$.

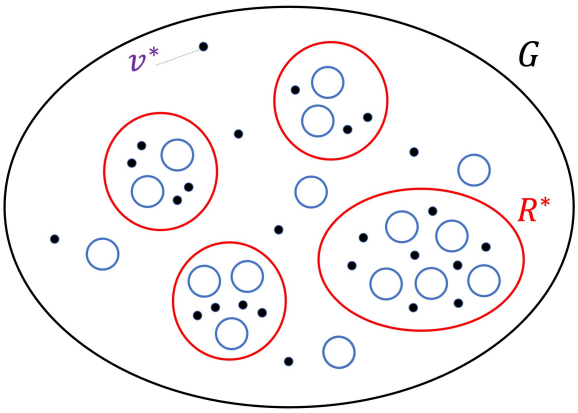
We denote by $\mathcal{C}'$ the set of all clusters $C \in \mathcal{C}$, such that $C \subseteq G \setminus \left(\bigcup_{R\in\mathcal{R}} V(R)\right)$. Observe that $\mathcal{R} \cup \mathcal{C}'$ is a set of mutually disjoint clusters of graph $G$ (see Figure 10(a)). It will be convenient for us to work with a slightly different contracted graph, that we denote by $\hat{H} = G_{|(\mathcal{R}\cup\mathcal{C}')}$. Note that every vertex $u \in V(\hat{H})$ that is different from a special vertex $v^*$, is either a regular vertex (that is, it is a vertex of $G$), or a supernode corresponding to a cluster of $\mathcal{C}' \cup \mathcal{R}$. If supernode $u$ represents a cluster of $\mathcal{C}'$, then we call it a $C$-node, and otherwise we call it an $R$-node (see Figure 10(b)). In order to prove Lemma 7.14, we will mostly work with graph $\hat{H}$. Note that $v^* \in V(\hat{H})$. We denote by $u^*$ the $R$-node representing the distinguished cluster $R^* \in \mathcal{R}$. We will maintain a collection $\mathcal{W}$ of clusters in graph $\hat{H}$, that we call $W$-clusters, and define next.

**Definition 7.15 (Valid set of $W$-clusters)** *A set $\mathcal{W}$ of disjoint clusters of graph $\hat{H}$ is a* valid set of $W$-clusters *if:*

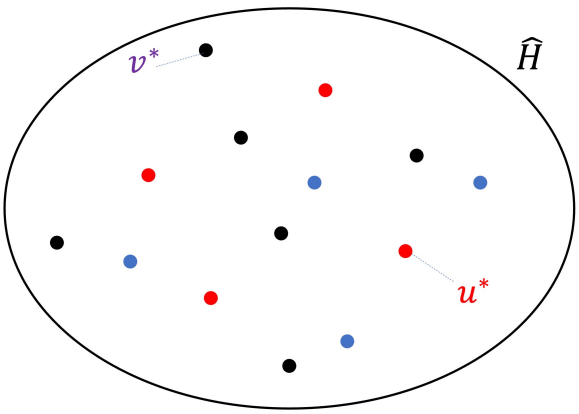
- *for every cluster $W \in \mathcal{W}$, every vertex of $W$ is an $R$-node or a regular vertex, and $W$ does not contain the special vertex $v^*$ or the $R$-node $u^*$ representing cluster $R^*$;*
- *for every cluster $W \in \mathcal{W}$, $|E_{\hat{H}}(W)| \geq |\delta_{\hat{H}}(W)|/(64\log m)$; and*
- *every cluster $W \in \mathcal{W}$ has the $\alpha'$-bandwidth property in graph $\hat{H}$, where $\alpha' = 1/(c\log^{2.5} m)$, for some large enough constant $c$.*

We will use the following lemma in order to prove Lemma 7.14.

**Lemma 7.16** *There is an efficient randomized algorithm, that, given a valid $W$-clustering $\mathcal{W}$ of graph $\hat{H}$, either (i) establishes that $(\mathcal{R}, \{\mathcal{D}'(R)\}_{R\in\mathcal{R}})$ is a type-2 legal clustering of $G$, by providing a nice witness structure for graph $G_{|\mathcal{R}}$, with respect to the set $\mathcal{C}''$ of clusters, containing every cluster $C \in \mathcal{C}$ with $C \subseteq G \setminus \left(\bigcup_{R\in\mathcal{R}} V(R)\right)$; or (ii) computes another helpful clustering $(\tilde{\mathcal{R}}, \{\mathcal{D}'(R)\}_{R\in\tilde{\mathcal{R}}})$ of*




(a) A schematic view of graph $G$. Clusters of $\mathcal{R}$ are shown in red, clusters of $\mathcal{C}$ are shown in blue, clusters of $\mathcal{C}'$ are the blue clusters that are disjoint from the red clusters, and vertices of $V(G) \setminus \left(\bigcup_{C \in \mathcal{C}} V(C)\right)$ are shown in black.




(b) A schematic view of graph $\hat{H}$. Regular vertices are shown in black, $C$-nodes are shown in blue and $R$-nodes are shown in red.

Figure 10: Graphs $G$ and $\hat{H}$.

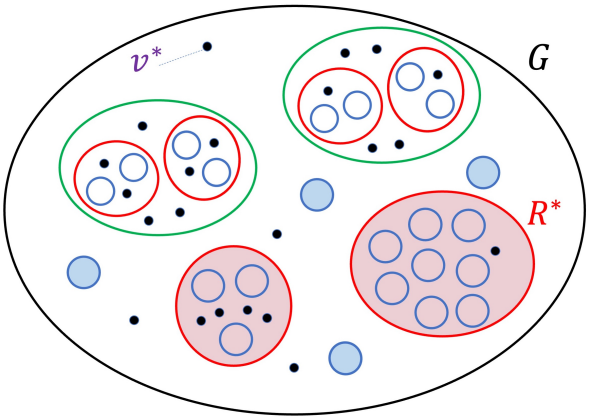
*graph $G$ with respect to vertex $v^*$ and set $\mathcal{C}$ of basic clusters, such that $R^* \in \mathcal{R}$ and $\tilde{\mathcal{R}}$ is a better clustering than $\mathcal{R}$; or (iii) computes a new valid set $\mathcal{W}'$ of $W$-clusters in the current graph $\hat{H}$, such that $|E(\hat{H}_{|\mathcal{W}'})| < |E(\hat{H}_{|\mathcal{W}})|$; or (iv) returns FAIL. The latter may only happen with probability at most $1/m^{11}$.*

Lemma 7.14 easily follows from Lemma 7.16. We start with $\mathcal{W} = \emptyset$, which is a valid set of $W$-clusters for $\hat{H}$, and then iterate. In every iteration, we apply the algorithm from Lemma 7.16 to the current valid set $\mathcal{W}$ of $W$-clusters. If the algorithm establishes that $\mathcal{R}$ is a type-2 legal clustering of $G$, then we terminate the algorithm and return the correpsonding witness structure for graph $G_{|\mathcal{R}}$. If the algorithm computes another helpful clustering $(\tilde{\mathcal{R}}, \{\mathcal{D}'(R)\}_{R \in \tilde{\mathcal{R}}})$ of graph $G$ with respect to vertex $v^*$ and set $\mathcal{C}$ of basic clusters with $R^* \in \mathcal{R}$, such that $\tilde{\mathcal{R}}$ is a better clustering than $\mathcal{R}$, then we terminate the algorithm and return the clustering $(\tilde{\mathcal{R}}, \{\mathcal{D}'(R)\}_{R \in \tilde{\mathcal{R}}})$. If the algorithm from Lemma 7.16 returns FAIL, then we terminate and algorithm and return FAIL as well. Otherwise, the algorithm from Lemma 7.16 computes a valid set $\mathcal{W}'$ of $W$-clusers in the current graph $\hat{H}$, such that $|E(\hat{H}_{|\mathcal{W}'})| < |E(\hat{H}_{|\mathcal{W}})|$. We then replace $\mathcal{W}$ with $\mathcal{W}'$ and continue to the next iteration. Since the number of edges in graph $\hat{H}_{|\mathcal{W}}$ decreases in every iteration, we are guaranteed that, after at most $m$ iterations the above algorithm terminates. Since the algorithm from Lemma 7.16 only returns FAIL with probability at most $1/m^{11}$, the total probability that our algorithm returns FAIL is at most $1/m^{10}$. From now on we focus on the proof of Lemma 7.16.

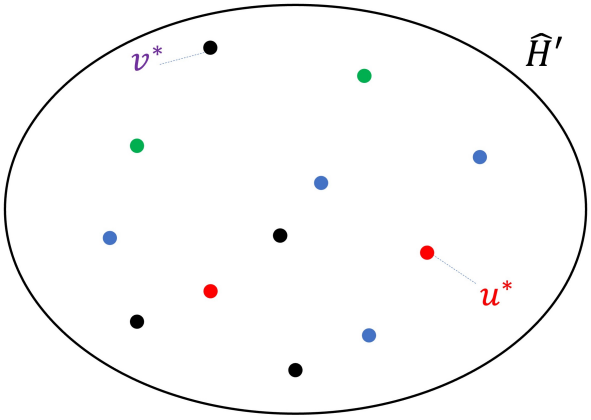
### 7.2.3 Proof of Lemma 7.16

Observe that so far, we have constructed a 3-level hierarchical clustering of the graph $G$. The first level consists of the set $\mathcal{C}$ of basic clusters of graph $G$. At the second level, there is a set $\mathcal{R}$ of clusters of graph $G$. Recall that, for every basic cluster $C \in \mathcal{C}$, either $C \subseteq G \setminus \left(\bigcup_{R \in \mathcal{R}} V(R)\right)$, or there is some cluster $R \in \mathcal{R}$, with $C \subseteq R$. As before, we denote by $\mathcal{C}' \subseteq \mathcal{C}$ the set of all basic clusters $C$ with $C \subseteq G \setminus \left(\bigcup_{R \in \mathcal{R}} V(R)\right)$. We can use the valid set $\mathcal{W}$ of $W$-clusters in graph $\hat{H}$, in order to construct another set $\mathcal{W}'$ of clusters in the original graph $G$, as follows. Recall that every cluster $W \in \mathcal{W}$ may only contain $R$-nodes or regular vertices of $\hat{H}$. For each such cluster $W$, let $\mathcal{R}(W) \subseteq \mathcal{R}$ be the set of all clusters $R \in \mathcal{R}$ with $v_R \in V(W)$. We then let $W'$ be a subgraph of $G$ induced by

vertex set $\left(\bigcup_{R\in\mathcal{R}(W)} V(R)\right) \cup (V(G)\cap V(W))$. In other words, $V(W')$ contains all regular vertices of $W$, and all vertices lying in clusters of $\mathcal{R}(W)$. We will refer to $W'$ as the *cluster of $G$ defined by $W$*. Finally, let $\mathcal{W}' = \{W' \mid W\in\mathcal{W}\}$. Note that every basic cluster $C\in\mathcal{C}'$ must be disjoint from clusters of $\mathcal{W}'$. We denote by $\mathcal{R}' = \mathcal{R}\setminus\left(\bigcup_{W\in\mathcal{W}}\mathcal{R}(W)\right)$. Note that each cluster $R\in\mathcal{R}'$ is contained in $G\setminus\left(\bigcup_{W'\in\mathcal{W}'} V(W')\right)$, while for each cluster $R\in\mathcal{R}\setminus\mathcal{R}'$, there is some cluster $W'\in\mathcal{W}'$ with $R\subseteq W'$. Therefore, $\mathcal{C}'\cup\mathcal{R}'\cup\mathcal{W}'$ is a collection of disjoint clusters of graph $G$ (see Figure 11(a)). Recall that we are guaranteed that every cluster $C\in\mathcal{C}'$ has the $\alpha_0$-bandwidth property, where $\alpha_0 = 1/\log^3 m$, and every cluster $R\in\mathcal{R}'$ has the $\alpha_1$-bandwidth property, where $\alpha_1 = 1/\log^6 m$. Lastly, every cluster $W\in\mathcal{W}$ has the $\alpha'$-bandwidth property (for $\alpha' = 1/(c\log^{2.5} m)$, where $c$ is a large enough constant) in graph $\hat{H}$. From Corollary 4.40, every cluster $W'\in\mathcal{W}'$ has the $\alpha_1\cdot\alpha' = 1/(c\log^{8.5} m)$-bandwidth property in graph $G$.




(a) A schematic view of graph $G$. Clusters of $\mathcal{W}'$ are shown in green. Clusters of $\mathcal{R}$ are shown in red (with clusters of $\mathcal{R}'$ shaded). Clusters of $\mathcal{C}$ are shown in blue (with clusters of $\mathcal{C}'$ shaded). Regular vertices lying outside of clusters of $\mathcal{C}$ are shown in black. Note that, if there exist clusters $C\in\mathcal{C}$ and $W'\in\mathcal{W}'$ with $C\subseteq W'$, then there exists a cluster $R\in\mathcal{R}$ with $C\subseteq R\subseteq W'$. Also, for every cluster $W'\in\mathcal{W}'$, $R^*\not\subseteq W'$ and $v^*\notin W'$ hold.




(b) A schematic view of graph $\hat{H}'$. Regular vertices are shown in black, $C$-nodes are shown in blue, $R$-nodes are shown in red, and $W$-nodes are shown in green.

Figure 11: Graphs $G$ and $\hat{H}'$.

In order to prove Lemma 7.14, it will be convenient for us to work with graph $\hat{H}'$, that is a contracted graph of $\hat{H}$, with respect to set $\mathcal{W}$ of clusters, that is, $\hat{H}' = \hat{H}_{|\mathcal{W}}$. Since graph $\hat{H}$ is itself a contracted graph of $G$ with respect to $\mathcal{R}\cup\mathcal{C}'$, it is easy to verify that $\hat{H}' = G_{|\mathcal{C}'\cup\mathcal{R}'\cup\mathcal{W}'}$ (see Figure 11(b)). The vertices of graph $\hat{H}'$ are partitioned into four types. The first type is regular vertices, which are also the vertices of the original graph $G$; note that the special vertex $v^*$ belongs to graph $\hat{H}$ as a regular vertex. The second type is supernodes corresponding to clusters of $\mathcal{C}'$, that we refer to as $C$-nodes. The third type is supernodes corresponding to clusters of $\mathcal{R}'$, that we call $R$-nodes, and it includes the vertex $u^*$, representing the cluster $R^*$. The fourth type is supernodes corresponding to clusters of $\mathcal{W}'$, that we call $W$-nodes. We denote the set of all regular vertices of $\hat{H}'$, excluding the special vertex $v^*$, by $U^*$. We denote the set of all $R$-nodes, excluding the vertex $u^*$, by $U^R$.

The remainder of the proof of Lemma 7.14 consists of three steps. In the first step, we perform some manipulations that will allow us to either compute a new valid set $\tilde{\mathcal{W}}$ of $W$-clusters in the current graph $\hat{H}$, such that $|E(\hat{H}_{|\tilde{\mathcal{W}}})| < |E(\hat{H}_{|\mathcal{W}})|$, or to organize the vertices of $U^*\cup U^R$ into a nice layered structure. We also define a collection $\mathcal{J}$ of clusters of the graph $\hat{H}'$ in this step. In the second step,

we will define another contracted graph $\check{H}$ with respect to the clustering $\mathcal{J}$, and explore some of its properties. In particular, we define the notion of a "simplifying cluster" in $\check{H}$, and show an algorithm that, given a simplifying cluster in $\check{H}$, produces a helpful clustering $\tilde{\mathcal{R}}$ of $G$ that is better than the current clustering $\mathcal{R}$. Lastly, in the third setp, we either compute a nice witness structure in graph $G_{|\mathcal{R}}$ as required, or compute a simplifying cluster in graph $\check{H}$, which in turn allows us to produce a helpful clustering $\tilde{\mathcal{R}}$ in graph $G$ that is better than $\mathcal{R}$. We now describe these steps one by one.

**Step 1: Layering the Vertices of $U^* \cup U^R$ and Clustering $\mathcal{J}$**

Consider the graph $\hat{H}'$, and let $C_0$ be the subgraph of $\hat{H}'$, induced by vertex set $V(\hat{H}') \setminus (U^* \cup U^R)$. We use the algorithm from Theorem 4.20, to compute a layered $\alpha'$-well-linked decomposition $(\mathcal{S}, (\mathcal{L}_1, \ldots, \mathcal{L}_h))$ of $\hat{H}'$ with respect to $C_0$, where $\alpha' = \Theta(1/\log^{2.5} m)$ is the parameter that was used in the definition of a valid set of $W$-clusters, such that $h \leq \log m$. We say that a cluster $S \in \mathcal{S}$ is a *singleton cluster*, if it contains a single vertex of $\hat{H}'$. Assume first that $\mathcal{S}$ contains a non-singleton cluster $S$. Recall that $S$ has the $\alpha'$-bandwidth property in $\hat{H}'$ (from Property L2 of layered well-linked decomposition; see Section 4.2.6), and it only contains verices of $U^* \cup U^R$. Therefore, $S$ is also a cluster of graph $\hat{H}$, and it has the $\alpha'$-bandwidth property in $\hat{H}$. Moreover, from Property L3 of the layered well-linked decomposition, $|E_{\hat{H}}(S)| = |E_{\hat{H}'}(S)| \geq |\delta_{\hat{H}'}(S)|/(64 \log m) = |\delta_{\hat{H}}(S)|/(64 \log m)$ holds. Since $S$ is disjoint from clusters in set $\mathcal{W}$, we get that $\tilde{\mathcal{W}} = \mathcal{W} \cup \{S\}$ is a valid set of $W$-clusters in graph $\hat{H}$, with $|E(\hat{H}_{|\tilde{\mathcal{W}}})| < |E(\hat{H}_{|\mathcal{W}})|$. We terminate the algorithm and return the new valid set $\tilde{\mathcal{W}}$ of $W$-clusters. Therfore, we will assume from now on that every cluster in set $\mathcal{S}$ is a singleton cluster. The partition $(\mathcal{L}_1, \ldots, \mathcal{L}_h)$ of the clusters of $\mathcal{S}$ into layers then immediately defines a partition $L_1, \ldots, L_h$ of vertices of $U^* \cup U^R$ into layers, where vertex $u$ lies in layer $U_i$ iff cluster $\{u\} \in \mathcal{S}$ lies in $\mathcal{L}_i$. For convenience, we denote by $L_0 = V(\hat{H}') \setminus (U^* \cup U^R)$. For every vertex $u \in U^* \cup U^R$ that lies in some layer $L_i$, for $1 \leq i \leq h$, we partition the edges of $\delta_{\hat{H}'}(u)$ into two subsets: set $\delta^{\text{down}}(u)$ contains all edges $(u, u')$ with $u' \in L_0 \cup L_1 \cup \cdots \cup L_{i-1}$, and set $\delta^{\text{up}}(u)$ contains all remaining edges of $\delta_{\hat{H}'}(u)$. Note that, from Property L4 of the layered well-linked decomposition, for every vertex $u \in U^* \cup U^R$, $|\delta^{\text{up}}(u)| < |\delta^{\text{down}}(u)|/\log m$.

In the remainder of the proof of Lemma 7.14, we will attempt to construct a nice witness structure for graph $G_{|\mathcal{R}}$, with respect to the set $\mathcal{C}'$ of clusters, containing every cluster $C \in \mathcal{C}$ with $C \subseteq G \setminus \left(\bigcup_{R \in \mathcal{R}} V(R)\right)$. If we fail to do so, then we will compute another helpful clustering $\tilde{\mathcal{R}}$ of graph $G$ with respect to vertex $v^*$ and set $\mathcal{C}$ of basic clusters, such that $R^* \in \mathcal{R}$ and $\tilde{\mathcal{R}}$ is a better clustering than $\mathcal{R}$.

In order to do so, we construct a collection $\mathcal{J}$ of clusters in graph $\hat{H}'$. Every cluster $J \in \mathcal{J}$ will contain exactly one vertex that is either a $C$-node or $W$-node, that we refer to as the *center of the cluster*, and possibly a number of additional vertices from $U^* \cup U^R$. Initially, for every vertex $u$ of $\hat{H}'$ that is either a $C$-node or a $W$-node, we construct a cluster $J(u) \in \mathcal{J}$, that only contains the vertex $u$ as its center node. We then iterate. As long as there exists a vertex $u' \in U^* \cup U^R$, such that at least $|\delta_{\hat{H}'}(u')|/128$ edges connect $u'$ to the vertices of some cluster $J \in \mathcal{J}$, we add vertex $u'$, together with all edges connecting $u'$ to $V(J)$, to cluster $J$. We also delete $u'$ from vertex set $U^*$ or $U^R$ in which it lies. If $u' \in L_i$, for some $1 \leq i \leq h$, then we delete $u'$ from $L_i$ and add it to $L_0$.

Consider the set $\mathcal{J}$ of clusters in graph $\hat{H}'$, that is obtained at the end of this procedure. It is immediate to verify that all clusters in $\mathcal{J}$ are mutually disjoint; every cluster $J \in \mathcal{J}$ contains a single center node that is a $C$-node or a $W$-node, and each remaining vertex of $J$ lies in $U^R \cup U^*$. We need the following observation, whose proof is deferred to Section G.5 of Appendix.

**Observation 7.17** *Every cluster $J \in \mathcal{J}$, has the $\Omega(1/\log m)$-bandwidth property in graph $\hat{H}'$.*

**Step 2: New Contracted Graph and Simplifying Clusters**

We start by revisiting the current hierarchical (4-level) clustering of $G$ and defining a new contracted graph. Recall that our starting point is a graph $G$, with a special vertex $v^*$, and a collection $\mathcal{C}$ of disjoint basic clusters in $G$, such that $v^*$ does not lie in any cluster of $\mathcal{C}$. Recall that every cluster in $\mathcal{C}$ has the $\alpha_0$-bandwidth property, where $\alpha_0 = 1/\log^3 m$. This is the first-level clustering.

The second level of clustering is the helpful clustering $\mathcal{R}$, which is also a collection of disjoint clusters, each of which has the $\alpha_1$-bandwidth property, where $\alpha_1 = 1/\log^6 m$. Recall that $v^*$ may not lie in any cluster of $\mathcal{R}$, and, for every cluster $C \in \mathcal{C}$, either $C \subseteq G \setminus \left(\bigcup_{R\in\mathcal{R}} R\right)$; or there is some cluster $R \in \mathcal{R}$ with $C \subseteq R$. Recall that we have denoted by $\mathcal{C}' \subseteq \mathcal{C}$ the set of all clusters $C \in \mathcal{C}$ with $C \subseteq G \setminus \left(\bigcup_{R\in\mathcal{R}} R\right)$. Recall also that we have defined a distinguished cluster $R^* \in \mathcal{R}$.

The third level of clustering is a $W$-clustering $\mathcal{W}$, that is defined with respect to the contracted graph $\hat{H} = G_{|\mathcal{C}'\cup\mathcal{R}}$. Recall that for every cluster $W \in \mathcal{W}$, every vertex of $W$ is either a regular vertex or an $R$-node, and $W$ may not contain the special vertex $v^*$ or the $R$-node $u^*$ representing the distinguished cluster $R^*$. We have defined, for every cluster $W \in \mathcal{W}$, the corresponding cluster $W' \subseteq G$, that is, intuitively, obtained from $W$ by un-contracting every cluster $R \in \mathcal{R}$ with $v_R \in V(W)$. We have then set $\mathcal{W}' = \{W' \mid W \in \mathcal{W}\}$, and we have established that every cluster $W' \in \mathcal{W}'$ has the $\Omega(1/\log^{8.5} m)$-bandwidth property in graph $G$. Observe that for every pair $C \in \mathcal{C}'$, $W' \in \mathcal{W}'$ of clusters, $C \cap W' = \emptyset$ must hold. For every pair $R \in \mathcal{R}$, $W' \in \mathcal{W}'$ of clusters, either $R \subseteq W'$, or $R \cap W'$ must hold. We denote by $\mathcal{R}'$ the set of all custers $R \in \mathcal{R}$ with $R \subseteq G \setminus \left(\bigcup_{W'\in\mathcal{W}'} W'\right)$. Observe that $R^* \in \mathcal{R}'$ must hold, and that $\mathcal{C}' \cup \mathcal{R}' \cup \mathcal{W}'$ is a collection of disjoint clusters in graph $G$. Graph $\hat{H}' = \hat{H}_{|\mathcal{W}}$ that we used in Step 1 is precisely the graph $G_{|\mathcal{C}'\cup\mathcal{R}'\cup\mathcal{W}'}$.

The fourth and the last level of clustering is defined by the collection $\mathcal{J}$ of clusters in graph $\hat{H}'$ that we have defined in Step 1. Recall that, for every cluster $J \in \mathcal{J}$ (that is a subgraph of $\hat{H}'$), there is a unique center vertex, that is either a $C$-node or a $W$-node, and the remaining vertices of $J$ are regular vertices or $R$-nodes; however, $J$ may not contain the special vertex $v^*$ or the $R$-node $u^*$ representing the distinguished cluster $R^*$. Moreover, every $C$-node and every $W$-node is a center of some cluster in $J$.

As before, we will define, for every cluster $J \in \mathcal{J}$, a corresponding cluster $J'$ in graph $G$, in a natural way. We first define the vertex set $V(J')$, and then let $J'$ be the subgraph of $G$ induced by $V(J')$. First, we add to $V(J')$ every regular vertex that lies in $J$ – each such vertex is a vertex of $G$. Next, for every $R$-node $v_R \in J$, we add all vertices of cluster $R$ to $V(J')$; observe that $R \in \mathcal{R}' \setminus \{R^*\}$ must hold. Lastly, we consider the unique center vertex of $J$. If that vertex is a $C$-node, corresponding to a cluster $C \in \mathcal{C}'$, then we add all vertices of $C$ to $V(J')$. Otherwise, the vertex is a $W$-node, representing some cluster $W' \in \mathcal{W}'$. We then add to $V(J')$ all vertices of $V(W')$. Lastly, we set $J' = G[V(J')]$. We denote by $\mathcal{J}' = \{J' \mid J \in \mathcal{J}\}$ the set of clusters in graph $G$ corresponding to the cluster set $\mathcal{J}$ in $\hat{H}'$. Observe that for every cluster $C \in \mathcal{C}'$, there is a unique cluster $J'(C) \in \mathcal{J}'$ containing $C$; we call $C$ the *center-cluster* of $J'(C)$. Similarly, for every cluster $W' \in \mathcal{W}'$, there is a unique cluster $J'(W) \in \mathcal{J}'$ containing $W'$; we similarly call $W'$ the *center-cluster* of $J'(W)$. Lastly, for every pair of clusters $R \in \mathcal{R}'$, $J' \in \mathcal{J}'$, either $R \subseteq J'$ or $R \cap J' = \emptyset$ holds. We denote by $\mathcal{R}'' \subseteq \mathcal{R}'$ the set of all custers $R \in \mathcal{R}'$, with $R \subseteq G \setminus \left(\bigcup_{J'\in\mathcal{J}} J'\right)$ (see Figure 12). Note that $R^* \in \mathcal{R}''$. Observe also that $\mathcal{R}'' \cup \mathcal{J}'$ defines a collection of disjoint clusters in graph $G$. Note that the special vertex $v^*$ does not lie in any cluster of $\mathcal{R}'' \cup \mathcal{J}'$, and that every cluster of $\mathcal{C}$, $\mathcal{R}$, and $\mathcal{W}'$ is contained in exactly one cluster of $\mathcal{R}'' \cup \mathcal{J}'$.

Since every cluster in $\mathcal{C}$ has the $\alpha_0 = 1/\log^3 m$-bandwidth property; every cluster $W \in \mathcal{W}'$ has the $\Omega(1/\log^{8.5} m)$-bandwidth property; and every cluster $R \in \mathcal{R}$ has the $1/\log^6 m$-bandwidth property in graph $G$, while, from Observation 7.17, every cluster $J \in \mathcal{J}$, has the $\Omega(1/\log m)$-bandwidth property in graph $\hat{H}' = G_{|\mathcal{C}'\cup\mathcal{R}'\cup\mathcal{W}'}$, from Corollary 4.40, we get that every cluster $J' \in \mathcal{J}'$ has the $\Omega(1/\log^{9.5} m)$-

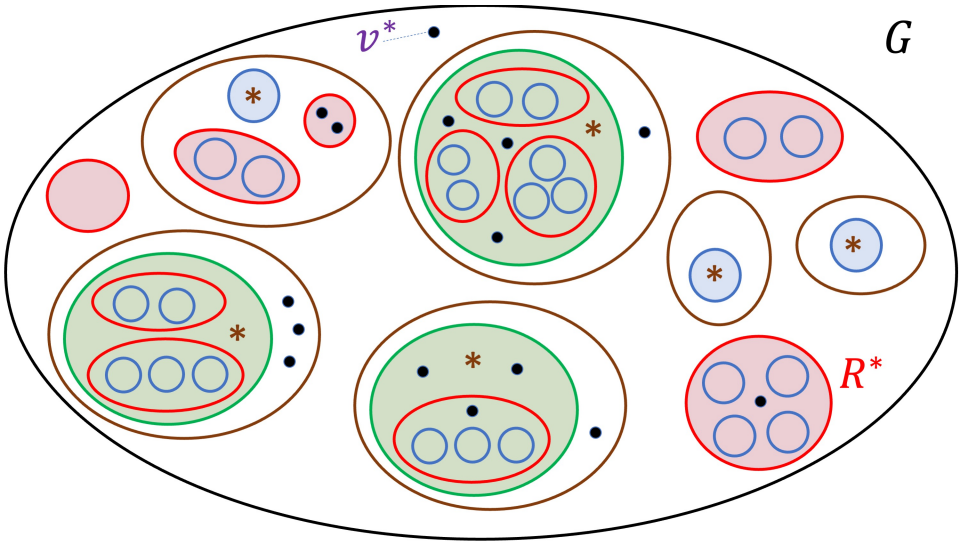



Figure 12: An illustration of a $\mathcal{J}$-clustering. Clusters of $\mathcal{C}$ are shown in blue (with clusters of $\mathcal{C}'$ shaded). Clusters of $\mathcal{R}$ are shown in red (with clusters of $\mathcal{R}'$ shaded). Clusters of $\mathcal{W}'$ are shown in green. Each cluster of $\mathcal{W}'$ may contain clusters of $\mathcal{R}$, but if a cluster $C \in \mathcal{C}$ is contained in $W'$, then there exists $R \in \mathcal{R}$ with $C \subseteq R \subseteq W'$. Vertices of $G$ that do not lie in clusters of $\mathcal{C}$ are shown in black. Clusters of $\mathcal{J}'$ are shown in brown. Each cluster of $\mathcal{J}'$ contains a cluster of $\mathcal{W}'$ or $\mathcal{C}'$ as its center cluster (indicated by $*$). Each cluster of $\mathcal{W}' \cup \mathcal{C}'$ is a center-cluster of some cluster of $\mathcal{J}'$. In addition to the center-cluster, a cluster of $\mathcal{J}'$ may contain clusters of $\mathcal{R}$ and regular vertices. Some clusters of $\mathcal{R}'$ and some regular vertices may not lie in any cluster of $\mathcal{J}'$.

bandwidth property. This property will be useful for us later, so we summarize it in the following observation.

**Observation 7.18** *Every cluster $J' \in \mathcal{J}'$ has the $\Omega(1/\log^{9.5} m)$-bandwidth property in graph $G$.*

In the remainder of the proof of Lemma 7.16 we consider the contracted graph $\breve{H} = G_{|\mathcal{J}' \cup \mathcal{R}''}$, which is exactly the contracted graph of $\hat{H}'$ with respect to cluster set $\mathcal{J}$, that is, $\breve{H} = \hat{H}'_{|\mathcal{J}}$. The set of vertices of $\breve{H}$ consists of three subsets: the set $V(G) \cap V(\breve{H})$ of regular vertices; the set $\{v_R \mid R \in \mathcal{R}''\}$ of supernodes corresponding to clusters of $\mathcal{R}''$ that we call $R$-nodes; and the set $\{v_{J'} \mid J' \in \mathcal{J}'\}$ of supernodes corresponding to clusters of $\mathcal{J}'$ that we call $J$-nodes. For convenience, we denote by $\hat{U}^*$ the set of all regular vertices of $\breve{H}$ excluding the special vertex $v^*$, and we denote by $\hat{U}^R$ the set of all $R$-nodes of $\breve{H}$ excluding the node $u^*$ that represents the distinguished cluster $R^*$. The algorithm from Step 1 ensures the following property of graph $\breve{H}$:

H1. for every vertex $u \in \hat{U}^* \cup \hat{U}^R$ and $J$-node $v_{J'}$, the number of edges connecting $u$ to $v_{J'}$ in $\breve{H}$ is at most $|\delta_{\breve{H}}(u)|/128$.

Indeed, if the above property does not hold for a vertex $u \in \hat{U}^* \cup \hat{U}^R$ and a $J$-node $v_{J'}$, then vertex $u$ should have been added to the cluster $J \in \mathcal{J}$ that corresponds to cluster $J' \in \mathcal{J}'$ by the algorithm that constructed the clusters in $\mathcal{J}$.

Additionally, the algorithm from Step 1 defines a partition $(L_1, L_2, \ldots, L_h)$ of the set $\hat{U}^* \cup \hat{U}^R$ of vertices, with $h \leq \log m$. Let $L_0$ be the set of vertices of $\breve{H}$ containing all $J$-nodes and the special vertices $v^*, u^*$; equivalently, $L_0 = V(\breve{H}) \setminus (\hat{U}^* \cup \hat{U}^R)$. Recall that for all $1 \leq i \leq h$, for every vertex $u \in L_i$, we have partitioned the edge set $\delta_{\breve{H}}(u)$ into two subsets: set $\delta^{\text{down}}(u)$ contains all edges

connecting $u$ to vertices of $L_0 \cup \cdots \cup L_{i-1}$, while set $\delta^{\text{up}}(u)$ contains all remaining edges of $\delta_{\check{H}}(u)$. Recall that we have also ensured that the following property holds:

H2. for every vertex $u \in \hat{U}^* \cup \hat{U}^R$, $|\delta^{\text{up}}(u)| < |\delta^{\text{down}}(u)|/\log m$.

Next, we define the notion of a simplifying cluster in graph $\check{H}$. We will then show that, given a simplifying cluster in $\check{H}$, we can efficiently compute a helpful clustering $\tilde{\mathcal{R}}$ of graph $G$ with respect to vertex $v^*$ and set $\mathcal{C}$ of basic clusters, such that $R^* \in \tilde{\mathcal{R}}$ and $\tilde{\mathcal{R}}$ is a better clustering than $\mathcal{R}$.

**Definition 7.19 (Simplifying Cluster)** *Let $S$ be a vertex-induced subgraph of $\check{H}$. We say that $S$ is a* simplifying cluster *if:*

- *vertices $v^*, u^*$ do not lie in $S$;*
- *there is a set $\mathcal{P}(S)$ of paths in graph $\check{H}$ (external $S$-router), routing the edges of $\delta_{\check{H}}(S)$ to a single vertex of $\check{H} \setminus S$, such that all paths in $\mathcal{P}(S)$ are internally disjoint from $S$, and they cause congestion at most $\beta' = O(\log m)$; and*
- *either $S$ contains at least one $J$-node, or $|E_{\check{H}}(S)| \geq |\delta_{\check{H}}(S)|/\log m$.*

We will use the following simple observation.

**Observation 7.20** *There is an efficient algorithm, that, given a cluster $S \subseteq \check{H}$, establishes whether $S$ is a simplifying cluster.*

**Proof:** In order to establish whether $S$ is a simplifying cluster, we need to check whether $S$ contains a $J$-node, or $|E_{\check{H}}(S)| \geq |\delta_{\check{H}}(S)|/\log m$ holds, which can be done efficiently. Additionally, we need to check whether there is a set of paths in graph $\check{H}$, routing the edges of $\delta_{\check{H}}(S)$ to a single vertex of $\check{H} \setminus S$, such that the paths are internally disjoint from $S$ and cause congestion at most $\beta'$. The latter can be done efficiently by computing maximum flow between the vertices of $S$ and each vertex of $\check{H} \setminus S$ in turn. □

In the next claim we show that, if we are given a simplifying cluster $S$ in $\check{H}$, then we can efficiently compute a helpful clustering $\tilde{\mathcal{R}}$ of graph $G$ with respect to $v^*$ and $\mathcal{C}$, such that $R^* \in \tilde{\mathcal{R}}$ and $\tilde{\mathcal{R}}$ is a better clustering than $\mathcal{R}$. The proof of the claim is somewhat technical and is deferred to Appendix G.6

**Claim 7.21** *There is an efficient algorithm, that, given a simplifying cluster $S$ of $\check{H}$, computes a helpful clustering $(\tilde{\mathcal{R}}, \{\mathcal{D}'(R)\}_{R \in \tilde{\mathcal{R}}})$ of graph $G$ with respect to the special vertex $v^*$ and the set $\mathcal{C}$ of basic clusters, such that $R^* \in \tilde{\mathcal{R}}$, and $\tilde{\mathcal{R}}$ is a better clustering than $\mathcal{R}$.*

Let $\tau$ be the Gomory-Hu tree of the graph $\check{H}$. We root the tree at the special vertex $v^*$, and, for every vertex $u \in V(\tau)$, we denote by $\tau_u$ the subtree of $\tau$ rooted at vertex $u$.

Assume first that there is some vertex $u \in V(\tau)$, such that the special $R$-node $u^*$ corresponding to the distinguished cluster $R^*$ does not lie in $\tau_u$, but some $J$-node $v_{J'}$ lies in $\tau_u$. In this case, we let $S$ be a subgraph of $\check{H}$ that is induced by $V(\tau_u)$. We claim that $S$ is a simplifying cluster. Indeed, from the construction, neither of $v^*, u^*$ may lie in $S$, and at least one $J$-node lies in $S$. Let $u'$ be the parent-vertex of $u$ in the tree $\tau$. Then from the properties of Gomory-Hu tree (see Corollary 4.10), $(V(S), V(\check{H}) \setminus V(S))$ is a minimum cut separating $u$ from $u'$ in $\check{H}$. From the max-flow / min-cut theorem, there is a collection $\mathcal{P}$ of $|\delta_{\check{H}}(S)|$ edge-disjoint paths connecting $u$ to $u'$ in $\check{H}$. Clearly, each edge $e \in \delta_{\check{H}}(S)$ is contained in exactly one path of $\mathcal{P}$, that we denote by $P(e)$. Let $P'(e)$ be the subpath of $P(e)$ that starts at edge $e$ and terminates at $u'$. Then $P'(e)$ must be internally disjoint

from $S$. Therefore, $\mathcal{P}(S) = \{P'(e) \mid e \in \delta_{\check{H}}(S)\}$ is a set of edge-disjoint paths in graph $\check{H}$, routing the edges of $\delta_{\check{H}}(S)$ to vertex $u' \in \check{H} \setminus S$, and the paths in $\mathcal{P}(S)$ are internally disjoint from $S$. We conclude that $S$ is a simplifying cluster. We can now use the algorithm from Claim 7.21 to compute a helpful clustering $(\tilde{\mathcal{R}}, \{\mathcal{D}'_R\}_{R\in\tilde{\mathcal{R}}})$ of graph $G$ with respect to the special vertex $v^*$ and the set $\mathcal{C}$ of basic clusters, such that $R^* \in \tilde{\mathcal{R}}$, and $\tilde{\mathcal{R}}$ is a better clustering than $\mathcal{R}$.

Therefore, we assume from now on that, for every vertex $u \in V(\tau)$, if $u^*$ does not lie in $\tau_u$, then $\tau_u$ does not contain any $J$-node, and so $V(\tau_u) \subseteq \hat{U}^* \cup \hat{U}^R$. Let $P^*$ denote the path connecting $v^*$ to $u^*$ in the tree $\tau$. We denote the sequence of vertices on the path by $v^* = u_1, u_2, \ldots, u_r = u^*$. For all $1 \leq i \leq r$, we define a cluster $S_i$ of $\check{H}$, associated with vertex $u_i$, as follows. We let $S_r$ be the subgraph of $\check{H}$ induced by the vertices of $\tau_{u_r}$. Consider now some index $1 \leq i < r$. Let $\left\{x_i^0, x_i^1, x_i^2, \ldots, x_i^{q_i}\right\}$ be the set of all child-vertices of $u_i$ in the tree $\tau$, and assume that $x_i^0 = u_{i+1}$. We then let $S_i$ be the subgraph of $\check{H}$ induced by the set $\{u_i\} \cup V(\tau_{x_i^1}) \cup V(\tau_{x_i^2}) \cup \cdots \cup V(\tau_{x_i^{q_i}})$ of vertices. In other words, we include in vertex set $V(S_i)$ the vertices lying in all subtrees of the children of $u_i$, except for the vertices lying in the subtree of $u_{i+1}$. From our assumption, for all $1 \leq i \leq r$, the only vertex of $S_i$ that may be a $J$-node is the vertex $u_i$; all other vertices of $S_i$ are $R$-nodes or regular vertices (and it is also possible that $u_i$ is an $R$-node or a regular vertex).

For all $1 \leq i \leq r$, we also define a subgraph $S'_i \subseteq S_i$, that is constructed as follows. We start by constructing the set $V(S'_i)$ of vertices. Initially, we let $V(S'_i) = \{u_i\}$. While there is any vertex $u \in S_i \setminus S'_i$, such that at the number of edges connecting $u$ to vertices of $V(S'_i)$ is at least $|\delta_{\check{H}}(u)|/128$, then we add $u$ to $V(S'_i)$. Once this algorithm terminates, we let $S'_i$ be the subgraph of $\check{H}$ induced by the set $V(S'_i)$ of vertices. Recall that we have established that, if $v$ is a vertex of $S_i \setminus S'_i$, for some $1 \leq i \leq r$, then $v \in \hat{U}^* \cup \hat{U}^R$ must hold. The following observation easily follows from the construction of $J$-clusters.

**Observation 7.22** *Consider any index $1 < i < r$, for which $u_i$ is a $J$-node. Then $S'_i = \{u_i\}$.*

**Proof:** Let $J \in \mathcal{J}$ be the cluster of $\hat{H}'$ that node $u_i$ represents (recall that we can think of graph $\check{H}$ as a contracted graph of $\hat{H}'$ with respect to cluster set $\mathcal{J}$). Assume for contradiction that $S'_i$ contains at least one vertex in addition to $u_i$, and let $v$ be the first vertex that was added to cluster $S'_i$. Then the number of edges connecting $v$ to $u_i$ is at least $|\delta_{\check{H}}(v)|/128$. But then $v$ is also a vertex of graph $\hat{H}'$, in which it serves as either an $R$-node distinct from $u^*$, or a regular vertex distinct from $v^*$. Moreover, the number of edges connecting $v$ to vertices of $J$ is at least $|\delta_{\hat{H}'}(v)|/128$. Therefore, $v$ should have been added to cluster $J$ when it was constructed, a contradiction. □

Additionally, we get the following observation, whose proof is identical to the proof of Observation 7.17 and is omitted here.

**Observation 7.23** *For all $1 \leq i \leq r$, cluster $S'_i$ has the $\Omega(1/\log m)$-bandwidth property in graph $\check{H}$.*

For all $1 \leq i \leq r$, we employ the algorithm from Observation 7.20 in order to establish whether $S'_i$ is a simplifying cluster. Additionally, for all $1 \leq i < r$, we use the algorithm from Observation 7.20 in order to establish whether the subgraph of $\check{H}$ induced by vertex set $V(S_i) \cup V(S_{i+1})$ is a simplifying cluster. If the algorithm from Observation 7.20 establishes that any of the above clusters is a simplifying cluster, then we can use the algorithm from Claim 7.21 to compute a helpful clustering $(\tilde{\mathcal{R}}, \{\mathcal{D}'(R)\}_{R\in\tilde{\mathcal{R}}})$ of graph $G$ with respect to the special vertex $v^*$ and the set $\mathcal{C}$ of basic clusters, such that $R^* \in \tilde{\mathcal{R}}$, and $\tilde{\mathcal{R}}$ is a better clustering than $\mathcal{R}$. Therefore, we assume from now on that, for all $1 \leq i \leq r$, cluster $S'_i$ is not a simplifying cluster, and for all $1 \leq i < r$, the subgraph of $\check{H}$ induced by $V(S_i) \cup V(S_{i+1})$ is not a simplifying cluster.

We will show, in Step 3, an efficient algorithm that constructs a nice witness structure for graph $G_{|\mathcal{R}}$, with respect to the set $\mathcal{C}'$ of clusters, that contains every cluster $C \in \mathcal{C}$ with $C \subseteq G \setminus \left(\bigcup_{R\in\mathcal{R}} V(R)\right)$.

**Step 3: Constructing the Nice Witness Structure**

The goal of this step is to construct a nice witness structure for graph $G_{|\mathcal{R}}$, with respect to the set $\mathcal{C}''$ of clusters, that contains every cluster $C \in \mathcal{C}$ with $C \subseteq G \setminus \left(\bigcup_{R \in \mathcal{R}} V(R)\right)$.

Intuitively, we will use the clusters $S_1, \ldots, S_r$ that we just defined in order to define the spine $\tilde{\mathcal{S}} = \left\{\tilde{S}_1, \ldots, \tilde{S}_r\right\}$ of the nice witness structure in the natural way: cluster $\tilde{S}_i$ will be obtained from $S_i$ by first replacing every $R$-node and every $J$-node of $S_i$ with the corresponding cluster $R \in \mathcal{R}''$ or $J' \in \mathcal{J}'$, and then contracting the $R$-clusters back. Similarly, we will use the clusters $S'_1, \ldots, S'_r$ in order to define the verterbrae $\tilde{S}'_1, \ldots, \tilde{S}'_r$ of the nice witness structure.

We partition the set $E(\check{H})$ of edges into two disjoint subsets, $E'$ and $E''$, as follows. Set $E'$ contains all edges of $\bigcup_{i=1}^{r} E(S'_i)$, and, additionally, for all $1 \leq i < r$, it contains every edge $e = (u, v)$ with $u \in S'_i$, $v \in S'_{i+1}$. Set $E''$ contains all remaining edges of $E(\check{H})$. Additionally, we let $\hat{E} \subseteq E''$ be the set of all edges $(u, v) \in E''$, where $u$ and $v$ lie in different sets of $\{S_1, \ldots, S_r\}$.

Next, we develop some tools that will allow us to define the set $\mathcal{P} = \left\{P(e) \mid e \in \hat{E}\right\}$ of nice guiding paths for the nice witness structure that we construct. Recall that in Step 1 of the algorithm, we have partitioned the set $U^* \cup U^R$ of vertices of graph $\hat{H}'$ into layers $L_1, \ldots, L_h$, where $h \leq \log m$. Recall that $U^*$ is the set of all regular vertices (excluding $v^*$), and $U^R$ is the set of all $R$-nodes (excluding $u^*$) of graph $\hat{H}'$. Recall that $\check{H} = \hat{H}'_{|\mathcal{J}}$, that is, graph $\check{H}$ can be obtained from $\hat{H}'$ by contracting all clusters of $\mathcal{J}$. Therefore, if we denote by $\hat{U}^*$ the set of all regular vertices of $\check{H}$ (excluding $v^*$), and by $\hat{U}^R$ the set of all $R$-nodes of $\check{H}$ (excluding $u^*$), then $\hat{U}^* \subseteq U^*$, and $\hat{U}^R \subseteq U^R$. Therefore, partition $(L_1, \ldots, L_h)$ of $U^* \cup U^R$ naturally defines a partition $(L'_1, \ldots, L'_h)$ of $\hat{U}^R \cup \hat{U}^*$. Recall that, for all $1 \leq i \leq r$, all vertices of $S_i \setminus S'_i$ lie in $\hat{U}^* \cup \hat{U}^R$. We denote by $L'_0 = V(\check{H}) \setminus \left(\bigcup_{j=1}^{h} L'_j\right)$. As before, for all $1 \leq j \leq r$, for every vertex $v \in L'_j$, we partition the set $\delta_{\check{H}}(v)$ of edges into two subsets: set $\delta^{\text{down}}(v)$ containing all edges that connect $v$ to vertices of $L'_0 \cup \cdots \cup L'_{j-1}$, and set $\delta^{\text{up}}(v)$ containing all remaining edges incident to $v$. From Property H2 of graph $\check{H}$, for every vertex $v \in \hat{U}^* \cup \hat{U}^R$, $|\delta^{\text{up}}(v)| < |\delta^{\text{down}}(v)|/\log m$.

For all $1 \leq i \leq r$ and $1 \leq j \leq h$, we denote by $U_{i,j} = L'_j \cap V(S_i)$ – the set of all vertices of $S_i$ that lie in layer $L'_j$.

Consider some pair $1 \leq i \leq r$, $1 \leq j \leq h$ of indices, and some vertex $v \in U_{i,j} \setminus \{u_i\}$. We partition the edges of $\delta^{\text{down}}(v)$ into four subsets, $\delta^{\text{down,left}}(v), \delta^{\text{down,right}}(v)$, $\delta^{\text{down,straight}'}(v)$, and $\delta^{\text{down,straight}''}(v)$, as follows. Let $e = (u, v)$ be an edge of $\delta^{\text{down}}(v)$, and assume that $u \in U_{i',j'}$. Since $e \in \delta^{\text{down}}(v)$, $j' < j$ must hold. If, additionally, $i' < i$ holds, then we add $e$ to $\delta^{\text{down,left}}(v)$, Similarly, if $i' > i$, then we add $e$ to $\delta^{\text{down,right}}(v)$. If $i' = i$, and $u \in S'_i$, then $e$ is added to $\delta^{\text{down,straight}''}(v)$, and otherwise it is added to $\delta^{\text{down,straight}'(v)}$. We will use the following simple observation, whose proof appears in Appendix G.7.

**Observation 7.24** *$S'_1 = S_1$, and $S'_r = S_r$. Additionally, for every vertex $v \in V(\check{H}) \setminus (\bigcup_{i=1}^{r} S'_i)$:*

- $|\delta^{\text{down,right}}(v)| + |\delta^{\text{down,left}}(v)| + |\delta^{\text{down,straight}'}(v)| \geq 63|\delta(v)|/64$;
- $|\delta^{\text{down,left}}(v)| \leq 2(|\delta^{\text{down,right}}(v)| + |\delta^{\text{down,straight}'}(v)|)$; *and*
- $|\delta^{\text{down,right}}(v)| \leq 2(|\delta^{\text{down,left}}(v)| + |\delta^{\text{down,straight}'}(v)|)$.

We will also use the following simple observation, whose proof appears in Appendix G.8

**Observation 7.25** *There is an efficient algorithm that defines, for every vertex $v \in V(\check{H}) \setminus (\bigcup_{i=1}^{r} S'_i)$, two mappings: mapping $f^{\text{right}}(v)$, that maps every edge of $\delta^{\text{down,straight}''}(v) \cup \delta^{\text{up}}(v)$ to a distinct edge of*

$\delta^{\text{down,right}}(v) \cup \delta^{\text{down,straight}'}(v)$, *and another mapping* $f^{\text{left}}(v)$, *that maps every edge of* $\delta^{\text{down,straight}''}(v) \cup \delta^{\text{up}}(v)$ *to a distinct edge of* $\delta^{\text{down,left}}(v) \cup \delta^{\text{down,straight}'}(v)$.

Next, we define the notion of a *left-monotone* and a *right-monotone* path.

**Definition 7.26 (Left-Monotone and Right-Monotone Paths)** *Let* $P = (x_1, x_2, \ldots, x_q)$ *be a path in graph* $\check{H}$. *For all* $1 \leq a \leq q$, *assume that* $x_a \in U_{i_a, j_a}$. *We say that path* $P$ *is* left-monotone *if either* $q = 1$ *(that is,* $P$ *consists of a single vertex), or all of the following conditions holds:*

*M1.* $j_1 > j_2 > \cdots > j_q$;

*M2. for all* $1 \leq a < q$, *vertex* $x_a \in S_{i_a} \setminus S'_{i_a}$, *and* $x_q \in S'_{i_q}$; *and*

*M3.* $i_1 \geq i_2 \geq \cdots \geq i_q$, *and* $i_q < i_1$.

*Similarly, we say* $P$ *is* right-monotone *if either* $q = 1$, *or properties M1 and M2 hold for it, together with the following property:*

*M'3.* $i_1 \leq i_2 \leq \cdots \leq i_q$, *and* $i_q > i_1$

Observe that the vertices on a left-monotone path must appear in the decreasing order of their layers, and in the non-increasing order of the sets $S_i$ to which they belong. Similarly, vertices on a right-monotone path appear in the decreasing order of their layers, and in the non-decreasing order of the sets $S_i$ to which they belong. The following lemma will allow us to construct prefix- and suffix-paths for each edge $e \in \hat{E}$, by constructing a left-monotone and a right-monotone path for each such edge in graph $\check{H}$; the proof is deferred to Appendix G.9.

**Lemma 7.27** *There is an efficient algorithm that constructs, for every edge* $e = (u, v) \in \hat{E}$ *two paths* $P(e, u)$ *and* $P(e, v)$ *in graph* $\check{H}$, *such that, if* $u \in S_i$, $v \in S_{i'}$, *and* $i < i'$, *then path* $P(e, u)$ *is left-monotone and path* $P(e, v)$ *is right-monotone. Moreover, the set* $\left\{P(e, v), P(e, u) \mid e = (u, v) \in \hat{E}\right\}$ *of paths causes congestion* $O(\log m)$.

Consider now some index $1 \leq i < r$. We let $\hat{E}_i \subseteq \hat{E}$ contain all edges $e = (u, v) \in \hat{E}$, such that, if $u \in S_{i'}$, $v \in S_{i''}$, and $i' < i''$, then $i' \leq i$ and $i'' \geq i + 1$ must hold. We also denote by $E_i \subseteq E'$ the set of all edges $e = (u, v)$ with $u \in S'_i$ and $v \in S'_{i+1}$ (see Figure 13). Note that $E_i \cap \hat{E}_i = \emptyset$. The next lemma is crucial to the algorithm for constructing a nice witness structure in graph $G_{|\mathcal{R}}$.

**Lemma 7.28** *For all* $1 < i < r$, *vertex* $u_i$ *is a J-node, and for all* $1 \leq i < r$, $|\hat{E}_i| \leq 1000 \cdot |E_i|$.

The proof of Lemma 7.28 is deferred to Section 7.2.4.

From now on, we denote $H = G_{|\mathcal{R}}$, and we denote by $\mathcal{C}' \subseteq \mathcal{C}$ the set of all basic clusters $C \in \mathcal{C}$ with $C \subseteq G \setminus \left(\bigcup_{R \in \mathcal{R}} V(R)\right)$; equivalently, $\mathcal{C}'$ contains every cluster $C \in \mathcal{C}$ that is contained in $H$. It now remains to construct a nice witness structure in graph $H$ with respect to the set $\mathcal{C}'$ of clusters. We start by constructing the backbone and the vertebrae of the witness structure, and by defining the partition $(\tilde{E}', \tilde{E}'')$ of the edges of $H$. We then construct the prefix and the suffix of path $P(e)$ for each edge $e \in \hat{E}$. Lastly, we construct the mid-segment of each such path.

**The Backbone and the Vertebrae of the Witness Structure.** We use the clusters in set $\mathcal{S} = \{S_1, \ldots, S_r\}$, and in set $\mathcal{S}' = \{S'_1, \ldots, S'_r\}$ in order to define the backbone and the vertebrae of the nice witness structure, respectively. Recall that every vertex of graph $\check{H}$ is either a regular vertex

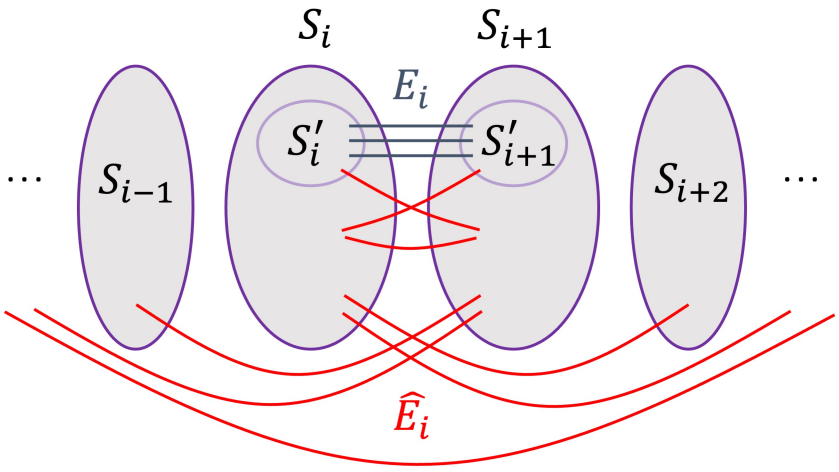



Figure 13: An illustration of edge sets $E_i$ and $\hat{E}_i$.

(that is, it lies in both $G$ and $H$); or it is an $R$-node $v_R$ representing some cluster $R \in \mathcal{R}$ (in which case it lies in $H = G_{|\mathcal{R}}$); or it is a $J$-node $v_{J'}$ for some cluster $J' \in \mathcal{J}'$. As we have established, for all $1 < i < r$, vertex $u_i$ is a $J$-node, and all other vertices of $\check{H}$ are either regular vertices of $R$-nodes.

Consider now any such $J$-node $v_{J'}$, and the corresponding cluster $J' \in \mathcal{J}'$. Recall that for every cluster $R \in \mathcal{R}$, either $R \subseteq J'$, or $R \cap J' = \emptyset$ holds. Denote by $\mathcal{R}(J') \subseteq \mathcal{R}$ the set of all clusters $R \in \mathcal{R}$ with $R \subseteq J'$. Let $J'' = J'_{|\mathcal{R}(J')}$ be the graph obtained from $J'$ by contracting every cluster $R \in \mathcal{R}(J')$ into a supernode. Then $J'' \subseteq H$, and we will think of $J''$ as the cluster of $H$ that vertex $v_{J'} \in V(\check{H})$ represents. We denote by $\mathcal{J}'' = \{J'' \mid J' \in \mathcal{J}'\}$ the resulting set of clusters in graph $H$. Note that equivalently we could define the graph $\check{H}$ as a graph that is obtained from $H$ by contracting every cluster in $\mathcal{J}''$, that is, $\check{H} = H_{|\mathcal{J}''}$. Recall that we have established, in Observation 7.18, that every cluster $J' \in \mathcal{J}'$ has the $\Omega(1/\log^{9.5} m)$-bandwidth property in graph $G$. It then immediately follows that every cluster $J'' \in \mathcal{J}''$ has the $\Omega(1/\log^{9.5} m)$-bandwidth property in graph $H$.

We start by defining the sequence $\tilde{\mathcal{S}}' = \left\{\tilde{S}'_1, \ldots, \tilde{S}'_r\right\}$ of the vertebrae of the nice witness structure. Consider an index $1 \leq i \leq r$. If $i = 1$, then $u_1 = v^*$, and so every vertex of set $S'_1$ is a vertex of $H$. We then set $\tilde{S}'_1 = S'_1$. If $i = r$, then $u_r = u^*$. As before, every vertex of $S'_r$ is then a vertex of $H$, and we set $\tilde{S}'_r = S'_r$. Lastly, assume that $1 < i < r$. From Lemma 7.28, $u_i$ is a $J$-node, and, from Observation 7.22, $S'_i = \{u_i\}$. Assume that $u_i = v_{J'}$, where $J' \in \mathcal{J}'$. We then let $\tilde{S}'_i$ be the cluster $J''$ of $H$ that corresponds to vertex $v_{J'}$. This completes the definition of the sequence $\tilde{\mathcal{S}}' = \left\{\tilde{S}'_1, \ldots, \tilde{S}'_r\right\}$ of the vertebrae of the nice witness structure. Consider again some index $1 \leq i \leq r$. If $i \in \{1, r\}$, then, from the construction, for every cluster $C \in \mathcal{C}'$, $C \cap \tilde{S}'_i = \emptyset$. This is because every cluster of $\mathcal{C}'$ must be contained in some cluster of $\mathcal{J}'$. Otherwise, assume that $u_i = v_{J'}$, where $J' \in \mathcal{J}'$. If the center-cluster of $J'$ is a basic cluster $C \in \mathcal{C}'$, then $C \subseteq \tilde{S}'_i$, and for every other cluster $C' \in \mathcal{C}'$, $C \cap \tilde{S}'_i = \emptyset$. Otherwise, the center-cluster of $J'$ is a cluster $W' \in \mathcal{W}'$. In this case, no cluster of $\mathcal{C}'$ may be contained in $\tilde{S}'_i$.

Consider again some index $1 \leq i \leq r$. Recall that we have established, in Observation 7.23, that cluster $S'_i$ has the $\Omega(1/\log m)$-bandwidth property in graph $\check{H}$. We have also established above that every cluster in $J'' \in \mathcal{J}''$ has the $\Omega(1/\log^{9.5} m)$-bandwidth property. From Corollary 4.40, cluster $\tilde{S}'_i$ of $H$ has the $\Omega(1/\log^{10.5} m) \geq \alpha^*$-bandwidth property, since $\alpha^* = \Omega(1/\log^{12} m)$. To conclude, we have shown that, for all $1 \leq i \leq r$, cluster $\tilde{S}'_i$ of $H$ has the $\alpha^*$-bandwidth property. We have also shown that, for all $1 \leq i \leq r$, there is at most one cluster $C \in \mathcal{C}'$ with $C \subseteq \tilde{S}'_i$. It is easy to verify

that, if such cluster $C$ exists, then $E(\tilde{S}'_i) \subseteq E(C) \cup E(G_{|\mathcal{C}'})$, and otherwise $E(\tilde{S}'_i) \subseteq E(G_{|\mathcal{C}'})$. This is because for every cluster $C \in \mathcal{C}'$, and for all $1 \leq i \leq r$, either $C \subseteq \tilde{S}'_i$, or $C \cap \tilde{S}'_i = \emptyset$ holds.

We now define the backbone $\tilde{\mathcal{S}} = \left\{\tilde{S}_1, \ldots, \tilde{S}_r\right\}$ of the nice witness structure. Fix an index $1 \leq i \leq r$. If $i \in \{1, r\}$, then, from Observation 7.24, $S'_i = S_i$. We then set $\tilde{S}_i = \tilde{S}'_i$. Assume now that $1 < i < r$. Recall that in this case, $S'_i = \{u_i\}$ holds, and $u_i$ is a $J$-node, from Lemma 7.28. Note that every vertex of $S_i \setminus \{u_i\}$ is either a regular vertex or an $R$-node, and so it must lie in graph $H$. We define the set $V(\tilde{S}_i)$ of vertices to contain all regular vertices and all $R$-nodes that lie in $S_i \setminus \{u_i\}$, and all vertices of $\tilde{S}'_i$. We then let $\tilde{S}_i$ be the subgraph of $H$ induced by the set $V(\tilde{S}_i)$ of vertices. In other words, we can think of cluster $\tilde{S}_i$ as being obtained from cluster $S_i$ of $\breve{H}$, by un-contracting the $J$-node $u_i$ (into the corresponding cluster of $\mathcal{J}''$). This completes the definition of the backbone of the nice witness structure. Since every vertex of $S_i \setminus S'_i$ is either a regular vertex or an $R$-node, either there is a single cluster $C \in \mathcal{C}'$ with $C \subseteq \tilde{S}'_i$, in which case then $E(\tilde{S}_i) \subseteq E(C) \cup E(G_{|\mathcal{C}'})$; or no such cluster exists, in which case $E(\tilde{S}_i) \subseteq E(G_{|\mathcal{C}'})$. Since vertex sets $V(S_1), \ldots, V(S_r)$ partition $V(\breve{H})$, it is easy to verify that vertex sets $V(\tilde{S}_1), \ldots, V(\tilde{S}_r)$ partition $V(H)$.

Recall that the second ingredient of the nice witness structure is a partition of the edges of $E(H)$ into two disjoint subsets, $\tilde{E}'$ and $\tilde{E}''$, that are defined as follows. Set $\tilde{E}'$ contains all edges of $\bigcup_{i=1}^{r} E(\tilde{S}'_i)$, and, additionally, for all $1 \leq i < r$, it contains every edge $e = (u, v)$ with $u \in \tilde{S}'_i$, $v \in \tilde{S}'_{i+1}$. Since, as observed already, $\breve{H} = H_{|\mathcal{J}''}$, it is easy to verify that $E' \subseteq \tilde{E}'$. Recall that we have denoted, for all $1 \leq i < r$, by $E_i \subseteq E'$ the set of all edges $e = (u, v)$ of $\breve{H}$ with $u \in S'_i$ and $v \in S'_{i+1}$. It is easy to verify that $E_i \subseteq E(H)$, and moreover, it is precisely the set of all edges $(u, v)$ in $H$ with $u \in \tilde{S}'_i$ and $v \in \tilde{S}'_{i+1}$. In particular, $E_i \subseteq \tilde{E}'$. The second edge set in the partition of $E(H)$ contains all remaining edges, $\tilde{E}'' = E(H) \setminus \tilde{E}'$. From the fact that $\breve{H} = H_{|\mathcal{J}''}$, and since, for all $1 \leq i \leq r$, $S_i \setminus S'_i$ may only contain regular vertices or $R$-nodes, we get that $E'' = \tilde{E}''$ holds. Lastly, we defined the set $\hat{E} \subseteq E''$ of all edges $e = (u, v) \in E''$ of graph $\breve{H}$, where $u$ and $v$ lie in different clusters of $\{S_1, \ldots, S_r\}$. It is immediate to verify that this is exactly the set of edges in graph $\breve{H}$, containing all edges $e = (u, v) \in \tilde{E}''$ where $u$ and $v$ lie in different clusters of $\left\{\tilde{S}_1, \ldots, \tilde{S}_r\right\}$. Recall that we have defined, for all $1 \leq i < r$, the set $\hat{E}_i \subseteq \hat{E}$ of edges in graph $\breve{H}$, that contains all edges $e = (u, v) \in \hat{E}$, such that, if $u \in S_{i'}$ and $v \in S_{i''}$ with $i' < i''$, then $i' \leq i$ and $i'' \geq i + 1$ hold. It is easy to verify that $\hat{E}_i$ is also precisely the set of all edges $e = (u, v) \in \hat{E}$ in graph $H$, such that, if $u \in \tilde{S}_{i'}$ and $v \in \tilde{S}_{i''}$ with $i' < i''$, then $i' \leq i$ and $i'' \geq i + 1$ hold. As before, $E_i \cap \hat{E}_i = \emptyset$.

In order to complete the construction of the nice witness structure, it now remains to define the paths in set $\hat{\mathcal{P}} = \left\{P(e) \mid e \in \hat{E}\right\}$. Recall that each such path $P(e)$ consists of three subpaths, prefix $P^1(e)$, suffix $P^3(e)$, and mid-segment $P^2(e)$. We first construct the prefixes and the suffixes of the paths in $\hat{\mathcal{P}}$, and then construct the mid-segment of each such path.

**Prefixes and Suffixes of Paths in $\hat{\mathcal{P}}$.** Consider an edge $e = (u, v) \in \hat{E}$ in graph $H$. Assume that $u \in \tilde{S}_i$, $v \in \tilde{S}_{i'}$, and $i < i'$. We now define vertices $u', v'$ of graph $\breve{H}$ that correspond to $u$ and $v$. If $u$ is also a vertex of cluster $S_i$ in $\breve{H}$, then we set $u' = u$. Otherwise, $u_i$ must be a $J$-node corresponding to some cluster $J' \in \mathcal{J}'$, with vertex $u$ lying in the corresponding cluster $J'' \in \mathcal{J}''$. In this case, we set $u' = u_i$. We define vertex $v'$ in graph $\breve{H}$, that corresponds to vertex $v$ in graph $H$ similarly, so $v' \in S_{i'}$. Observe that $(u', v')$ is an edge of $\breve{H}$, that lies in the edge set $\hat{E}$, and it corresponds to edge $e$ in $H$; we do not distinguish between the two edges. Consider now the left-monotone path $P(e, u')$ in graph $\breve{H}$ given by Lemma 7.27, and denote $P(e, u') = (u' = x_1, x_2, \ldots, x_q)$. For all $1 \leq a \leq q$, assume that $x_a \in U_{i_a, j_a}$. Recall that, from the definition of the left-monotone path, $i_1 \geq i_2 \geq \cdots \geq i_q$, and, if $P(e, u')$ contains more than one vertex, then $i_q < i_1 = i$ holds. Additionally, for all $1 \leq a < q$, vertex $x_a \notin S'_{i_a}$, while $x_q \in S'_{i_q}$. In particular, every inner vertex on path $P(e, u')$ is an $R$-node or a

regular vertex of $\check{H}$, and hence it lies in graph $H$. Clearly, every edge of path $P(e,u')$ is an edge of $H$ that lies in edge set $\tilde{E}''$. Therefore, path $P(e,u')$ is contained in graph $H$. We set the prefix $P^1(e)$ of the path $P(e)$ to be $P(e,u')$. We also denote by $i^{\text{left}}(e)=i_q$, and we denote by $e^{\text{left}}$ the last edge on path $P^1(e)$. Observe that $e^{\text{left}}\in\delta_H(\tilde{S}'_{i^{\text{left}}(e)})$. We define the suffix $P^3(e)$ using the right-monotone path $P(e,v)$ similarly. We denote by $e^{\text{right}}$ the last edge on that path, and by $i^{\text{right}}(e)$ the index $i^*$ such that the last vertex of path $P^3(e)$ belongs to $\tilde{S}'_{i^*}$. From the definition of monotone paths, if $u\notin\tilde{S}'_i$, then $i^{\text{left}}(e)<i$, and, if $v\notin\tilde{S}'_{i'}$, then $i^{\text{right}}(e)>i'$. Lastly, we define the *span* of edge $e$ to be $\text{span}(e)=\left\{i^{\text{left}}(e),(i^{\text{left}}(e)+1),\ldots,(i^{\text{right}}(e)-1)\right\}$. Recall that the congestion caused by the set $\left\{P(e,v),P(e,u)\mid e=(u,v)\in\hat{E}\right\}$ of paths in graph $\check{H}$ is $O(\log m)$, so the congestion caused by the set $\left\{P^1(e),P^3(e)\mid e\in\hat{E}\right\}$ of paths in graph $H$ is also $O(\log m)$.

**Mid-Segments of Paths in $\hat{\mathcal{P}}$.** We now focus on constructing the mid-segment $P^2(e)$ of the nice guiding path $P(e)$ for every edge $e\in\hat{E}$. In order to do so, fix some index $1\leq i<r$, and let $\hat{E}'_i$ be the set of all edges $e\in\hat{E}$, such that $i\in\text{span}(e)$. Note that edge $e$ may only lie in $\hat{E}'_i$ if either (i) $e\in\hat{E}_i$; or (ii) some edge $e'\in\hat{E}_i$ belongs to path $P^1(e)$; or (iii) some edge $e''\in\hat{E}_i$ belongs to path $P^3(e)$. Since the paths in set $\left\{P^1(e),P^3(e)\mid e\in\hat{E}\right\}$ cause congestion at most $O(\log m)$ in graph $H$, from Lemma 7.28, $|\hat{E}'_i|\leq O(\log m)\cdot|E_i|$. Therefore, we can define an arbitrary mapping $f_i:\hat{E}'_i\to E_i$, such that, for every edge $e\in E_i$, at most $O(\log m)$ edges of $\hat{E}'_i$ are mapped to $e$.

In order to define the mid-segment of every path in $\left\{P(e)\mid e\in\hat{E}\right\}$, we proceed as follows. For all $1\leq i<r$, we will define a collection $M_i$ of pairs of edges in $\delta_H(\tilde{S}'_i)$, so that every edge of $\delta_H(\tilde{S}'_i)$ participates in at most $O(\log m)$ such pairs. We will later exploit the bandwidth property of cluster $\tilde{S}'_i$ in order to connect every pair of edges in $M_i$ with a path. We start with $M_i=\emptyset$ for all $1\leq i\leq r$, and then gradually add edge pairs to the sets $M_i$.

Consider again some edge $e\in\hat{E}$, and recall that $\text{span}(e)=\left\{i^{\text{left}}(e),(i^{\text{left}}(e)+1),\ldots,(i^{\text{right}}(e)-1)\right\}$. For convenience, denote $i^{\text{left}}(e)$ by $i'$ and $i^{\text{right}}(e)$ by $i''$. Recall that the last edge on path $P^1(e)$, that we denoted by $e^{\text{left}}$, is an edge that is incident to cluster $\tilde{S}_{i'}$ in $H$. Let $e^{i'}$ be the edge of $E_{i'}$ to which edge $e$ is mapped by $f_{i'}$. We then add the pair $(e^{\text{left}},e^{i'})$ to $M_{i'}$.

Consider now any index $i'<i<i''-1$. Let $e^{i-1}\in E_{i-1}$ be the edge to which $e$ is mapped by $f_{i-1}$, and let $e^i\in E_i$ be the edge to which $e$ is mapped by $f_i$. We then add the pair $(e^{i-1},e^i)$ to $M_i$. Lastly, we add the edge pair $(e^{i''-1},e^{\text{right}})$ to $M_{i''}$.

We will define a path $Q^{i'}(e)$ in graph $H$, whose first edge is $e^{\text{left}}$ and last edge is $e^{i'}$, such that all inner vertices of $Q^{i'}(e)$ lie in $\tilde{S}'_{i'}$. Additionally, for all $i'<i<i''-1$, we will define a path $Q^i(e)$ in graph $H$, whose first edge is $e^{i-1}$ and last edge is $e^i$, such that all inner vertices of $Q^i(e)$ lie in $\tilde{S}'_i$. Laslty, we will define a path $Q^{i''}(e)$, whose first edge is $e^{i''-1}$, last edge is $e^{\text{right}}$, and all inner vertices lie in $\tilde{S}'_{i''}$. The final path $P^2(e)$ is then obtained by concatenating the paths $Q^{i'}(e),\ldots,Q^{i''}(e)$, and omitting the first and the last edge from the resulting path.

In order to define the paths of $\left\{Q^i(e)\mid e\in\hat{E};i^{\text{left}}(e)\leq i<i^{\text{right}}(e)\right\}$, we consider the clusters $\tilde{S}'_i\in\tilde{\mathcal{S}}'$ one by one. Consider any such cluster $\tilde{S}'_i$. Recall that we have defined a collection $M_i$ of pairs of edges from $\delta_H(\tilde{S}'_i)$, such that every edge of $\delta_H(\tilde{S}'_i)$ appears in at most $O(\log m)$ pairs. Using a standard greedy algorithm, we can compute $z=O(\log m)$ collections $M_i^1,\ldots,M_i^z$ of pairs of edges, such that $\bigcup_{j=1}^z M_i^j=M_i$, and, for all $1\leq j\leq z$, every edge of $\delta_H(\tilde{S}'_i)$ participates in at most one pair of $M_i^j$. By applying the algorithm from Corollary 4.25 to the augmented cluster $(\tilde{S}'_i)^+$, we obtain, for each $1\leq j\leq z$, a collection $\mathcal{Q}_i^j=\left\{\hat{Q}(e,e')\mid(e,e')\in M_i^j\right\}$ of paths, where each path $Q(e,e')$ has $e$ as its first edge, $e'$ as its last edge, and all internal vertices of the path lie in $\tilde{S}'_i$. Moreover, since

cluster $\tilde{S}'_i$ has $\alpha^*$-bandwidth property, with high probability, the paths in $\mathcal{Q}_i^j$ cause edge-congestion at most $O(\log^4 m/\alpha^*) \leq O(\log^{16} m)$, since $\alpha^* = \Omega(1/\log^{12} m)$. By letting $\mathcal{Q}_i = \bigcup_{j=1}^{z} \mathcal{Q}_i^j$, we obtain a collection $\mathcal{Q}_i = \left\{\hat{Q}(e,e') \mid (e,e') \in M_i\right\}$ of paths, where for every edge pair $(e,e') \in M_i$, path $Q(e,e')$ has $e$ as its first edge, $e'$ as its last edge, and every inner vertex on the path lies in $\tilde{S}'_i$. The total edge-congestion caused by paths in $\mathcal{Q}_i$ is then bounded by $O(\log^{17} m)$ with high probability. This completes the definition of the nice routing paths $\hat{\mathcal{P}} = \left\{P(e) \mid e \in \hat{E}\right\}$ in graph $H$. From the above discussion, the paths in $\hat{\mathcal{P}}$ cause edge-congestion $O(\log^{18} m)$ with high probability. If the congestion caused by the paths in $\hat{\mathcal{P}}$ is greater than $\Theta(\log^{18} m)$, we return FAIL. Otherwise, we have established that $(\mathcal{R}, \{\mathcal{D}'(R)\}_{R\in\mathcal{R}})$ is a type-2 legal clustering in $G$ with respect to $v^*$ and $\mathcal{C}'$, by providing a nice witness structure in graph $H = G_{|\mathcal{R}}$ with respect to set $\mathcal{C}'$ of basic clusters. In order to complete the proof of Lemma 7.16 and Theorem 7.3, it now remains to prove Lemma 7.28, which we do next.

### 7.2.4 Proof of Lemma 7.28

Throughout the proof, we will only consider the graph $\check{H}$, so we will omit subscript $\check{H}$ from various notations, such as, for example, $\delta_{\check{H}}(v)$ for vertices $v \in \check{H}$. We start by considering the edges connecting different clusters in $\{S_1, \ldots, S_r\}$, and by establishing some useful relationships between them.

**Edges Connecting Clusters in $\{S_1, \ldots, S_r\}$**

Fix an index $1 \leq i < r$. We denote by $E'_i = E(S_i, S_{i+1})$, and by $\tilde{E}_i^{\text{over}}$ the set of all edges $e = (u,v)$, such that, if $u \in S_j, v \in S_{j'}$, then $j < i$ and $j' > i+1$ holds. For all $1 < i \leq r$, we denote by $\tilde{E}_i^{\text{left}}$ the set of all edges $e = (u,v)$ with $u \in S_i$, such that, if $v \in S_j$, then $j < i-1$. Similarly, for all $1 \leq i < r$, we denote by $\tilde{E}_i^{\text{right}}$ the set of all edges $e = (u,v)$ with $u \in S_i$, such that, if $v \in S_j$, then $j > i+1$ holds (see Figure 14). Notice that $\delta(S_i) = E'_{i-1} \cup E'_i \cup \tilde{E}_i^{\text{left}} \cup \tilde{E}_i^{\text{right}}$. Notice also that, by the definition, if $i \in \{1,2\}$, then $E_i^{\text{left}} = \emptyset$; if $i \in \{1, r-1, r\}$, then $\tilde{E}_i^{\text{over}} = \emptyset$, and, if $i \in \{r-1, r\}$, then $\tilde{E}_i^{\text{right}} = \emptyset$. We prove the following observation that helps us relate the sizes of all these edge sets. The proof is deferred to Appendix G.10.

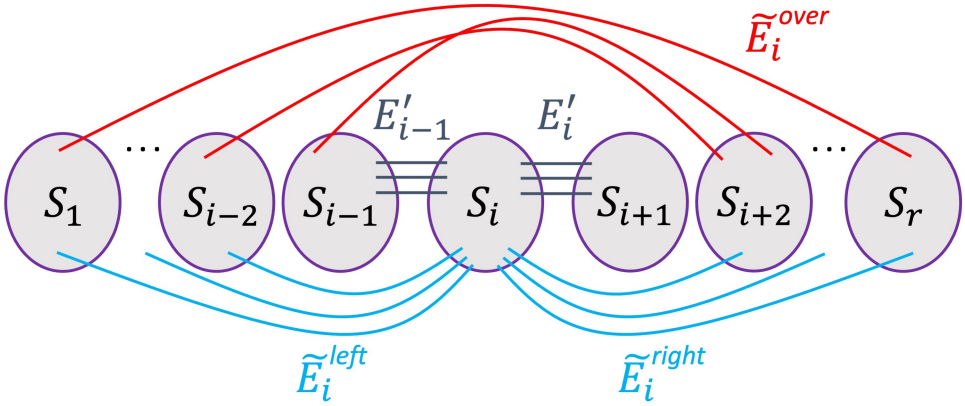



Figure 14: Edge sets $\tilde{E}_i^{\text{left}}, \tilde{E}_i^{\text{left}}$ and $\tilde{E}_i^{\text{over}}$.

**Observation 7.29** *For all $1 < i < r$, the following hold:*

- $|\tilde{E}_i^{\text{over}}| \leq |E'_i|$.

- $|\tilde{E}_{i+1}^{\text{left}}| \leq |E_i'| + |\tilde{E}_i^{\text{right}}|$
- $|\tilde{E}_i^{\text{right}}| \leq |E_i'| + |\tilde{E}_{i+1}^{\text{left}}|$.

For all $1 \leq i \leq r$, we denote $S_i'' = S_i \setminus S_i'$. Observation 7.29 allows us to bound the cardinality of the set $\tilde{E}_i^{\text{over}} \subseteq \hat{E}_i$ of edges in terms of the cardinality of the set $E_i'$ of edges. Note that the set $E_i'$ of edges can be thought of as the union of four subsets: set $E_i$, and sets $E(S_i', S_{i+1}'')$, $E(S_i'', S_{i+1}')$, and $E(S_i'', S_{i+1}'')$ (see Figure 15). The latter three sets are all contained in $\hat{E}_i$. We will bound the cardinalities of these subsets in terms of $|E_i|$ in turn. We start by considering edge sets incident to clusters of $\{S_i''\}_{1<i<r}$.

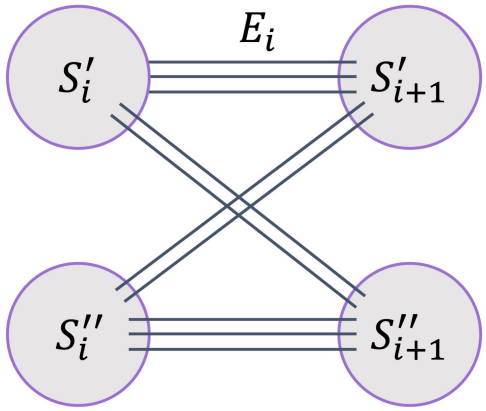



Figure 15: Edges in set $E_i'$ are shown in black.

**Edges Incident to Clusters of $\{S_i''\}_{1<i<r}$.**

Consider an index $1 < i < r$ (recall that $S_1'' = S_r'' = \emptyset$ from Observation 7.24). We partition the edges of $\delta(S_i'')$ into three subsets: set $\delta^{\text{down}}(S_i'') = E(S_i', S_i'')$; set $\delta^{\text{left}}(S_i'')$ containing all edges $(u, v)$ with $u \in S_i''$ and $v \in V(S_1) \cup \cdots \cup V(S_{i-1})$; and set $\delta^{\text{right}}(S_i'')$ containing the remaining edges (all edges $(u, v)$ with $u \in S_i''$ and $v \in V(S_{i+1}) \cup \cdots \cup V(S_r)$) (see Figure 16).

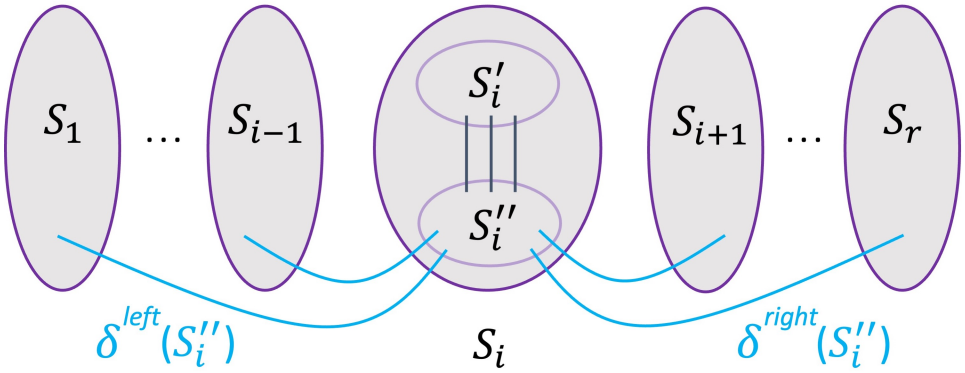



Figure 16: Edge sets $\delta^{\text{left}}(S_i'')$, $\delta^{\text{right}}(S_i'')$ and $\delta^{\text{down}}(S_i'')$ (shown in black).

We next show that for all $1 < i < r$, $|\delta^{\text{down}}(S_i'')| \leq 0.1 \min\left\{|\delta^{\text{right}}(S_i'')|, |\delta^{\text{left}}(S_i'')|\right\}$, and that the sizes of the edge sets $\delta^{\text{right}}(S_i''), \delta^{\text{left}}(S_i'')$ are close to each other, in the following two claims, whose proofs are deferred to Appendix G.11 and Appendix G.12, respectively.

**Claim 7.30** *For all $1 < i < r$, $|\delta^{\text{down}}(S_i'')| \leq 0.1 \cdot \min\left\{|\delta^{\text{right}}(S_i'')|, |\delta^{\text{left}}(S_i'')|\right\}$ holds. Additionally, there is a set $\mathcal{P}^{\text{left}} = \left\{P^{\text{left}}(e) \mid e \in \delta^{\text{down}}(S_i'')\right\}$ of edge-disjoint paths in $\check{H}$, where, for each edge $e \in \delta^{\text{down}}(S_i'')$, path $P^{\text{left}}(e)$ has $e$ as its first edge, some edge of $\delta^{\text{left}}(S_i'')$ as its last edge, and all inner vertices of $P^{\text{left}}(e)$ are contained in $S_i''$. Similarly, there is a set $\mathcal{P}^{\text{right}} = \left\{P^{\text{right}}(e) \mid e \in \delta^{\text{down}}(S_i'')\right\}$ of edge-disjoint paths in $\check{H}$, where, for each edge $e \in \delta^{\text{down}}(S_i'')$, path $P^{\text{right}}(e)$ has $e$ as its first edge, some edge of $\delta^{\text{right}}(S_i'')$ as its last edge, and all inner vertices of $P^{\text{right}}(e)$ are contained in $S_i''$.*

**Claim 7.31** *For all $1 < i < r$, $|\delta^{\text{right}}(S_i'')| \leq 1.1|\delta^{\text{left}}(S_i'')|$, and similarly, $|\delta^{\text{left}}(S_i'')| \leq 1.1|\delta^{\text{right}}(S_i'')|$.*

Next, we consider edges incident to the clusters of $\{S_1', \ldots, S_r'\}$.

### Edges Incident to the Clusters of $\{S_1', \ldots, S_r'\}$

Consider some index $1 \leq i \leq r$, and consider the edges incident to the cluster $S_i'$ in graph $\check{H}$ (see Figure 17). Recall that we have denoted by $E_{i-1} = E(S_{i-1}', S_i')$, and by $E_i = E(S_i, S_{i+1})$. We have also denoted by $\delta^{\text{down}}(S_i'') = E(S_i', S_i'')$. The remaining edges that are incident to $S_i'$ can be partitioned into two subsets: set $\delta^{\text{left}}(S_i')$, containing all edges $(u, v)$, with $u \in S_i'$ and $v \in (S_1 \cup \cdots \cup S_{i-2}) \cup S_{i-1}''$; and set set $\delta^{\text{right}}(S_i')$, containing all edges $(u, v)$, with $u \in S_i'$ and $v \in S_{i+1}'' \cup (S_{i+2} \cup \cdots \cup S_r)$. Next, for all $1 < i < r$, we bound the cardinality of edge set $\delta^{\text{left}}(S_i'')$ in terms of the cardinality of $\delta^{\text{left}}(S_i')$, and similarly we bound $|\delta^{\text{right}}(S_i'')|$ in terms of $|\delta^{\text{right}}(S_i')|$, in the following claim whose proof appears in Appendix G.13.

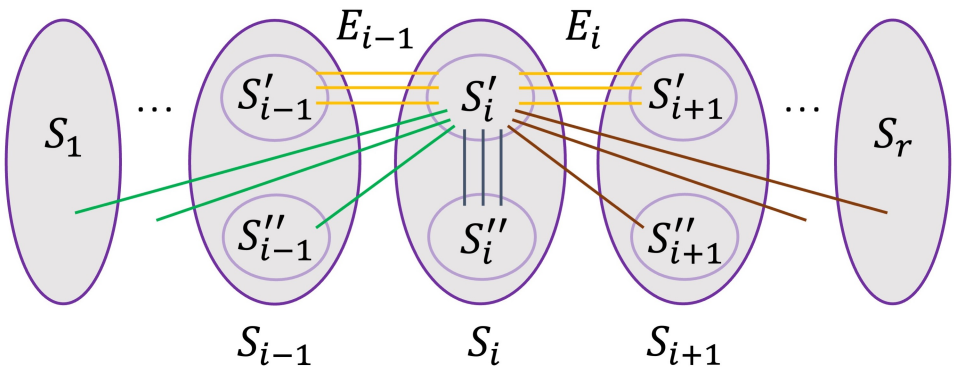



Figure 17: Edges in set $\delta^{\text{left}}(S_i')$ are shown in green, edges of $\delta^{\text{right}}(S_i')$ are shown in brown, and edges of $\delta^{\text{down}}(S_i'')$ in black.

**Claim 7.32** *For all $1 < i < r$:*

- $|\delta^{\text{right}}(S_i'')| \leq 1.3|E_i| + 1.3|\delta^{\text{right}}(S_i')|$; *and*
- $|\delta^{\text{left}}(S_{i+1}'')| \leq 1.3|E_i| + 1.3|\delta^{\text{left}}(S_{i+1}')|$.

### Accounting So Far

We now summarize what we have shown so far. Fix some index $1 \leq i < r$. Recall that set $\hat{E}_i$ of edges contains every edge $e = (u, v) \in E(\check{H})$, such that, if $u \in S_j$ and $v \in S_{j'}$, then $j \leq i$ and $j' \geq i + 1$ holds, but it excludes the edges in the set $E_i = E(S_i', S_{i+1}')$. Therefore, $\hat{E}_i$ is the union of the following subsets (see Figure 18):

- edge set $\tilde{E}_i^{\text{over}}$, connecting vertices of $V(S_1), \ldots, V(S_{i-1})$ to vertices of $V(S_{i+2}), \ldots, V(S_r)$ (see Figure 14);
- edges that lie in $\delta^{\text{right}}(S_i'') \cup \delta^{\text{left}}(S_{i+1}'')$ (see Figure 16);
- edges that lie in $\delta^{\text{right}}(S_i') \cup \delta^{\text{left}}(S_{i+1}')$ (see Figure 17).

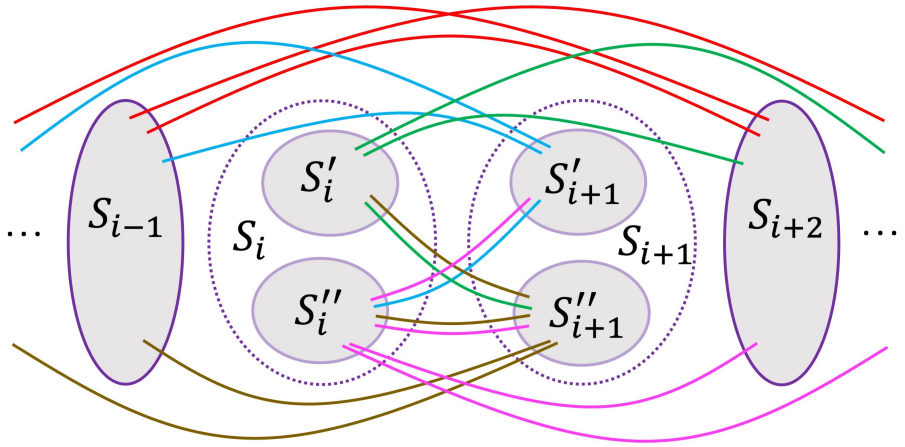



Figure 18: The set $\hat{E}_i$ of edges, with the edges of $\tilde{E}_i^{\text{over}}$ shown in red; the edges of $\delta^{\text{right}}(S_i'')$ and $\delta^{\text{left}}(S_{i+1}'')$ in pink and brown respectively; and the edges of $\delta^{\text{right}}(S_i')$ and $\delta^{\text{left}}(S_{i+1}')$ in green and blue, respectively. Note that the edges of $E(S_i'', S_{i+1}'')$ belong to both $\delta^{\text{right}}(S_i'')$ and $\delta^{\text{left}}(S_i'')$. Also, the edges of $E(S_i', S_{i+1}'')$ belong to both $\delta^{\text{left}}(S_{i+1}'')$ and $\delta^{\text{right}}(S_i')$. Similarly, the edges of $E(S_i'', S_{i+1}')$ belong to both $\delta^{\text{right}}(S_i'')$ and $\delta^{\text{left}}(S_{i+1}'))$.

In Observation 7.29, we have established that $|\tilde{E}_i^{\text{over}}| \leq |E_i'|$, where $E_i' = E(S_i, S_{i+1})$. Notice that all edges of $E_i'$ are contained in $E_i \cup \delta^{\text{right}}(S_i'') \cup \delta^{\text{left}}(S_{i+1}'')$ (see Figure 15), so we get that $|\tilde{E}_i^{\text{over}}| \leq |E_i| + |\delta^{\text{right}}(S_i'')| + |\delta^{\text{left}}(S_{i+1}'')|$. From Claim 7.32, $|\delta^{\text{right}}(S_i'')| \leq 1.3|E_i| + 1.3|\delta^{\text{right}}(S_i')|$, and $|\delta^{\text{left}}(S_{i+1}'')| \leq 1.3|E_i| + 1.3|\delta^{\text{left}}(S_{i+1}')|$. Therefore, altogether, we have shown so far that:

$$\begin{aligned} |\hat{E}_i| &\leq |\tilde{E}_i^{\text{over}}| + |\delta^{\text{right}}(S_i'')| + |\delta^{\text{left}}(S_{i+1}'')| + |\delta^{\text{right}}(S_i')| + |\delta^{\text{left}}(S_{i+1}')| \\ &\leq |E_i| + 2|\delta^{\text{right}}(S_i'')| + 2|\delta^{\text{left}}(S_{i+1}'')| + |\delta^{\text{right}}(S_i')| + |\delta^{\text{left}}(S_{i+1}')| \\ &\leq 7|E_i| + 7|\delta^{\text{right}}(S_i')| + 7|\delta^{\text{left}}(S_{i+1}')|. \end{aligned} \tag{6}$$

Therefore, it now remains to bound $|\delta^{\text{right}}(S_i')|$ and $|\delta^{\text{left}}(S_{i+1}')|$ in terms of $|E_i|$. We start with the following claim that allows us to establish some useful connection between the cardinalities of the three edge sets. The proof appears in Appendix G.14

**Claim 7.33** *For all $1 \leq i < r$: $|\delta^{\text{right}}(S_i')| \leq 2.5|E_i| + 2.5|\delta^{\text{left}}(S_{i+1}')|$, and $|\delta^{\text{left}}(S_{i+1}')| \leq 2.5|E_i| + 2.5|\delta^{\text{right}}(S_i')|$.*

Next, we show that for all $1 < i < r$, vertex $u_i$ must be a $J$-node.

**Proving that $u_2, \ldots, u_{r-1}$ are $J$-nodes.**

We start with the following simple claim, whose proof appears in Appendix G.15.

**Claim 7.34** *Consider an index $1 < i < r$, and assume that vertex $u_i$ is not a $J$-node. Then $|\bigcup_{v\in S'_i}\delta(v)| \leq \left(1+\frac{130}{\log m}\right)|\delta(u_i)|$. Moreover, if $u_i \in L'_j$, for some $1 \leq j \leq h$, then every vertex of $S'_i \setminus \{u_i\}$ lies in $L'_{j+1}\cup\cdots\cup L'_h$.*

We are now ready to prove that, for all $1 < i < r$, vertex $u_i$ must be a $J$-node.

**Lemma 7.35** *For all $1 < j < r$, vertex $u_i$ is a $J$-node.*

**Proof:** Assume for contradiction that the lemma is false. We fix an index $1 < i^* < r$, such that $u_{i*}$ is not a $J$-node, and subject to this, $|\delta(u_{i^*})|$ is maximized, breaking ties arbitrarily.

We first assume that there is some index $a$, such that at least $|\delta(u_{i^*})|/16$ edges connect $u_{i^*}$ to edges of $S''_a$. We show that in this case, $|\delta^{\text{left}}(S''_a)|, |\delta^{\text{right}}(S''_a)|$ are both large, and $u_a$ must be a $J$-node. (Note that it is impossible that $a \in \{1, r\}$, since $S''_1 = S''_r = \emptyset$, as we have established in Observation 7.24.) The proof of the following claim is deferred to Appendix G.16.

**Claim 7.36** *Suppose there is an index $1 \leq a \leq r$ (where possibly $a = i^*$), such that at least $|\delta(u_{i^*})|/16$ edges connect $u_{i^*}$ to vertices of $S''_a$. Then, $|\delta^{\text{left}}(S''_a)|, |\delta^{\text{right}}(S''_a)| \geq |\delta(u_{i^*})| \cdot \frac{\log m}{256}$, and moreover, $u_a$ is a $J$-node.*

Consider again the vertex $u_{i^*}$, and assume that $u_{i^*} \in L'_j$, for some $1 \leq j \leq h$. Recall that, from Claim 7.34, every vertex of $V(S'_{i^*}) \setminus \{u_{i^*}\}$ lies in $L'_{j+1}\cup\cdots\cup L'_h$. Therefore, all edges connecting $u_{i^*}$ to vertices of $V(S'_{i^*}) \setminus \{u_{i^*}\}$ lie in $\delta^{\text{up}}(u_{i^*})$, and their number is bounded by $|\delta^{\text{up}}(u_{i^*})| \leq |\delta(u_{i^*})|/\log m$. From Claim 7.36, the number of edges connecting $u_{i^*}$ to vertices of $S''_{i^*}$ must be bounded by $|\delta(u_{i^*})|/16$ (as $u_{i^*}$ is not a $J$-node). The remaining edges of $\delta(u_{i^*})$ must connect $u_{i^*}$ to vertices of $\bigcup_{a\neq i^*} V(S_a)$. Denote by $E^*$ the set of all edges connecting $u_{i^*}$ to vertices of $\bigcup_{a>i^*} V(S_a)$, and denote by $E^{**}$ the set of all edges connecting $u_{i^*}$ to vertices of $\bigcup_{a<i^*} V(S_a)$. From the above discusison, $|E^*\cup E^{**}| \geq 7|\delta(u_{i^*})|/8$, and so either $|E^*| \geq |\delta(u_{i^*})|/4$ or $|E^{**}| \geq |\delta(u_{i^*})|/4$ must hold. We assume w.l.o.g. that it is the former. Next we consider three cases. The first case is when neither $u_{i^*+1}$ or $u_{i^*+2}$ are $J$-nodes; the second case is when $u_{i^*+1}$ is a $J$-node; and the third case is when $u_{i^*+2}$ is a $J$-node. We show that neither of these cases is possible, by showing a simplifying cluster that should have been considered by our algorithm. For simplicity of notation, in the remainder of the proof, we denote $i^*$ by $i$.

**Case 1: neither of $u_{i+1}$, $u_{i+2}$ is a $J$-node.** Consider the set $E^*$ of edges; recall that these are all edges connecting $u_i$ to vertices of $\bigcup_{a>i} S_a$. We need the following observation:

**Observation 7.37** *At least $|\delta(u_i)|/16$ edges connect $u_i$ to vertices of $\bigcup_{a>i+2} V(S_a)$.*

(Note that in particular it follows from the observation that $r \geq i + 3$ must hold).

**Proof:** We partition the edges of $E^*$ into five subsets. The first subset, $E^*_1$, contains all edges of $E^*$ connecting $u_i$ to vertices of $S''_{i+1}$, and the second subset, $E^*_2$, contains all edges of $E^*$ connecting $u_i$ to vertices of $S''_{i+2}$. Notice that, from Claim 7.36, since we have assumed that neither of $u_{i+1}$, $u_{i+2}$ is a $J$-node, $|E^*_1|, |E^*_2| \leq |\delta(u_i)|/16$. We let $E^*_3$ be the set of all edges of $E^*$ connecting $u_i$ to vertices of $\bigcup_{a>i+2} V(S_a)$. Lastly, we let $E^*_4$ and $E^*_5$ be the sets of all edges of $E^*$ connecting $u_i$ to vertices of $S'_{i+1}$ and of $S'_{i+2}$, respectively. Assume for contradiction that $|E^*_3| < |\delta(u_i)|/16$. Then, since $|E^*| \geq |\delta(u_i)|/4$, either $|E^*_4| \geq |\delta(u_i)|/32$ or $|E^*_5| \geq |\delta(u_i)|/32$ must hold. We assume first that $|E^*_4| \geq |\delta(u_i)|/32$. Since $|\delta^{\text{up}}(u_i) \leq |\delta(u_i)|/\log m$, $|E^*_4 \cap \delta^{\text{down}}(u_i)| \geq |\delta(u_i)|/64$. Notice that, if $e = (u_i, v)$ is an edge of $E^*_4 \cap \delta^{\text{down}}(u_i)$, then $v \in S'_{i+1}$, and $e \in \delta^{\text{up}}(v)$. Since, for every vertex $v$, $|\delta^{\text{up}}(v)| \leq |\delta(v)|/\log m$, we get that:

$$|\bigcup_{v\in S'_{i+1}}\delta(v)|\geq |E^*_4\cap\delta^{\text{down}}(u_i)|\cdot\log m\geq\frac{|\delta(u_i)|\cdot\log m}{32}.$$

On the other hand, from Claim 7.34, $|\bigcup_{v\in S'_{i+1}}\delta(v)|\leq\left(1+\frac{130}{\log m}\right)|\delta(u_{i+1})|<2|\delta(u_{i+1})|$. Therefore, we get that $|\delta(u_{i+1})|>\frac{|\delta(u_i)|\cdot\log m}{64}>|\delta(u_i)|$, contradicting the choice of index $i$.
In the case where $|E^*_5|\geq|\delta(u_i)|/32$, the analysis is identical. □

Consider a cluster $S^*$, which is a subgraph of $\check{H}$ induced by $V(S_{i+1})\cup V(S_{i+2})$. In the following claim, whose proof is deferred to Appendix G.17, we prove that $S^*$ is a simplifying cluster, reaching a contradiction.

**Claim 7.38** *Cluster $S^*$ is a simplifying cluster.*

**Case 2: $u_{i+1}$ is a $J$-node.** Recall that in this case, from Observation 7.22, $S'_{i+1}=\{u_{i+1}\}$. We show that cluster $S^*=\{u_{i+1}\}$ is a simplifying cluster in the following claim, whose proof is similar to but slightly more involved than the proof of Claim 7.38, and is deferred to Appendix G.18.

**Claim 7.39** *Cluster $S^*$ is a simplifying cluster.*

This is a constradiction, since our algorithm must have identified that $S^*=S'_{i+1}$ is a simplifying cluster.

**Case 3: Neither Case 1 nor Case 2 happened.** Since Cases 1 and 2 did not happen, vertex $u_{i+1}$ is not a $J$-node. We start with the following simple observation.

**Observation 7.40** *The number of edges connecting $u_i$ to vertices of $S_{i+1}$ is at most $|\delta(u_i)|/8$.*

**Proof:** Assume otherwise. Since Case 3 happened, $u_{i+1}$ is not a $J$-node, and so, from Claim 7.36, at most $|\delta(u_i)|/16$ edges may connect $u_i$ to vertices of $S''_{i+1}$. Therefore, the number of edges connecting $u_i$ to $S'_{i+1}$ must be at least $|\delta(u_i)|/16$. But then at least $|\delta(u_i)|/32$ edges of $\delta^{\text{down}}(u_i)$ connect $u_i$ to vertices of $S'_{i+1}$. For each such vertex $e'=(u,v)$, $e'\in\delta^{\text{up}}(v)$ must hold. Since, for every vertex $v$, $\delta^{\text{up}}(v)\leq|\delta(v)|/\log m$, we get that $|\bigcup_{v\in S'_{i+1}}\delta(v)|\geq\frac{|\delta(u_i)|\log m}{16}$ must hold. However, from Claim 7.34, $|\delta(u_{i+1})|\geq\frac{|\bigcup_{v\in S'_{i+1}}\delta(v)|}{2}\geq\frac{|\delta(u_i)|\log m}{32}>|\delta(u_i)|$, contradicting the choice of the index $i^*=i$. □

Recall that we have assumed that $|E^*|\geq|\delta(u_i)|/4$, where $E^*$ is the set of all edges connecting $u_i$ to vertices of $\bigcup_{a>i}V(S_a)$. Since, from Observation 7.40, at most $|\delta(u_i)|/8$ edges connect $u_i$ to vertices of $S_{i+1}$, it must be the case that $i+2\leq r$, and at least $|\delta(u_i)|/8$ edges connect $u_i$ to vertices of $\bigcup_{a>i+1}V(S_a)$. Since we have assumed that Case 1 did not happen, vertex $u_{i+2}$ must be a $J$-node, and so, from Observation 7.22, $S'_{i+2}=\{u_{i+2}\}$. We let $S^*=\{u_{i+2}\}$, and we show, in the next claim, that cluster $S^*$ is a simplifying cluster. The proof of the claim is deferred to Appendix G.19.

**Claim 7.41** *Cluster $S^*$ is a simplifying cluster.*

This is a constradiction, since our algorithm must have identified that $S^*=S'_{i+2}$ is a simplifying cluster. □

In order to complete the proof of Lemma 7.28, it is enough to prove that for all $1\leq i<r$, $|\hat{E}_i|\leq 1000|E_i|$. Assume for contradiction that there is some index $1\leq i<r$, for which $|\hat{E}_i|>1000|E_i|$ holds. Recall that we have shown already, in Equation (6), that:

$$|\hat{E}_i| \leq 7|E_i| + 7|\delta^{\text{right}}(S'_i)| + 7|\delta^{\text{left}}(S'_{i+1})|.$$

If $|\hat{E}_i| > 1000|E_i|$, then either $|\delta^{\text{right}}(S'_i)| > 64|E_i|$, or $|\delta^{\text{left}}(S'_{i+1})| > 64|E_i|$. We assume without loss of generality that it is the former; the other case is symmetric. Recall that, since $u_i$ is a $J$-node, $S'_i = \{u_i\}$. From the definition, set $\delta^{\text{right}}(S'_i)$ contains all edges $(u_i, v)$ with $v \in S''_{i+1} \cup (S_{i+2} \cup \cdots \cup S_r)$. Note that, from Observation 7.24, $S'_r = S_r$, and so $\delta^{\text{right}}(S'_{r-1}) = \emptyset$. Therefore, we can assume that $i < r-1$. We now prove that $S^* = S'_{i+1} = \{u_{i+1}\}$ is a simplifying cluster, in the following claim, whose proof is very similar to the analysis of Case 2 in the proof of Lemma 7.35, and is deferred to Appendix G.20.

**Claim 7.42** *Cluster $S^*$ is a simplifying cluster.*

We reach a contradiction, since our algorithm must have established that cluster $S^* = S'_{i+1}$ is a simplifying cluster.

## 7.3 Disengagement of Nice Instances – Proof of Theorem 7.4

In this section we provide the proof of Theorem 7.4. Recall that we are given as input an instance $I' = (G', \Sigma')$ of the MCNwRS problem, that we will denote by $I = (G, \Sigma)$, in order to simplify the notation. Additionally, we are given a set $\mathcal{C}'$ of disjoint clusters of $G'$; in order to simplify the notation, we will denote $\mathcal{C}'$ by $\mathcal{C}$. Lastly, we are given a nice witness structure $(\tilde{\mathcal{S}}, \tilde{\mathcal{S}}', \hat{\mathcal{P}})$ for graph $G$ with respect to the set $\mathcal{C}$ of clusters, where $\tilde{\mathcal{S}} = \left\{\tilde{S}_1, \ldots, \tilde{S}_r\right\}$ is the backbone of the witness structure, with vertex sets in $\left\{V(\tilde{S}_i)\right\}_{1 \leq i \leq r}$ partitioning $V(G)$. For convenience, we will denote the set $\tilde{\mathcal{S}}' = \left\{\tilde{S}'_1, \ldots, \tilde{S}'_r\right\}$ of the vertebrae of the nice witness structure by $\mathcal{S} = \{S_1, \ldots, S_r\}$. Recall that each cluster $S_i \in \mathcal{S}$ has the $\alpha^*$-bandwidth property, for $\alpha^* = \Omega(1/\log^{12} m)$.

Recall that we are given a partition of the edges of $G$ into two subsets: set $\tilde{E}'$, containing all edges of $\bigcup_{1 \leq i \leq r} E(S_i)$, and all edges of $\bigcup_{1 \leq i < r} E(S_i, S_{i+1})$; and set $\tilde{E}'' = E(G) \setminus \tilde{E}'$. Recall that set $\hat{E} \subseteq \tilde{E}''$ contains all edges $(v, u) \in \tilde{E}''$, where $v$ and $u$ lie in different clusters of $\tilde{\mathcal{S}}$, and the set $\hat{\mathcal{P}}$ of paths contains, for each edge $e \in \hat{E}$, a path $P(e)$ that consists of three subpaths: $P^1(e), P^2(e)$, and $P^3(e)$, that are called the prefix, the mid-part and the suffix of $P(e)$, respectively. We denote by $\hat{\mathcal{P}}^1 = \left\{P^1(e) \mid e \in \hat{E}\right\}$, $\hat{\mathcal{P}}^2 = \left\{P^2(e) \mid e \in \hat{E}\right\}$, and $\hat{\mathcal{P}}^3 = \left\{P^3(e) \mid e \in \hat{E}\right\}$, the sets of paths containing all prefixes, all mid-parts, and all suffixes of the paths in $\hat{\mathcal{P}}$, respectively. Throughout, we will use a parameter $\hat{\eta} = 2^{O((\log m)^{3/4} \log\log m)}$.

In order to compute a decomposition of instance $I$ into subinstances, we need to define, for every edge $e \in \hat{E}$, a cycle $W(e)$, called an *auxiliary cycle* that has some useful properties. As an intuition, we could obtain a cycle $W(e)$ by taking the union of the nice guiding path $P(e) \in \hat{\mathcal{P}}$ with the edge $e$. The structure of the nice guiding paths ensures that the cycle $W(e)$ has a single contiguous segment $P^2(e)$ that visits a contiguous subset of the vertebrae in the order of their indices. The resulting set $\left\{W(e) \mid e \in \hat{E}\right\}$ of cycles is close to having the properties that we need, except that we would like to ensure that these cycles are non-transeversal (or close to being non-transversal) with respect to $\Sigma$. We discuss the construction of the family $\left\{W(e) \mid e \in \hat{E}\right\}$ of cycles with these properties below. Next, we define a laminar family $\mathcal{L} = \{U_1, \ldots, U_r\}$ of clusters of graph $G$, where for all $1 \leq i \leq r$, $U_i$ is the subgraph of $G$ induced by vertex set $V(S_1) \cup \cdots \cup V(S_i)$. We define, for every vertebra $S_i \in \mathcal{S}$, an internal $S_i$-router $\mathcal{Q}(S_i)$, and use these routers, together with the auxiliary cycles in $\left\{W(e) \mid e \in \hat{E}\right\}$ in order to define an internal $U_i$-router and an external $U_i$-router for every cluster $U_i \in \mathcal{L}$. The final decomposition $\mathcal{I}_2$ of instance $I$ into subinstances is simply a decomposition via the

laminar family $\mathcal{L}$ defined in Section 5.1. Recall that for each cluster $U_z \in \mathcal{I}_2$, there is a unique instance $I_z = (G_z, \Sigma_z) \in \mathcal{I}_2$, where graph $G_z$ is obtained from $G$ by contracting all vertices of $S_1 \cup \cdots \cup S_{z-1}$ into a special vertex $v_z^*$, and all vertices of $S_{z+1} \cup \cdots \cup S_r$ into a special vertex $v_z^{**}$ (for $z = 1$, $G_1$ is obtained from $G$ by contracting all vertices of $S_2 \cup \cdots \cup S_r$ into a special vertex $v_1^{**}$, and for $z = r$, graph $G_z$ is obtained from $G$ by contracting all vertices of $S_1 \cup \cdots \cup S_{r-1}$ into a special vertex $v_r^*$). For each $1 \leq z < r$, we use the internal $U_z$-router that we computed, in order to define a circular ordering of the edges of $\delta_G(U_z)$, that will in turn be used in order to define the rotation systems $\{\Sigma_z\}_{z=1}^r$ associated with each subinstance. The techniques developed in Section 5.1 prove that there is an efficient algorithm that combines solutions to the resulting subinstances into a solution to instance $I$ that has a relatively low cost. However, since the depth of the laminar family $\mathcal{L}$ may be quite high, we cannot use the tools from Section 5 in order to bound $\sum_{z=1}^r |E(G_z)|$ and $\sum_{z=1}^r \mathsf{OPT}_{\mathsf{cnwrs}}(I_z)$. Instead, we show a simple direct bound on $\sum_{z=1}^r |E(G_z)|$, and a more involved proof for bounding $\sum_{z=1}^r \mathsf{OPT}_{\mathsf{cnwrs}}(I_z)$. The latter proof exploits the internal and external $U_i$-routers that we construct, for $1 \leq i \leq r$, which in turn are based on the auxiliary cycles $\left\{W(e) \mid e \in \hat{E}\right\}$, in order to show the existence of a low-cost solution to each instance $I_z \in \mathcal{I}_2$. In order to ensure that the costs of these solutions is sufficiently low, it is crucial that the cycles in $\left\{W(e) \mid e \in \hat{E}\right\}$ are *almost* non-transversal with respect to $\Sigma$: that is, for every pair $W(e), W(e')$ of cycles, there is at most one vertex $v$, such that $W(e)$ and $W(e')$ intersect transversally at $v$. In order to define the set $\mathcal{W} = \left\{W(e) \mid e \in \hat{E}\right\}$ of cycles, we define two collections of paths: path set $\mathcal{P}^{\mathsf{out}} = \left\{P^{\mathsf{out}}(e) \mid e \in \hat{E}\right\}$, which is obtained by modifying the paths of $\hat{\mathcal{P}}^1 \cup \hat{\mathcal{P}}^3$, and path set $\mathcal{P}^{\mathsf{in}} = \left\{P^{\mathsf{in}}(e) \mid e \in \hat{E}\right\}$. For every edge $e \in \hat{E}$, the first and the last edges on paths $P^{\mathsf{out}}(e)$ and $P^{\mathsf{in}}(e)$ are identical. Path $P^{\mathsf{out}}(e)$ contains the edge $e$, and all its edges lie in $\tilde{E}''$. All inner edges of path $P^{\mathsf{in}}(e)$ lie in $\tilde{E}'$, and the path visits a consequtive subset of clusters of $\mathcal{S}$ in their natural order. The auxiliary cycle $W(e)$ is obtained by taking the union of the paths $P^{\mathsf{out}}(e)$ and $P^{\mathsf{in}}(e)$.

The remainder of the proof of Theorem 7.4 consists of four steps. In the first step, we construct the set $\mathcal{P}^{\mathsf{out}} = \left\{P^{\mathsf{out}}(e) \mid e \in \hat{E}\right\}$ of paths. In the second step, we construct the set $\mathcal{P}^{\mathsf{in}} = \left\{P^{\mathsf{in}}(e) \mid e \in \hat{E}\right\}$ of paths and the collection $\mathcal{W} = \left\{W(e) \mid e \in \hat{E}\right\}$ of auxiliary cycles. In the third step, we construct the laminar family $\mathcal{L} = \{U_1, \ldots, U_r\}$ of clusters, and, for all $1 \leq z \leq r$, an internal $U_z$-router $\mathcal{Q}(U_z)$ and an external $U_z$-router $\mathcal{Q}'(U_z)$. In the fourth and the final step, we compute the collection $\mathcal{I}_2$ of subinstances of $I$ and analyze its properties. We now describe each of the steps in turn.

### 7.3.1 Step 1. Constructing the Paths of $\mathcal{P}^{\mathsf{out}}$

In this step, we construct the set $\mathcal{P}^{\mathsf{out}} = \left\{P^{\mathsf{out}}(e) \mid e \in \hat{E}\right\}$ of paths, by slightly modifying the prefixes and the suffixes of the paths in $\hat{\mathcal{P}}$ to make them non-transversal.

Throughout, we denote $V' = \bigcup_{S \in \mathcal{S}} V(S)$ and $V'' = V(G) \setminus V'$. Consider an edge $e = (u, v) \in \hat{E}$, and assume that $u \in S_i, v \in S_j$, and $i < j$. For convenience, we will call $u$ the *left endpoint of edge* $e$, and $v$ the *right endpoint of edge* $e$. We also define two sets of indices associated with edge $e$. The first set of indices is $\mathrm{span}(e) = \{i, i+1, \ldots, j-1\}$. In order to define the second set of indices, assume that the last vertex on path $P^1(e)$ (vertex that lies in $V'$) belongs to cluster $S_{i'}$, while the last vertex on path $P^3(e)$ belongs to cluster $S_{j'}$. From the definition of nice guiding paths, $i' \leq i < j \leq j'$ must hold. We then let $\mathrm{span}'(e) = \{i', i'+1, \ldots, j'-1\}$.

It will be convenient for us to define the notion of left-monotone and right-monotone paths. The definition is similar to the one used in Section 7.2.3, but not identical.

**Definition 7.43** *Let $R$ be a (directed) path in graph $G$ that contains at least one edge, let $(v_1,\ldots,v_z)$ be the sequence of vertices appearing on $R$, and, for all $1\leq a\leq z$, assume that $v_a\in\tilde{S}_{i_a}$. We say that $R$ is a* left-monotone path *if:*

- $v_z\in V'$;
- *for* $1<a<z$, $v_a\in V''$; *and*
- $i_1\geq i_2\geq\cdots\geq i_z$.

*Similarly, we say that $R$ is a* right-monotone path*, if:*

- $v_z\in V'$;
- *for* $1<a<z$, $v_a\in V''$; *and*
- $i_1\leq i_2\leq\cdots\leq i_z$.

For each edge $e\in\hat{E}$, we view the paths $P^1(e)\in\mathcal{P}^1$, $P^3(e)\in\mathcal{P}^3$ as directed paths that originate at an endpoint of edge $e$ and terminate at a vertex of $V'$ (notice that it is possible that one or even both endpoints of $e$ lie in $V'$). For every vertex $v\in V'$, we denote by $n_1(v)$ the total number of paths in $\hat{\mathcal{P}}^1$ that terminate at $v$, and by $n_3(v)$ the total number of paths in $\hat{\mathcal{P}}^3$ that terminate at $v$. Let $\eta=O(\log^{18}m)$ be such that the set $\hat{\mathcal{P}}$ of paths causes congestion at most $\eta$ in $G$. We use the following claim in order to construct the paths of $\mathcal{P}^{\mathsf{out}}$; the proof of the claim uses standard techniques and is deferred to Appendix G.21.

**Claim 7.44** *There is an efficient algorithm to compute two sets $\mathcal{P}^{\mathsf{out},\mathrm{left}}=\left\{P^{\mathsf{out},\mathrm{left}}(e)\mid e\in\hat{E}\right\}$ and $\mathcal{P}^{\mathsf{out},\mathrm{right}}=\left\{P^{\mathsf{out},\mathrm{right}}(e)\mid e\in\hat{E}\right\}$ of simple paths in graph $G$, each of which causes congestion at most $\eta$, such that the paths in set $\mathcal{P}^{\mathsf{out},\mathrm{left}}$ are non-transversal with respect to $\Sigma$, and so are the paths in set $\mathcal{P}^{\mathsf{out},\mathrm{right}}$. Additionally, for every edge $e\in\hat{E}$, path $P^{\mathsf{out},\mathrm{left}}(e)$ has $e$ as its first edge, and it is left-monotone, while path $P^{\mathsf{out},\mathrm{right}}(e)$ has $e$ as its first edge, and is right-monotone. Moreover, for every vertex $v\in V'$, exactly $n_1(v)$ paths of $\mathcal{P}^{\mathsf{out},\mathrm{left}}$ terminate at $v$, and exactly $n_3(v)$ paths of $\mathcal{P}^{\mathsf{out},\mathrm{right}}$ terminate at $v$.*

Consider now some edge $e=(u,v)\in\hat{E}$, and assume that $u\in\tilde{S}_i$, $v\in\tilde{S}_j$, and $i<j$ holds. Consider the paths $P^{\mathsf{out},\mathrm{left}}(e),P^{\mathsf{out},\mathrm{right}}(e)$. Assume that the last vertex on path $P^{\mathsf{out},\mathrm{left}}$ is $u'$, and the last vertex on path $P^{\mathsf{out},\mathrm{right}}$ is $v'$. We let $P^{\mathsf{out}}(e)$ be the path obtained by concatenating path $P^{\mathsf{out},\mathrm{left}}(e)$ with the reversed path $P^{\mathsf{out},\mathrm{right}}(e)$, after deleting the extra copy of edge $e$. We view path $P^{\mathsf{out}}(e)$ as being directed rom $u'$ to $v'$. Therefore, path $P^{\mathsf{out}}(e)$ originates at vertex $u'$ and termiantes at vertex $v'$, and it contains the edge $e$. All inner vertices on $P^{\mathsf{out}}(e)$ belong to $V''$. We will sometimes refer to $u'$ and to $v'$ as the first and the last endpoints of path $P^{\mathsf{out}}(e)$. We will also refer to the edge of $P^{\mathsf{out}}(e)$ that is incident to $u'$ as the *first edge* of path $P^{\mathsf{out}}(e)$, and to the edge of $P^{\mathsf{out}}(e)$ that is incident to $v'$ as the *last edge* of path $P^{\mathsf{out}}(e)$. Assume that $u'\in V(S_{i''})$ and $v'\in V(S_{j''})$. We define another set of indices associated with edge $e$: $\mathrm{span}''(e)=\{i'',i''+1,\ldots,j''-1\}$. Notice that $\mathrm{span}(e)\subseteq\mathrm{span}''(e)$ must hold by the definition of left-monotone and right-monotone paths. Lastly, we set $\mathcal{P}^{\mathsf{out}}=\left\{P^{\mathsf{out}}(e)\mid e\in\hat{E}\right\}$.

For an index $1\leq i<r$, let $\hat{E}_i\subseteq\hat{E}$ be the set of all edges $e\in\hat{E}$, with $i\in\mathrm{span}(e)$. We need the following simple claim.

**Claim 7.45** *For every index $1\leq i<r$, the paths in set $\left\{P^{\mathsf{out}}(e)\mid e\in\hat{E}_i\right\}$ are non-transversal with respect to $\Sigma$. Moreover, for each edge $e=(u,v)\in\hat{E}_i$ whose left endpoint is $u$, every vertex of $P^{\mathsf{out},\mathrm{left}}(e)\setminus\{v\}$ lies in $\bigcup_{z\leq i}V(\tilde{S}_z)$, and every vertex of $P^{\mathsf{out},\mathrm{right}}(e)\setminus\{u\}$ lies in $\bigcup_{z>i}V(\tilde{S}_z)$.*

**Proof:** The fact that, for every edge $e \in \hat{E}_i$, every vertex of $P^{\mathsf{out,left}}(e) \setminus \{v\}$ lies in $\bigcup_{z \le i} V(\tilde{S}_z)$, and every vertex of $P^{\mathsf{out,right}}(e) \setminus \{u\}$ lies in $\bigcup_{z > i} V(\tilde{S}_z)$ follows immediately from the definition of left-monotone and right-monotone paths and edge set $\hat{E}_i$. Consider now any pair $e, e' \in \hat{E}_i$ of edges, and a vertex $v$ that is an inner vertex of both $P^{\mathsf{out}}(e)$ and $P^{\mathsf{out}}(e')$. Assume that $v \in V(\tilde{S}_j)$, for some index $1 \le j \le r$. From the definition of left-monotone and right-monotone paths, either $j \le i$ and $v$ is an inner vertex of both $P^{\mathsf{out,left}}(e)$ and $P^{\mathsf{out,left}}(e')$; or $j > i$, and $v$ is an inner vertex of both $P^{\mathsf{out,right}}(e)$ and $P^{\mathsf{out,right}}(e')$. In either case, Claim 7.44 ensures that the intersection of $P^{\mathsf{out}}(e)$ and $P^{\mathsf{out}}(e')$ at $v$ is non-transversal. □

For every index $1 \le t < r$, we denote by $N_t$ the number of all edges $e \in \hat{E}$, with $t \in \mathrm{span}'(e)$, and we denote by $N'_t$ be the number of all edges $e \in \hat{E}$, with $t \in \mathrm{span}''(e)$. We also denote by $E_t \subseteq \tilde{E}'$ the set of all edges with one endpoint in $S_t$ and another in $S_{t+1}$. We need the following claim, whose proof appears in Appendix G.22.

**Claim 7.46** *For all $1 \le t < r$, $N'_t = N_t$, and $N_t \le O(\log^{18} m) \cdot |E_t|$.*

### 7.3.2 Step 2: Constructing the Paths of $\mathcal{P}^{\mathsf{in}}$ and the Auxiliary Cycles

Consider some edge $e = (u, v) \in \hat{E}$, and assume that $u$ is the left endpoint of $e$. Assume that $\mathrm{span}''(e) = \{i'', i''+1, \ldots, j''-1\}$, and denote by $\hat{e}_{i''-1}$ the first edge on path $P^{\mathsf{out}}(e)$, and by $\hat{e}_{j''}$ the last edge on path $P^{\mathsf{out}}(e)$. In this step we will construct another path $P^{\mathsf{in}}(e)$, whose first edge is $\hat{e}_{i''-1}$ and last edge is $\hat{e}_{j''}$. In order to do so, we will select, for all $i'' \le z < j''$, some edge $\hat{e}_z \in E_z$, that we assign to the edge $e$, and we will compute a path $R_z(e)$ (that we call a *segment*), whose first edge is $\hat{e}_{z-1}$, last edge is $\hat{e}_z$, and all inner vertices are contained in $S_z$. The final path $P^{\mathsf{in}}(e)$ will be obtained by concatenating the segments $R_{i''}(e), \ldots, R_{j''}(e)$. Note that path $P^{\mathsf{in}}(e)$ has $\hat{e}_{i''-1}$ and $\hat{e}_{j''}$ as its first and last edges. By concatenating the paths $P^{\mathsf{in}}(e)$ and $P^{\mathsf{out}}(e)$, we will then obtain the auxiliary cycle $W(e)$. Our goal in constructing the set $\mathcal{P}^{\mathsf{in}} = \left\{P^{\mathsf{in}}(e) \mid e \in \hat{E}\right\}$ of paths is to ensure that these paths cause low congestion, and that these paths are *mostly* non-transversal with respect to $\Sigma$. In fact, we will ensure that, for every pair $P, P' \in \mathcal{P}^{\mathsf{in}}$ of such paths, there is at most one vertex $v$, such that the intersection of $P$ and $P'$ at $v$ is transversal. Intuitively, the resulting auxiliary cycles in $\left\{W(e) \mid e \in \hat{E}\right\}$ will be exploited in order to show the existence of cheap solutions to the subinstances of the input instance $I$ that we compute. Each transversal intersection between a pair of such cycles may give rise to a crossing in these solutions, which motivates the requirement that the paths in $\mathcal{P}^{\mathsf{in}}$ have few transversal intesrections is low.

Consider again an edge $e \in \hat{E}$, and assume that $\mathrm{span}''(e) = \{i'', i''+1, \ldots, j''-1\}$. Recall that we have already defined edges $\hat{e}_{i''-1} \in \delta(S_{i''-1})$ and $\hat{e}_{j''} \in \delta(S_{j''-1})$. We construct a collection $\tilde{\mathcal{R}}(e) = \left\{R_{i''}(e), \ldots, R_{j''}(e)\right\}$ of paths, and define, for all $i'' \le z < j''$, edge $\hat{e}_z \in E_z$, such that, for all $i'' \le z \le j''$, path $R_z(e)$ connects edge $\hat{e}_{z-1}$ to edge $\hat{e}_z$, and its inner vertices lie in $S_z$. In order to do so, we initially set $\tilde{\mathcal{R}}(e) = \emptyset$ for every edge $e \in \hat{E}$. We then process indices $1 \le z \le r$ one by one. When index $z$ is processed, we will define, for every edge $e \in \hat{E}$ with $z \in \mathrm{span}''(e)$ or $z-1 \in \mathrm{span}''(e)$, the segment $R_z(e)$; if $z \in \mathrm{span}''(e)$, we will also define the edge $\hat{e}_z \in E_z$, which is the last edge on path $R_z(e)$. We will ensure that every edge $e' \in E_z$ is assigned to at most $O(\log^{18} m)$ edges of $\hat{E}$. We now describe an interation where index $1 \le z \le r$ is processed.

**Iteration Description.** We fix an index $1 \le z \le r$, and describe an iteration for processing index $z$. Let $A_z \subseteq \hat{E}$ be the set of all edges $e \in \hat{E}$, with $z \in \mathrm{span}''(e)$. Note that for every edge $e \in A_z$, the corresponding edge $\hat{e}_{z-1} \in E_{z-1}$ is already fixed. Let $A'_z \subseteq \hat{E} \setminus A_z$ be the set of all edges $e \in \hat{E}$, such that $z-1 \in \mathrm{span}''(e)$ but $z \notin \mathrm{span}''(e)$. Notice that, if $e \in A'_z$, then both edges $\hat{e}_{z-1}, \hat{e}_z \in \delta_G(S_z)$ are already fixed (in this case, edge $\hat{e}_z$ is the last edge on path $P^{\mathsf{out}}(e)$).

Consider the augmentation $S_z^+$ of the cluster $S_z$, that we denote for convenience by $H$. Recall that, in order to obtain graph $H$, we start by subdividing every edge $e' \in \delta_G(S_z)$ by vertex $t_{e'}$, and then denote by $T = \{t_{e'} \mid e' \in \delta_G(S_z)\}$ the set of newly added vertices, that we call *terminals*. We then let $H$ be the subgraph of the resulting graph induced by $V(S_z) \cup T$. Recall that, from the definition of nice witness structure, cluster $S_z$ has the $\alpha^* = \Omega(1/\log^{12} m)$-bandwidth property in $G$, and so, from Observation 4.16, vertex set $T$ is $\alpha^*$-well-linked in $H$.

We now define a collection $M$ of pairs of terminals, that we call *demand pairs*, that are associated with the edges of $A'_z$. Consider an edge $e \in A'_z$, and recall that edges $\hat{e}_{z-1}, \hat{e}_z \in \delta_G(S_z)$ are already defined. The demand pair associated with edge $e$ is $(x_e, y_e)$, where $x_e = t_{\hat{e}_{z-1}}$ (the terminal vertex associated with edge $\hat{e}_{z-1}$), and $y_e = t_{\hat{e}_z}$ (the terminal vertex associated with edge $\hat{e}_z$). We then set $M = \{(x_e, y_e) \mid e \in A'_z\}$. Recall that the edges of $\mathcal{P}^{\mathsf{out}}$ cause congestion at most $\eta = O(\log^{18} m)$, and every edge of $E_{z-1}$ is assigned to at most $\eta$ edges of $\hat{E}$. Therefore, a terminal $t_{e'}$ may participate in at most $\eta$ pairs in $M$. Using a standard greedy algorithm, we can now compute $2\eta$ sets of terminal pairs $M_1, \ldots, M_{2\eta}$, such that $M = \bigcup_{a=1}^{2\eta} M_a$, and, for all $1 \leq a \leq 2\eta$, each terminal participates in at most one pair in $M_a$. For each $1 \leq a \leq 2\eta$, we use the algorithm from Corollary 4.25, to compute a collection $\mathcal{R}(M_a) = \{R(x,y) \mid (x,y) \in M_a\}$ of paths in graph $H$, where for every pair $(x,y) \in M_a$, path $R(x,y)$ connects $x$ to $y$. Since the vertices of $T$ are $\alpha^*$-well-linked in $G$, with high probability, the paths in $\mathcal{R}(M_a)$ cause congestion $O(\log^4 m/\alpha^*) = O(\log^{16} m)$. If the paths in $\mathcal{R}(M_a)$ cause a higher congestion, then we terminate the algorithm and return FAIL. Note that the set $\bigcup_{a=1}^{2\eta} \mathcal{R}(M_a)$ of paths in graph $H$ naturally defines a set $\mathcal{R}'' = \{R(e) \mid e \in A'_z\}$ of paths in graph $G$, where, for every edge $e \in A'_z$, path $R(e)$ has $\hat{e}_{z-1}$ as its first edge and $\hat{e}_z$ as its last edge, while all inner vertices of $R(e)$ lie in $S_z$. From the above discussion, the paths in $\mathcal{R}''$ cause congestion at most $\eta' = 2\eta \cdot O(\log^{16} m) = O(\log^{34} m)$.

Next, we consider the set $A_z \subseteq E_{z-1}$ of edges. Let $X$ be a multiset of vertices of $T$ that contains, for every edge $e' \in A_z$, the corresponding vertex $t_{e'}$. Since every edge of $E_{z-1}$ may only be assigned to at most $\eta$ edges of $\hat{E}$, a vertex may appear in set $X$ at most $\eta$ times. Recall that $|A_z| = N'_z$, and, from Claim 7.46, $N'_z \leq \eta \cdot |E_z|$. Therefore, we can define a multiset $Y$ that contains $|A|$ elements, each of which is a vertex from $\{t_{e'} \mid e' \in E_z\}$, such that at most $\eta$ copies of each such vertex $t_{e'}$ appear in set $Y$. We let $M'$ be an arbitrary matching between elements of $X$ and elements of $Y$. Using the same procedure as the one employed for the edges of $A'_z$, we construct a set $\mathcal{R}' = \{R(e) \mid e \in A_z\}$ of paths in graph $G$, where, for every edge $e \in A_z$, path $R(e)$ has $\hat{e}_{z-1}$ as its first edge and some edge of $E_z$ as its last edge, while all inner vertices of $R(e)$ lie in $S_z$. Additionally, the paths in $\mathcal{R}''$ cause congestion at most $\eta'$, and every edge of $E_z$ appears on at most $\eta$ edges of $\mathcal{R}$. (As before, if the paths in set $\mathcal{R}''$ cause a higher congestion, we terminate the algorithm and return FAIL.) Note that we can assume without loss of generality that all paths in $\mathcal{R}' \cup \mathcal{R}''$ are simple paths.

To summarize, we have now constructed two sets $\mathcal{R}' = \{R(e) \mid e \in A_z\}, \mathcal{R}'' = \{R(e) \mid e \in A'_z\}$ of paths, with the following properties:

I1. Paths in each of the sets $\mathcal{R}', \mathcal{R}''$ cause congestion at most $\eta' = O(\log^{34} m)$;

I2. For every edge $e \in A_z$, path $R(e) \in \mathcal{R}'$ has $\hat{e}_{z-1}$ as its first edge, some edge of $E_z$ as its last edge, and all inner vertices of $R(e)$ lie in $S_z$;

I3. Every edge of $E_z$ participates in at most $\eta$ paths of $\mathcal{R}'$;

I4. For every edge $e \in A'_z$, path $R(e) \in \mathcal{R}''$ has $\hat{e}_{z-1}$ as its first edge, $\hat{e}_z$ as its last edge, and all inner vertices of $R(e)$ lie in $S_z$; and

I5. All paths in set $\mathcal{R}' \cup \mathcal{R}''$ are simple.

Next, we will iteratively modify the paths in $\mathcal{R}' \cup \mathcal{R}''$, while ensuring that Properties I1–I5 hold at the end of each iteration. In every iteration, we will attempt to reduce the number of transversal

intersections between the paths of $\mathcal{R}' \cup \mathcal{R}''$. In fact we will guarantee that, after each iteration, either $\sum_{R \in \mathcal{R}' \cup \mathcal{R}''} |E(R)|$ decreases, or $\sum_{R \in \mathcal{R}' \cup \mathcal{R}''} |E(R)|$ remains unchanged, and the number of triples in set $\Pi^T(\mathcal{R}' \cup \mathcal{R}'')$ strictly decreases (see definition immediately after Definition 4.5)

We now describe a single iteration. Assume first that there are two paths $R(e), R(e') \in \mathcal{R}'$, and some vertex $v$ that is an inner vertex of both $R(e)$ and $R(e')$, such that the intersection of $R(e)$ with $R(e')$ at $v$ is transversal. In this case, we splice paths $R(e)$ and $R(e')$ at vertex $v$ (see Section 4.1.4), obtaining two new paths. The first path, that replaces $R(e)$ in $\mathcal{R}'$, originates at edge $\hat{e}_{z-1}$ and terminates at the edge of $E_z$ that served as the last edge of $R(e')$. The second path, that replaces $R(e')$ in $\mathcal{R}'$, originates at edge $e'_{z-1}$, and terminates at the edge of $E_z$ that served as the last edge of the original path $R(e)$. Both paths only contain vertices of $S_z$ as inner vertices. From Observation 4.6, either at least one of the two new paths $R(e), R(e')$ becomes a non-simple path; or both paths remain simple paths, but $|\Pi^T(\mathcal{R}' \cup \mathcal{R}'')|$ decreases. Notice that, in the latter case, $\sum_{R \in \mathcal{R}' \cup \mathcal{R}''} |E(R)|$ remains unchanged. If the former case happens, then we remove cycles from paths $R(e), R(e')$, until they become simple paths. In this case, $\sum_{R \in \mathcal{R}' \cup \mathcal{R}''} |E(R)|$ decreases. In either case, it is easy to verify that Invariants I1–I5 continue to hold. We then proceed to the next iteration.

Assume now that there is a path $R(e) \in \mathcal{R}' \cup \mathcal{R}''$ and another path $R(e') \in \mathcal{R}''$, and two distinct vertices $v, v'$, both of which are inner vertices on both $R(e)$ and $R(e')$, such that $R(e)$ and $R(e')$ intersect transversally at both $v$ and $v'$. Let $Q$ be the subpath of $R(e)$ between $v$ and $v'$, and let $Q'$ be the subpath of $R(e')$ between $v$ and $v'$. We splice the paths $R(e)$ and $R(e')$ at both $v$ and $v'$. Equivalently, we modify path $R(e)$ by replacing its segment $Q$ with $Q'$, and we modify path $R(e')$ by replacing its segment $Q'$ with $Q$. Note that the first and the last edge on each path remains the same, and the congestion caused by the set $\mathcal{R}' \cup \mathcal{R}''$ of paths remains the same. If any of the resulting paths $R(e), R(e')$ becomes a non-simple path, then we delete cycles from it, until it becomes a simple path. In this case, $\sum_{R \in \mathcal{R}' \cup \mathcal{R}''} |E(R)|$ decreases. Otherwise, we can use Observation 4.6 to conclude that $|\Pi^T(\mathcal{R}' \cup \mathcal{R}'')$ has decreased. Indeed, it is easy to verify that, for any vertex $v'' \in V(S_z) \setminus \{v, v'\}$, the number of triples $(R_1, R_2, v'') \in \Pi^T(\mathcal{R}' \cup \mathcal{R}'')$ did not grow. The number of triples $(R_1, R_2, v) \in \Pi^T(\mathcal{R}' \cup \mathcal{R}'')$, and the number of triples $(R_1, R_2, v') \in \Pi^T(\mathcal{R}' \cup \mathcal{R}'')$ have both decreased (as can be seen by applying Observation 4.6 to the set of paths that contains, for every path $R^* \in \mathcal{R}' \cup \mathcal{R}''$ with $v \in R^*$, a subpath of $R^*$ consisting of the two edges of $R^*$ incident to $v$, and doing the same for vertex $v'$). This completes the description of an iteration. It is easy to verify that Invariants I1–I5 continue to hold.

The algorithm for processing the index $z$ terminates when the path set $\mathcal{R}'$ becomes non-traversal with respect to $\Sigma$, and, for every pair $R \in \mathcal{R}' \cup \mathcal{R}''$, $R' \in \mathcal{R}''$ of paths, there is at most one vertex $v$ such that the intersection of $R$ and $R'$ at $v$ is transversal. We then denote $\mathcal{R}_z = \mathcal{R}' \cup \mathcal{R}''$. For every edge $e \in A_z \cup A'_z$, we set $R_z(e) = R(e)$ (the unique path in $\mathcal{R}_z$ that originates at edge $\hat{e}_{z-1}$). If $e \in A'_z$, then path $R(e)$ is guaranteed to terminate at edge $\hat{e}_z$, from Invariant I4. If $e \in A_z$, then we let $\hat{e}_z$ be the last edge on path $R_z(e)$. We then add path $R_z(e)$ to set $\tilde{\mathcal{R}}(e)$.

Once all indices $1 \leq z \leq r$ are processed, we obtain, for every edge $e \in \hat{E}$, the desired set $\tilde{\mathcal{R}}(e)$ of paths. If $\mathrm{span}(e) = \{i'', \ldots, j'' - 1\}$, then $\tilde{\mathcal{R}}(e) = \big\{R_{i''}(e), \ldots, R_{j''}(e)\big\}$. We then let $P^{\mathsf{in}}(e)$ be the path obtained by concatenating the paths in $\tilde{\mathcal{R}}(e)$. Recall that the first edge on $P^{\mathsf{in}}(e)$ is $\hat{e}_{i''-1}$, which is the first edge of $P^{\mathsf{out}}(e)$, and similary, the last edge on $P^{\mathsf{in}}(e)$ is $\hat{e}_{j''}$, the last edge of $P^{\mathsf{out}}(e)$. We obtain the auxiliary cycle $W(e)$ by concatenating the paths $P^{\mathsf{in}}(e)$ and $P^{\mathsf{out}}(e)$ (after deleting the extra copies of edges $\hat{e}_{i''-1}, \hat{e}_{j''}$). It is immediate to verify that cycle $W(e)$ is a simple cycle. Lastly, we set $\mathcal{P}^{\mathsf{in}} = \Big\{P^{\mathsf{in}}(e) \mid e \in \hat{E}\Big\}$ and $\mathcal{W} = \Big\{W(e) \mid e \in \hat{E}\Big\}$. We refer to $\mathcal{W}$ as the *set of auxiliary cycles*. From the above discussion and Claim 7.44, we obtain the following immediate observation:

**Observation 7.47** *Every edge $e \in \bigcup_{z=1}^{r} E(S_z)$ appears on at most $\eta' = O(\log^{34} m)$ cycles of $\mathcal{W}$. Every edge $e \in E(G) \setminus (\bigcup_{z=1}^{r} E(S_z))$ appears on at most $\eta = O(\log^{18} m)$ cycles of $\mathcal{W}$.*

Recall that we denoted, for each index $1 \leq z < r$, by $\hat{E}_z \subseteq \hat{E}$ the set of all edges $e \in \hat{E}$, with

$z \in \text{span}(e)$. We need the following observation.

**Observation 7.48** *For every index $1 \leq z < r$, for every pair $e, e' \in \hat{E}_z$ of distinct edges, there is at most one vertex $v \in V(W(e)) \cap V(W(e'))$, such that the intersection of cycles $W(e)$ and $W(e')$ is transversal at $v$. If such a vertex $v$ exists, then $v \in S_j$ for some index $z < j < r$, and either $j-1$ is the last index in both $\text{span}''(e), \text{span}''(e')$, or $j-1$ is the last index in one of these sets, while $j$ belongs to another.*

**Proof:** Fix an index $1 \leq z < r$ and a pair $e, e' \in \hat{E}_z$ of edges. Let $v$ be any vertex that lies on both $W(e)$ and $W(e')$. We consider two cases. The first case is when $v \in V''$, that is, there is some index $1 \leq z' \leq r$, such that $v \in V(\tilde{S}_z) \setminus V(S_z)$. Since all inner vertices of paths $P^{\mathsf{in}}(e), P^{\mathsf{in}}(e')$ lie in $V'$, vertex $v$ must be an inner vertex of both $P^{\mathsf{out}}(e)$ and $P^{\mathsf{out}}(e')$. From Claim 7.45, the intersection of $P^{\mathsf{out}}(e)$ and $P^{\mathsf{out}}(e')$ at vertex $v$ is non-transversal. Therefore, the intersection of $W(e)$ and $W(e')$ at $v$ is non-transversal.

The second case is when there is some index $1 \leq z' \leq r$, such that $v \in V(S_{z'})$. Assume that the intersection of $W(e)$ and $W(e')$ at $v$ is transversal. From our construction of the path set $\mathcal{R}_{z'}$, it must be the case that at least one of the edges $e, e'$ lies in $A'_{z'}$. Assume without loss of generality that this edge is $e$. Notice that, from the construction of set $A'_{z'}$, the last index of $\text{span}''(e)$ must be $z'-1$. Since $e' \in A_{z'} \cup A'_{z'}$, either $z' \in \text{span}''(e')$, or the last index in $\text{span}''(e')$ is $z'-1$. Therefore, if we denote by $j$ the last index of $\text{span}''(e')$, then $j \geq z'-1$ must hold. From our construction, $v$ is the only vertex of $S_{z'}$, such that the intersection of $P^{\mathsf{in}}(e)$ and $P^{\mathsf{in}}(e')$ at $v$ is transversal. Moreover, since $j \geq z'-1$, and $z'-1$ is the last index in $\text{span}''(e)$, for every index $z'' \neq z'$, for every vertex $v' \in S_{z''}$ that lies on both $P^{\mathsf{in}}(e)$ and $P^{\mathsf{in}}(e')$, the intersection of the two paths at $v'$ must be non-transversal. □

### 7.3.3 Step 3: Laminar Family $\mathcal{L}$ of Clusters, and Internal and External Routers for Clusters of $\mathcal{L}$

We define a laminar family $\mathcal{L} = \{U_1, \ldots, U_r\}$ of clusters of $G$, that will be later used in order to compute a decomposition of the input instance $I$ into subinstances.

For each $1 \leq i \leq r$, we define cluster $U_i$ to be the subgraph of $G$ induced by $\bigcup_{1 \leq z \leq i} V(\tilde{S}_z)$. For convenience, we also denote by $\overline{U}_i$ the subgraph of $G$ induced by $\bigcup_{i < z \leq r} V(\tilde{S}_z)$. We then define a laminar family $\mathcal{L} = \{U_1, \ldots, U_r\}$ of clusters of $G$. Notice that $U_r = G$, and $U_1 \subseteq U_2 \subseteq \cdots \subseteq U_r$.

We now fix an index $1 \leq i \leq r$ and consider the set $\delta_G(U_i) = E(U_i, \overline{U}_i)$ of edges. We can partition this edge set into two subsets: set $E_i = E(S_i, S_{i+1})$, and set $\hat{E}_i$, containing all remaining edges. Notice that $\hat{E}_i$ is precisely the set of all edges $e \in \hat{E}$ with $i \in \text{span}(e)$.

For all $1 \leq i \leq r$, we will define an internal router $\mathcal{Q}(U_i)$, and an external router $\mathcal{Q}'(U_i)$ for cluster $U_i$. The internal routers $\mathcal{Q}(U_i)$ of the clusters $U_i \in \mathcal{L}$ will be used in order to compute the decomposition of $I$ into subinstances. Both the internal and the external routers $\mathcal{Q}(U_i), \mathcal{Q}'(U_i)$ will be used in order to argue that the resulting instances have a relatively cheap solution. In order to define these routers, we first need to define, for all $1 \leq i \leq r$, an internal router $\mathcal{Q}(S_i)$ for cluster $S_i \in \mathcal{S}$. The algorithm for computing these routers is randomized, and is provided next.

**Algorithm for Computing Internal Routers for the Vertebrae.**

Recall that we have defined a parameter $\hat{\eta} = 2^{O((\log m)^{3/4} \log\log m)}$. We also use a new parameter $\beta^* = 2^{O(\sqrt{\log m} \cdot \log\log m)}$.

We now provide a randomized algorithm that computes, for each cluster $S_i \in \mathcal{S}$ an internal $S_i$-router $\mathcal{Q}(S_i)$. Additionally, we compute a partition $(\mathcal{S}^{\text{bad}}, \mathcal{S}^{\text{light}})$ of the clusters in $\mathcal{S}$. Initially, we set

$\mathcal{S}^{\text{bad}} = \mathcal{S}^{\text{light}} = \emptyset$. Recall that, from the definition of the nice witness structure, every cluster $S_i \in \mathcal{S}$ has the $\alpha^*$-bandwidth property, for $\alpha^* = \Omega(1/\log^{12} m)$.

For each $1 \leq i \leq r$ in turn, we apply Algorithm AlgClassifyCluster from Theorem 6.1 to instance $I = (G, \Sigma)$ of MCNwRS and cluster $J = S_i$, with parameter $p = 1/m^{100}$. If the algorithm returns a distribution $\mathcal{D}(S_i)$ over internal $S_i$-routers in $\Lambda(S_i)$, such that $S_i$ is $\beta^*$-light with respect to $\mathcal{D}(S_i)$, then we sample an internal $S_i$-router $\mathcal{Q}(S_i)$ from the distribution $\mathcal{D}(S_i)$, and we let $u_i$ be the vertex of $S_i$ that serves as the common endpoint of all paths in $\mathcal{Q}(S_i)$. We refer to vertex $u_i$ as the *center vertex of* $S_i$. We also add cluster $S_i$ to set $\mathcal{S}^{\text{light}}$ in this case. Otherwise, Algorithm AlgClassifyCluster returns FAIL. In this case, we add cluster $S_i$ to $\mathcal{S}^{\text{bad}}$, and we apply the algorithm from Corollary 4.28 to graph $G$ and cluster $S_i$. Let $\mathcal{D}(S_i)$ be the distribution over the set $\Lambda_G(S_i)$ of internal $S_i$-routers that the algorithm returns. We then let $\mathcal{Q}(S_i)$ be an internal $S_i$-router sampled from the distribution $\mathcal{D}(S_i)$. From Corollary 4.28, for each edge $e \in E(S_i)$, $\mathbf{E}\left[\text{cong}(\mathcal{Q}(S_i), e)\right] \leq O(\log^4 m/\alpha^*) = O(\log^{16} m)$.

**Bad Event $\mathcal{E}$.** For an index $1 \leq i \leq r$, we say that bad event $\mathcal{E}_i$ happens, if $S_i$ is not a $\hat{\eta}$-bad cluster, but Algorithm AlgClassifyCluster returned FAIL when applied to it. From Theorem 6.1, $\Pr[\mathcal{E}_i] \leq 1/m^{100}$. We also let $\mathcal{E}$ be the bad event that event $\mathcal{E}_i$ happens for any index $1 \leq i \leq r$. From the union bound, $\Pr[\mathcal{E}] \leq 1/m^{99}$.

Consider again any index $1 \leq i \leq r$. We will now slightly modify the paths in set $\mathcal{Q}(S_i)$, to ensure that they are non-traversal with respect to the rotation system $\Sigma$. In order to do so, we start by subdividing every edge $e \in \delta_G(S_i)$ with a vertex $t(e)$, and we let $X_i = \{t(e) \mid e \in \delta_G(S_i)\}$ be the resulting set of new vertices. We truncate the paths of $\mathcal{Q}(S_i)$, so that each such path originates at a distinct vertex of $X_i$ and terminates at vertex $u_i$. We then let $Y_i$ be the multiset of vertices containing the last vertex on every path of $\mathcal{Q}(S_i)$ (so $Y_i$ contains $|\mathcal{Q}(S_i)|$ copies of vertex $u_i$). We apply the algorithm from Lemma 4.7 to the resulting instance of MCNwRS, set $\mathcal{Q}(S_i)$ of paths and vertex multisets $X_i, Y_i$, to obtain another set $\tilde{\mathcal{Q}}(S_i)$ of paths that are non-transversal with respect to $\Sigma$. Recall that for every edge $e \in E(G)$, $\text{cong}_G(\tilde{\mathcal{Q}}(S_i), e) \leq \text{cong}_G(\mathcal{Q}(S_i), e)$, so in particular all inner vertices of the paths in $\tilde{\mathcal{Q}}(S_i)$ lie in $S_i$. The path set $\tilde{\mathcal{Q}}(S_i)$ naturally defines a set of paths (internal $S_i$-router) in graph $G$, that route the edges of $\delta_G(S_i)$ to vertex $u_i$, and are non-transversal with respect to $\Sigma$. For convenience of notation, we denote this internal $S_i$-router by $\mathcal{Q}(S_i)$ from now on. The following observation follows immediately from the above discussion, Theorem 6.1 and the definition of $\beta^*$-light and $\hat{\eta}$-bad clusters (see Definition 5.4 and Definition 5.5 in Section 5.2), and from the fact that $\beta^* \leq \hat{\eta}$.

**Observation 7.49** *For every cluster $S_i \in \mathcal{S}^{\text{light}}$, for every edge $e \in E(S_i)$: $\mathbf{E}\left[\left(\text{cong}_G(\mathcal{Q}(S_i), e)\right)^2\right] \leq \hat{\eta}$. Additionally, for every cluster $S_i \in \mathcal{S}^{\text{bad}}$, for every edge $e \in E(S_i)$: $\mathbf{E}\left[\text{cong}_G(\mathcal{Q}(S_i), e)\right] \leq O(\log^{16} m)$. Moreover, if $\mathcal{E}$ did not happen, then every cluster $S_i \in \mathcal{S}^{\text{bad}}$ is $\hat{\eta}$-bad, that is:*

$$\mathsf{OPT}_{\mathsf{cnwrs}}(S_i, \Sigma(S_i)) + |E(S_i)| \geq \frac{|\delta_G(S_i)|^2}{\hat{\eta}},$$

*where $\Sigma(S_i)$ is the rotation system for graph $S_i$ induced by $\Sigma$. Lastly, $\mathbf{Pr}\left[\mathcal{E}\right] \leq 1/m^{99}$.*

### Internal and External Routers for Clusters of $\Lambda$

Fix an index $1 \leq i \leq r$. We now define an internal router $\mathcal{Q}(U_i)$ and an external router $\mathcal{Q}'(U_i)$ for cluster $U_i \in \Lambda$. We will ensure that all paths of $\mathcal{Q}(U_i)$ terminate at the center vertex $u_i$ of $S_i$, and all paths of $\mathcal{Q}'(U_i)$ terminate at the center vertex $u_{i+1}$ of $S_{i+1}$. In order to do so, we consider the edges $e \in \delta_G(U_i)$ one by one. For each such edge $e$, we define a path $Q(e)$, whose first edge is $e$, last vertex is $u_i$, and all inner vertices lie in $U_i$, and we define a path $Q'(e)$, whose first edge is $e$, last vertex is $u_{i+1}$, and all inner vertices lie in $U_{i+1}$. We will then set $\mathcal{Q}(U_i) = \{Q(e) \mid e \in \delta(U_i)\}$, and $\mathcal{Q}'(U_i) = \{Q'(e) \mid e \in \delta(U_i)\}$.

We now fix an edge $e \in \delta_G(U_i)$, and define the two paths $Q(e), Q'(e)$. Recall that $\delta_G(U_i) = E_i \cup \hat{E}_i$. Assume first that $e \in E_i$. In this case, $e \in \delta_G(S_i)$ and $e \in \delta_G(S_{i+1})$ must hold. We let $Q(e)$ be the unique path of the internal $S_i$-router $\mathcal{Q}(S_i)$ whose first edge is $e$, and we let $Q'(e)$ be the unique path of the internal $S_{i+1}$-router $\mathcal{Q}(S_{i+1})$, whose first edge is $e$. As required, path $Q(e)$ connects $e$ to $u_i$ and only contains vertices of $U_i$ as inner vertices, while path $Q'(e)$ connects $e$ to $u_{i+1}$, and only contains vertices of $\overline{U}_i$ as inner vertices.

Assume now that $e \in \hat{E}_i$. We denote $e = (u, v)$, and we assume that $u$ is the left endpoint of $e$. Since $e \in \hat{E}_i$, $i \in \text{span}(e)$, and, since $\text{span}(e) \subseteq \text{span}''(e)$, we get that $i \in \text{span}''(e)$. From the construction of the path $P^{\text{in}}(e)$, it must contain an edge $\hat{e}_i \in E_i$. Additionally, it must contain some edge $\hat{e}_{i-1} \in \delta_G(S_i) \setminus \{\hat{e}_i\}$ and some edge $\hat{e}_{i+1} \in \delta_G(S_{i+1}) \setminus \{\hat{e}_i\}$ (if $e \in \delta_G(S_i)$, then $\hat{e}_{i-1} = e$, and if $e \in \delta_G(S_{i+1})$, then $\hat{e}_{i+1} = e$); see Figure 19. Let $\rho(e) \subseteq P^{\text{in}}(e)$ be the subpath of $P^{\text{in}}(e)$ that starts at edge $\hat{e}_{i-1}$ and terminates at edge $\hat{e}_{i+1}$. Consider now the graph that is obtained from the auxiliary cycle $W(e)$ by deleting the edge $e$, and all edges of $\rho(e)$ excluding $\hat{e}_{i-1}$ and $\hat{e}_{i+1}$. Once we delete all isolated vertices in the resulting graph, we obtain two contiguous paths. The first path, that we denote by $P$, originates at $u$ and has edge $\hat{e}_{i-1} \in \delta_G(S_i)$ as its last edge; all edges and vertices of $P$ lie in $U_i$ (if $e = \hat{e}_{i-1}$, then $P = \{u\}$). The second path, that we denote by $P'$, originates at $v$ and has $\hat{e}_{i+1} \in \delta_G(S_{i+1})$ as its last edge; all edges and vertices of $P'$ lie in $\overline{U}_i$ (if $e = \hat{e}_{i+1}$, then $P' = \{v\}$). We let $Q(e)$ be the path obtained by concatenating the edge $e$ with the path $P$, and the unique path of the internal $S_i$-router $\mathcal{Q}(S_i)$ that originates at edge $\hat{e}_{i-1}$ (see Figure 20). Clearly, path $Q(e)$ has $e$ as its first edge, $u_i$ as its last vertex, and all its inner vertices lie in $U_i$. Similarly, we let $Q'(e)$ be the path obtained by concatenating the edge $e$ with the path $P'$, and the unique path of the internal $S_{i+1}$-router $\mathcal{Q}(S_{i+1})$ that originates at edge $\hat{e}_{i+1}$. Clearly, path $Q'(e)$ has $e$ as its first edge, $u_{i+1}$ as its last vertex, and all its inner vertices lie in $\overline{U}_i$.

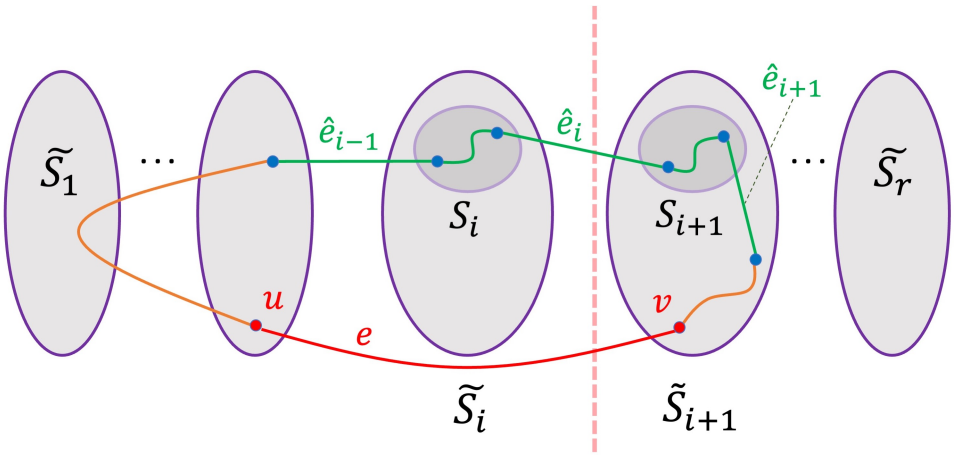



Figure 19: Construction of paths $Q(e)$ and $Q'(e)$ for an edge $e \in \hat{E}_i$. Path $\rho(e)$ is shown in green, and the cut $(U_i, V \setminus U_i)$ is shown in a pink dashed line.

Once every edge of $\delta_G(U_i)$ is processed, we set $\mathcal{Q}(U_i) = \{Q(e) \mid e \in \delta(U_i)\}$ and $\mathcal{Q}'(U_i) = \{Q'(e) \mid e \in \delta(U_i)\}$. It is immediate to verify that $\mathcal{Q}(U_i)$ is an internal $U_i$-router, while $\mathcal{Q}'(U_i)$ is an external $U_i$-router. Note that the construction of both routers is randomized, and the only randomized component in the construction is the selection of the internal routers for the vertebrae. We need the following simple observation.

**Observation 7.50** *For all $1 \leq i < r$, the set $\mathcal{Q}(U_i)$ of paths is non-transversal with respect to $\Sigma$. Additionally, for every pair $Q'(e), Q'(e') \in \mathcal{Q}'(U_i)$ of paths, there is at most one vertex $v$, such that $Q'(e)$ and $Q'(e')$ have a transversal intersection at $v$. If such a vertex $v$ exists, then $v$ is the unique vertex, such that the auxiliary cycles $W(e), W(e')$ have a transversal intersection at $v$.*

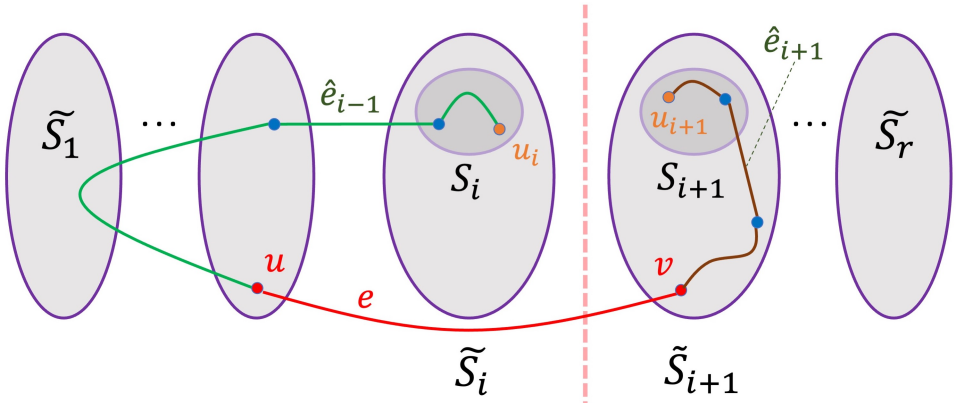



Figure 20: Paths $Q(e)$ is the union of edge $e$ and the green path; path $Q'(e)$ is the union of edge $e$ and the brown path.

**Proof:** We start by considering a pair of paths $Q(e), Q(e') \in \mathcal{Q}(U_i)$. Let $v$ be any vertex that serves as an inner vertex on both paths. We consider three cases. First, if $v \in V''$, then $e, e' \in \hat{E}_i$ must hold, and, from Observation 7.48, the intersection of $Q(e)$ and $Q(e')$ at $v$ is non-transversal. The second case is when $v \in V(S_i)$. In this case, there must be two paths $\tilde{Q}_1, \tilde{Q}_2 \in \mathcal{Q}(S_i)$, such that $\tilde{Q}_1 \subseteq Q(e)$, $\tilde{Q}_2 \subseteq Q(e')$, and $v$ is an inner vertex on both $\tilde{Q}_1$ and $\tilde{Q}_2$ (note that it is possible that $\tilde{Q}_1 = \tilde{Q}_2$). In this case, from the construction of the internal $S_i$-router $\mathcal{Q}(S_i)$, the intersection of $Q(e)$ and $Q(e')$ at $v$ is non-transversal. The third case is when $v \in V' \setminus V(S_i)$. In this case, $e, e' \in \hat{E}_i$ must hold. Assume that $v \in V(S_z)$ for some index $z < i$. Clearly, $z$ may not be the last index of $\text{span}''(e)$ or of $\text{span}''(e')$. Therefore, from Observation 7.48, the intersection of $Q(e)$ and of $Q(e')$ at $v$ is non-transversal. We conclude that the set $\mathcal{Q}(U_i)$ of paths is non-transversal with respect to $\Sigma$.

Consider now some pair $Q'(e), Q'(e') \in \mathcal{Q}'(U_i)$ of paths. Let $v$ be any vertex that serves as an inner vertex on both paths. We again consider three cases. First, if $v \in V''$, then $e, e' \in \hat{E}_i$ must hold, and, from Observation 7.48, the intersection of $Q'(e)$ and $Q'(e')$ at $v$ is non-transversal. The second case is when $v \in V(S_{i+1})$. In this case, there must be two paths $\tilde{Q}_1, \tilde{Q}_2 \in \mathcal{Q}(S_{i+1})$, such that $\tilde{Q}_1 \subseteq Q'(e)$, $\tilde{Q}_2 \subseteq Q'(e')$, and $v$ is an inner vertex on both $\tilde{Q}_1$ and $\tilde{Q}_2$. In this case, from the construction of the internal $S_{i+1}$-router $\mathcal{Q}(S_{i+1})$, the intersection of $Q'(v)$ and $Q'(v')$ at $v$ is non-transversal. The third case is when $v \in V' \setminus V(S_i)$. In this case, $e, e' \in \hat{E}_i$ must hold, and $v$ also lies on cycles $W(e)$ and $W(e')$. Moreover, the intersection of $W(e)$ and $W(e')$ must be transversal at $v$. From Observation 7.48, there may be at most one vertex $v'$, such that the intersection of $W(e)$ and $W(e')$ at $v'$ is transversal. We conclude that there is at most one vertex $v \in V(Q'(e)) \cap V(Q'(e'))$, such that the intersection of $Q'(e)$ and $Q'(e')$ at $v$ is transversal. If such a vertex $v$ exists, then $v$ is the unique vertex, such that the auxiliary cycles $W(e), W(e')$ have a transversal intersection at $v$. □

From Observation 7.47, and the construction of the internal and external $U_i$-routers, we obtain the following immediate observation.

**Observation 7.51** *For all $1 \leq i < r$, an edge $e \in E(S_i)$ may appear on at most $O(\log^{34} m) \cdot \text{cong}_G(\mathcal{Q}(S_i), e)$ paths of $\mathcal{Q}(U_i)$, and an edge $e \in E(U_i) \setminus E(S_i)$ may appear on at most $O(\log^{34} m)$ paths of $\mathcal{Q}(U_i)$. Similarly, an edge $e \in E(S_{i+1})$ may appear on at most $O(\log^{34} m) \cdot \text{cong}_G(\mathcal{Q}(S_{i+1}), e)$ paths of $\mathcal{Q}'(U_i)$, and an edge $e \in E(\overline{U}_i) \setminus E(S_{i+1})$ may appear on at most $O(\log^{34} m)$ paths of $\mathcal{Q}'(U_i)$.*

We will also use the following simple corollary of the observation.

**Corollary 7.52** *For all $1 \leq z \leq r$, $|\delta_G(U_z)| \leq |\delta_G(S_z)| \cdot O(\log^{34} m)$, and $|\delta_G(U_z)| \leq |\delta_G(S_{z+1})| \cdot O(\log^{34} m)$.*

**Proof:** Recall that we have defined a collection $\mathcal{Q}(U_z)$ of paths, routing the edges of $\delta_G(U_z)$ to vertex $u_z$. Each such path must contain an edge of $\delta_G(S_z)$. Moreover, from Observation 7.51, an edge of $\delta_G(S_z)$ may lie on at most $O(\log^{34} m)$ paths of $\mathcal{Q}(U_z)$. Therefore, $|\delta_G(U_z)| \leq |\delta_G(S_z)| \cdot O(\log^{34} m)$.

Similarly, we have defined a collection $\mathcal{Q}'(U_z)$ of paths, routing the edges of $\delta_G(U_z)$ to vertex $u_{z+1}$. Each such path must contain an edge of $\delta_G(S_{z+1})$. From Observation 7.51, an edge of $\delta_G(S_{z+1})$ may lie on at most $O(\log^{34} m)$ paths of $\mathcal{Q}'(U_z)$. Therefore, $|\delta_G(U_z)| \leq |\delta_G(S_{z+1})| \cdot O(\log^{34} m)$. □

### 7.3.4 Step 4: Constructing the Collection of Subinstances

Consider again an index $1 \leq i < r$, and the internal $U_i$-router $\mathcal{Q}(U_i) = \{Q(e) \mid e \in \delta_G(U_i)\}$. Recall that all paths in $\mathcal{Q}(U_i)$ terminate at vertex $u_i$. Denote $\delta_G(u_i) = \left\{e^i_1, \ldots, e^i_{|\delta(u_i)|}\right\}$, where the edges are indexed according to their order in $\mathcal{O}_{u_i} \in \Sigma$. For all $1 \leq j \leq |\delta(u_i)|$, let $A^i_j \subseteq \delta_G(U_i)$ be the set of all edges $e \in \delta_G(U_i)$, such that the uniue path $Q(e) \in \mathcal{Q}(U_i)$ that has $e$ as its first edge contains edge $e^i_j$ as its last edge. We now define an ordering $\tilde{\mathcal{O}}_i$ of the edges of $\delta_G(U_i)$: the edges in sets $A^i_1, A^i_2, \ldots, A^i_{|\delta(u_i)|}$ appear in the order of the indices of their sets, and, for each $1 \leq j \leq |\delta(u_i)|$, the ordering of the edges in set $A^i_j$ is arbitrary. Notice that the resulting ordering $\tilde{\mathcal{O}}_i$ of the edges of $\delta_G(U_i)$ is precisely the ordering $\mathcal{O}^{\text{guided}}(\mathcal{Q}(U_i), \Sigma)$, that is guided by the internal $U_i$-router $\mathcal{Q}(U_i)$ (see the definition in Section 5.2). For $i = r$, $\delta_G(U_i) = \emptyset$, and the ordering $\tilde{\mathcal{O}}_r$ of the edges of $\delta_G(U_i)$ is the trivial one.

We let $\mathcal{I}_2$ be a collection of subinstances of $I$ obtained by computing a laminar family-based decomposition of $I$ (defined in Section 5.1) via the laminar family $\mathcal{L} = \{U_i\}_{1 \leq i \leq r}$ of clusters, and the orderings $\tilde{\mathcal{O}}_i$ of the edge sets $\delta(U_i)$ for all $1 \leq i \leq r$. We denote $\mathcal{I}_2 = \{I_1, \ldots, I_r\}$, where, for $1 \leq z \leq r$, instance $I_z = (G_z, \Sigma_z)$ is the instance associated with the cluster $U_z$. Recall that instance $I_z$ is defined as follows. Assume first that $1 < z < r$. Then graph $G_z$ is obtained from graph $G$ by first contracting all vertices of $U_{z-1}$ into vertex $v^*_z$, and then contracting all vertices of $U_{z+1}$ into vertex $v^{**}_z$. Notice that, equivalently, graph $G_z$ consists of the cluster $\tilde{S}_z \in \tilde{\mathcal{S}}$, the two special vertices $v^*_z, v^{**}_z$, and possibly some additional edges that are incident to these two special vertices.

The rotation system $\Sigma_z$ is defined as follows. Observe first that, for every vertex $v \in V(G_z) \setminus \{v^*_z, v^{**}_z\}$, $\delta_{G_z}(v) = \delta_G(v)$. The ordering $\mathcal{O}_v \in \Sigma_z$ of the edges incident to $v$ in $\Sigma_z$ remains the same as in $\Sigma$. Notice that $\delta_{G_z}(v^*_z) = \delta_G(U_{z-1})$. We let the ordering $\mathcal{O}_{v^*_z} \in \Sigma_z$ of the edges of $\delta_{G_z}(v^*_z)$ be the ordering $\tilde{\mathcal{O}}_{z-1}$ that we defined above. Lastly, observe that $\delta_{G_z}(v^{**}_z) = \delta_G(U_z)$. We let the ordering $\mathcal{O}_{v^{**}_z} \in \Sigma_z$ of the edges of $\delta_{G_z}(v^{**}_z)$ be the ordering $\tilde{\mathcal{O}}_z$ that we defined above. For $z = 1$, instance $I_1 = (G_1, \Sigma_1)$ is defined similarly, except that the instance does not contain vertex $v^*_1$. Instance $I_r = (G_r, \Sigma_r)$ is also defined similarly, except that it does not contain vertex $v^{**}_r$.

We now verify that the resulting collection $\mathcal{I}_2$ of instances has all required properties. Fix some index $1 \leq z \leq r$. Recall that, from the definition of a nice witness structure, there is at most one cluster $C \in \mathcal{C}$ with $C \subseteq \tilde{S}_z$. Recall also that, for each cluster $C \in \mathcal{C}$, there is exactly one cluster $S_i \in \mathcal{S}$ that contains $C$. If some cluster $C \in \mathcal{C}$ is contained in $\tilde{S}_z$, then $E(\tilde{S}_z) \subseteq E(C) \cup E(G_{|\mathcal{C}})$ must hold, and so $E(G_z) \subseteq E(C) \cup E(G_{|\mathcal{C}})$ as well. Otherwise, $E(\tilde{S}_z) \subseteq E(G_{|\mathcal{C}})$, and so $E(G_z) \subseteq E(G_{|\mathcal{C}})$.

From Lemma 5.3, there is an efficient algorithm, that, given, for each instance $I_z \in \mathcal{I}$, a solution $\varphi_z$, computes a solution for instance $I$ of value at most $\sum_{t=1}^{r} \mathsf{cr}(\varphi_z)$. Next, we bound the total number of edges in all resulting instances.

**Observation 7.53** $\sum_{1 \leq z \leq r} |E(G_z)| \leq O(|E(G)| \cdot \log^{34} m)$.

**Proof:** Fix an index $1 \leq z \leq r$. From our construction, $|E(G_z)| \leq |E(\tilde{S}_z)| + |\delta_G(U_{z-1})| + |\delta_G(U_z)|$. From Corollary 7.52, $|\delta_G(U_{z-1})| \leq |\delta_G(S_{z-1})| \cdot O(\log^{34} m)$, and $|\delta_G(U_z)| \leq |\delta_G(S_{z+1})| \cdot O(\log^{34} m)$.

Therefore, $|E(G_z)| \leq |E(\tilde{S}_z)| + (|\delta_G(S_{z-1})| + |\delta_G(S_{z+1})|) \cdot O(\log^{34} m)$. Summing up over all indices $z$, we get that:

$$\sum_{z=1}^{r} |E(G_z)| \leq \sum_{z=1}^{r} |E(\tilde{S}_z)| + O(\log^{34} m) \cdot \sum_{z=1}^{r} |\delta_G(S_z)| \leq O(|E(G)| \cdot \log^{34} m).$$

□

Recall that we have used a randomized algorithm to compute, for all $1 \leq i \leq r$, an internal $U_i$-router $\mathcal{Q}(U_i)$ and an external $U_i$-router $\mathcal{Q}'(U_i)$. In order to complete the proof of Theorem 7.4, it is now enough to show that the expected total optimal solution costs of all instances in $\mathcal{I}_2$ (over the random choices performed by the algorithm that computed the internal and the external $U_i$-routers) is bounded by $2^{O((\log m)^{3/4} \log\log m)} \cdot (\mathsf{OPT}_{\mathsf{cnwrs}}(I) + |E(G)|)$. The following claim, whose proof appears in Section 7.4 will finish the proof of Theorem 7.4.

**Claim 7.54** $\mathbf{E}\left[\sum_{z=1}^{r} \mathsf{OPT}_{\mathsf{cnwrs}}(I_z)\right] \leq 2^{O((\log m)^{3/4} \log\log m)} \cdot (\mathsf{OPT}_{\mathsf{cnwrs}}(I) + |E(G)|)$.

## 7.4 Proof of Claim 7.54

Notice that graphs $G_1, G_r$ have a somewhat different structure than graphs of $\{G_z\}_{1<z<r}$: specifically, graph $G_1$ does not contain vertex $v_1^*$, and graph $G_r$ does not contain vertex $v_r^{**}$. It would be convenient for us to modify these two graphs so that we can treat all resulting graphs uniformly. In order to do so, we add a new dummy vertex $v_1^*$ to graph $G_1$, and connect it with an edge to an arbitrary vertex $v_1 \in S_1$. We modify the rotation $\mathcal{O}_{v_1} \in \Sigma_1$ to include the new edge $(v_1, v_1^*)$ at an arbitrary position in the rotation. Notice that any solution to the resulting new instance $I_1 = (G_1, \Sigma_1)$ of $\mathsf{MCNwRS}$ immediately provides a solution to the original instance $I_1 = (G_1, \Sigma_1)$, of the same cost. We similarly modify graph $G_r$, by adding a new dummy vertex $v_r^{**}$, which is connected with an edge to an arbitrary vertex $v_r \in S_r$. We modify the rotation $\mathcal{O}_{v_r} \in \Sigma_r$ as before. In order to be consistent, we also add the vertices $v_1^*, v_r^{**}$, and edges $(v_1^*, v_1)$ and $(v_r^{**}, v_r)$ to the original graph $G$, and modify the rotations $\mathcal{O}_{v_1}, \mathcal{O}_{v_r} \in \mathcal{O}$, so that they remain consistent with the rotations $\mathcal{O}_{v_1} \in \Sigma_1$ and $\mathcal{O}_{v_r} \in \Sigma_r$, respectively. Notice that this modification does not increase $\mathsf{OPT}_{\mathsf{cnwrs}}(I)$. We also modify the nice witness structure, by adding two new clusters $\tilde{S}_0 = \{v_1^*\}$ and $\tilde{S}_{r+1} = \{v_r^{**}\}$ to $\tilde{\mathcal{S}}$, and their subclusters $S_0 = \{v_1^*\}$ and $S_{r+1} = \{v_r^{**}\}$. We note that all the above modifications are only performed for ease of exposition and are not strictly necessary.

Let $\varphi^*$ be an optimal solution to instance $I$ of $\mathsf{MCNwRS}$. We can assume that no pair of edges cross twice, and that the image of each edge does not cross itself in $\varphi^*$. We denote by $\chi^*$ the set of all unordered pairs $(e, e')$ of edges of $G$, such that the images of $e$ and $e'$ cross in $\varphi^*$. For all $1 \leq z \leq r$, we denote by $\chi_z^* \subseteq \chi^*$ the set of all unordered pairs $(e, e')$ of edges of $G$ whose images cross, such that either $e \in E(\tilde{S}_z) \cup \delta_G(\tilde{S}_z)$, or $e' \in E(\tilde{S}_z) \cup \delta_G(\tilde{S}_z)$, or both. We will use the drawing $\varphi^*$ in order to construct, for each $1 \leq z \leq r$, a solution $\varphi_z$ to instance $I_z = (G_z, \Sigma_z)$

For each $1 \leq z \leq r$, we construct a solution $\varphi_z$ to instance $I_z$, and then argue that the total expected costs of all these solutions is relatively small. We now fix an index $1 \leq z \leq r$, and focus on constructing solution $\varphi_z$ to instance $I_z$. The construction of the solution consists of four steps. In the first step, we construct an auxiliary graph $H_z$ and its drawing $\psi_z$. This graph and its drawing are used in the second step, in order to construct an initial drawing $\varphi_z'$ of graph $G_z$. The number of crossings in drawing $\varphi_z'$ may be quite large, and we modify the drawing in order to lower the number of crossings in the third step. The resulting drawing, $\varphi_z''$, will have a sufficiently low expected number of crossings, but unfortunately it may not obey the rotation $\mathcal{O}_{v_z^{**}} \in \Sigma_z$. In the fourth and the last step, we modify this drawing in order to obtain a feasible solution $\varphi_z$ to instance $I_z$ of $\mathsf{MCNwRS}$, while only slightly increasing the number of crossings. We now fix an index $1 \leq z \leq r$, and describe a construction of a solution $\varphi_z$ for instance $I_z$ of $\mathsf{MCNwRS}$ step by step.

### 7.4.1 Step 1: Computing Auxiliary Graph $H_z$ and Its Drawing

Recall that graph $G_z$ is obtained from graph $G$ by contracting all vertices of $\bigcup_{1\leq i<z} V(\tilde{S}_i)$ into the special vertex $v_z^*$, and then contracting all vertices of $\bigcup_{z<i\leq r}$ into the special vertex $v_z^{**}$. Clearly, every edge of $G_z$ corresponds to some edge of $G$, and we do not distinguish between these edges. In order to simplify the notation, when the index $z$ is fixed, we denote vertices $v_z^*$ and $v_z^{**}$ by $v^*$ and $v^{**}$, respectively. Notice that $\delta_{G_z}(v^*) = \delta_G(U_{z-1}) = E_{z-1} \cup \hat{E}_{z-1}$, while $\delta_{G_z}(v^{**}) = \delta_G(U_z) = E_z \cup \hat{E}_z$. In order to obtain the drawing $\varphi_z$ of graph $G_z$, we will exploit the internal $U_{z-1}$-router $\mathcal{Q}(U_{z-1})$, that routes the edges of $\delta_G(U_{z-1})$ to vertex $u_{z-1}$, and the external $U_z$-router $\mathcal{Q}'(U_z)$, that routes the edges of $\delta_G(U_z)$ to vertex $u_{z+1}$. For each edge $e \in \delta_G(U_{z-1})$, we denote by $Q(e)$ the unique path of $\mathcal{Q}(U_{z-1})$ whose first edge is $e$, and for each edge $e \in \delta_G(U_z)$, we denote by $Q'(e)$ the unique path of $\mathcal{Q}'(U_z)$ whose first edge is $e$.

Denote $\mathcal{Q}_z^* = \mathcal{Q}(U_{z-1}) \cup \mathcal{Q}(U_z)$. For every edge $e \in E(G)$, we define a value $N_z(e)$, as follows. If $e \in E(G) \setminus E(G_z)$, then $N_z(e) = \mathrm{cong}_G(\mathcal{Q}_z^*, e)$ – the number of paths in $\mathcal{Q}_z^*$ that contain the edge $e$. For each edge $e \in \delta_G(U_{z-1}) \cup \delta_G(U_z) \cup E(\tilde{S}_z)$, we set $N_z(e) = 1$. The will use the following observation.

**Observation 7.55** *Let $e$ be an edge of $E(G) \setminus E(G_z)$. If $e \notin E(S_{z-1}) \cup E(S_{z+1})$, then $N_z(e) \leq O(\log^{34} m)$; otherwise, $\mathbf{E}[N_z(e)] \leq \hat{\eta}$. Moreover, if $e \in E(S_{z-1})$ and $S_{z-1} \in \mathcal{S}^{\mathrm{light}}$, then $\mathbf{E}\left[(N_z(e))^2\right] \leq \hat{\eta}$. Similarly, if $e \in E(S_{z+1})$ and $S_{z+1} \in \mathcal{S}^{\mathrm{light}}$, then $\mathbf{E}\left[(N_z(e))^2\right] \leq \hat{\eta}^2$. (All expectations here are over the selections of the internal routers $\mathcal{Q}(S_{z-1})$ and $\mathcal{Q}(S_{z+1})$).*

**Proof:** Consider an edge $e \in E(G) \setminus E(G_z)$. Notice that either $e \in E(U_{z-1})$, or $e \in E(\overline{U}_z)$ must hold. We assume that it is the former; the other case is symmetric. In this case, $N_z(e) = \mathrm{cong}_G(\mathcal{Q}(U_{z-1}), e)$, and, from Observation 7.51, $N_z(e) \leq O(\log^{34} m)$.

Assume now that $e \in E(U_{z-1})$. From Observation 7.51, edge $e$ may appear on at most $\mathrm{cong}_G(\mathcal{Q}(S_{z-1}), e) \cdot O(\log^{34} m)$ paths of $\mathcal{Q}(U_{z-1})$, that is, $N_z(e) \leq \mathrm{cong}_G(\mathcal{Q}(S_{z-1}), e) \cdot O(\log^{34} m)$. From Observation 7.49, for $1 \leq i \leq r$, if $S_i \in \mathcal{S}^{\mathrm{light}}$, then $\mathbf{E}\left[\left(\mathrm{cong}_G(\mathcal{Q}(S_i), e)\right)^2\right] \leq \hat{\eta}$, while, if $S_i \in \mathcal{S}^{\mathrm{bad}}$, then $\mathbf{E}\left[\mathrm{cong}(\mathcal{Q}(S_i), e)\right] \leq O(\log^{16} m) \leq \hat{\eta}$. The observation now follows immediately. □

We now construct an auxiliary graph $H_z$, and its drawing $\psi_z$. In order to do so, we start with $H_z = G$, and $\psi_z = \varphi^*$. We call the edges of $E(\tilde{S}_z) \cup \delta_G(\tilde{S}_z)$ *primary edges*, and the remaining edges of $G$ *secondary edges*. We now process every secondary edge $e$ one by one. If edge $e$ does not participate in any path of $\mathcal{Q}_z^*$ (that is, $N_z(e) = 0$), then we delete $e$ from $H_z$ and we delete its image from $\psi_z$. Otherwise, we replace $e$ with a set $J(e)$ of $N_z(e)$ parallel copies of $e$ in graph $H_z$, and we replace the image of $e$ in $\psi_z$ with images of these copies, that follow the original image of $e$ in parallel to it, without crossing each other. For convenience, for each edge $e \in E(\tilde{S}_z) \cup \delta_G(U_{z-1}) \cup \delta_G(U_z)$, we define $J(e) = \{e\}$, and we think of the graph $H_z$ as having a single copy of the edge $e$ (the edge $e$ itself). This completes the definition of the graph $H_z$ and its drawing $\psi_z$.

For every edge $e \in E(G) \setminus E(G_z)$, we can now assign, to every path of $\mathcal{Q}_z^*$ containing $e$, a distinct copy of this edge from $J(e)$. If edge $e \notin \delta_G(u_{z-1})$, we assign each copy of $e$ in $J(e)$ to a distinct path of $\mathcal{Q}_z^*$ containing $e$ arbitrarily. If edge $e \in \delta_G(u_{z-1})$, then we perform the assignment more carefully. Intuitively, this assignment is performed in a way that is consistent with the ordering $\tilde{\mathcal{O}}_{z-1}$ of the edges of $\delta_G(U_{z-1})$ that we have defined, and the ordering of the paths of set $\mathcal{Q}(U_{z-1}) = \{Q(e) \mid e \in \delta_G(U_{z-1})\}$ that it induces.

**Assigning the copies of edges of $\delta_G(u_{z-1})$ to paths.** Consider the set $\delta_G(U_{z-1})$ of edges. Recall that we have defined an ordering $\tilde{\mathcal{O}}_{z-1}$ of the edges of $\delta_G(U_{z-1})$, which is precisely the ordering $\mathcal{O}^{\mathrm{guided}}(\mathcal{Q}(U_{z-1}), \Sigma)$, that is guided by the internal $U_{z-1}$-router $\mathcal{Q}(U_{z-1})$. Denote $\delta_G(U_{z-1}) = \{\hat{a}_1, \hat{a}_2, \ldots, \hat{a}_q\}$, where the edges are indexed according to the ordering $\tilde{\mathcal{O}}_{z-1}$.

Recall the procedure that we used in order to define the ordering $\tilde{\mathcal{O}}_{z-1}$ of the edges of $\delta_G(U_{z-1})$ (for convenience we omit the superscript $z-1$): we have denoted $\delta_G(u_{z-1}) = \left\{e_1, \ldots, e_{|\delta(u_{z-1})|}\right\}$, where the edges are indexed according to their order in the rotation $\mathcal{O}_{u_{z-1}} \in \Sigma$. For all $1 \leq j \leq |\delta(u_{z-1})|$, we denoted by $A_j \subseteq \delta_G(U_{z-1})$ the set of all edges $e' \in \delta_G(U_{z-1})$, such that the unique path $Q(e') \in \mathcal{Q}(U_{z-1})$ originating at edge $e'$ terminates at edge $e_j$.

We have defined the ordering $\tilde{\mathcal{O}}_{z-1} = (\hat{a}_1, \hat{a}_2, \ldots, \hat{a}_q)$ of the edges of $\delta_G(U_{z-1})$ as follows: the edges that lie in sets $A_1, A_2, \ldots, A_{|\delta(u_{z-1})|}$ appear in the order of the indices of their sets, and, for each $1 \leq j \leq |\delta(u_{z-1})|$, the ordering of the edges within each set $A_j$ is arbitrary; denote this latter ordering by $\hat{\mathcal{O}}_j = \left\{a_1^j, a_2^j, \ldots, a_{q_j}^j\right\}$. The current drawing $\psi_z$ of graph $H_z$ naturally defines a circular ordering of the edges of $\delta_H(u_{z-1})$, which is precisely the order in which the images these edges enter the image of $u_{z-1}$. In this circular ordering, the edges of each set $J(e_1), J(e_2), \ldots, J(e_{|\delta_G(u_{z-1})|})$ appear consecutively, in the order of the indices of their sets. For each index $1 \leq j \leq |\delta_G(u_{z-1})|$, the above circular ordering defines an ordering $\hat{\mathcal{O}}'_j$ of the edges of $J(e_j)$.

Consider now some edge $e_j \in \delta_G(u_{z-1})$, and assume that $|J(e_j)| = q_j$. On the one hand, we have defined the ordering $\hat{\mathcal{O}}'_j$ of the edges of $J(e_j)$ – the order in which the images of these edges in $\psi_z$ enter the image of $u_{z-1}$. On the other hand, we have defined an ordering $\hat{\mathcal{O}}_j = \left\{a_1^j, a_2^j, \ldots, a_{q_j}^j\right\}$ of the edges of $A_j$ – that is, the edges $e' \in \delta_G(u_{z-1})$, whose corresponding path $Q(e')$ contains edge $e_j$. For all $1 \leq h \leq q_j$, we then assign the $h$th edge of $J(e_j)$ in the ordering $\hat{\mathcal{O}}'_j$ to path $Q(a_h^j)$. This completes the assignment of edges of $H_z$ that are incident to vertex $u_{z-1}$ to the paths of $\mathcal{Q}(U_{z-1})$.

For every edge $\hat{a}_i \in \delta_G(U_{z-1})$, we can now obtain a path $\hat{Q}(\hat{a}_i)$ in graph $H_z$, that originates at edge $\hat{a}_i$ and terminates at vertex $u_{z-1}$, with all inner vertices of $\hat{Q}(a_i)$ lying in $V(U_{z-1})$, by starting from the path $Q(\hat{a}_i) \in \mathcal{Q}(U_{z-1})$, and replacing every edge $e' \in E(G) \setminus E(G_z)$ with the copy of $e'$ that is assigned to path $Q(\hat{a}_i)$. Denote the resulting set of paths in graph $H_z$ by $\hat{\mathcal{Q}}_z = \left\{\hat{Q}(\hat{a}_i) \mid \hat{a}_i \in \delta_G(U_{z-1})\right\}$. For each edge $\hat{a}_i \in \delta_G(U_{z-1})$, denote by $\hat{a}'_i$ the last edge on path $\hat{Q}(\hat{a}_i)$. Then the paths of $\hat{\mathcal{Q}}_z$ are mutually edge-disjoint, and they route the edges of $\delta_G(U_{z-1})$ to vertex $u_{i-1}$ in $H_z$. All inner vertices on the paths of $\hat{\mathcal{Q}}_z$ lie in $V(U_{z-1})$. Moreover, the images of edges $\hat{a}'_1, \ldots, \hat{a}'_q$ enter the image of $u_{i-1}$ in the drawing $\psi_z$ of $H_z$ in the circular order of their indices (and recall that edges $\hat{a}_1, \ldots, \hat{a}_q$ are indexed in the order of their appearance in $\tilde{\mathcal{O}}_{z-1}$).

Similarly, for every edge $a \in \delta_G(U_z)$, we can now obtain a path $\hat{Q}'(a)$ in graph $H_z$, that originates at edge $a$ and terminates at vertex $u_{z+1}$, with all inner vertices of $\hat{Q}'(a)$ lying in $V(\overline{U}_z)$, by starting from the path $Q'(a) \in \mathcal{Q}'(U_z)$, and replacing every edge $e' \in E(G) \setminus E(G_z)$ with the copy of $e'$ that is assigned to path $Q(a)$. Denote the resulting set of paths in graph $H_z$ by $\hat{\mathcal{Q}}'_z = \left\{\hat{Q}'(a) \mid a \in \delta_G(U_z)\right\}$. Then the paths of $\hat{\mathcal{Q}}'_z$ are mutually edge-disjoint, and they route the edges of $\delta_G(U_z)$ to vertex $u_{i+1}$. All inner vertices on paths of $\hat{\mathcal{Q}}'_z$ lie in $V(\overline{U}_z)$. This completes the construction of graph $H_z$ and its drawing $\psi_z$. We now analyze the number of crossings in this graph.

**Bounding the Number of Crossings in $\psi_z$.** Recall that $\delta_G(U_{z-1}) = E_{z-1} \cup \hat{E}_{z-1}$, while $\delta_G(U_z) = E_z \cup \hat{E}_z$. We denote $E_z^{\text{over}} = \hat{E}_{z-1} \cap \hat{E}_z$; note that every edge $e \in E_z^{\text{over}}$ has one endpoint in $\bigcup_{1 \leq i < z} V(\tilde{S}_i)$, and another endpoint in $\bigcup_{z < i \leq r} V(\tilde{S}_i)$ (see Figure 21). We also denote by $E_z^{\text{left}} = \hat{E}_{z-1} \setminus E_z^{\text{over}}$, and by $E_z^{\text{right}} = \hat{E}_z \setminus E_z^{\text{over}}$. Notice that every edge $e \in E_z^{\text{left}}$ has one endpoint in $\bigcup_{1 \leq i < z} V(\tilde{S}_i)$, and another endpoint in $V(\tilde{S}_z)$, while every edge $e \in E_z^{\text{right}}$ has one endpoint in $V(\tilde{S}_z)$ and another endpoint in $\bigcup_{z < i \leq r} V(\tilde{S}_i)$ (see Figure 21). From the above definitions, $\delta_{G_z}(v^*) = \delta_G(U_{z-1}) = E_{z-1} \cup E_z^{\text{left}} \cup E_z^{\text{over}}$, and $\delta_{G_z}(v^{**}) = \delta_G(U_z) = E_z \cup E_z^{\text{right}} \cup E_z^{\text{over}}$.

We now reorganize the paths in $\hat{\mathcal{Q}}_z \cup \hat{\mathcal{Q}}'_z$ as follows. We let $\mathcal{R}'_1 = \left\{\hat{Q}(e) \mid e \in E_{z-1} \cup E_z^{\text{left}}\right\}$, $\mathcal{R}''_1 =$

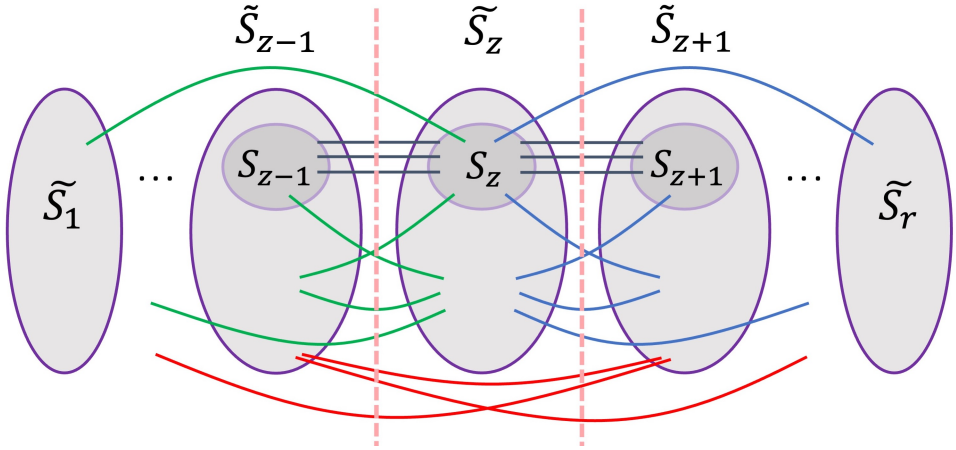



Figure 21: Set $E_z^{\text{over}}$ of edges is shown in red, set $E_z^{\text{right}}$ in blue, and set $E_z^{\text{left}}$ in green. The left pink dashed line shows the cut $(U_{z-1}, V \setminus U_{z-1})$, and the right pink dashed line shows the cut $(U_z, V \setminus U_z)$.

$\left\{\hat{Q}'(e) \mid e \in E_z \cup E_z^{\text{right}}\right\}$, and $\mathcal{R}_1^{(z)} = \mathcal{R}_1' \cup \mathcal{R}_1''$. For each edge $e \in E_{z-1} \cup E_z^{\text{left}} \cup E_z^{\text{right}} \cup E_z$, we denote by $R(e) \in \mathcal{R}_1^{(z)}$ the unique path that has edge $e$ as its first edge. For every edge $e \in E_z^{\text{over}}$, we let $R(e)$ be the concatenation of the paths $\hat{Q}(e) \in \hat{\mathcal{Q}}_z$ and $\hat{Q}'(e) \in \hat{\mathcal{Q}}_z'$, so that path $R(e)$ is a simple path connecting vertices $u_{z-1}$ and $u_{z+1}$, and it contains the edge $e$. We then set $\mathcal{R}_2^{(z)} = \{R(e) \mid e \in E_z^{\text{over}}\}$.

For every secondary edge $e'$ in graph $G$, we denote by $N_z'(e')$ the number of paths in set $\mathcal{R}_1^{(z)}$ that contain a copy of $e'$, and we denote by $N_z''(e')$ the number of paths in set $\mathcal{R}_2^{(z)}$ that contain a copy of $e'$. Note that, equivalently, $N_z'(e')$ is the total number of paths in $\mathcal{Q}(U_{z-1}) \cup \mathcal{Q}'(U_z)$ that originate at edges of $E_{z-1} \cup E_z^{\text{left}} \cup E_z^{\text{right}} \cup E_z$ and contain $e'$, while $N_z''(e')$ is the total number of paths in $\mathcal{Q}(U_{z-1}) \cup \mathcal{Q}'(U_z)$ that originate at edges of $E_z^{\text{over}}$ and contain $e'$. For a primary edge $e'$, we set $N_z'(e') = 1$ and $N_z''(e') = 0$. Clearly, $N_z(e') = N_z'(e') + N_z''(e')$ holds for every edge $e'$. Intuitively, for each edge $e'$, the value $\sum_{z=1}^{r} N_z'(e')$ is relatively small, while the value $\sum_{z=1}^{r} N_z''(e')$ may be quite large. Indeed, recall that the paths in set $\mathcal{Q}(U_{z-1}) \cup \mathcal{Q}'(U_z)$ can be thought of as constructed by composing subpaths of cycles of $\left\{W(e') \mid e' \in \hat{E}_{z-1} \cup \hat{E}_z\right\}$ with the internal routers $\mathcal{Q}(S_{z-1})$ and $\mathcal{Q}(S_{z+1})$. Consider an edge $e \in \hat{E}$, and the corresponding cycle $W(e)$. Assume that $\text{span}(e) = \{i, \ldots, j-1\}$. Then there is only one index $z$ for which $e \in E_z^{\text{left}}$ – index $z = j$. Similarly, there is only one index $z$ for which $e \in E^{\text{right}}(z)$ – index $z = i$. Therefore, cycle $W(e)$ contributes its subpath to set $\mathcal{R}_1^{(z)}$ only for indices $z = i$ and $z = j$. On the other hand, cycle $W(e)$ may contribute a subpath to set $\mathcal{R}_2^{(z)}$ for every index $i < z < j$. Because of this, we will try to bound the number of crossings in the final drawing $\varphi_z$ that we construct for instance $I_z$ in terms of the values $\{N_z'(e')\}_{e' \in E(G)}$. For convenience, when the index $z$ is fixed, we omit the superscript $(z)$ in the notation $\mathcal{R}_1^{(z)}$ and $\mathcal{R}_2^{(z)}$.

Notice that, from our assumption about drawing $\varphi^*$, no pair of edges in drawing $\psi_z$ of $H_z$ may cross more than once, and no edge has its image cross itself. Consider any crossing $(e_1, e_2)$ in drawing $\psi_z$. Assume that $e_1$ is a copy of edge $e_1' \in E(G)$, and that $e_2$ is a copy of edge $e_2' \in E(G)$. Then the images of edges $e_1', e_2'$ must cross in $\varphi^*$, and we say that crossing $(e_1', e_2')$ in $\varphi^*$ is *responsible* for crossing $(e_1, e_2)$ in $\psi_z$.

We classify the crossings in drawing $\psi_z$ into several types, and we bound the number of crossings of each of these types separately. Consider now a crossing $(e_1, e_2)$ in drawing $\psi_z$ of graph $H_z$. Let $e_1', e_2'$ be the edges of $G$, such that $e_1 \in J(e_1')$ and $e_2 \in J(e_2')$, so crossing $(e_1', e_2')$ of $\varphi^*$ is responsible for crossing $(e_1, e_2)$.

**Type-1 Crossings.** We say that crossing $(e_1, e_2)$ in $\psi_z$ is a *type-1 crossing* if at least one of the two edges $e'_1, e'_2$ lies in $E(\tilde{S}_z) \cup \delta_G(\tilde{S}_z)$. We assume w.l.o.g. that $e'_1 \in E(\tilde{S}_z) \cup \delta_G(\tilde{S}_z)$. Notice that crossing $(e'_1, e'_2)$ of $\varphi^*$ may be responsible for at most $N_z(e'_2)$ type-1 crossings in $\psi_z$. From Observation 7.55, $\mathbf{E}\left[N_z(e'_2)\right] \leq \hat{\eta}$. We note that random variable $N_z(e'_2)$ may only depend on the random selections of the internal routers $\mathcal{Q}(S_{z-1})$ and $\mathcal{Q}(S_{z+1})$, and in particular it is independent of the random selection of the internal router $\mathcal{Q}(S_z)$. The expected number of crossings for which crossing $(e'_1, e'_2)$ is responsible is then bounded by $\hat{\eta}$. From our definition, crossing $(e'_1, e'_2)$ must lie in $\chi^*_z$. Therefore, the total expected number of type-1 crossings is bounded by $|\chi^*_z| \cdot \hat{\eta}$. We note that the random variable corresponding to the total number of type-1 crossings only depends on the random selections of the internal routers $\mathcal{Q}(S_{z-1})$ and $\mathcal{Q}(S_{z+1})$, and it is independent of the random selection of the internal router $\mathcal{Q}(S_z)$. We will use this fact later.

**Type-2 Crossings.** We say that a crossing $(e_1, e_2)$ in $\psi_z$ is a *type-2 crossing* if it is not a type-1 crossing, and, additionally, one of the two edges (say $e_1$) lies on a path of $\mathcal{R}'_1$, while the other edge (edge $e_2$) lies on a path of $\mathcal{R}''_1$. Notice that, in this case, $e_1 \in E(U_{z-1})$ and $e_2 \in E(\overline{U}_z)$ must hold. A crossing $(e'_1, e'_2)$ of $\psi^*$ may be responsible for at most $N'_z(e'_1) \cdot N'_z(e'_2)$ type-2 crossings of $\psi_z$. Moreover, the random variables $N'_z(e'_1), N'_z(e'_2)$ are independent from each other. From Observation 7.55, we can bound $\mathbf{E}\left[N'_z(e'_1) \cdot N'_z(e'_2)\right] \leq \mathbf{E}\left[N'_z(e'_1)\right] \cdot \hat{\eta}$. Therefore, we get that the total expected number of type-2 crossings is bounded by:

$$\sum_{(e'_1,e'_2)\in\chi^*} \left(\mathbf{E}\left[N'_z(e'_1)\right] + \mathbf{E}\left[N'_z(e'_2)\right]\right) \cdot \hat{\eta}.$$

**Type-3 Crossings.** We say that a crossing $(e_1, e_2)$ in $\psi_z$ is a *type-3 crossing* if it is not a type-1 or a type-2 crossing, and, additionally, one of the two edges (say $e_1$) lies on a path of $\mathcal{R}'_1$, while the other edge (edge $e_2$) lies on a path of $\mathcal{R}'_1 \cup \mathcal{R}_2$. We denote by $\tilde{\chi}_{z-1}$ the set of all crossings $(e'_1, e'_2)$ of $\varphi^*$, where $e'_1, e'_2 \in E(S_{z-1})$.

Consider any crossing $(e'_1, e'_2)$ of $\varphi^*$ that does not lie in $\tilde{\chi}_{z-1}$. This crossing may be responsible for at most $N'_z(e'_1) \cdot N_z(e'_2) + N_z(e'_1) \cdot N'_z(e'_2)$ type-3 crossings in $\psi_z$. Notice that random variables $N'_z(e'_1)$, $N_z(e'_2)$ are independent from each other, as are random variables $N_z(e'_1)$, $N'_z(e'_2)$. From Observation 7.55, we can bound the expected number of type-3 crossings for which crossing $(e'_1, e'_2)$ of $\varphi^*$ is responsible by $\mathbf{E}\left[N'_z(e'_1)\right] \cdot \hat{\eta} + \mathbf{E}\left[N'_z(e'_2)\right] \cdot \hat{\eta}$. Therefore, the total number of type-3 crossings, for which crossings of $\chi^* \setminus \tilde{\chi}_{z-1}$ are responsible is bounded by:

$$\sum_{(e'_1,e'_2)\in\chi^*\setminus\tilde{\chi}_{z-1}} \left(\mathbf{E}\left[N'_z(e'_1)\right] + \mathbf{E}\left[N'_z(e'_2)\right]\right) \cdot \hat{\eta}.$$

A crossing $(e'_1, e'_2) \in \tilde{\chi}_{z-1}$ may be responsible for up to $N'_z(e'_1) \cdot N_z(e'_2) + N_z(e'_1) \cdot N'_z(e'_2) \leq 2N_z(e'_1) \cdot N_z(e'_2)$ type-3 crossings of $\psi_z$. But now the random variables $N_z(e'_1), N_z(e'_2)$ are no longer indepdendent. We can, however, bound $N_z(e'_1) \cdot N_z(e'_2) \leq (N_z(e'_1))^2 + (N_z(e'_2))^2$. From Observation 7.51, for an edge $e \in E(S_{z-1})$, $N_z(e) \leq O(\log^{34} m) \cdot \mathrm{cong}_G(\mathcal{Q}(S_{z-1}), e)$. Therefore, the total expected number of type-3 crossings is at most:

$$\begin{aligned}\hat{\eta} \cdot \sum_{(e'_1,e'_2)\in\chi^*\setminus\tilde{\chi}_{z-1}} &\left(\mathbf{E}\left[N'_z(e'_1)\right] + \mathbf{E}\left[N'_z(e'_2)\right]\right) \\ &+ O(\log^{68} m) \cdot \sum_{(e'_1,e'_2)\in\tilde{\chi}_{z-1}} \left(\mathbf{E}\left[(\mathrm{cong}_G(\mathcal{Q}(S_{z-1}), e'_1)^2\right] + \mathbf{E}\left[(\mathrm{cong}_G(\mathcal{Q}(S_{z-1}), e'_2)^2\right]\right).\end{aligned}$$

**Type-4 Crossings.** We say that a crossing $(e_1, e_2)$ in $\psi_z$ is a *type-4 crossing* if it is not a crossing of one of the first three types, and, additionally, one of the two edges (say, edge $e_1$) lies on a path of $\mathcal{R}_1''$, while the other edge (edge $e_2$) lies on a path of $\mathcal{R}_1'' \cup \mathcal{R}_2$. We denote by $\tilde{\chi}_{z+1}$ the set of all crossings $(e_1', e_2')$ of $\varphi^*$, where $e_1', e_2' \in E(S_{z+1})$.

Consider any crossing $(e_1', e_2')$ of $\varphi^*$ that does not lie in $\tilde{\chi}_{z+1}$. As before, this crossing may be responsible for at most $N_z'(e_1') \cdot N_z(e_2') + N_z(e_1') \cdot N_z'(e_2')$ type-4 crossings in $\psi_z$. Using the same analysis as for type-3 crossings, we can bound the expected number of type-4 crossings for which crossing $(e_1', e_2')$ of $\varphi^* \setminus \tilde{\chi}_{z+1}$ is responsible by $\mathbf{E}\left[N_z'(e_1')\right] \cdot \hat{\eta} + \mathbf{E}\left[N_z'(e_2')\right] \cdot \hat{\eta}$. The total number of type-4 crossings, for which crossings of $\chi^* \setminus \tilde{\chi}_{z+1}$ are responsible is bounded by:

$$\sum_{(e_1', e_2') \in \chi^* \setminus \tilde{\chi}_{z+1}} \left(\mathbf{E}\left[N_z'(e_1')\right] + \mathbf{E}\left[N_z'(e_2')\right]\right) \cdot \hat{\eta}.$$

As before, a crossing $(e_1', e_2') \in \tilde{\chi}_{z+1}$ may be responsible for up to $N_z'(e_1') \cdot N_z(e_2') + N_z(e_1') \cdot N_z'(e_2') \leq 2N_z(e_1') \cdot N_z(e_2') \leq 2(N_z(e_1'))^2 + 2(N_z(e_2'))^2$ type-4 crossings of $\psi_z$. Using the same reasoning as in type-3 crossings, the total expected number of type-4 crossings is bounded by:

$$\hat{\eta} \cdot \sum_{(e_1', e_2') \in \chi^* \setminus \tilde{\chi}_{z+1}} \left(\mathbf{E}\left[N_z'(e_1')\right] + \mathbf{E}\left[N_z'(e_2')\right]\right) + O(\log^{68} m) \cdot \sum_{(e_1', e_2') \in \tilde{\chi}_{z+1}} \left(\mathbf{E}\left[(\mathrm{cong}_G(\mathcal{Q}(S_{z-1}), e_1')^2\right] + \mathbf{E}\left[(\mathrm{cong}_G(\mathcal{Q}(S_{z+1}), e_2')^2\right]\right).$$

**Type-5 Crossings** All remaining crossings of $\psi_z$ are type-5 crossings. For each such crossing $(e_1, e_2)$, it must be the case that each of the edges $e_1, e_2$ belongs to a path of $\mathcal{R}_2$. We do not bound the number of type-5 crossings, as we will eventually eliminate all such crossings.

### 7.4.2 Step 2: Initial Drawing of $G_z$

For every edge $e \in E_{z-1} \cup E_z^{\text{left}}$, we denote by $\Gamma(e)$ the curve corresponding to the image of path $R(e) \in \mathcal{R}_1'$ in the drawing $\psi_z$ of $H_z$. Note that, if $v$ is an endpoint of the path $R(e)$ that lies in $V(\tilde{S}_z)$, then curve $\Gamma(e)$ connects the image of $v$ to the image of vertex $u_{z-1}$ in $\psi_z$. For every edge $e \in E_z \cup E_z^{\text{right}}$, we denote by $\Gamma(e)$ the curve corresponding to the image of the path $R(e) \in \mathcal{R}_1''$ in the drawing $\psi_z$ of $H_z$. Note that, if $v$ is an endpoint of $e$ that lies in $V(\tilde{S}_z)$, then curve $\Gamma(e)$ connects the image of $v$ to the image of vertex $u_{z+1}$ in $\psi_z$. Lastly, for every edge $e \in E_z^{\text{over}}$, we let $\Gamma(e)$ be the image of the path $R(e) \in \mathcal{R}_2$ in $\psi_z$. Notice that curve $\Gamma(e)$ connects the image of $u_{z-1}$ to the image of $u_{z+1}$ in $\psi_z$. From the constructions of the paths in $\mathcal{R}_1' \cup \mathcal{R}_2$, if we denote $\delta_G(U_{z-1}) = \{\hat{a}_1, \ldots, \hat{a}_q\}$, where the edges are indexed in the order of their apperance in the ordering $\tilde{\mathcal{O}}_{z-1}$, the curves $\Gamma(\hat{a}_1), \ldots, \Gamma(\hat{a}_q)$ enter the image of vertex $u_{z-1}$ in this circular order.

In order to obtain the initial drawing $\varphi_z'$ of $G_z$, we start with the drawing $\psi_z$ of graph $H_z$, and we delete from it the images of all vertices except those lying in $V(\tilde{S}_z) \cup \{u_{z-1}, u_{z+1}\}$, and the images of all edges except those lying in $E(\tilde{S}_z)$; we view the image of vertex $u_{z-1}$ as the image of the special vertex $v^*$, and the image of vertex $u_{z+1}$ as the image of the secial vertex $v^{**}$. We then add to this drawing the curves in $\{\Gamma(e) \mid e \in \delta_G(U_{z-1}) \cup \delta_G(U_z)\}$. Each such curve $\Gamma(e)$ becomes an image of the corresponding edge $e$. Notice that the edges of $\delta_G(U_{z-1})$ become incident to $v^*$ in graph $G_z$; the edges of $\delta_G(U_z)$ become incident to $v^{**}$, and the edges of $E_z^{\text{over}}$ connect $v^*$ to $v^{**}$. From the above discussion, the circular order in which the images of the edges in $\delta_{G_z}(v^*)$ enter the image of $v^*$ in the current drawing is exactly the ordering $\tilde{\mathcal{O}}_{z-1}$, which is precisely the ordering $\mathcal{O}_{v^*} \in \Sigma_z$. However, the images

of the edges of $\delta_{G_z}(v^{**})$ may not enter the image of vertex $v^{**}$ in the correct order. We will fix this in subsequent steps. There is one major problem with the current drawing of the graph $G_z$: it is possible that some point $p$ lies on a large number of curves in set $\{\Gamma(e) \mid e \in \delta_G(U_{z-1}) \cup \delta_G(U_z)\}$, and it is an inner point on each such curve. This may only happen if $p$ corresponds to an image of some vertex $v$, where $v \in V(U_{z-1}) \setminus \{u_{z-1}\}$, or $v \in V(\overline{U}_z) \setminus \{u_{z+1}\}$. We will now "fix" the images of the edges of $\delta_{G_z}(v^*) \cup \delta_{G_z}(v^{**})$ by slightly "nudging" them in the vicinity of each such vertex, to ensure that all resulting curves are in general position. We do so by performing a nudging operation (see Section 4.4.3).

We process every vertex $v \in \big(V(U_{z-1}) \cup V(\overline{U}_z)\big) \setminus \{u_{z-1}, u_{z+1}\}$ one by one. Consider an iteration when any such vertex $v$ is processed. Let $A(v)$ be the set of all edges $e \in \delta_G(U_{z-1}) \cup \delta_G(U_z)$, such that curve $\Gamma(e)$ contains the image of vertex $v$ (in $\psi_z$). We denote $A(v) = \{a_1, \ldots, a_k\}$. Consider the tiny $v$-disc $D(v) = D_{\psi_z}(v)$ in the drawing $\psi_z$ of graph $H_z$. For all $1 \leq i \leq k$, we let $s_i, t_i$ be the two points at which curve $\Gamma(a_i)$ intersects the boundary of the disc $D(v)$. Note that all points $s_1, t_1, \ldots, s_k, t_k$ must be distinct. We use the algorithm from Claim 4.34 in order to construct a collection $\{\gamma_1, \ldots, \gamma_k\}$ of curves, such that, for all $1 \leq i \leq k$, curve $\gamma_i$ has $s_i$ and $t_i$ as its endpoints, and is completely contained in $D(v)$. Recall that the claim ensures that, for every pair $1 \leq i < j \leq k$ of indices, if the two pairs $(s_i, t_i), (s_j, t_j)$ of points cross, then curves $\gamma_i, \gamma_j$ intersect at exactly one point; otherwise, curves $\gamma_i, \gamma_j$ do not intersect. For all $1 \leq i \leq k$, we modify the curve $\Gamma(a_i)$ as follows: we replace the segment of the curve between points $s_i, t_i$ with the curve $\gamma_i$.

Once every vertex $v \in \big(V(U_{z-1}) \cup V(\overline{U}_z)\big) \setminus \{u_{z-1}, u_{z+1}\}$ is processed in this way, the curves in set $\{\Gamma(e) \mid e \in \delta_G(U_{z-1}) \cup \delta_G(U_z)\}$ are in general position, and we obtain a valid drawing of the graph $G_z$, that we denote by $\varphi'_z$. The modification of the curves in $\{\Gamma(e) \mid e \in \delta_G(U_{z-1}) \cup \delta_G(U_z)\}$ do not affect the endpoints of the curves, and so, for every vertex $x \in V(G_z) \setminus \{v^{**}\}$, the images of the edges of $\delta_{G_z}(x)$ enter the image of $x$ in the order consistent with the rotation $\mathcal{O}_x \in \Sigma_z$. We now bound the number of crossings in drawing $\varphi'_z$.

Consider some pair of edges $e, e' \in E(G_z)$ that cross at some point $p$ in the drawing $\varphi'_z$. We say that this crossing is *primary* iff point $p$ does not belong to any of the discs in the set

$$\big\{D(v) \mid v \in \big(V(U_{z-1}) \cup V(\overline{U}_z)\big) \setminus \{u_{z-1}, u_{z+1}\}\big\};$$

otherwise we say that the crossing is *secondary*.

Notice that every primary crossing in $\varphi'_z$ corresponds to a unique crossing in the drawing $\psi_z$ of the graph $H_z$. Recall that we have partitioned all such crossings into five types, and we have bounded the number of crossings of each of the first four types. This partition naturally defines a partition of all primary crossings in $\varphi'_z$ into five types. Specifically, primary crossings of type 1 are all crossings $(e, e')$ where at least one of the edges $e, e'$ lies in $E(\tilde{S}_z) \cup \delta_G(\tilde{S}_z)$. Primary crossings of type 2 are primary crossings between curves $\Gamma(e), \Gamma(e')$, where $e \in E_{z-1} \cup E_z^{\text{left}}$, while $e' \in E_z \cup E_z^{\text{right}}$. Primary crossings of type 3 are primary crossings between curves $\Gamma(e), \Gamma(e')$, where $e \in E_{z-1} \cup E_z^{\text{left}}$ and $e' \in E_{z-1} \cup E_z^{\text{left}} \cup E_z^{\text{over}}$, while primary crossings of type 4 are primary crossings between curves $\Gamma(e), \Gamma(e')$, where $e \in E_z \cup E_z^{\text{right}}$, while $e' \in E_z \cup E_z^{\text{right}} \cup E_z^{\text{over}}$. Lastly, primary crossings of type 5 are primary crossings between curves $\Gamma(e), \Gamma(e')$, where $e, e' \in E_z^{\text{over}}$. The number of primary crossings of the first four types is bounded as before.

We now consider secondary crossings of $\varphi_z$. Notice that each such crossing must be between a pair of curves $\Gamma(e), \Gamma(e')$, where $e, e' \in \delta_G(U_{z-1}) \cup \delta_G(U_z)$.

Consider a pair of curves $\Gamma(e), \Gamma(e')$, where $e, e' \in \delta_G(U_{z-1}) \cup \delta_G(U_z)$, and some point $p$ at which the curves cross, such that the crossing is secondary. Let $v \in \big(V(U_{z-1}) \cup V(\overline{U}_z)\big) \setminus \{u_{z-1}, u_{z+1}\}$ be the vertex such that $p$ lies in the interior of disc $D(v)$. Denote by $s, t$ the points on the boundary of $D$ that lie on $\Gamma(e)$, and define $s', t'$ similary for $\Gamma(e')$. From Claim 4.34, curves $\Gamma(e), \Gamma(e')$ may only cross inside the disc $D(v)$ if the pairs $(s, t), (s', t')$ of points on the boundary of $D$ cross. Consider now

the paths $R(e) \in \mathcal{R}_1 \cup \mathcal{R}_2$ and $R(e') \in \mathcal{R}_1 \cup \mathcal{R}_2$. Denote by $e_1, e_2$ the two edges on path $R(e)$ that immediately precede and immediately follow vertex $v$, and define edges $e'_1, e'_2$ similarly for path $R(e')$. We now consider three cases.

First, if $v \in V(S_{z-1})$, then there must be two paths $Q, Q' \in \mathcal{Q}(S_{z-1})$, such that $Q \subseteq R(e)$ and $Q' \subseteq R(e')$, where $Q, Q'$ both contain the vertex $v$. Since the paths of $\mathcal{Q}(S_{z-1})$ are non-transversal with respect to $\Sigma$, the only way for the two pairs $(s,t), (s',t')$ of points to cross is if the set $\{e_1, e_2, e'_1, e'_2\}$ contains copies of fewer than four distinct edges of $\delta_G(v)$. In other words, for some edge $e^* \in \delta_G(v)$, both $R(e)$ and $R(e')$ contain a copy of $e^*$. In this case, we say that edge $e^*$ is *responsible* for this secondary crossing between $\Gamma(e)$ and $\Gamma(e')$. The second case is when $v \in V(S_{z+1})$. The analysis of this case is similar to the previous case: there must be an edge $e^* \in \delta_G(v)$ such that both $R(e)$ and $R(e')$ contain a copy of $e^*$. We say that $e^*$ is responsible for this crosisng.

We now consider the third case, when $v \notin V(S_{z-1}) \cup V(S_{z+1})$. We consider the paths $R(e), R(e') \in \mathcal{R}_1 \cup \mathcal{R}_2$, and we define the edges $e_1, e_2, e'_1, e'_2$ as before. Since $v \notin V(S_{z-1}) \cup V(S_{z+1})$, it must be the case that $v \in W(e) \cap W(e')$. If there is an edge $e^* \in \delta_G(v)$, such that both $R(e)$ and $R(e')$ contain a copy of $e^*$, then we designate $e^*$ to be responsible for this crossing as before. Otherwise, the edges in set $\{e_1, e_2, e'_1, e'_2\}$ are copies of four distinct edges of $\delta_G(V)$. In this case, the cycles $W(e)$ and $W(e')$ must have a transversal intersection at vertex $v$, from Observation 7.50, and $e, e' \in \hat{E}_z$ must hold. In this case, we say that the transversal intersection of $W(e)$ and $W(e')$ at $v$ is responsible for the crossing.

We now classify the secondary crossings into three types and bound the number of crossings of the first two types. We will eventually eliminate all crossings of the third type.

**Type-1 secondary crossing.** Consider a secondary crossing between a pair $\Gamma(e), \Gamma(e')$ of curves, for $e, e' \in \delta_G(U_{z-1}) \cup \delta_G(U_z)$, and a secondary crossing of the two curves at some point $p$. We say that the crossing is of type 1 if $e \in E_{z-1} \cup E_z^{\text{left}}$ and $e' \in E_{z-1} \cup E_z^{\text{left}} \cup E_z^{\text{over}}$.

**Type-2 secondary crossing.** We say that a secondary crossing between a pair $\Gamma(e), \Gamma(e')$ of curves, for $e, e' \in \delta_G(U_{z-1}) \cup \delta_G(U_z)$, is of type 2 if $e \in E_z \cup E_z^{\text{right}}$ and $e' \in E_z \cup E_z^{\text{right}} \cup E_z^{\text{over}}$.

**Type-3 secondary crossing.** All remaining secondary crossings are of tye 3. Consider any such crossing between a pair $\Gamma(e), \Gamma(e')$ of curves, for $e, e' \in \delta_G(U_{z-1}) \cup \delta_G(U_z)$. Notice that it is impossible that one of the two edges $e, e'$ lies in $E_{z-1} \cup E_z^{\text{left}}$, while the other lies in $E_z \cup E_z^{\text{right}}$, since, in such a case, paths $R(e), R(e')$ cannot share any edges. Therefore, $e, e' \in E_z^{\text{over}}$ must hold.

We now bound the expected number of type-1 secondary crossing. Consider any such crossing between a pair $\Gamma(e), \Gamma(e')$ of curves, and assume that the crossing point $p$ lies in disc $D(v)$, for some vertex $v$. From the definition of a type-1 crossing, $v \in V(U_{z-1}) \setminus \{u_{z-1}\}$ must hold. In this case, it is impossible that a pair of auxiliary cycles $W(e), W(e')$ with $e, e' \in \hat{E}_z$ have a transversal intersection at vertex $v$, from Observation 7.48. Therefore, some edge of $E_G(U_{z-1})$ must be responsible for this crossing. It is immediate to verify that every edge $e \in E_G(U_{z-1})$ may be responsible for at most $N'_z(e) \cdot N_z(e)$ type-1 secondary crossings. If $e \notin E(S_{z-1})$, then, from Observation 7.55, $N_z(e) \leq O(\log^{34} m)$. If $e \in E(S_{z-1})$, then, from Observation 7.51, $N_z(e) \leq O(\log^{34} m) \cdot \text{cong}_G(\mathcal{Q}(S_{z-1}), e)$. Therefore, the total expected number of type-1 secondary crossings is bounded by:

$$O(\log^{34} m) \cdot \sum_{e \in E(U_{z-1}) \setminus E(S_{z-1})} N'_z(e) + O(\log^{68} m) \cdot \sum_{e \in E(S_{z-1})} \mathbf{E}\left[ \left(\text{cong}_G(\mathcal{Q}(S_{z-1}), e)\right)^2 \right].$$

Next, we bound the expected number of type-2 secondary crossings. This time some of the crossigns are charged to individual edges (that is, some edge of $E(\overline{U}_z)$ is responsible for the crossing), and some

crossings are charged to transversal intersections of pairs of cycles $W(e'), W(e'')$, where $e', e'' \in \hat{E}_z$. The expected number of edges of the former type is bounded using the same reasoning as for type-1 crossings, and their expected number is at most:

$$O(\log^{34} m) \cdot \sum_{e \in E(\overline{U}_z) \setminus E(S_{z+1})} N'_z(e) + O(\log^{68} m) \cdot \sum_{e \in E(S_{z+1})} \mathbf{E}\left[\left(\mathrm{cong}_G(\mathcal{Q}(S_z), e)\right)^2\right].$$

Let $\Pi_z^T$ denote the set of triples $(e, e', v)$, where $e \in E_z^{\mathrm{right}}$, $e' \in \hat{E}_z$, and $v$ is a vertex that lies on both $W(e)$ and $W(e')$, such that cycles $W(e)$ and $W(e')$ have a transversal intersection at $v$. Clearly, the number of type-2 secondary crossings that are charged to transversal intersections of pairs of cycles is bounded by $|\Pi_z^T|$. Overall, we get that the total expected number of type-2 secondary crossings is bounded by:

$$O(\log^{34} m) \cdot \sum_{e \in E(\overline{U}_z) \setminus E(S_{z+1})} N'_z(e) + O(\log^{68} m) \cdot \sum_{e \in E(S_{z+1})} \mathbf{E}\left[\left(\mathrm{cong}_G(\mathcal{Q}(S_z), e)\right)^2\right] + |\Pi_z^T|.$$

This completes the analysis of the initial drawing $\varphi'_z$ of graph $G_z$. Notice that we did not analyze the number of type-5 primary crossings and the number of type-3 secondary crossings. These are all crossings between the images of the edges of $E_z^{\mathrm{over}}$. Unfortunately, a crossing of the original drawing $\varphi^*$ of $G$ may give rise to many crossings between edges of $E_z^{\mathrm{over}}$ in drawings $\varphi'_z$ of graphs $G_z$, for $1 \leq z \leq r$. In the next step, we will slightly modify the drawing $\varphi'_z$ in order to eliminate all such crossings. Notice that our current bounds on the expected number of crossings in $\varphi'_z$ contain terms like $\sum_{e \in E(S_{z-1})} \mathbf{E}\left[\left(\mathrm{cong}_G(\mathcal{Q}(S_{z-1}), e)\right)^2\right]$. If cluster $S_{z-1}$ lies in set $\mathcal{S}^{\mathrm{light}}$, then this expression can be bounded by $|E(S_{z-1})| \cdot \hat{\eta}$. However, if $S_{z-1} \in \mathcal{S}^{\mathrm{bad}}$ then this bound may no longer be valid. In such a case we will perform an additional uncrossing operation of the images of edges of $\delta_G(U_{z-1})$ in order to decrease this number of crossings. We also perform such an operation on the images of the edges of $\delta_G(U_z)$ if $S_{z+1} \in \mathcal{S}^{\mathrm{bad}}$.

### 7.4.3 Step 3: Modified Drawing of $G_z$

In this step we modify the drawing $\varphi'_z$ of $G_z$ to obtain a new modified drawing $\varphi''_z$, by performing one or more uncrossing operations. We first consider the cases where $S_{z-1} \in \mathcal{S}^{\mathrm{bad}}$ or $S_{z+1} \in \mathcal{S}^{\mathrm{bad}}$ hold, and perform some initial uncrossings to decrease the number of type-3 primary and type-1 secondary crossings (in case where $S_{z-1} \in \mathcal{S}^{\mathrm{bad}}$), and the number type-4 primary and type-2 secondary crossings (in case where $S_{z-1} \in \mathcal{S}^{\mathrm{bad}}$). After that we perform one more uncrossing operation that will eliminate all type-5 primary and type-3 secondary crossings.

Recall that the expected number of type-3 primary crossings and type-1 secondary crossings (that is, all crossings between images of edge pairs $e, e'$ where $e \in E_{z-1} \cup E_z^{\mathrm{left}}$ and $e' \in E_{z-1} \cup E_z^{\mathrm{left}} \cup E_z^{\mathrm{over}}$) is at most:

$$\hat{\eta} \cdot \sum_{(e'_1,e'_2)\in\chi^*\setminus\tilde{\chi}_{z-1}} \left(\mathbf{E}\left[N'_z(e'_1)\right] + \mathbf{E}\left[N'_z(e'_2)\right]\right) + O(\log^{34} m) \cdot \sum_{e\in E(U_{z-1})\setminus E(S_{z-1})} N'_z(e) + O(\log^{68} m) \cdot \sum_{(e'_1,e'_2)\in\tilde{\chi}_{z-1}} \left(\mathbf{E}\left[(\mathrm{cong}_G(\mathcal{Q}(S_{z-1}), e'_1)^2\right] + \mathbf{E}\left[(\mathrm{cong}_G(\mathcal{Q}(S_{z-1}), e'_2)^2\right]\right) + O(\log^{68} m) \cdot \sum_{e\in E(S_{z-1})} \mathbf{E}\left[(\mathrm{cong}_G(\mathcal{Q}(S_{z-1}), e))^2\right].$$

From Observation 7.49, if $S_{z-1} \in \mathcal{S}^{\mathrm{light}}$, then for every edge $e \in E(S_{z-1})$, $\mathbf{E}\left[(\mathrm{cong}_G(\mathcal{Q}(S_{z-1}), e)^2\right] \leq \hat{\eta}$. Recalling that $\tilde{\chi}_{z-1} \subseteq \chi^*_{z-1}$, we get that, if $S_{z-1} \in \mathcal{S}^{\mathrm{light}}$, then the expected number of type-3 primary and type-1 secondary crossings is bounded by:

$$\hat{\eta} \cdot \sum_{(e'_1,e'_2)\in\chi^*\setminus\tilde{\chi}_{z-1}} \left(\mathbf{E}\left[N'_z(e'_1)\right] + \mathbf{E}\left[N'_z(e'_2)\right]\right) + O(\log^{34} m) \cdot \sum_{e\in E(U_{z-1})\setminus E(S_{z-1})} N'_z(e) + O(\hat{\eta}^2) \cdot (|\chi^*_{z-1}| + |E(S_{z-1}|). \tag{7}$$

Recall that the expected number of type-4 primary crossings and type-2 secondary crossings (that is, all crossings between images of edge pairs $e, e'$ where $e \in E_z \cup E_z^{\mathrm{right}}$ and $e' \in E_z \cup E_z^{\mathrm{right}} \cup E_z^{\mathrm{over}}$) is at most:

$$\hat{\eta} \cdot \sum_{(e'_1,e'_2)\in\chi^*\setminus\tilde{\chi}_{z+1}} \left(\mathbf{E}\left[N'_z(e'_1)\right] + \mathbf{E}\left[N'_z(e'_2)\right]\right) + O(\log^{34} m) \cdot \sum_{e\in E(\overline{U}_z)\setminus E(S_{z+1})} N'_z(e) + O(\log^{68} m) \cdot \sum_{(e'_1,e'_2)\in\tilde{\chi}_{z+1}} \left(\mathbf{E}\left[(\mathrm{cong}_G(\mathcal{Q}(S_{z+1}), e'_1)^2\right] + \mathbf{E}\left[(\mathrm{cong}_G(\mathcal{Q}(S_{z+1}), e'_2)^2\right]\right) + O(\log^{68} m) \cdot \sum_{e\in E(S_{z+1})} \mathbf{E}\left[(\mathrm{cong}_G(\mathcal{Q}(S_z), e))^2\right] + |\Pi^T_z|$$

Here, $\Pi^T_z$ is the set of triples $(e, e', v)$, where $e \in E_z^{\mathrm{right}}$, $e' \in \hat{E}_z$, and cycles $W(e)$ and $W(e')$ have a transversal intersection at vertex $v$.

From Observation 7.49, if $S_{z+1} \in \mathcal{S}^{\mathrm{light}}$, then for every edge $e \in E(S_{z+1})$, $\mathbf{E}\left[(\mathrm{cong}_G(\mathcal{Q}(S_{z+1}), e)^2\right] \leq \hat{\eta}$. Recalling that $\tilde{\chi}_{z+1} \subseteq \chi^*_{z+1}$, we get that, if $S_{z+1} \in \mathcal{S}^{\mathrm{light}}$, then the expected number of type-4 primary and type-2 secondary crossings is bounded by:

$$\hat{\eta} \cdot \sum_{(e'_1,e'_2)\in\chi^*\setminus\tilde{\chi}_{z+1}} \left(\mathbf{E}\left[N'_z(e'_1)\right] + \mathbf{E}\left[N'_z(e'_2)\right]\right) + O(\log^{34} m) \cdot \sum_{e\in E(\overline{U}_z)\setminus E(S_{z+1})} N'_z(e) + O(\hat{\eta}^2) \cdot (|\chi^*_{z+1}| + |E(S_{z+1})|) + |\Pi^T_z|. \tag{8}$$

We now consider four cases, depending on whether the clusters $S_{z-1}, S_{z+1}$ lie in $\mathcal{S}^{\text{light}}$ or $\mathcal{S}^{\text{bad}}$.

**Case 1: $S_{z-1} \in \mathcal{S}^{\text{bad}}$ and $S_{z+1} \in \mathcal{S}^{\text{light}}$.** When $S_{z-1} \in \mathcal{S}^{\text{bad}}$, we no longer have a bound on $\mathbf{E}\left[(\text{cong}_G(\mathcal{Q}(S_{z-1}), e)^2\right]$ for edges $e \in E(S_{z-1})$, and so the bound from Equation (7) on the number of type-3 primary and type-1 secondary crossings is no longer valid. Instead, we will perform a type-1 uncrossing of the images of the edges of $E_{z-1} \cup E_z^{\text{left}} \cup E_z^{\text{over}}$. Let $\Gamma_1$ denote the set of curves reprsenting the images of these edges in $\varphi'_z$. Let $\Gamma_2$ denote the set of curves reprsenting the images of all remaining edges of $G_z$ in $\varphi'_z$. We apply the algorithm from Theorem 4.33 to compute a new collection $\Gamma'_1$ of curves, where, for each edge $e \in E_{z-1} \cup E_z^{\text{left}} \cup E_z^{\text{over}}$, there is a curve $\gamma(e) \in \Gamma'_1$ connecting the images of the endpoints of $e$. Intuitively, the algortihm for type-1 uncrossing proceeds in iterations, as long as there is a pair $\gamma_1, \gamma_2 \in \Gamma_1$ of curves that cross at least twice. Assume that $p$ and $q$ are two points lying on both $\gamma_1$ and $\gamma_2$. The algorithm then uncrosses the two curves, by "swapping" the segments of these curves that connect $p$ and $q$ (see Figure 43 in Appendix D.17 for an illustration.) At the end of this procedure, every pair of curves in $\Gamma'_1$ may cross each other at most once. For each curve $\gamma \in \Gamma_2$, the number of crossings between $\gamma$ and the curves in $\Gamma'_1$ is no higher than the number of crossings between $\gamma$ and the curves in $\Gamma_1$. We modify the images of the edges in $E_{z-1} \cup E_z^{\text{left}} \cup E_z^{\text{over}}$, so that for each such edge $e$, its new image is the curve $\gamma(e) \in \Gamma'_1$. Observe that the total number of primary crossings of types 1,2, and 4 does not increase, and neither does the number of secondary crossings of type 2. We note however that a primary crossing of type 2 (a crossing between images of edges $e, e'$ where $e \in E_{z-1} \cup E_z^{\text{left}}$ and $e' \in E_z \cup E_z^{\text{right}}$) may become a primary crossing of type 4 (a crossing between images of edges $e, e'$ where $e' \in E_z \cup E_z^{\text{right}}$ and $e \in E_z^{\text{over}}$), and vice versa. The total number of type-3 primary crossings and of type-1 secondary crossings is now bounded by: $\left(|E_{z-1}| + |E_z^{\text{left}}| + |E_z^{\text{over}}|\right)^2 \leq |\delta_G(U_{z-1})|^2$.

From Corollary 7.52, $|\delta_G(U_{z-1})| \leq |\delta_G(S_{z-1})| \cdot O(\log^{34} m)$. Recall that, if $S_{z-1} \in \mathcal{S}^{\text{bad}}$, and the bad event $\mathcal{E}$ does not happen, then, from Observation 7.49, $\mathsf{OPT}_{\mathsf{cnwrs}}(S_{z-1}, \Sigma(S_{z-1})) + |E(S_{z-1})| \geq \frac{|\delta_G(S_{z-1})|^2}{\hat{\eta}}$, where $\Sigma(S_{z-1})$ is the rotation system for graph $S_{z-1}$ induced by $\Sigma$. Therefore, $|\delta_G(S_{z-1})|^2 \leq \hat{\eta} \cdot \left(|\chi^*_{z-1}| + |E(S_{z-1})|\right)$ must hold.

Overall, if $S_{z-1} \in \mathcal{S}^{\text{bad}}$, and event $\mathcal{E}$ did not happen, then the total number of type-3 primary crossings and of type-1 secondary crossings is now bounded by:

$$|\delta_G(U_{z-1})|^2 \leq \hat{\eta}^2 \cdot |\delta_G(S_{z-1})|^2 \leq \hat{\eta}^2 \cdot \left(|\chi^*_{z-1}| + |E(S_{z-1})|\right).$$

Combining this with the bound from Equation 7, we get that, regardless of whether Case 1 happened or not, if event $\mathcal{E}$ did not happen, then, after the current modification, the total expected number of type-3 primary crossings and of type-1 secondary crossings is bounded by:

$$\begin{aligned}\hat{\eta} \cdot \sum_{(e'_1, e'_2) \in \chi^* \setminus \tilde{\chi}_{z-1}} & \left(\mathbf{E}\left[N'_z(e'_1)\right] + \mathbf{E}\left[N'_z(e'_2)\right]\right) \\ & + O(\log^{34} m) \cdot \sum_{e \in E(U_{z-1}) \setminus E(S_{z-1})} N'_z(e) \\ & + \hat{\eta}^2 \cdot (|\chi^*_{z-1}| + |E(S_{z-1}|).\end{aligned} \tag{9}$$

**Case 2: $S_{z-1} \in \mathcal{S}^{\text{light}}$ and $S_{z+1} \in \mathcal{S}^{\text{bad}}$.** We now consider the case where $S_{z-1} \in \mathcal{S}^{\text{light}}$ and $S_{z+1} \in \mathcal{S}^{\text{bad}}$. The modification that we perform is almost identical to that performed in the case where $S_{z-1} \in \mathcal{S}^{\text{bad}}$, except that now we uncross the images of the edges in $\delta_G(U_z)$.

As before, when $S_{z+1} \in \mathcal{S}^{\text{bad}}$, we no longer have a bound on $\mathbf{E}\left[(\text{cong}_G(\mathcal{Q}(S_{z+1}),e)^2\right]$ for edges $e \in E(S_{z+1})$, and so the bound from Equation (8) on the number of type-4 primary and type-2 secondary crossings is no longer valid. Instead, we will perform a type-1 uncrossing of the images of the edges of $E_z \cup E_z^{\text{right}} \cup E_z^{\text{over}}$, similarly to the first case. Let $\Gamma_1$ denote the set of curves reprsenting the images of these edges in the current drawing $\varphi'_z$. Let $\Gamma_2$ denote the set of curves reprsenting the images of all remaining edges of $G_z$ in $\varphi'_z$. We apply the algorithm from Theorem 4.33 to compute a new collection $\Gamma'_1$ of curves, where, for each edge $e \in E_z \cup E_z^{\text{right}} \cup E_z^{\text{over}}$, there is a curve $\gamma(e) \in \Gamma'_1$ connecting the images of the endpoints of $e$. Recall that every pair of curves in $\Gamma'_1$ may cross at most once, and, for each curve $\gamma \in \Gamma_2$, the number of crossings between $\gamma$ and the curves in $\Gamma'_1$ is no higher than the number of crossings between $\gamma$ and the curves in $\Gamma_1$. We modify the images of the edges in $E_z \cup E_z^{\text{right}} \cup E_z^{\text{over}}$, so that for each such edge $e$, its new image is the curve $\gamma(e) \in \Gamma'_1$. As before, the total number of primary crossings of types 1,2, and 3 does not increase, and neither does the number of secondary crossings of type 1. As before, a primary crossing of type 2 (a crossing between images of edges $e, e'$ where $e \in E_{z-1} \cup E_z^{\text{left}}$ and $e' \in E_z \cup E_z^{\text{right}}$) may become a primary crossing of type 3 (a crossing between images of edges $e, e'$ where $e' \in E_{z-1} \cup E_z^{\text{left}}$ and $e \in E_z^{\text{over}}$), and vice versa. The total number of type-4 primary crossings and of type-2 secondary crossings is now bounded by: $\left(|E_z| + |E_z^{\text{right}}| + |E_z^{\text{over}}|\right)^2 \leq |\delta_G(U_z)|^2$. Using the same arguments as in the first case, and the second statement from Corollary 7.52, we conclude that $|\delta_G(U_z)| \leq |\delta_G(S_{z+1})| \cdot O(\log^{34} m)$. As before, if $S_{z+1} \in \mathcal{S}^{\text{bad}}$, and the bad event $\mathcal{E}$ does not happen, then, from Observation 7.49, $\mathsf{OPT}_{\mathsf{cnwrs}}(S_{z+1}, \Sigma(S_{z+1})) + |E(S_{z+1})| \geq \frac{|\delta_G(S_{z-1})|^2}{\hat{\eta}}$, where $\Sigma(S_{z+1})$ is the rotation system for graph $S_{z+1}$ induced by $\Sigma$. As before, we get that $|\delta_G(S_{z+1})|^2 \leq \hat{\eta} \cdot \left(|\chi^*_{z+1}| + |E(S_{z+1})|\right)$.

Overall, if $S_{z+1} \in \mathcal{S}^{\text{bad}}$, and event $\mathcal{E}$ did not happen, then the total number of type-4 primary type-2 secondary crossings is now bounded by:

$$|\delta_G(U_{z+1})|^2 \leq O(\log^{68} m) \cdot |\delta_G(S_{z+1})|^2 \leq \hat{\eta}^2 \cdot \left(|\chi^*_{z+1}| + |E(S_{z+1})|\right).$$

Combining this with the bound from Equation 8, we get that, regardless of whether Case 2 happened or not, if event $\mathcal{E}$ did not happen, then, after the current modification, the total expected number of type-4 primary crossings and of type-2 secondary crossings is bounded by:

$$\begin{aligned}\hat{\eta} \cdot \sum_{(e'_1,e'_2) \in \chi^* \setminus \tilde{\chi}_{z+1}} & \left(\mathbf{E}\left[N'_z(e'_1)\right] + \mathbf{E}\left[N'_z(e'_2)\right]\right) \\ & + O(\log^{34} m) \cdot \sum_{e \in E(\overline{U}_z) \setminus E(S_{z+1})} N'_z(e) \\ & + \hat{\eta}^2 \cdot (|\chi^*_{z+1}| + |E(S_{z+1})|) + |\Pi_z^T|.\end{aligned} \tag{10}$$

**Case 3: $S_{z-1}, S_{z+1} \in \mathcal{S}^{\text{light}}$, and accounting so far.** If $S_{z-1}, S_{z+1} \in \mathcal{S}^{\text{light}}$, then we do not perform any modifications for now. We now bound the total number of crossings in the current drawing $\varphi'_z$ of graph $G_z$ for cases 1–3, excluding the crossings between pairs of edges in $E_z^{\text{over}}$. If Case 3 happened, then the number of crossings did not increase in this step. If Case 1 happened, then the total number of primary crossings of types 1,2 and 4, and secondary crossings of type 2 did not change, and the number of primary crossings of type 3 and secondary crossings of type 1 is bounded by Equation (9). Similarly, If Case 2 happened, then the total number of primary crossings of types 1,2 and 3, and secondary crossings of type 1 did not change, and the number of primary crossings of type 4 and secondary crossings of type 2 is bounded by Equation (10). Therefore, if any of the cases 1–3 happened, and event $\mathcal{E}$ did not happen, then the total expected number of crossings in the current drawing $\varphi'_z$ of graph $G_z$, excluding the crossings between pairs of edges in $E_z^{\text{over}}$, is at most:

$$\begin{aligned}\hat{\eta}^2\left(|\chi^*_{z-1}|+|\chi^*_z|+|\chi^*_{z+1}|+|E(S_{z-1})|+|E(S_{z+1})|\right)\\ +\hat{\eta}\sum_{(e,e')\in\chi^*}\left(\mathbf{E}\left[N'_z(e'_1)\right]+\mathbf{E}\left[N'_z(e'_2)\right]\right)\\ +\hat{\eta}\cdot\sum_{e\in E(U_{z-1})\setminus E(S_{z-1})}N'_z(e)+\hat{\eta}\cdot\sum_{e\in E(\overline{U}_z)\setminus E(S_{z+1})}N'_z(e)+|\Pi^T_z|.\end{aligned}\tag{11}$$

**Case 4:** $S_{z-1},S_{z+1}\in\mathcal{S}^{\text{bad}}$**.** In this case, we will perform a type-1 uncrossing of the images of the edges of $E_{z-1}\cup E_z^{\text{left}}\cup E_z^{\text{over}}\cup E_z^{\text{right}}\cup E_z=\delta_G(U_{z-1})\cup\delta_G(U_z)$. Let $\Gamma_1$ denote the set curves reprsenting the images of these edges in $\varphi'_z$. Let $\Gamma_2$ denote the set of curves reprsenting the images of all remaining edges of $G_z$ in $\varphi'_z$. We apply the algorithm from Theorem 4.33 to compute a new collection $\Gamma'_1$ of curves, where, for each edge $e\in\delta_G(U_{z-1})\cup\delta_G(U_z)$, there is a curve $\gamma(e)\in\Gamma'_1$ connecting the images of the endpoints of $e$. We are guaranteed that every pair of curves in $\Gamma'_1$ may cross each other at most once, and, for each curve $\gamma\in\Gamma_2$, the number of crossings between $\gamma$ and the curves in $\Gamma'_1$ is no greater than the number of crossings between $\gamma$ and the curves in $\Gamma_1$. We modify the images of the edges in $\delta_G(U_{z-1})\cup\delta_G(U_z)$, so that for each such edge $e$, its new image is the curve $\gamma(e)\in\Gamma'_1$. Note that the total number of type-1 primary crossings does not change. The total number of all other crossings is bounded by $(|\delta_G(U_{z-1})|+|\delta_G(U_z)|)^2\leq O(|\delta_G(U_{z-1})|^2)+O(|\delta_G(U_z)|^2)$. Using the same reasoning as in Cases 1 and 2, if event $\mathcal{E}$ did not happen, then:

$$|\delta_G(U_{z-1})|^2\leq\hat{\eta}^2\cdot\left(|\chi^*_{z-1}|+|E(S_{z-1})|\right),$$

and

$$|\delta_G(U_z)|^2\leq\hat{\eta}^2\cdot\left(|\chi^*_{z+1}|+|E(S_{z+1})|\right),$$

Therefore, if event $\mathcal{E}$ did not happen, the total expected number of crossings in the current drawing is bounded by:

$$\hat{\eta}^2\cdot\left(|\chi^*_{z+1}|+|\chi^*_z|+|\chi^*_{z+1}|+|E(S_{z-1})|+|E(S_{z+1})|\right).\tag{12}$$

**Uncrossing the Edges of** $E_z^{\text{over}}$**.** So far we have constructed a drawing $\varphi'_z$ of graph $G_z$ and bounded the expected number of crossings in $\varphi'_z$, excluding the crossings between the images of the edges in $E_z^{\text{over}}$. In this step, we eliminate all crossings of the latter type, by performing a type-2 uncrossing of the images of the edges in $E_z^{\text{over}}$. Specifically, we let $\tilde{\mathcal{Q}}$ be the set of paths in graph $G_z$ that contains, for each edge $e\in E_z^{\text{over}}$, a path $\tilde{Q}(e)$, that consists of the edge $e$ only. Recall that each edge $e\in E_z^{\text{over}}$ connects the special vertices $v^*$, $v^{**}$ to each other. We view each such path $\tilde{Q}(e)$ as being directed from $v^*$ to $v^{**}$. We then apply the algorithm from Theorem 4.37 that performs a type-2 uncrossing on the images of the paths in $\tilde{\mathcal{Q}}$. Let $\Gamma$ be the resulting set of curves that it produces. Recall that, for every edge $e\in E_z^{\text{over}}$, there must be a curve $\gamma(e)\in\Gamma$, that contains the segment of the image of edge $e$ that lies in the disc $D(v^*)$. We replace the current image of the edge $e$ with the curve $\gamma(e)$. Once the images of all edges $e\in E_z^{\text{over}}$ are modified, we obtain the final modified drawing $\varphi''_z$ of graph $G_z$. The algorithm from Theorem 4.37 ensures that the images of the edges in $E_z^{\text{over}}$ do not cross each other. Since the curves in $\Gamma$ are aligned with the graph that consists of the edges of $E_z^{\text{over}}$, we are guaranteed that, for each edge $e\in E(G_z)\setminus E_z^{\text{over}}$, the number of crossings in which edge $e$ participates does not increase. The algorithm from Theorem 4.37, and the type-1 uncrossings that we performed in Cases 1 – 3 ensure that the order in which the images of the edges of $\delta_{G_z}(v^*)$ enter the image of $v^*$

does not change, and remain consistent with the rotation $\mathcal{O}_{v^*} \in \Sigma_z$. To summarize, we have obtained a drawing $\varphi_z''$ of graph $G_z$, such that, for every vertex $v \in V(G_z) \setminus \{v^{**}\}$, the images of the edges of $\delta_{G_z}(v)$ enter the image of $v$ in the order consistent with the rotation $\mathcal{O}_v \in \Sigma_z$, and the total expected number of crossings in $\varphi_z''$ is bounded by:

$$\begin{aligned}\hat{\eta}^2 \left(|\chi^*_{z-1}| + |\chi^*_z| + |\chi^*_{z+1}| + |E(S_{z-1})| + |E(S_{z+1})|\right)& \\ + \hat{\eta} \cdot \sum_{(e,e') \in \chi^*} \left(\mathbf{E}\left[N'_z(e'_1)\right] + \mathbf{E}\left[N'_z(e'_2)\right]\right)& \\ + \hat{\eta} \cdot \sum_{e \in E(U_{z-1}) \setminus E(S_{z-1})} N'_z(e) + \hat{\eta} \cdot \sum_{e \in E(\overline{U}_z) \setminus E(S_{z+1})} N'_z(e) + |\Pi^T_z|.&\end{aligned} \tag{13}$$

In the next and the final step, we obtain the final solution $\varphi_z$ to instance $I_z = (G_z, \Sigma_z)$ of MCNwRS, by modifying the current drawing $\varphi_z''$ of graph $G_z$ inside the tiny $v^{**}$-disc $D(v^{**})$.

### 7.4.4 Step 4: the Final Drawing of Graph $G_z$

In this step we slightly modify the current drawing $\varphi_z''$ of graph $G_z$ in order to obtain the final drawing $\varphi_z$ of $G_z$, which is a valid solution to instance $I_z = (G_z, \Sigma_Z)$ of MCNwRS.

Consider the tiny $v^{**}$-disc $D = D_{\varphi_z''}(v^{**})$. Denote $\delta_{G_z}(v^{**}) = \{e_1, \ldots, e_h\}$, and, for all $1 \leq i \leq h$, let $p_i$ be the point on the image of the edge $e_i$ in $\varphi_z''$ that lies on the boundary of the disc $D$. We assume that the edges are indexed so that the points $p_1, \ldots, p_h$ are encountered in this order when traversing the boundary of $D$ in the clock-wise direction. We denote by $\mathcal{O}$ this ordering of the edges $e_1, \ldots, e_h$. Let $\mathcal{O}'$ be the ordering $\mathcal{O}_{v^{**}} \in \Sigma_z$ of the edges of $\delta_{G_z}(v^{**})$. We use the algorithm from Corollary 4.32 to compute a collection $\Gamma = \{\gamma(e_i) \mid 1 \leq i \leq h\}$ of curves, such that, for each edge $e_i$, curve $\gamma(e_i)$ only differs from the image of the edge $e_i$ in the current drawing $\varphi_z''$ of $G_z$ inside the disc $D$, and the curves of $\Gamma$ enter the image of $v^{**}$ in the order $\mathcal{O}'$. We then replace, for each edge $e_i \in \delta_{G_z}(v^{**})$, the current image of the edge $e_i$ with the curve $\gamma(e_i)$. As the result, we obtain a valid solution $\varphi_z$ to instance $I_z = (G_z, \Sigma_z)$ of MCNwRS, as the images of the edges in $\delta_{G_z}(v^{**})$ now enter the image of $v^{**}$ in the correct order. Corollary 4.32 guarantees that the number of crossings between the curves in $\Gamma$ within the disc $D$ is bounded by $O(\mathsf{dist}(\mathcal{O}, \mathcal{O}'))$, and these are the only new crossings. Therefore, the number of crossings grows by at most $O(\mathsf{dist}(\mathcal{O}, \mathcal{O}'))$. In the next claim we bound $\mathsf{dist}(\mathcal{O}, \mathcal{O}')$.

**Claim 7.56** *If event $\mathcal{E}$ did not happen, then the expectation of $\mathsf{dist}(\mathcal{O}, \mathcal{O}')$ is bounded by:*

$$\begin{aligned}\hat{\eta}^{O(1)} \cdot \left(\sum_{e \in E(G)} \mathbf{E}\left[N'_z(e)\right] + \sum_{(e,e') \in \chi^*} \left(\mathbf{E}\left[N'_z(e)\right] + \mathbf{E}\left[N'_z(e')\right]\right)\right)& \\ + \hat{\eta}^{O(1)} \cdot \left(|\chi^*_{z-1}| + |\chi^*_z| + |\chi^*_{z+1}| + |E(S_{z-1})| + |E(\tilde{S}_z)| + |E(S_{z+1})| + |\delta_G(\tilde{S}_z)| + |\delta_G(S_{z-1})|\right)& \\ + \hat{\eta} \cdot \mathsf{cr}(\varphi_z'') + |\Pi^T_z|.&\end{aligned}$$

We prove Claim 7.56 below, after we complete the proof of Claim 7.54 using it. For convenience, we denote by $E^*_z = E(S_{z+1}) \cup E(S_{z-1}) \cup E(\tilde{S}_z) \cup \delta_G(\tilde{S}_z) \cup \delta_G(S_{z-1})$. Combining the bound from Equation (13) with the bound from Claim 7.56, we get that, if Event $\mathcal{E}$ did not happen, then $\mathbf{E}\left[\mathsf{OPT}_{\mathsf{cnwrs}}(I_z)\right]$ is bounded by:

$$\hat{\eta}^{O(1)} \left(|\chi^*_{z-1}| + |\chi^*_z| + |\chi^*_{z+1}| + |E^*_z| + \sum_{(e,e') \in \chi^*} \left(\mathbf{E}\left[N'_z(e'_1)\right] + \mathbf{E}\left[N'_z(e'_2)\right]\right) + \sum_{e \in E(G)} \mathbf{E}\left[N'_z(e)\right] + |\Pi^T_z|\right).$$

Note that an edge $e \in E(G)$ may belong to at most $O(1)$ sets in $\{E^*_z\}_{z=1}^r$. Also, a crossing $(e,e') \in \chi^*$ may belong to at most two sets in $\{\chi^*_z\}_{z=1}^r$ Therefore, we get that:

$$\begin{aligned}\mathbf{E}\left[\sum_{z=1}^{r}\mathsf{OPT}_{\mathsf{cnwrs}}(I_z)\right] &\leq \hat{\eta}^{O(1)}(|E(G)|+|\chi^*|)\\ &\quad+\hat{\eta}^{O(1)}\cdot\sum_{(e,e')\in\chi^*}\sum_{z=1}^{r}\left(\mathbf{E}\left[N'_z(e'_1)\right]+\mathbf{E}\left[N'_z(e'_2)\right]\right)\\ &\quad+\hat{\eta}^{O(1)}\cdot\sum_{e\in E(G)}\sum_{z=1}^{r}\mathbf{E}\left[N'_z(e)\right]\\ &\quad+\hat{\eta}^{O(1)}\cdot\sum_{z=1}^{r}|\Pi^T_z|\end{aligned} \tag{14}$$

We use the following two observations, whose proofs appear in Appendix G.23 and Appendix G.24, respectively, in order to complete the proof of Claim 7.54.

**Observation 7.57** *For every edge $e \in E(G)$, $\sum_{z=1}^r \mathbf{E}\left[N'_z(e)\right] \leq O(\hat{\eta})$.*

**Observation 7.58** $\sum_{z=1}^r |\Pi^T_z| \leq (|E(G)|+|\chi^*|)\cdot O(\log^{68} m)$.

Combining Equation (14) with Observations 7.57 and 7.58, and recalling that $\hat{\eta} = 2^{O((\log m)^{3/4}\log\log m)}$, we get that, if event $\mathcal{E}$ did not happen:

$$\mathbf{E}\left[\sum_{z=1}^{r}\mathsf{OPT}_{\mathsf{cnwrs}}(I_z)\right] \leq 2^{O((\log m)^{3/4}\log\log m)}\cdot(|E(G)|+|\chi^*|).$$

Recall that $\mathbf{Pr}\left[\mathcal{E}\right] \leq 1/m^{99}$, and, if $\mathcal{E}$ happens, $\sum_{z=1}^r \mathsf{OPT}_{\mathsf{cnwrs}}(I_z) \leq m^3$ must hold. Therefore, overall, $\mathbf{E}\left[\sum_{z=1}^r \mathsf{OPT}_{\mathsf{cnwrs}}(I_z)\right] \leq 2^{O((\log m)^{3/4}\log\log m)}\cdot(|E(G)|+|\chi^*|)$. In order to complete the proof of Claim 7.54, it is now enough to prove Claim 7.56, which we do next.

## 7.5 Proof of Claim 7.56

Assume first that $S_{z+1} \in \mathcal{S}^{\text{bad}}$. Clearly, $\mathsf{dist}(\mathcal{O},\mathcal{O}') \leq |\delta_{G_z}(v^{**})|^2 \leq |\delta_G(U_z)|^2$. From Corollary 7.52, $|\delta_G(U_z)| \leq |\delta_G(S_{z+1})|\cdot O(\log^{34} m)$. If the bad event $\mathcal{E}$ does not happen, then, from Observation 7.49, $\mathsf{OPT}_{\mathsf{cnwrs}}(S_{z+1},\Sigma(S_{z+1})) + |E(S_{z+1})| \geq \frac{|\delta_G(S_{z+1})|^2}{\hat{\eta}}$, where $\Sigma(S_{z+1})$ is the rotation system for graph $S_{z-1}$ induced by $\Sigma$. Therefore:

$$\mathsf{dist}(\mathcal{O},\mathcal{O}') \leq |\delta_G(U_z)|^2 \leq O(\log^{68} m)\cdot|\delta_G(S_{z+1})|^2 \leq \hat{\eta}^2\cdot\left(|\chi^*_{z+1}|+|E(S_{z+1})|\right).$$

Assume now that $S_z \in \mathcal{S}^{\text{bad}}$. As before, $\mathsf{dist}(\mathcal{O},\mathcal{O}') \leq |\delta_G(U_z)|^2$. From Corollary 7.52, $|\delta_G(U_z)| \leq |\delta_G(S_z)|\cdot O(\log^{34} m)$. If the bad event $\mathcal{E}$ does not happen, then, from Observation 7.49, $\mathsf{OPT}_{\mathsf{cnwrs}}(S_z,\Sigma(S_z)) + |E(S_z)| \geq \frac{|\delta_G(S_z)|^2}{\hat{\eta}}$, where $\Sigma(S_z)$ is the rotation system for graph $S_z$ induced by $\Sigma$. Therefore:

$$\mathsf{dist}(\mathcal{O},\mathcal{O}') \leq |\delta_G(U_z)|^2 \leq O(\log^{68} m)\cdot|\delta_G(S_z)|^2 \leq \eta^2\cdot\left(|\chi^*_z|+|E(S_z)|\right).$$

We assume from now on that $S_z, S_{z+1} \in \mathcal{S}^{\text{light}}$. In order to complete the proof of Claim 7.56, we will define, for every edge $e \in \delta_G(U_z)$, a curve $\gamma(e)$, such that all curves in the resulting set $\Gamma^* =$

$\{\gamma(e) \mid e \in \delta_G(U_z)\}$ are in general position; each one of the curves originates at the image of vertex $v^{**}$ in the drawing $\varphi''_z$ of $G_z$; and each one of the curves terminates at the image of vertex $u_z$ in the drawing $\varphi''_z$. We will ensure that the order in which the curves in set $\Gamma^*$ enter the image of $v^{**}$ is precisely the ordering $\mathcal{O}$ of their corresponding edges, while the order in which they enter the image of $u_z$ is precisely the ordering $\mathcal{O}'$ of their corresponding edges. We will then bound the number of crossings between the curves in $\Gamma^*$, thereby bounding $\mathsf{dist}(\mathcal{O}, \mathcal{O}')$.

In order to define the set $\Gamma^*$ of curves, we define, for every edge $e \in \delta_G(U_z)$, a path $\tilde{R}(e)$ in graph $G_z$, that connects vertex $v^{**}$ to vertex $u_z$, and originates at edge $e$. For each edge $e \in \delta_G(U_z)$, the curve $\gamma(e)$ is then obtained by slightly altering the image of the path $\tilde{R}(e)$ in the drawing $\varphi''_z$, in order to ensure that all resulting curves in $\Gamma^*$ are in general position. We start by defining the set $\tilde{\mathcal{R}} = \left\{\tilde{R}(e) \mid e \in \delta_G(U_z)\right\}$ of paths.

**Set $\tilde{\mathcal{R}} = \left\{\tilde{R}(e) \mid e \in \delta_G(U_z)\right\}$ of paths.** Consider an edge $e \in \delta_G(U_z)$. Assume first that $e \in E_z$. Denote $e = (u, v)$, where $u \in S_z$ and $v \in S_{z+1}$ (see Figure 22). Note that edge $e$ belongs to graph $G_z$, where it connects vertex $u$ to vertex $v^{**}$. Let $\tilde{Q}(e)$ be the unique path of the internal $U_z$-router $\mathcal{Q}(U_z)$ that originates at edge $e$; recall that the path terminates at vertex $u_z$, and, from the construction of the path set $\mathcal{Q}(U_z)$, path $\tilde{Q}(e)$ is also the unique path of the internal $S_z$-router $\mathcal{Q}(S_z)$ that originates at edge $e$. Therefore, all internal vertices of path $\tilde{Q}(e)$ lie in $S_z$, and path $\tilde{Q}(e)$ is contained in graph $G_z$. We then let $\tilde{R}(e)$ be the path $\tilde{Q}(e)$ in graph $G_z$ (that now connects vertex $u_z$ to vertex $v^{**}$.)

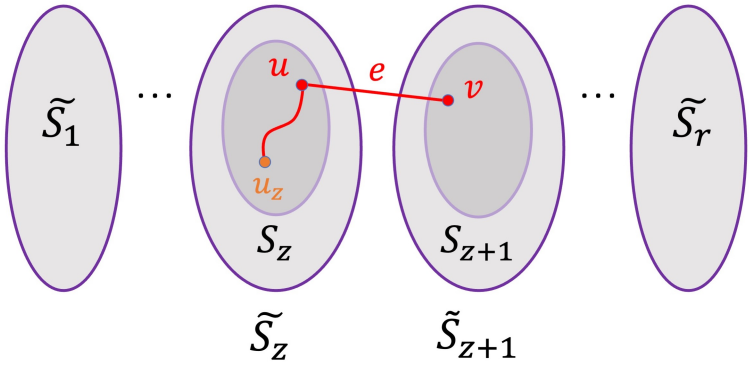



Figure 22: Definition of path $\tilde{R}(e)$ when $e \in E_z$.

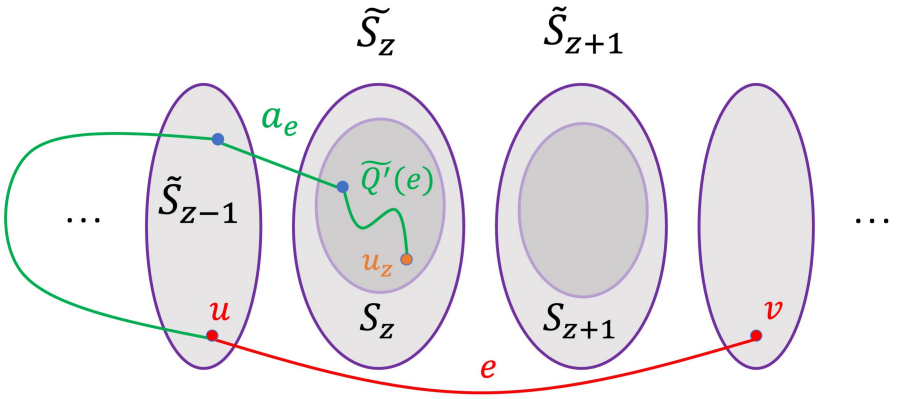



Figure 23: Construction of path $\tilde{R}(e)$ when $e \in E_z^{\text{over}}$. Path $\tilde{Q}(e)$ is shown in green.

Next, we consider an edge $e \in E_z^{\text{over}}$. Assume that $e = (u, v)$, where $u \in \bigcup_{i<z} V(\tilde{S}_i)$ and $v \in \bigcup_{i>z} V(\tilde{S}_i)$

(see Figure 23). Consider the unique path $\tilde{Q}(e) \in \mathcal{Q}(U_z)$ that originates at edge $e$. Recall that the path terminates at vertex $u_z$, and it must contain some edge $a_e \in \delta_G(S_z)$ (in fact it may only contain one such edge). Note that, in graph $G_z$, all vertices of $U_{z-1}$ were contracted into vertex $v^*$, and so both edges $e$ and $a_e$ are incident to vertex $v^*$ in $G_z$. The subpath of the path $\tilde{Q}(e)$ from edge $a_e$ to vertex $u_z$ is precisely the unique path of the internal router $\mathcal{Q}(S_z)$ that originates at edge $a_e$, which we denote by $\tilde{Q}'(e)$. We then let $\tilde{R}(e)$ be the path obtained by appending the edge $e$ at the beginning of the path $\tilde{Q}'(e)$. Note that path $\tilde{R}(e)$ is contained in graph $G_z$, and it connects vertex $v^{**}$ to vertex $u_z$. In fact, path $\tilde{R}(e)$ is a concatenation of edge $(v^*, v^{**})$ and path $\tilde{Q}'(e)$.

Lastly, we consider an edge $e \in E_z^{\text{right}}$. Assume that $e = (u, v)$, where $u \in V(\tilde{S}_z)$, and $v \in \bigcup_{i>z} V(\tilde{S}_i)$. Note that edge $e$ is also present in graph $G_z$, where it now connects vertex $u$ to vertex $v^{**}$. Consider the unique path $\tilde{Q}(e) \in \mathcal{Q}(U_z)$ that originates at edge $e$, and recall that this path terminates at vertex $u_z$. We now consider two cases. First, if path $\tilde{Q}(e)$ is contained in cluster $\tilde{S}_z$, then it is contained in the current graph $G_z$, except that now it connects vertex $v^{**}$ to vertex $u_z$. We then set $\tilde{R}(e) = \tilde{Q}(e)$ (see Figure 24). We denote by $a_e$ the unique edge of $\tilde{R}(e)$ that lies in $\delta_G(S_z)$. Otherwise, let $u''$ be the first vertex on path $\tilde{Q}(e)$ that does not belong to $\tilde{S}_z$, and let $u'$ be the vertex preceding $u''$ on the path (see Figure 25). Denote $a_e^* = (u', u'')$. Note that edge $a_e^*$ also lies in graph $G_z$, where it connects vertex $u'$ to vertex $v^*$. Moreover, path $\tilde{Q}(e)$ must now contain some edge $a_e \in \delta_G(S_z)$. Since, in graph $G_z$, all vertices of $U_{z-1}$ were contracted into the vertex $v^*$, edge $a_e$ is now incident to vertex $v^*$. The subpath of the path $\tilde{Q}(e)$ from edge $a_e$ to vertex $u_z$, that we denote by $\tilde{Q}'(e)$, is precisely the unique path of the internal $S_z$-router $\mathcal{Q}(S_z)$ that originates at edge $a_e$. We then let $\tilde{R}(e)$ be the path obtained by concatenating the subpath of $\tilde{Q}(e)$ from edge $e$ to edge $a_e^*$ (that, in graph $G_z$, connects $v^{**}$ to $v^*$), and the path $\tilde{Q}'(e)$ (that originates at $v^*$ in $G_z$). Note that path $\tilde{R}(e)$ is contained in graph $G_z$, and it connects vertex $v^{**}$ to vertex $u_z$. Since $\delta_G(U_z) = E_z \cup E_z^{\text{over}} \cup E_z^{\text{right}}$, we have now constructed a path $\tilde{R}(e)$ for each edge $e \in \delta_G(U_z)$.

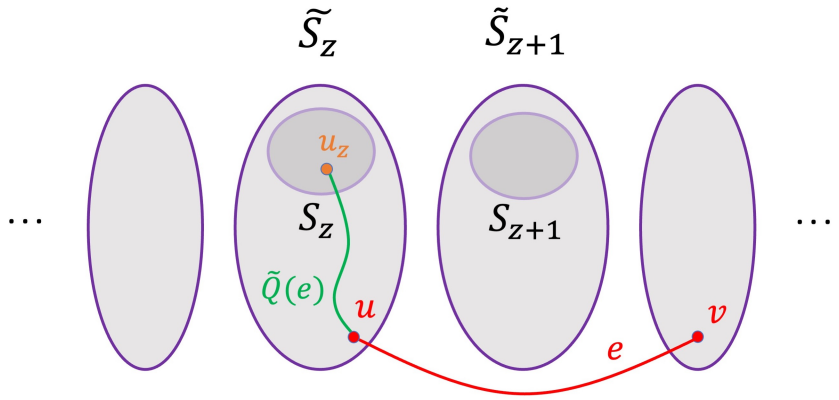



Figure 24: Construction of path $\tilde{R}(e)$ when $e \in E_z^{\text{right}}$ and $\tilde{Q}(e) \subseteq \tilde{S}_z$. Path $\tilde{Q}(e)$ is shown in green.

Consider the final set $\tilde{\mathcal{R}} = \left\{\tilde{R}(e) \mid e \in \delta_G(U_z)\right\}$ of paths that we have defined in graph $G_z$. Notice that, for each edge $e \in \delta_G(U_z)$, there is some edge $a_e \in \delta_G(S_z)$ that lies both on the path $\tilde{\mathcal{Q}}(e) \in \mathcal{Q}(U_z)$, and on path $\tilde{R}(e)$ (in the case where $e \in E_z$, we set $a_e = e$; for the other two cases, we have defined the edge $a_e$ explicitly). Moreover, the unique path of the internal $S_z$-router $\mathcal{Q}(S_z)$ that originates at edge $a_e$ is a subpath of $\tilde{R}(e)$. Therefore, the last edge on path $\tilde{R}(e)$ is identical to the last edge on the unique path $\tilde{Q}(e) \in \mathcal{Q}(U_z)$ that originates at edge $e$. Recall that we have defined the ordering $\mathcal{O} = \tilde{\mathcal{O}}_z$ of the edges of $\delta_G(U_z) = \delta_{G_z}(v^{**})$ to be $\mathcal{O}^{\text{guided}}(\mathcal{Q}(U_z), \Sigma)$ – the ordering that is guided by the internal $U_z$-router $\mathcal{Q}(U_z)$ (see definition in Section 5.2). Since the rotation $\mathcal{O}_{u_z}$ in $\Sigma$ and $\Sigma_z$ is identical, equivalently, $\mathcal{O}' = \mathcal{O}^{\text{guided}}(\tilde{\mathcal{R}}, \Sigma_z)$, that is, ordering $\mathcal{O}'$ can be defined as an ordering that is guided by the set $\tilde{\mathcal{R}}$ of paths in graph $G_z$, with respect to the rotation system $\Sigma_z$.

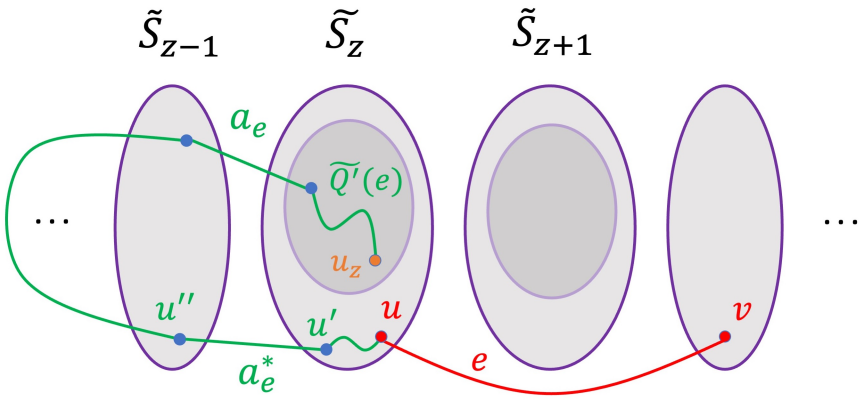



Figure 25: Construction of path $\tilde{R}(e)$ when $e \in E_z^{\text{right}}$ and $\tilde{Q}(e)$ is not contained in $\tilde{S}_z$. Path $\tilde{Q}(e)$ is shown in green.

Notice that for every edge $e \in \delta_G(U_z)$, an edge $e'$ may lie on path $\tilde{R}(e)$ only if $e'$ lies on path $\tilde{Q}(e) \in \mathcal{Q}(U_z)$. From our construction, if $e' \in \delta_{G_z}(U_z)$, then $e'$ may lie on at most one path of $\tilde{\mathcal{R}}$ – the path $\tilde{R}(e')$. From Observation 7.51, an edge $e' \in E(G_z) \setminus E(S_z)$ may participate in at most $O(\log^{34} m)$ paths of $\tilde{\mathcal{R}}$, and an edge $e' \in E(S_z)$ may participate in at most $O(\log^{34} m) \cdot \text{cong}_G(\mathcal{Q}(S_z), e')$ paths of $\tilde{\mathcal{R}}$.

Next, we construct an auxiliary graph $H'_z$, by replicating some edges of $G_z$ and deleting some other edges, similarly to our construction of graph $H_z$. We will use the paths of $\tilde{\mathcal{R}}$ in order to define a collection of edge-disjoint paths $\tilde{\mathcal{R}}'$ in the resulting graph $H'_z$, which will in turn be used in order to construct the collection $\Gamma^*$ of curves.

**Graph $H'_z$ and its drawing $\psi'_z$** For every edge $e \in E(G_z)$, we let $\tilde{N}(e)$ be the number of paths in $\tilde{\mathcal{R}}$ that contain the edge $e$. From the discussion so far, we obtain the following immediate observation.

**Observation 7.59** *For each edge $e \in E(S_z)$, $\tilde{N}(e) \leq O(\log^{34} m) \cdot \text{cong}_G(\mathcal{Q}(S_z), e')$, and for each edge $e \in E(G_z) \setminus E(S_z)$, $\tilde{N}(e) \leq O(\log^{34} m)$.*

In order to construct the graph $H'_z$, we start with the set $V(H'_z) = V(G_z)$ of vertices. For every edge $e \in E(G_z)$ with $\tilde{N}(e) > 0$, we add a collection $J'(e)$ of $\tilde{N}(e)$ parallel copies of the edge $e$ to graph $H'_z$. We also assign each copy of edge $e$ in set $J'(e)$ to a distinct path of $\tilde{\mathcal{R}}$ that contains the edge $e$, arbitrarily. As in Step 1 of the algorithm for computing a drawing of graph $G_z$, we can now define a collection $\tilde{\mathcal{R}}' = \left\{\tilde{R}'(e) \mid e \in \delta_G(U_z)\right\}$ of edge-disjoint paths in graph $H'_z$, as follows: for each edge $e \in \delta_G(U_z)$, path $\tilde{R}'(e)$ is obtained from path $\tilde{R}(e)$ by replacing each edge $e' \in \tilde{R}(e)$ with the copy of edge $e'$ that is assigned to path $\tilde{R}(e)$.

Drawing $\varphi''_z$ of graph $G_z$ naturally defines drawing $\psi'_z$ of graph $H'_z$: for each edge $e \in E(G_z)$ with $\tilde{N}(e) > 0$, we draw all copies of $e$ to appear in parallel to the image of $e$, without crossing each other.

As in Step 1 of the algorithm for constructing a drawing for graph $G_z$, we can assign the copies of the edges incident to vertex $u_z$ more carefully, to ensure that the images of the paths in $\tilde{\mathcal{R}}'$ enter the image of $u_z$ according to the ordering $\tilde{\mathcal{O}}_z = \mathcal{O}'$. In other words, if we denote $\delta_G(U_z) = \{\tilde{a}'_1, \tilde{a}'_2, \ldots \tilde{a}'_h\}$, and the edges are indexed in the order of their appearance in the ordering $\tilde{\mathcal{O}}_z = \mathcal{O}'$, and if, for all $1 \leq i \leq h$, we denote by $\tilde{a}''_i$ the last edge on the path $\tilde{R}'(\tilde{a}'_i)$, then the images of the edges $\tilde{a}''_1, \ldots, \tilde{a}''_h$ enter the image of vertex $u_z$ in the natural order of their indices. Note that the images of edges $\tilde{a}'_1, \ldots, \tilde{a}'_h$ enter the image of $v^{**}$ in the ordering $\mathcal{O}$.

We now bound the number of crossings in the drawing $\psi'_z$ of graph $H'_z$. Consider any crossing in $\psi'_z$

between a pair of edges $e'_1, e'_2$. Assume that $e'_1$ is a copy of edge $e_1 \in E(G_z)$, and $e'_2$ is a copy of edge $e_2 \in E(G_z)$. Clearly, the images of the edges $e_1$ and $e_2$ must cross in drawing $\varphi''_z$, and we say that this crossing is responsible for the crossing $(e'_1, e'_2)$. It is easy to see that a crossing $(e_1, e_2)$ in drawing $\varphi_z$ may be responsible for at most $\tilde{N}(e_1) \cdot \tilde{N}(e_2)$ crossings in $\psi'_z$. If neither of the edges $e_1, e_2$ lie in $E(S_z)$, then, from Observation 7.59, $\tilde{N}(e_1), \tilde{N}(e_2) \leq O(\log^{34} m)$. Therefore, the total number of crossings of $\psi'_z$, for which crossings $(e_1, e_2)$ of $\varphi''_z$ with $e_1, e_2 \notin E(S_z)$ are responsible is at most: $O(\log^{68} m) \cdot \mathsf{cr}(\varphi''_z)$.

If exactly one of the two edges (say $e_1$) lies in $E(S_z)$, then, from Observation 7.59, $\tilde{N}(e_1) \leq O(\log^{34} m) \cdot \mathrm{cong}_G(\mathcal{Q}(S_z), e_1)$, while $\tilde{N}(e_2) \leq O(\log^{34} m)$. Moreover, crossing $(e_1, e_2)$ must be a type-1 primary crossing of $\varphi''_z$. Recall that the expected number of type-1 crossings in $\varphi''_z$ is bounded by $|\chi^*_z| \cdot \hat{\eta}$. Recall also that the random variable corresponding to the total number of type-1 primary crossings only depends on the random choices of the internal routers $\mathcal{Q}(S_{z-1})$ and $\mathcal{Q}(S_{z+1})$, and it is independent of the random choice of the internal router $\mathcal{Q}(S_z)$. For each edge $e \in E(S_z)$, $\mathbf{E}\left[\mathrm{cong}_G(\mathcal{Q}(S_z), e)\right] \leq \hat{\eta}$ (from Observation 7.49 and our assumption that $S_z \in \mathcal{S}^{\text{light}}$), and random variable $\mathrm{cong}_G(\mathcal{Q}(S_z), e)$ only depends on the selection of the internal $S_z$-router $\mathcal{Q}(S_z)$. Since the random variable representing the number of type-1 primary crossings of $\varphi''_z$ is independent from the random variables $\{\mathrm{cong}_G(\mathcal{Q}(S_z), e)\}_{e \in E(S_z)}$, we get that the total expected number of crossings of $\psi''_z$, for which crossings $(e_1, e_2)$ of $\varphi''_z$, with exactly one of $e_1, e_2$ lying in $E(S_z)$ are responsible, is at most: $|\chi^*_z| \cdot \hat{\eta}^{O(1)}$.

Lastly, assume that both edges $e_1, e_2 \in E(S_z)$. Then, from Observation 7.59, $\tilde{N}(e_1) \leq O(\log^{34} m) \cdot \mathrm{cong}_G(\mathcal{Q}(S_z), e_1)$ and $\tilde{N}(e_2) \leq O(\log^{34} m) \cdot \mathrm{cong}_G(\mathcal{Q}(S_z), e_2)$. The number of crossigns in drawing $\psi'_z$ for which crossing $(e_1, e_2)$ is responsible is then bounded by:

$$\begin{aligned} &O(\log^{68} m) \cdot \mathrm{cong}_G(\mathcal{Q}(S_z), e_1) \cdot \mathrm{cong}_G(\mathcal{Q}(S_z), e_2) \\ &\qquad \leq O(\log^{68} m) \left( (\mathrm{cong}_G(\mathcal{Q}(S_z), e_1))^2 + (\mathrm{cong}_G(\mathcal{Q}(S_z), e_2))^2 \right). \end{aligned}$$

From Observation 7.49, and since we have assumed that $S_z \in \mathcal{S}^{\text{light}}$, for every edge $e \in E(S_z)$, $\mathbf{E}\left[(\mathrm{cong}_G(\mathcal{Q}(S_z), e))^2\right] \leq \hat{\eta}$. As before, random variable $(\mathrm{cong}_G(\mathcal{Q}(S_z), e))^2$ only depends on the random selection of the internal $S_z$-router $\mathcal{Q}(S_z)$. Clearly, the expected number of crossings of $\psi'_z$ for which crossing $(e_1, e_2)$ is responsible is at most $O(\hat{\eta}^2)$. Note also that crossing $(e_1, e_2)$ must a type-1 primary crossing of $\varphi''_z$, and the expected number of such crossings is bounded by $|\chi^*_z| \cdot \hat{\eta}$. As before, the random variable corresponding to the number of type-1 primary crossings of $\varphi''_z$ does not depend on the selection of the internal $S_z$-router $\mathcal{Q}(S_z)$. Therefore, the total expected number of crossings of $\psi''_z$, for which crossings $(e_1, e_2)$ of $\varphi''_z$, with $e_1, e_2 \in E(S_z)$ are responsible is at most: $|\chi^*_z| \cdot \hat{\eta}^{O(1)}$.

Overall, the total expected number of crossings in drawing $\psi'_z$ of graph $H'_z$ is bounded by:

$$\mathsf{cr}(\varphi''_z) \cdot O(\log^{68} m) + |\chi^*_z| \cdot \hat{\eta}^{O(1)}.$$

**Constructing the set $\Gamma^*$ of curves.** For every edge $e \in \delta_G(U_z)$, we initially let $\gamma(e)$ be the image of the path $\tilde{R}'(e)$ in the drawing $\psi'_z$ of graph $H'_z$. From our construction, the curves in set $\Gamma^* = \{\gamma(e) \mid e \in \delta_G(U_z)\}$ all originate at the image of vertex $v^{**}$, and terminate at the image of vertex $u_z$ in $\psi'_z$. Moreover, from our construction, the order in which the curves of $\Gamma^*$ enter the image of $v^{**}$ is according to the ordering $\mathcal{O}$ of the edges of $\delta_G(U_z)$, while the order in which the curves of $\Gamma^*$ enter the image of $u_z$ is according to the ordering $\mathcal{O}'$ of the edges of $\delta_G(U_z)$. However, the curves of $\Gamma^*$ are not in general position, as a point $p$ may serve as an inner point on more than 2 such curves; this, however, may only happen if $p$ is an image of some vertex $v \in V(G_z) \setminus \{u_z, v^{**}\}$. We will now "nudge" the curves in the vicinity of each such vertex to ensure that the resulting set of curves is in general position. The nudging procedure is identical to that we have employed in Step 2 (see Section 7.4.2).

We process every vertex $v \in V(G_z) \setminus \{u_z, v^{**}\}$ one by one. Consider an iteration when any such vertex $v$ is processed. Let $A(v) \subseteq \delta_G(U_z)$ be the set of all edges $e \in \delta_G(U_z)$, such that curve $\gamma(e)$ contains the image of vertex $v$ (in $\psi'_z$). We denote $A(v) = \{a_1, \ldots, a_k\}$. Consider the tiny $v$-disc $D(v) = D_{\psi'_z}(v)$ in the drawing $\psi'_z$ of graph $H'_z$. For all $1 \leq i \leq k$, we let $s_i, t_i$ be the two points at which curve $\gamma(a_i)$ intersects the boundary of the disc $D(v)$. Note that all points $s_1, t_1, \ldots, s_k, t_k$ must be distinct. We use the algorithm from Claim 4.34 in order to construct a collection $\{\gamma'_1, \ldots, \gamma'_k\}$ of curves, such that, for all $1 \leq i \leq k$, curve $\gamma'_i$ has $s_i$ and $t_i$ as its endpoints, and is completely contained in $D(v)$. Recall that the claim ensures that, for every pair $1 \leq i < j \leq k$ of indices, if the two pairs $(s_i, t_i), (s_j, t_j)$ of points cross, then curves $\gamma_i, \gamma_j$ intersect at exactly one point; otherwise, curves $\gamma_i, \gamma_j$ do not intersect. For all $1 \leq i \leq k$, we modify the curve $\Gamma(a_i)$ as follows: we replace the segment of the curve between points $s_i, t_i$ with the curve $\gamma_i$.

Once every vertex $v \in V(G_z) \setminus \{u_z, v^{**}\}$ is processed, we obtain the final collection $\Gamma^* = \{\gamma(e) \mid e \in \delta_G(U_z)\}$ of curves, which are now in general position. The order in which these curves enter the images of vertices $u_z$ and $v^{**}$ did not change, but we may have added some new crossings over the course of this modification of the curves in $\Gamma^*$. For convenience, we say that a crossing between a pair of curves in $\Gamma^*$ is *primary* if this crossing existed before this last modification, and otherwise it is called *secondary*. For each point $p$ corresponding to a secondary crossing, there must be a vertex $v$, with $p \in D(v)$. The expected number of all primary crossings remains unchanged, and is bounded by $\mathsf{cr}(\varphi''_z) \cdot O(\log^{68} m) + |\chi^*_z| \cdot \hat{\eta}^{O(1)}$. We now bound the expected number of all secondary crossings.

Consider a pair $e_1, e_2 \in \delta_G(U_z)$ of distinct edges, and assume that there is some vertex $v \in V(G_z) \setminus \{u_z, v^{**}\}$, such that the curves $\gamma(e_1), \gamma(e_2)$ cross at some point $p \in D(v)$. Using the same arguments as before, this may only happen in one of two cases: either (i) some edge $e \in \delta_{G_z}(v)$ lies on both $\tilde{R}(e_1)$ and $\tilde{R}(e_2)$; or (ii) paths $\tilde{R}(e_1), \tilde{R}(e_2)$ have a transversal intersection at vertex $v$. In the former case, we say that the crossing is a type-1 secondary crossing, and that edge $e$ is responsible for it, while in the second case we say that the crossing is a type-2 secondary crossing, and that the transversal intersection of paths $\tilde{R}(e_1), \tilde{R}(e_2)$ at vertex $v$ is responsible for it.

Clearly, for every edge $e \in E(G_z)$, the total number of type-1 secondary intersections for which $e$ may be responsible is at most $(\tilde{N}(e))^2$. If $e \in \delta_{G_z}(v^{**})$, then $\tilde{N}(e) = 1$, and $e$ may not be responsible for any type-1 secondary crossings. If $e \in E(G_z) \setminus (E(S_z) \cup \delta_G(v^{**}))$, then, from Observation 7.59, $\tilde{N}(e) \leq O(\log^{34} m)$. Otherwise, if $e \in E(S_z)$, then, from Observation 7.59, $\tilde{N}(e) \leq O(\log^{34} m) \cdot \mathrm{cong}_G(\mathcal{Q}(S_z), e)$. Moreover, since we have assumed that $S_z \in \mathcal{S}^{\mathrm{light}}$, from Observation 7.49, $\mathbf{E}\left[\left(\mathrm{cong}_G(\mathcal{Q}(S_z), e)\right)^2\right] \leq \hat{\eta}$. Overall, the total expected number of type-1 secondary crossings is bounded by: $\hat{\eta} \cdot |E(G_z) \setminus \delta_{G_z}(v^{**})| \leq \hat{\eta} \cdot (|E(\tilde{S}_z)| + |\delta_G(U_{z-1}|)$. Since, from Corollary 7.52, $|\delta_G(U_{z-1})| \leq |\delta_G(S_z)| \cdot O(\log^{34} m)$, and $\delta_G(S_z) \subseteq E(\tilde{S}_z) \cup \delta_G(\tilde{S}_z)$, we get that the total expected number of type-1 secondary crossings is at most: $\hat{\eta} \cdot \Big(|E(\tilde{S}_z)| + |\delta_G(\tilde{S}_z)|\Big)$.

We now turn to bound the number of type-2 secondary crossings. Consider any such crossing between a pair of curves $\gamma(e_1), \gamma(e_2)$, and assume that this crossing is charged to transversal intersection of the paths $\tilde{R}(e_1), \tilde{R}(e_2)$ at vertex $v$ with respect to $\Sigma_z$. We claim that in this case, $v = v^*$ must hold. Indeed, assume otherwise. As vertex $v^{**}$ may not serve as an inner vertex on any path of $\tilde{\mathcal{R}}'$, it must be the case that $v \in V(\tilde{S}_z)$. If $v \in V(S_z)$, then there must be two paths $Q, Q' \in \mathcal{Q}(S_z)$, such that $Q \subseteq \tilde{R}(e_1)$ and $Q' \subseteq \tilde{R}(e_2)$. From the construction of the internal router $\mathcal{Q}(S_z)$, all paths in $\mathcal{Q}(S_z)$ are non-transversal with respect to $\Sigma$, so it is impossible that $Q$ and $Q'$ have a transversal intersection at $v$, and the same is true for paths $\tilde{R}(e_1)$ and $\tilde{R}(e_2)$. Otherwise, $v \in V(\tilde{S}_z) \setminus V(S_z)$. In this case, $v$ must be a vertex that lies on each of the two paths $\tilde{Q}(e_1), \tilde{Q}(e_2)$ of the internal router $\mathcal{Q}(U_z)$, and moreover, the two paths must have a transversal intersection at $v$. But that is impossible from Observation 7.50. Therefore, $v = v^*$ must hold.

We denote by $\Pi$ the set of all pairs $(e_1, e_2) \in \delta_G(U_z)$ of edges, such that paths $\tilde{R}(e_1), \tilde{R}(e_2)$ have

a transversal intersection at vertex $v^*$, with respect to $\Sigma_Z$. Clearly, the number of type-2 secondary crossings between the curves of $\Gamma^*$ is bounded by $|\Pi|$. We use the following claim, whose proof appears in Section 7.6, in order to bound $\mathbf{E}\left[|\Pi|\right]$.

**Claim 7.60**

$$\mathbf{E}\left[|\Pi|\right] \leq \hat{\eta}^2 \cdot \left( \sum_{e \in E(G)} \mathbf{E}\left[N'_z(e)\right] + \sum_{(e,e') \in \chi^*} \left(\mathbf{E}\left[N'_z(e)\right] + \mathbf{E}\left[N'_z(e')\right]\right) \right) + \hat{\eta}^2 \cdot \Big( |E(S_{z-1})| + |E(\tilde{S}_z)| + |\delta_G(S_{z-1})| + |\chi^*_{z-1}| + |\chi^*_z| \Big) + |\Pi^T_z|.$$

In order to complete the proof Claim 7.56, it now remains to bound the expected number of crossings between the curves of $\Gamma^*$. Recall that the expected number of all primary crossings between the curves of $\Gamma^*$ is bounded by $\mathsf{cr}(\varphi''_z) \cdot O(\log^{68} m) + |\chi^*_z| \cdot \hat{\eta}^{O(1)}$, while the expected number of type-1 secondary crossings is at most $\hat{\eta} \cdot \Big( |E(\tilde{S}_z)| + |\delta_G(\tilde{S}_z)| \Big)$. The expected number of type-2 secondary crossings is bounded by $\mathbf{E}\left[|\Pi|\right]$. We conclude that, if $S_z, S_{z+1} \in \mathcal{S}^{\text{light}}$, then the expected number of all crossings between the curves in $\Gamma^*$ is bounded by:

$$\hat{\eta}^{O(1)} \cdot \left( \sum_{e \in E(G)} \mathbf{E}\left[N'_z(e)\right] + \sum_{(e,e') \in \chi^*} \left(\mathbf{E}\left[N'_z(e)\right] + \mathbf{E}\left[N'_z(e')\right]\right) \right) + \eta^{O(1)} \cdot \Big( |\chi^*_{z-1}| + |\chi^*_z| + |E(S_{z-1})| + |E(\tilde{S}_z)| + |\delta_G(\tilde{S}_z)| + |\delta_G(S_{z-1})| \Big) + \hat{\eta} \cdot \mathsf{cr}(\varphi''_z) + |\Pi^T_z|.$$

In order to complete the proof of Claim 7.56, it now remains to prove Claim 7.60, which we do next.

## 7.6 Proof of Claim 7.60

Assume first that $S_{z-1} \in \mathcal{S}^{\text{bad}}$. Since we have assumed that Event $\mathcal{E}$ did not happen, from Observation 7.49, $\mathsf{OPT}_{\mathsf{cnwrs}}(S_{z-1}, \Sigma(S_{z-1})) + |E(S_{z-1})| \geq \frac{|\delta_G(S_{z-1})|^2}{\hat{\eta}}$, where $\Sigma(S_{z-1})$ is the rotation system for graph $S_{z-1}$ induced by $\Sigma$. We then get that $|\delta_G(S_{z-1})|^2 \leq \hat{\eta} \cdot (|\chi^*_{z-1}| + |E(S_{z-1})|)$. On the other hand, if $(e_1, e_2) \in \Pi$, then paths $\tilde{Q}(e_1), \tilde{Q}(e_2) \in \mathcal{Q}(U_z)$ must each contain an edge of $\delta_G(S_{z-1})$. Since, from Observation 7.51, each edge of $\delta_G(S_{z-1})$ may appear on at most $O(\log^{34} m)$ paths of $\mathcal{Q}(U_{z-1})$, we get that $|\Pi| \leq O(\log^{68} m) \cdot |\delta_G(S_{z-1})|^2 \leq \hat{\eta}^2 \cdot (|\chi^*_{z-1}| + |E(S_{z-1})|)$. From now on we assume that $S_{z-1} \in \mathcal{S}^{\text{light}}$.

Consider a pair of edges $(e_1, e_2) \in \Pi$. Note that both $\tilde{R}(e_1)$ and $\tilde{R}(e_2)$ must contain the vertex $v^*$, and $e_1, e_2 \in \delta_G(U_z)$ must hold. Recall that $\delta_G(U_z) = E_z \cup E^{\text{right}}_z \cup E^{\text{over}}_z$, and that, for each edge $e \in E_z$, path $\tilde{R}(e)$ may not contain vertex $v^*$. Therefore, $e_1, e_2 \in E^{\text{right}}_z \cup E^{\text{over}}_z$ must hold. Note that, for an edge $e \in E^{\text{right}}_z$, it is possible that path $\tilde{R}(e)$ does not contain the vertex $v^*$. For convenience, we denote by $E^{\text{right}'}_z$ the set of all edges $e \in E^{\text{right}}_z$ for which $v^* \in \tilde{R}(e)$. We denote by $\Pi^1 \subseteq \Pi$ the set of all pairs $(e_1, e_2)$ where at least one of the two edges $e_1, e_2$ lies in $E^{\text{right}'}_z$, and we denote by $\Pi^2 = \Pi \setminus \Pi^1$. Clearly, for every pair $(e_1, e_2) \in \Pi^2$, $e_1, e_2 \in E^{\text{over}}_z$ must hold. We will now define, for each edge $e \in E^{\text{right}'}_z \cup E^{\text{over}}_z$ three special edges $a^*_e, a_e$, and $\hat{a}_e$ associated with $e$, a new cycle $\hat{W}(e)$ in graph $G$, and some additional structures.

Consider first an edge $e \in E^{\text{over}}_z$. Denote $e = (u, v)$, and assume that $u$ is the left endpoint of the edge. Then, from the definition of edge set $E^{\text{over}}_z$, $u \in \bigcup_{i<z} V(\tilde{S}_i)$ and $v \in \bigcup_{i>z} V(\tilde{S}_i)$ must hold (see

Figure 26). In particular, $e \in \delta_G(U_{z-1}) \cap \delta_G(U_z)$. We denote $a_e^* = e$. Clearly, $v^* \in \tilde{R}(e)$ must hold. We let $a_e$ be the edge immediately following vertex $v^*$ on path $\tilde{R}(e)$. Observe that $a_e$ is also an edge of graph $G$, where it must belong to edge set $E_{z-1}$ (see Figure 26 and Figure 23). We denote the endpoints of edge $a_e$ graph $G$ by $a_e = (x_e, y_e)$, with $x_e \in V(S_{z-1})$ and $y_e \in V(S_z)$. Since edge $a_e$ lies on path $\tilde{Q}(e)$, that path visits the cluster $S_{z-1}$. We denote by $\hat{x}_e$ the first vertex on path $Q(e)$ that belongs to cluster $S_{z-1}$, by $\hat{a}_e$ the edge preceding vertex $\hat{x}_e$ on the path, and by $\hat{y}_e$ the other endpoint of edge $\hat{a}_e$ (see Figure 26). From the construction of the set $\mathcal{Q}(U_{z-1})$ of paths, and the auxiliary cycles in $\mathcal{W}$, edges $\hat{a}_e$ and $a_e$ must lie on the cycle $W(e)$. We denote by $W'(e)$ the subpath of the auxiliary cycle $W(e)$ that connects vertex $\hat{y}_e$ to vertex $y_e$, such that all inner vertices of $W'(e)$ lie in $S_{z-1}$. We denote by $\hat{W}^{\text{left}}(e)$ the subpath of $W(e)$ from $\hat{y}_e$ to $v$, that is internally disjoint from $W'(e)$, and by $\hat{W}^{\text{right}}(e)$ the subpath of $W(e)$ from $u$ to $y_e$ that is internally disjoint from $W'(e)$. Observe that edge $e$ lies on both $\hat{W}^{\text{right}}(e)$ and $\hat{W}^{\text{left}}(e)$. Let $P(e)$ be the path of the internal $S_{z-1}$-router $\mathcal{Q}(S_{z-1})$ that originates at edge $a_e$, and let $\hat{P}(e)$ be the path of $\mathcal{Q}(S_{z-1})$ that originates at edge $\hat{a}_e$. We now define a new cycle $\hat{W}(e)$ associated with the edge $e$ to be the concatenation of the paths $\hat{W}^{\text{left}}, \hat{P}(e), P(e)$, and $\hat{W}^{\text{right}}$ (after deleting the extra copy of the edge $e$, so that we obtain a cycle).

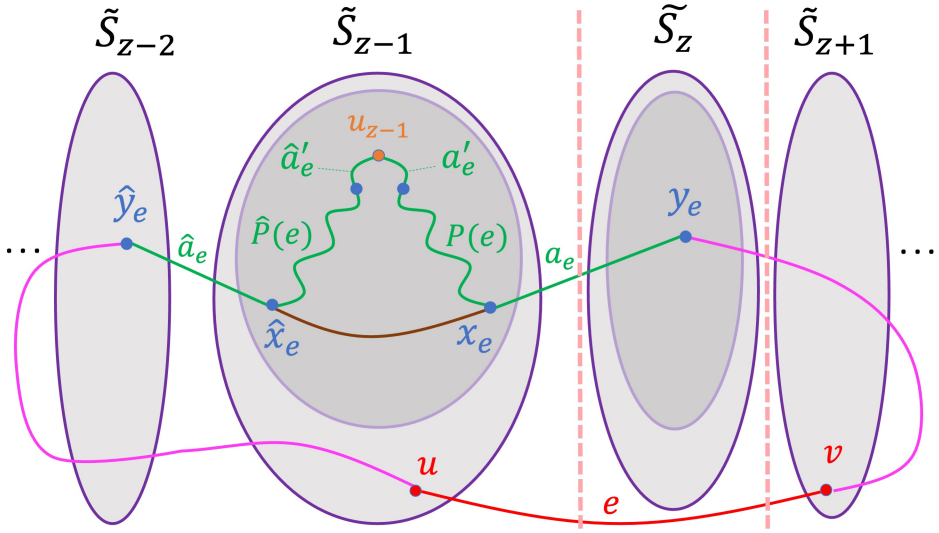



Figure 26: Definition of edges $a_e, \hat{a}_e$ and $a_e^*$ when edge $e \in E_z^{\text{over}}$. Path $W'_e$ is the concatenation of the brown path and edges $a_e, \hat{a}_e$. Paths $\hat{W}^{\text{left}}(e)$ (connecting $e$ to $\hat{y}_e$) and $\hat{W}^{\text{right}}(e)$ (connecting $e$ to $y_e$) are shown in pink; both these paths also contain edge $e = a_e^*$.

Next, we consider an edge $e \in E_z^{\text{right}'}$. Denote $e = (u, v)$, and assume that $u$ is the left endpoint of the edge. Then, from the definition of edge set $E_z^{\text{right}}$, $u \in V(\tilde{S}_z)$ and $v \in \bigcup_{i>z} V(\tilde{S}_i)$ must hold (see Figure 27). We let $a_e^*$ be the first edge on path $\tilde{R}(e)$ that is not contained in $E(\tilde{S}_z)$. Note that $a_e^*$ is also an edge of graph $G$, where it connects a vertex of $V(\tilde{S}_z)$, that we denote by $y_e^*$, to a vertex of $\bigcup_{i<z} V(\tilde{S}_i)$, that we denote by $x_e^*$. It is easy to see that edge $a_e^*$ must lie on the auxiliary cycle $W(e)$, and that it belongs to $\delta_G(U_{z-1})$, and more specifically to $E_z^{\text{left}}$. We will say that edge $e$ *owns* edge $a_e^*$, and that edge $a_e^*$ *belongs* to edge $e$. Note that an edge of $E_z^{\text{left}}$ may belong to a number of edges of $E_z^{\text{right}'}$. Since edge $a_e^*$ lies on the auxiliary cycle $W(e)$, it must be the case that $z - 1 \in \text{span}''(e)$, and so cycle $W(e)$ must contain an edge of $E_{z-1}$, that we denote by $a_e$ (see Figure 27). As before, we denote the endpoints of edge $a_e$ in graph $G$ by $a_e = (x_e, y_e)$, with $x_e \in V(S_{z-1})$ and $y_e \in V(S_z)$. We also denote by $P(e)$ the unique path of the internal $S_{z-1}$-router $\mathcal{Q}(S_{z-1})$ that originates at edge $a_e$. We define the path $\hat{W}^{\text{right}}(e)$ to be the subpath of the auxiliary cycle $W(e)$, between vertices $x_e^*$ and $y_e$, that is disjoint from cluster $S_{z-1}$. Notice that this path contains both edges $a_e^*$ and $e$. Path $\hat{P}(e)$

and edge $\hat{e}_a$ are defined slightly differently. Recall again that $a^*_e \in \delta_G(U_{z-1})$. We consider the unique path $Q(a^*_e)$ of the internal $U_{z-1}$-router $\mathcal{Q}(U_{z-1})$ that originates at edge $a^*_e$. Recall that path $Q(a^*_e)$ terminates at vertex $u_{z-1}$, so it must contain some edge of $\delta_G(S_{z-1})$, and, from the definition of the internal router $\mathcal{Q}(U_{z-1})$, exactly one edge of $\delta_G(S_{z-1})$ lies on path $Q(a^*_e)$. We denote that edge by $\hat{a}_e = (\hat{x}_e, \hat{y}_e)$, where $\hat{x}_e$ is the endpoint of the edge that lies in $S_{z-1}$. We let $\hat{W}^{\text{left}}(e)$ be the subpath of $Q(a^*(e))$ from vertex $y^*_e$ to vertex $\hat{y}_e$. Observe that $a^*_e \in W^{\text{left}}(e)$; and that path $W^{\text{left}}(e)$ is a subpath of both the auxiliary cycle $W(a^*(e))$, and of path $W^{\mathsf{out},\text{left}}(a^*(e))$. We denote by $\hat{P}(e)$ the unique path of the internal $S_{z-1}$-router $\mathcal{Q}(S_{z-1})$ that originates at edge $\hat{a}_e$. Lastly, we define a new cycle $\hat{W}(e)$ associated with edge $e$, to be the union of paths $\hat{W}^{\text{left}}(e), \hat{P}(e), P(e)$, and $\hat{W}^{\text{right}}(e)$, after we delete the extra copy of edge $a^*_e$ (see Figure 27).

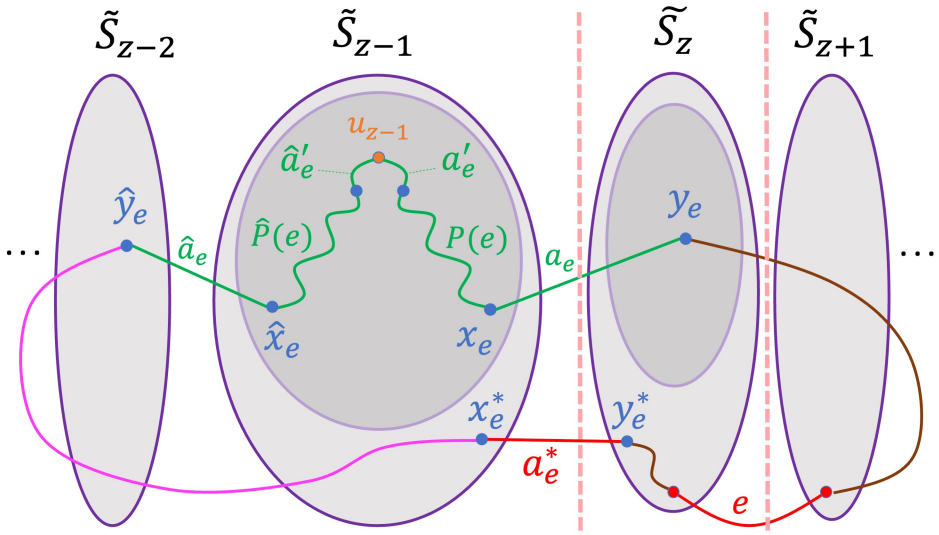



Figure 27: Definitions of edges $a_e, \hat{a}_e$ and $a^*_e$ when edge $e \in E_z^{\text{right}'}$. Path $\hat{W}^{\text{left}}(e)$ is the concatenation of the purple path and edge $a^*_e$. Path $\hat{W}^{\text{right}}(e)$ is the concatenation of edges $a^*_e, e, a_e$ and the two brown paths.

For consistency of notation, for an edge $e = (u, v) \in E_z^{\text{over}}$, where $u$ is the left endpoint of $e$, we will also say that $e$ owns the edge $a^*_e = e$, and that edge $a^*_e$ belongs to $e$. We will also denote $x^*_e = u$ and $y^*_e = v$.

For every edge $e \in E_z^{\text{over}} \cup E_z^{\text{right}'}$, we have now defined two paths $P(e), \hat{P}(e) \in \mathcal{Q}(S_{z-1})$. We denote by $a'_e$ the last edge on path $P(e)$, and by $\hat{a}'_e$ the last edge on path $\hat{P}(e)$; both these edges are incident to $u_{z-1}$ (see Figure 26 and Figure 27). We now provide several observations that will be useful for us later.

**Observation 7.61** *For every pair $(e_1, e_2) \in \Pi$, either edge set $\{a'_{e_1}, \hat{a}'_{e_1}, a'_{e_2}, \hat{a}'_{e_2}\}$ contains fewer than four distinct edges, or edges $\hat{a}'_{e_1}, \hat{a}'_{e_2}, a'_{e_1}, a'_{e_2}$ appear in this order in the rotation $\mathcal{O}_{u_{z-1}} \in \Sigma$.*

**Proof:** Since $(e_1, e_2) \in \Pi$, the paths $\tilde{R}(e_1), \tilde{R}(e_2)$ (that lie in graph $G_z$) have a transversal intersection at vertex $v^*$. Since vertex $v^*$ was obtained by contracting all vertices of $U_{z-1}$ in graph $G$, it is easy to verify that the edges of path $\tilde{R}(e_1)$ that immediately precede and follow vertex $v^*$ on the path are $a^*_{e_1}$ and $a_{e_1}$, respectively (see Figure 23 and Figure 25). Similarly, the edges of path $\tilde{R}(e_2)$ that immediately precede and follow vertex $v^*$ on the path are $a^*_{e_2}$ and $a_{e_2}$, respectively.

Since the paths $\tilde{R}(e_1), \tilde{R}(e_2)$ have a transversal intersection at vertex $v^*$, edges $a^*_{e_1}, a^*_{e_2}, a_{e_1}, a_{e_2}$ appear in this order in the circular ordering $\mathcal{O}_{v^*} \in \Sigma_z$ (up to reversing the ordering). We now recall how the ordering $\mathcal{O}_{v^*} \in \Sigma_z$ was constructed.

Recall that $\delta_{G_z}(v^*) = \delta_G(U_{z-1})$, and in particular, $a^*_{e_1}, a_{e_1}, a^*_{e_2}, a_{e_2} \in \delta_G(U_{z-1})$. The ordering $\mathcal{O}_{v^*} \in \Sigma_z$ was defined to be identical to the ordering $\tilde{\mathcal{O}}_{z-1}$ of the edges of $\delta_G(U_{z-1})$, which, in turn, is the ordering guided by the set $\mathcal{Q}(U_{z-1})$ of paths.

From our construction, it is immediate to verify that the last edge of the unique path in $\mathcal{Q}(U_{z-1})$ that originates at edge $a^*_{e_1}$ is $\hat{a}'_{e_1}$, and the last edge of the unique path in $\mathcal{Q}(U_{z-1})$ that originates at edge $a_{e_1}$ is $a'_{e_1}$ (see Figure 26 and Figure 27). Similarly, the last edge of the unique path in $\mathcal{Q}(U_{z-1})$ that originates at edge $a^*_{e_2}$ is $\hat{a}'_{e_2}$, and the last edge of the unique path in $\mathcal{Q}(U_{z-1})$ that originates at edge $a_{e_2}$ is $a'_{e_2}$. From the definition of the ordering $\tilde{\mathcal{O}}_{z-1}$, it must be the case that either set $\{a_{e_1}, \hat{a}_{e_1}, a_{e_2}, \hat{a}_{e_2}\}$ contains fewer than four distinct edges, or edges $\hat{a}'_{e_1}, \hat{a}'_{e_2}, a'_{e_1}, a'_{e_2}$ appear in this order in the rotation $\mathcal{O}_{u_{z-1}} \in \Sigma$. □

We will use the following simple observation in order to bound the congestion that is caused by the set $\hat{\mathcal{W}} = \left\{\hat{W}(e) \mid e \in E^{\text{over}}_z \cup E^{\text{right}'}_z\right\}$ of cycles.

**Observation 7.62** *Each edge $e \in E^{\text{left}}_z$ may belong to at most $O(\log^{34} m)$ edges of $E^{\text{right}'}_z$. Additionally, an edge $e \in E(G) \setminus E(S_{z-1})$ may lie on at most $O(\log^{68} m)$ cycles of $\hat{\mathcal{W}}$, while an edge $e \in E(S_{z-1})$ may lie on at most $O(\log^{68} m) \cdot \text{cong}_G(\mathcal{Q}(S_{z-1}), e)$ cycles of $\hat{\mathcal{W}}$. Lastly, an edge $e \in E(G) \setminus \left(E(\tilde{S}_z) \cup E(S_{z-1})\right)$ may lie on at most $O(\log^{34} m) \cdot N'_z(e)$ cycles of $\left\{\hat{W}(e') \mid e' \in E^{\text{right}'}_z\right\}$.*

**Proof:** In order to prove the first assertion, consider any edge $e \in E^{\text{left}}_z$. From our construction, $e$ may belong to an edge $e' \in E^{\text{right}'}_z$ only if $e \in W(e')$. From Observation 7.47, edge $e$ may lie on at most $O(\log^{34} m)$ cycles of $\mathcal{W}$, and so $e$ may belong to at most $O(\log^{34} m)$ edges of $E^{\text{right}'}_z$.

Consider now some edge $e \in E(G) \setminus E(S_{z-1})$. Notice that, if $e$ lies on a cycle $\hat{W}(e')$ for some edge $e' \in E^{\text{right}'}_z \cup E^{\text{over}}_z$, then either $e \in W(e')$, or $e \in W(a^*_{e'})$ must hold. Since, from Observation 7.47, edge $e$ may lie on at most $O(\log^{34} m)$ cycles of $\mathcal{W}$, and, as we have shown, every edge $e'' \in E^{\text{left}}_z$ may belong to at most $O(\log^{34} m)$ edges of $E^{\text{right}}_z$, we get that $e$ may lie on at most $O(\log^{68} m)$ cycles of $\hat{\mathcal{W}}$.

Consider now an edge $e \in E(S_{z-1})$. Notice that, if $e$ lies on a cycle $\hat{W}(e')$ for some edge $e' \in E^{\text{right}'}_z \cup E^{\text{over}}_z$, then either $e \in P(e')$, or $e \in \hat{P}(e')$ must hold. Consider some path $P \in \mathcal{Q}(S_{z-1})$ that contains the edge $e$, and let $a$ be the first edge on path $P$. Consider any edge $e' \in E^{\text{right}'}_z \cup E^{\text{over}}_z$, for which $P = P(e')$ or $P = \hat{P}(e')$ holds. Then $a = a_{e'}$ or $a = \hat{a}_{e'}$ must hold, and in particular, edge $a$ must lie on $\hat{W}(e')$. As we have shown, every edge $e \in \delta_G(S_{z-1})$ may lie on at most $O(\log^{68} m)$ cycles of $\hat{\mathcal{W}}$. Therefore, there are at most $O(\log^{68} m)$ edges $e' \in E^{\text{right}'}_z \cup E^{\text{over}}_z$, for which $P = P(e')$ or $P = \hat{P}(e')$ holds. We conclude that $e$ may lie on at most $O(\log^{68} m) \cdot \text{cong}_G(\mathcal{Q}(S_{z-1}), e)$ cycles of $\hat{W}$.

It now remains to prove the last assertion. Consider an edge $e \in E(G) \setminus \left(E(\tilde{S}_z) \cup E(S_{z-1})\right)$. Assume that $e \in \hat{W}(e')$ for some edge $e' \in E^{\text{right}'}_z$. From our construction, this may only happen if either $e$ lies on the unique path of $\mathcal{Q}(U_z)$ that originates at $e'$; or $e$ lies on the unique path of $\mathcal{Q}(U_{z-1})$ that originates at $a^*_{e'}$. Recall that, in the latter case, $a^*_{e'} \in E^{\text{left}}_z$ must hold. Recall that $N'_z(e)$ is the total number of paths in $\mathcal{Q}(U_{z-1}) \cup \mathcal{Q}'(U_z)$ that originate at edges of $E_{z-1} \cup E^{\text{left}}_z \cup E^{\text{right}}_z \cup E_z$ and contain $e$. Since each edge $a^* \in E^{\text{left}}_z$ may belong to at most $O(\log^{34} m)$ edges of $E^{\text{right}'}_z$, we get that, overall, edge $e$ may lie on at most $O(\log^{34} m) \cdot N'_z(e)$ cycles of $\left\{\hat{W}(e') \mid e' \in E^{\text{right}'}_z\right\}$. □

Recall that we have denoted by $\Pi^1 \subseteq \Pi$ the set of all edge pairs $(e_1, e_2) \in \Pi$, where at least one of the two edges lies in $E^{\text{right}'}_z$. We will always assume w.l.o.g. that $e_1 \in E^{\text{right}'}_z$ for each such pair. We bound the expected cardinalities of sets $\Pi^1$ and $\Pi^2$ separately in the following two claims.

**Claim 7.63** *The expected cardinality of set $\Pi^1$ is at most:*

$$\hat{\eta}^2\cdot\left(\sum_{e\in E(G)}\mathbf{E}\left[N'_z(e)\right]+\sum_{(e,e')\in\chi^*}\left(\mathbf{E}\left[N'_z(e)\right]+\mathbf{E}\left[N'_z(e')\right]\right)+|E(S_{z-1}|+|E(\tilde{S}_z)|+|\chi^*_{z-1}|+|\chi^*_z|\right)+|\Pi^T_z|.$$

**Proof:** We denote by $\Pi^1_1\subseteq\Pi^1$ the set of all pairs $(e_1,e_2)\in\Pi^1$, for which cycles $\hat{W}(e_1),\hat{W}(e_2)$ share at least one edge. We let $e$ be any edge in $E(\hat{W}(e_1))\cap E(\hat{W}(e_2))$, and we say that $e$ is responsible for the pair $(e_1,e_2)$. Consider now any pair of edges $(e_1,e_2)\in\Pi^1\setminus\Pi^1_1$, and their two corresponding cycles $\hat{W}(e_1),\hat{W}(e_2)$. Note that the two cycles do not share edges, and, from Observation 7.61, they have a transversal intersection at vertex $u_{z-1}$. Therefore, either there is a pair of edges $e'_1\in E(\hat{W}(e_1))$ and $e'_2\in E(\hat{W}(e_2))$ that cross in the drawing $\varphi^*$ of $G$; or there is a vertex $v\neq u_{z-1}$, such that $\hat{W}(e_1),\hat{W}(e_2)$ have a transversal intersection at $v$. In the former case, we say that crossing $(e'_1,e'_2)$ is responsible for the edge pair $(e_1,e_2)$. In the latter case, we say that the transversal intersection of $\hat{W}(e_1),\hat{W}(e_2)$ at $v$ is responsible for the pair $(e_1,e_2)$. We denote by $\Pi^1_2\subseteq\Pi^1\setminus\Pi^1_1$ the set of all pairs $(e_1,e_2)$, such that some crossing $(e'_1,e'_2)$ is responsible for $(e_1,e_2)$, and we denote by $\Pi^1_3=\Pi^1\setminus(\Pi^1_1\cup\Pi^1_2)$. We now bound the number of pairs in each one of the three sets one by one in the following three observations.

**Observation 7.64** $\mathbf{E}\left[|\Pi^1_1|\right]\leq\hat{\eta}^2\cdot\left(\sum_{e\in E(G)}\mathbf{E}\left[N'_z(e)\right]+|E(S_{z-1}|+|E(\tilde{S}_z)|\right)$.

**Proof:** Consider an edge $e\in E(G)$. We will bound the expected number of pairs $(e_1,e_2)\in\Pi^1_1$ of edges, for which edge $e$ is responsible. For each such pair, we assume w.l.o.g. that $e_1\in E_z^{\text{right}'}$. We distinguish between three cases.

The first case is when $e\in E(G)\setminus\left(E(\tilde{S}_z)\cup E(S_{z-1})\right)$. From Observation 7.62, $e$ may lie on at most $O(\log^{34}m)\cdot N'_z(e)$ cycles of $\left\{\hat{W}(e')\mid e'\in E_z^{\text{right}'}\right\}$, and on at most $O(\log^{68}m)$ cycles of $\hat{\mathcal{W}}$. Therefore, such an edge may be responsible for at most $O(\log^{102}m)\cdot N'_z(e)\leq\hat{\eta}\cdot N'_z(e)$ edge pairs in $\Pi^1_1$.

The second case is when $e\in E(\tilde{S}_z)$. In this case, from Observation 7.62, edge $e$ lies on at most $O(\log^{68}m)$ cycles of $\hat{\mathcal{W}}$. Therefore, such an edge may be responsible for at most $O(\log^{136}m)\leq\hat{\eta}$ edge pairs in $\Pi^1_1$, and overall, the edges of $\tilde{S}_z$ may be responsible for at most $\hat{\eta}\cdot|E_G(\tilde{S}_z)|$ edge pairs in $\Pi^1_1$.

The third and the last case is when $e\in E(S_{z-1})$. From Observation 7.62, such an edge may lie on at most $O(\log^{68}m)\cdot\text{cong}_G(\mathcal{Q}(S_{z-1}),e)$ cycles of $\hat{W}$, and so it may be responsible for at most $O(\log^{138}m)\cdot(\text{cong}_G(\mathcal{Q}(S_{z-1}),e))^2\leq\hat{\eta}\cdot(\text{cong}_G(\mathcal{Q}(S_{z-1}),e))^2$ edge pairs in $\Pi^1_1$. Since we have assumed that $S_{z-1}\in\mathcal{S}^{\text{light}}$, from Observation 7.49, $\mathbf{E}\left[(\text{cong}_G(\mathcal{Q}(S_{z-1}),e))^2\right]\leq\hat{\eta}$. Therefore, the expected number of edge pairs in $\Pi^1_1$ for which edge $e$ is responsible is at most $\hat{\eta}^2$, and the total expected number of edge pairs in $\Pi^1_1$ for which edges of $E(S_{z-1})$ are responsible is at most $\hat{\eta}^2\cdot|E(S_{z-1})|$. The bound now follows. □

**Observation 7.65** $\mathbf{E}\left[|\Pi^1_2|\right]\leq\hat{\eta}^2\cdot\left(\sum_{(e,e')\in\chi^*}\left(\mathbf{E}\left[N'_z(e)\right]+\mathbf{E}\left[N'_z(e')\right]\right)+|\chi^*_{z-1}|+|\chi^*_z|\right)$.

**Proof:** Consider a crossing $(e,e')\in\varphi^*$. We bound the number of pairs $(e_1,e_2)\in\Pi^1_2$ with $e_1\in E_z^{\text{right}'}$, for which the crossing $(e,e')$ is responsible. Recall that, if crossing $(e,e')$ is responsible for a pair $(e_1,e_2)\in\Pi^1_2$, then $e\in\hat{W}(e_1)$ and $e'\in\hat{W}(e_2)$ must hold.

We first consider the case where neither of the edges $e,e'$ lie in $E(\tilde{S}_z)\cup E(S_{z-1})$. In this case, from Observation 7.62, edge $e'$ may lie on at most $O(\log^{68}m)$ cycles of $\hat{\mathcal{W}}$, while edge $e$ may lie on at most $O(\log^{34}m)\cdot N'_z(e)$ cycles of $\left\{\hat{W}(e')\mid e'\in E_z^{\text{right}'}\right\}$. Therefore, crossing $(e,e')$ may be responsible for at most $O(\log^{102}m)\cdot N'_z(e)\leq\hat{\eta}\cdot N'_z(e)$ edge pairs in $\Pi^1_2$. Note that, if either of the edges $e,e'$

lies in $E(\tilde{S}_z) \cup E(S_{z-1})$, then crossing $(e, e')$ belongs to $\chi^*_{z-1} \cup \chi^*_z$. We conclude overall, all crossings $(e, e') \in \chi^* \setminus (\chi^*_{z-1} \cup \chi^*_z)$ may be responsible for at most $\sum_{(e,e') \in \chi^*} \hat{\eta} \cdot (N'_z(e) + N'_z(e'))$ pairs in $\Pi^1_2$.

Next, we consider the case where at least one of the edges $e, e'$ lies in $E(\tilde{S}_z) \cup E(S_{z-1})$, so crossing $(e, e')$ belongs to $\chi^*_{z-1} \cup \chi^*_z$. We let $\hat{N}(e)$ be the random variable indicating the number of cycles in $\hat{\mathcal{W}}$ containing edge $e$, and we define random variable $\hat{N}(e')$ for edge $e'$ similarly. Notice that random variables $\hat{N}(e), \hat{N}(e')$ may not be independent, if $e, e' \in E(S_{z-1})$. The total number of edge pairs in $\Pi^1_2$ for which crossing $(e, e')$ is responsible is bounded by $\hat{N}(e) \cdot \hat{N}(e') \leq (\hat{N}(e))^2 + (\hat{N}(e'))^2$. From Observation 7.62, combined with Observation 7.49 and our assumption that $S_{z-1} \in \mathcal{S}^{\text{light}}$, we get that $\mathbf{E}\left[(\hat{N}(e))^2\right], \mathbf{E}\left[(\hat{N}(e'))^2)\right] \leq \hat{\eta}^2$. We conclude that the expected number of pairs in $\Pi^1_2$ for which all crossings $(e, e') \in \chi^*_{z-1} \cup \chi^*_z$ are responsible is at most $\hat{\eta}^2 \cdot (|\chi^*_{z-1}| + |\chi^*_z|)$. □

**Observation 7.66** $|\Pi^1_3| \leq |\Pi^T_z|$.

**Proof:** Consider an edge pair $(e, e') \in \Pi^1_3$. Recall that there must be a vertex $v \neq u_{z-1}$, such that cycles $\hat{W}(e), \hat{W}(e')$ have a transversal intersection at $v$. Recall also that $e \in E^{\text{right}'}_z \subseteq E^{\text{right}}_z$. We claim that $v \in V(\overline{U}_z)$ must hold, and moreover, auxiliary cycles $W(e), W(e')$ must have a transversal intersection at vertex $v$.

Indeed, assume first that $v \in V(S_{z-1}) \setminus \{u_{z-1}\}$. In this case, vertex $v$ must lie on $P(e) \cup \hat{P}(e)$, and on $P(e') \cup \hat{P}(e')$. Since paths $P(e), \hat{P}(e), P(e'), \hat{P}(e')$ all belong to the internal $S_{z-1}$-router $\mathcal{Q}(S_{z-1})$, they cannot have a transversal intersection at any vertex.

Assume now that $v \in V(U_{z-1}) \setminus V(S_{z-1})$. In this case, by our construction, $v$ lies on the auxiliary cycle $W(a^*_e)$ of the edge $a^*_e \in E^{\text{left}}_z$ that belongs to $e$, and similarly, $v$ lies on the auxiliary cycle $W(a^*_{e'})$. Moreover, cycles $W(a^*_e), W(a^*_{e'})$ must have a transversal intersection at vertex $v$. From Observation 7.48, this is only possible if $v \in S_j$ for some index $1 < j < r$, and either $j-1$ is the last index in both $\text{span}''(a^*_e), \text{span}''(a^*_{e'})$, or $j-1$ is the last index in one of these sets, while $j$ belongs to another. This is impossible, since $v \in V(U_{z-1})$, and $a^*_e, a^*_{e'} \in \delta_G(U_{z-1})$.

We conclude that vertex $v$ may not lie in $U_{z-1}$. But then, from the construction of the cycles $\hat{W}(e), \hat{W}(e')$, $v$ must lie on both the auxiliary cycles $W(e), W(e')$, and the two cycles must have a transversal intersection at $v$. From Observation 7.48, this is only possible if $v \in S_j$ for some index $1 < j < r$, and either $j-1$ is the last index in both $\text{span}''(e), \text{span}''(e')$, or $j-1$ is the last index in one of these sets, while $j$ belongs to another. Since $e, e' \in \delta_G(U_z)$, we conclude that $v \in V(\overline{U}_z)$ must hold, and, from the above discusison, $W(e), W(e')$ must have a transversal intersection at vertex $v$. Recall that $\Pi^T_z$ is the set of triples $(\tilde{e}, \tilde{e}', \tilde{v})$, where $\tilde{e} \in E^{\text{right}}_z$, $\tilde{e}' \in \hat{E}_z$, and cycles $W(\tilde{e})$ and $W(\tilde{e}')$ have a transversal intersection at $\tilde{v}$. We conclude that, if the transversal crossing of cycles $\hat{W}(e), \hat{W}(e')$ at vertex $v$ is responsible for edge pair $(e, e')$, then triple $(e, e', v)$ must lie in $\Pi^T_z$, and so $|\Pi^1_3| \leq |\Pi^T_z|$. □

Combining the bounds from Observations 7.64–7.66, we get that the expected cardinality of set $\Pi^1$ is at most:

$$\hat{\eta}^2 \cdot \left( \sum_{e \in E(G)} \mathbf{E}\left[N'_z(e)\right] + \sum_{(e,e') \in \chi^*} \left(\mathbf{E}\left[N'_z(e)\right] + \mathbf{E}\left[N'_z(e')\right]\right) + |E(S_{z-1}| + |E(\tilde{S}_z)| + |\chi^*_{z-1}| + |\chi^*_z| \right) + |\Pi^T_z|.$$

□

We use the following claim to bound the expected cardinality of $\Pi^2$.

**Claim 7.67**

$$\mathbf{E}\left[|\Pi^2|\right] \leq \hat{\eta}^2 \cdot \left(|E(S_{z-1})| + |\delta_G(S_{z-1})| + |\chi^*_{z-1}|\right).$$

**Proof:** Recall that set $\Pi^2$ contains all edge pairs $(e, e') \in \Pi$, with $e, e' \in E_z^{\text{over}}$. Consider any edge $e \in E_z^{\text{over}}$. Recall that we denoted by $W'(e)$ the subpath of the auxiliary cycle $W(e)$ between vertices $y_e$ and $\hat{y}_e$ that intersects cluster $S_{z-1}$. We denote by $W''(e)$ the subpath of $W(e)$ between vertices $y_e$ and $\hat{y}_e$ that does not share edges with $W'(e)$. Equivalently, $W''(e)$ is the concatenation of the paths $\hat{W}^{\text{left}}(e)$ and $\hat{W}^{\text{right}}(e)$ (after the extra copy of edge $e$ is deleted). We denote by $H(e)$ the graph obtained from the union of the paths $W'(e), P(e)$ and $\hat{P}(e)$. The following observation, whose proof is deferred to Appendix G.25 is central to the proof of Claim 7.67.

**Observation 7.68** *Let $(e_1, e_2)$ be a pair of edges in $\Pi^2$. Then one of the following must happen: either (i) some edge lies in both $H(e_1)$ and $H(e_2)$; or (ii) there is a pair of edges $\tilde{e}_1 \in H(e_1)$, $\tilde{e}_2 \in H(e_2) \cup W''(e_2)$, whose images cross in the drawing $\varphi^*$ of graph $G$; or (iii) there is a pair of edges $\tilde{e}'_1 \in H(e_1) \cup W''(e_1)$, $\tilde{e}'_2 \in H(e_2)$, whose images cross in the drawing $\varphi^*$ of graph $G$.*

As before, we partition the set $\Pi^2$ of edge pairs into two subsets. The first set, $\Pi_1^2$, containing all pairs $(e_1, e_2) \in \Pi^2$, such that there is an edge $e \in E(H_1) \cap E(H_2)$. In this case, we say that edge $e$ is responsible for the pair $(e_1, e_2)$. Set $\Pi_2^2$ contains all remaining edge pairs $(e_1, e_2) \in \Pi^2$. From Observation 7.62, for each such pair $(e_1, e_2) \in \Pi_2^2$, there must be a crossing in $\chi^*$ between a pair of edges $\tilde{e}_1$ and $\tilde{e}_2$, such that either (i) $\tilde{e}_1 \in H(e_1)$ and $\tilde{e}_2 \in H(e_2) \cup W''(e_2)$; or (ii) $\tilde{e}'_1 \in H(e_1) \cup W''(e_1)$ and $\tilde{e}'_2 \in H(e_2)$. For convenience, we will always assume that it is the former. Since $H(e_1) \subseteq S_{z-1} \cup \delta_G(S_{z-1})$, crossing $(e_1, e_2)$ must lie in $\chi^*_{z-1}$, and we say that this crossing is responsible for the pair $(e_1, e_2) \in \Pi$. We bound the expected cardinalities of the sets $\Pi_1^2, \Pi_2^2$ separately, as before.

In order to bound $\mathbf{E}\left[|\Pi_1^2|\right]$, consider some edge $e \in E(S_{z-1}) \cup \delta_G(S_{z-1})$. Note that edge $e$ may only lie in graph $H(e')$, for an edge $e' \in E_z^{\text{over}}$, if $e \in W(e')$, or $e \in \hat{W}(e')$. From Observation 7.47, edge $e$ may appear on at most $O(\log^{34} m)$ auxiliary cycles of $\mathcal{W}$, and, from Observation 7.62, $e$ may lie on at most $O(\log^{68} m) \cdot \text{cong}_G(\mathcal{Q}(S_{z-1}), e)$ cycles of $\hat{W}$. From Observation 7.49, since we have assumed that $S_{z-1} \in \mathcal{S}^{\text{light}}$, we get that $\mathbf{E}\left[\left(\text{cong}_G(\mathcal{Q}(S_{z-1}), e)\right)^2\right] \leq \hat{\eta}$. Therefore, the expected number of pairs in $\Pi_1^2$, for which edge $e$ is responsible is at most:

$$O(\log^{136} m) \cdot \mathbf{E}\left[\left(\text{cong}_G(\mathcal{Q}(S_{z-1}, e))\right)^2\right] \leq \hat{\eta}^2.$$

We conclude that $\mathbf{E}\left[|\Pi_1^2|\right] \leq \hat{\eta}^2 \cdot (|E(S_{z-1})| + |\delta_G(S_{z-1})|)$.

In order to bound the expected cardinality of the set $\Pi_2^2$, consider some edge pair $(e_1, e_2) \in \Pi_2^2$, and the crossing $(\tilde{e}_1, \tilde{e}_2)$ that is responsible for it, where $\tilde{e}_1 \in H(e_1)$ and $\tilde{e}_2 \in H(e_2) \cup W''(e_2)$. Recall that $H(e_1) \subseteq E(S_{z-1}) \cup \delta_G(S_{z-1})$, and that crossing $(\tilde{e}_1, \tilde{e}_2)$ must lie in $\chi^*_{z-1}$. Consider now any crossing $(e, e') \in \chi^*_{z-1}$, and assume w.l.o.g. that $e \in E(S_{z-1}) \cup \delta_G(S_{z-1})$. If $e' \in E(S_{z-1}) \cup \delta_G(S_{z-1})$ as well, then for every pair $(e_1, e_2) \in \Pi_2^2$ for which crossing $(e, e')$ is responsible, $e \in H(e_1)$ and $e' \in H(e_2)$ must hold. As observed above, the total number of edges $e_1 \in \delta_G(U_{z-1})$ with $e \in H(e_1)$ is $O(\log^{68} m) \cdot \text{cong}_G(\mathcal{Q}(S_{z-1}), e)$, and similarly, the total number of edges $e_2 \in \delta_G(U_{z-1})$ with $e' \in H(e_2)$ is at most $O(\log^{68} m) \cdot \text{cong}_G(\mathcal{Q}(S_{z-1}), e')$. Therefore, the total number of edge pairs $(e_1, e_2) \in \Pi_2^2$ for which crossing $(e, e')$ is responsible is bounded by:

$$\begin{aligned} &O(\log^{136} m) \cdot \text{cong}_G(\mathcal{Q}(S_{z-1}), e) \cdot \text{cong}_G(\mathcal{Q}(S_{z-1}), e') \\ &\qquad \leq O(\log^{136} m) \cdot \left((\text{cong}_G(\mathcal{Q}(S_{z-1}), e))^2 + (\text{cong}_G(\mathcal{Q}(S_{z-1}), e'))^2\right). \end{aligned}$$

As before, from Observation 7.49 and the assumption that $S_{z-1} \in \mathcal{S}^{\text{light}}$:

$$\mathbf{E}\left[\left(\text{cong}_G(\mathcal{Q}(S_{z-1}), e)\right)^2\right], \mathbf{E}\left[\left(\text{cong}_G(\mathcal{Q}(S_{z-1}), e')\right)^2\right] \leq \hat{\eta}.$$

Therefore, the expected number of edge pairs $(e_1,e_2)\in\Pi_2^2$ for which crossing $(e,e')$ is responsible is at most $\hat{\eta}^2$.

Lastly, we consider a crossing $(e,e')\in\chi^*_{z-1}$ with $e\in E(S_{z-1})\cup\delta_G(S_{z-1})$ and $e'\notin E(S_{z-1})\cup\delta_G(S_{z-1})$. In this case, for every pair $(e_1,e_2)\in\Pi_2^2$ for which crossing $(e,e')$ is responsible, $e\in H(e_1)$, and $e'\in W''(e_2)\subseteq W(e_2)$ must hold. As before, the total number of edges $e_1\in\delta_G(U_{z-1})$ with $e\in H(e_1)$ is at most $O(\log^{68}m)\cdot\mathrm{cong}_G(\mathcal{Q}(S_{z-1}),e)$, and, from Observation 7.47, edge $e'$ appears on at most $O(\log^{34}m)$ cycles of $\mathcal{W}$. Therefore, the total expected number of edge pairs $(e_1,e_2)\in\Pi_2^2$ for which crossing $(e,e')$ is responsible is bounded by:

$$O(\log^{102}m)\cdot\mathbf{E}\left[\mathrm{cong}_G(\mathcal{Q}(S_{z-1}),e)\right]\leq\hat{\eta}^2,$$

from Observation 7.49. Overall, we get that $\mathbf{E}\left[|\Pi_2^2|\right]\leq\eta^2\cdot|\chi^*_{z-1}|$, and:

$$\mathbf{E}\left[|\Pi^2|\right]\leq\hat{\eta}^2\cdot\left(|E(S_{z-1})|+|\delta_G(S_{z-1})|+|\chi^*_{z-1}|\right).$$

□

Combining the bounds from Claims 7.63 and 7.67, we get that:

$$\begin{aligned}\mathbf{E}\left[|\Pi|\right]\leq\hat{\eta}^2\cdot&\left(\sum_{e\in E(G)}\mathbf{E}\left[N'_z(e)\right]+\sum_{(e,e')\in\chi^*}\left(\mathbf{E}\left[N'_z(e)\right]+\mathbf{E}\left[N'_z(e')\right]\right)\right)\\&+\hat{\eta}^2\cdot\left(|E(S_{z-1})|+|E(\tilde{S}_z)|+|\delta_G(S_{z-1})|+|\chi^*_{z-1}|+|\chi^*_z|\right)+|\Pi^T_z|,\end{aligned}$$

completing the proof of Claim 7.60.

# 8 Proof of Theorem 3.12

We assume that we are given a wide instance $I=(G,\Sigma)$ of MCNwRS, with $m=|E(G)|$, such that $\mu^{20}\leq m\leq m^*$. The high-level idea of the proof is to compute a collection $\mathcal{C}$ of disjoint clusters in graph $G$ that have the $\alpha_0$-bandwidth property for $\alpha_0=1/\log^3m$ with $|E(G_{|\mathcal{C}})|$ sufficiently small, and then to apply the algorithm for computing advanced disengagement from Theorem 7.1 to $\mathcal{C}$, to obtain a $\nu$-decomposition of $I$ into subinstances. In order to ensure that all subinstances have the required properties, we need to ensure that, if $C$ is a cluster of $\mathcal{C}$ with $|E(C)|>\frac{m}{\mu}$, then for any pair $u,v$ of disctinct vertices of $C$ whose degree in $C$ is at least $\frac{m}{\mu^6}$, there are at least $\frac{8m}{\mu^{50}}$ edge-disjoint paths connecting $u$ to $v$ in $C$. We start with the following lemma that allows us to compute the desired collection $\mathcal{C}$ of clusters.

**Lemma 8.1** *There is an efficient algorithm, that, given a wide instance $I=(G,\Sigma)$ of* MCNwRS*, with $m=|E(G)|$, such that $\mu^{20}\leq m\leq m^*$, computes a collection $\mathcal{C}$ of disjoint clusters of $G$, that have the following properties:*

- $\bigcup_{C\in\mathcal{C}}V(C)=V(G)$;
- *every cluster $C\in\mathcal{C}$ has the $\alpha_0$-bandwidth property, for $\alpha_0=\frac{1}{\log^3m}$;*
- *for every cluster $C\in\mathcal{C}$, for every pair $u,v$ of distinct vertices of $C$ with $\deg_G(v),\deg_G(u)\geq\frac{m}{\mu^6}$, there is a collection of at least $\frac{8m}{\mu^{50}}$ edge-disjoint paths in $C$ connecting $u$ to $v$; and*

- $|E^{\mathsf{out}}(\mathcal{C})| \leq \frac{m}{\mu^{27}}$.

**Proof:** The proof of the lemma uses somewhat standard techniques and is similar, for example, to the proof of Theorem 4.19. We denote by $U$ the set of all vertices of $G$ whose degree is at least $\frac{m}{\mu^6}$. Clearly, $|U| \leq 2\mu^6$. Our algorithm maintains a collection $\mathcal{R}$ of clusters of $G$. Throughout the algorithm, we ensure that the following invariants hold:

I1. all clusters in $\mathcal{R}$ are mutually disjoint; and

I2. $\bigcup_{R\in\mathcal{R}} V(R) = V(G)$.

For a given collection $\mathcal{R}$ of clusters with the above properties, we define a *budget* $b(e)$ for every edge $e \in E(G)$, as follows. If both endpoints of $e$ lie in the same cluster of $\mathcal{C}$, then we set the budget $b(e) = 0$. Assume now that the endpoints of $e$ lie in different clusters $R, R' \in \mathcal{R}$. We define $b_R(e) = 1 + 8\alpha_0 \cdot \beta_{\text{ARV}}(m) \cdot \log_{3/2}(|\delta_G(R)|)$, $b_{R'}(e) = 1 + 8\alpha_0 \cdot \beta_{\text{ARV}}(m) \cdot \log_{3/2}(|\delta_G(R')|)$, and $b(e) = b_R(e) + b_{R'}(e)$. Notice that $b(e) \leq 3$ always holds.

For every cluster $R \in \mathcal{R}$, we denote $U_R = U \cap V(R)$. For every vertex $u \in U$, we define a budget $b(u)$ of $u$ as follows. Assume that $u \in V(R)$ for some cluster $R \in \mathcal{R}$. Then $b(u) = 4|U_R| \cdot \frac{m}{\mu^{40}}$. We denote by $B = \sum_{e\in E(G)} b(e) + \sum_{u\in U} b(u)$ the *total budget in the system*. Clearly, throughout the algorithm, $B \geq \sum_{R\in\mathcal{R}} |\delta_G(R)|$ holds.

At the beginning of the algorithm, we set $\mathcal{R} = \{G\}$. Clearly, both invariants hold for $\mathcal{R}$. Moreover, the budget of every vertex $u \in U$ is at most $4|U| \cdot \frac{m}{\mu^{40}}$, so the total budget of all vertices in $U$ is at most $4|U|^2 \cdot \frac{m}{\mu^{40}} \leq \frac{16m}{\mu^{28}} < \frac{m}{\mu^{27}}$ (since $|U| \leq 2\mu^6$), while the budget of every edge of $G$ is 0. Therefore, at the beginning of the algorithm, $B \leq \frac{m}{\mu^{27}}$ holds. We will ensure that, throughout the algorithm, the total budget $B$ never increases. Since $B \geq \sum_{R\in\mathcal{R}} |\delta_G(R)|$ always holds, this will ensure that, at the end of the algorithm, $\sum_{R\in\mathcal{R}} |\delta_G(R)| \leq \frac{m}{\mu^{27}}$ will hold.

Throughout the algorithm, we maintain a partition of the set $\mathcal{R}$ of clusters into two subsets: set $\mathcal{R}^A$ of *active* clusters, and set $\mathcal{R}^I$ of *inactive* clusters. We will ensure that the following additional invariant holds:

I3. every cluster $R \in \mathcal{R}^I$ has the $\alpha_0$-bandwidth property; and

I4. for every cluster $R \in \mathcal{R}^I$, for every pair $u, v$ of distinct vertices of $U_R$, there is a collection of at least $\frac{8m}{\mu^{50}}$ edge-disjoint paths in $R$ connecting $u$ to $v$.

At the beginning of the algorithm, we set $\mathcal{R}^A = \mathcal{R} = \{G\}$ and $\mathcal{R}^I = \emptyset$. Clearly, all invariants hold then. We then proceed in iterations, as long as $\mathcal{R}^A \neq \emptyset$.

In order to execute an iteration, we select an arbitrary cluster $R \in \mathcal{R}^A$ to process. We will either establish that $R$ has the $\alpha_0$-bandwidth property in graph $G$, and that for every pair $u, v \in U_R$ of distinct vertices there is a collection of at least $\frac{8m}{\mu^{50}}$ edge-disjoint paths in $R$ connecting $u$ to $v$ (in which case $R$ is moved from $\mathcal{R}^A$ to $\mathcal{R}^I$); or we will modify the set $\mathcal{R}$ of clusters in a way that ensures that the total budget decreases by at least $1/m$. An iteration that processes a cluster $R \in \mathcal{R}^A$ consists of two steps. The purpose of the first step is to either establish the $\alpha_0$-bandwidth property of cluster $R$, or to replace it with a collection of smaller clusters in $\mathcal{R}^A$. The purpose of the second step is to either establish that, for every pair $u, v \in U_R$ of distinct vertices there is a collection of at least $\frac{8m}{\mu^{50}}$ edge-disjoint paths in $R$ connecting $u$ to $v$, or to modify the set $\mathcal{R}$ of clusters in a way that decreases the total budget by at least $1/m$. We now describe each of the two steps in turn.

**Step 1: Ensuring the Bandwidth Property.** Let $R^+$ be the augmentation of the cluster $R$ in graph $G$. Recall that $R^+$ is a graph that is obtained from $G$ through the following process. First, we subdivide every edge $e \in \delta_G(R)$ with a vertex $t_e$, and we let $T = \{t_e \mid e \in \delta_G(R)\}$ be the resulting set of vertices. We then let $R^+$ be the subgraph of the resulting graph induced by vertex set $V(R) \cup T$. We apply Algorithm $\mathcal{A}_{\text{ARV}}$ for computing approximate sparsest cut to graph $R^+$, with the set $T$ of vertices, to obtain a $\beta_{\text{ARV}}(m)$-approximate sparsest cut $(X, Y)$ in graph $R^+$ with respect to vertex set $T$. We now consider two cases. The first case happens if $|E_R(X, Y)| \geq \alpha_0 \cdot \beta_{\text{ARV}}(m) \cdot \min\{|X \cap T|, |Y \cap T|\}$. In this case, we are guaranteed that the minimum sparsity of any $T$-cut in graph $R^+$ is at least $\alpha_0$, or equivalently, set $T$ of vertices is $\alpha_0$-well-linked in $R^+$. From Observation 4.16, cluster $R$ has the $\alpha_0$-bandwidth property in graph $G$. In this case, we proceed to the second step of the algorithm.

Assume now that $|E_R(X, Y)| < \alpha_0 \cdot \beta_{\text{ARV}}(m) \cdot \min\{|X \cap T|, |Y \cap T|\}$. Since $\alpha_0 = 1/\log^3 m$, and $m$ is larger than a large enough constant (because $m \geq \mu^{20}$), and since $\beta_{\text{ARV}}(m) = O(\sqrt{\log m})$, we get that the sparsity of the cut $(X, Y)$ is less than 1. Consider now any vertex $t \in T$, and let $v$ be the unique neighbor of $t$ in $R^+$. We can assume w.l.o.g. that either $t, v$ both lie in $X$, or they both lie in $Y$. Indeed, if $t \in X$ and $v \in Y$, then moving vertex $t$ from $X$ to $Y$ does not increase the sparsity of the cut $(X, Y)$. This is because, for any two real numbers $1 \leq a < b$, $\frac{a-1}{b-1} \leq \frac{a}{b}$. Similarly, if $t \in Y$ and $v \in X$, then moving $t$ from $Y$ to $X$ does not increase the sparsity of the cut $(X, Y)$. Therefore, we assume from now on, that for every vertex $t \in T$, if $v$ is the unique neighbor of $t$ in $R^+$, then either both $v, t \in X$, or both $v, t \in Y$.

Consider now the partition $(X', Y')$ of $V(R)$, where $X' = X \setminus T$ and $Y' = Y \setminus T$. It is easy to verify that $|\delta_G(R) \cap \delta_G(X')| = |X \cap T|$, and $|\delta_G(R) \cap \delta_G(Y')| = |Y \cap T|$. Let $E' = E_G(X', Y')$, and assume w.l.o.g. that $|\delta_G(R) \cap \delta_G(X')| \leq |\delta_G(R) \cap \delta_G(Y')|$. Then $|E'| < \alpha_0 \cdot \beta_{\text{ARV}}(m) \cdot |\delta_G(R) \cap \delta_G(X')|$ must hold. We remove cluster $R$ from sets $\mathcal{R}$ and $\mathcal{R}^A$, and we add instead every connected component of graphs $G[X']$ and $G[Y']$ to both sets. It is immediate to verify that $\mathcal{R}$ remains a collection of disjoint clusters of $G$, and that $\bigcup_{R' \in \mathcal{R}} V(R') = V(G)$. Therefore, all invariants continue to hold. We now show that the total budget $B$ decreases by at least $1/m$ as the result of this operation.

Note that the only edges whose budgets may change as the result of this operation are edges of $\delta_G(R) \cup E'$. Observe that, for each edge $e \in \delta_G(R) \cap \delta_G(Y')$, its budget $b(e)$ may not increase. Since we have assumed that $|\delta_G(R) \cap \delta_G(X')| \leq |\delta_G(R) \cap \delta_G(Y')|$, and since $|E'| < |\delta_G(R)|/8$, we get that $|\delta_G(X')| \leq 2|\delta_G(R)|/3$. Therefore, for every edge $e \in \delta_G(X') \cap \delta_G(R)$, its budget $b(e)$ decreases by at least $8\alpha_0 \cdot \beta_{\text{ARV}}(m) \cdot \log_{3/2}(|\delta_G(R)|) - 8\alpha_0 \cdot \beta_{\text{ARV}}(m) \cdot \log_{3/2}(|\delta_G(X')|)$. Since $|\delta_G(X')| \leq 2|\delta_G(R)|/3$, we get that $\log_{3/2}(|\delta_G(R)|) \geq \log_{3/2}(3|\delta_G(X')|/2) \geq 1 + \log_{3/2}(|\delta_G(X')|$. We conclude that the budget $b(e)$ of each edge $e \in \delta_G(X') \cap \delta_G(R)$ decreases by at least $8\alpha_0 \cdot \beta_{\text{ARV}}(m)$. On the other hand, the budget of every edge $e \in E'$ increases by at most 3. Since $|E'| \leq \alpha_0 \cdot \beta_{\text{ARV}}(m) \cdot |\delta_G(R) \cap \delta_G(X')|$, we get that the decrease in the budget $B$ is at least:

$$\begin{aligned}
&8\alpha_0 \cdot \beta_{\text{ARV}}(m) \cdot |\delta_G(X') \cap \delta_G(R)| - 3|E'| \\
&\qquad \geq 8\alpha_0 \cdot \beta_{\text{ARV}}(m) \cdot |\delta_G(X') \cap \delta_G(R)| - 3\alpha_0 \cdot \beta_{\text{ARV}}(m) \cdot |\delta_G(R) \cap \delta_G(X')| \\
&\qquad \geq 5\alpha_0 \cdot \beta_{\text{ARV}}(m) \cdot |\delta_G(R) \cap \delta_G(X')| \\
&\qquad > 1/m,
\end{aligned}$$

since $\alpha_0 \geq 1/m$. Therefore, the total budget of all edges decreases by at least $1/m$. Since the clusters only become smaller, it is easy to verify that the budgets of the vertices of $U$ do not increase. To conclude, if $|E_R(X, Y)| < \alpha_0 \cdot \beta_{\text{ARV}}(m) \cdot \min\{|X \cap T|, |Y \cap T|\}$, then we have modified the set $\mathcal{R}$ of clusters, so that all invariants continue to hold, and the total budget $B$ decreases by at least $1/m$. In this case, we terminate the current iteration.

From now on we assume that $|E(X, Y)| > \alpha_0 \cdot \beta_{\text{ARV}}(m) \cdot \min\{|X \cap T|, |Y \cap T|\}$, which, as observed already, implies that cluster $R$ has the $\alpha_0$-bandwidth property. We now proceed to describe the second step of the algorithm.

**Step 2: Ensuring Connectivity of Vertices of $U$.** If, for every pair $u, v \in U_R$ of distinct vertices, there is a collection of at least $\frac{8m}{\mu^{50}}$ edge-disjoint paths in $R$ connecting $u$ to $v$, then we move cluster $R$ from $\mathcal{R}^A$ to $\mathcal{R}^I$ and terminate the current iteration. It is easy to verify that all invariants continue to hold.

Assume now that there is a pair $u, v \in U_R$ of distinct vertices, such that the largest collection of edge-disjoint paths in graph $R$ connecting $u$ to $v$ contains fewer than $\frac{8m}{\mu^{50}}$ paths. From the max-flow / min-cut theorem, there is a cut $(X, Y)$ of $R$, with $u \in X$, $v \in Y$, and $|E_R(X, Y)| < \frac{8m}{\mu^{50}}$. We assume w.l.o.g. that $|X \cap U_R| \leq |Y \cap U_R|$. We delete cluster $R$ from $\mathcal{R}$ and from $\mathcal{R}^A$, and we add instead every connected component of $G[X]$ and $G[Y]$ to both sets. We now show that the total budget decreases by at least $1/m$ as the result of this procedure.

Notice that for every vertex $x \in U$, the budget of $x$ did not increase. Moreover, if $x$ is a vertex of $X \cap U_R$, then its original budget was $4|U_R| \cdot \frac{m}{\mu^{40}}$, and its new budget is $4|U \cap X| \cdot \frac{m}{\mu^{40}} \leq 2|U_R| \cdot \frac{m}{\mu^{40}}$. Since $U \cap X \neq \emptyset$, we get that $\sum_{x \in U} b(x)$ decreased by at least $\frac{2m}{\mu^{40}}$.

Next, we consider the changes to the budgets of the edges. First, every edge in set $E' = E_R(X, Y)$ had budget 0 at the beginning of the iteration, and has budget at most 3 at the end of the iteration. Since $|E_R(X, Y)| \leq \frac{8m}{\mu^{50}}$, the increase in the budget of the edges of $E'$ is bounded by $\frac{24m}{\mu^{50}}$.

We now consider two cases. The first case happens if $|\delta_G(R)| \leq \frac{m}{3\mu^{40}}$. In this case, the increase in the budget of every edge $e \in \delta_G(R)$ is bounded by 3 (since edge budgets may not exceed 3), and so the total increase in the budgets of edges $e \in \delta_G(R)$ is bounded by $3|\delta_G(R)| \leq \frac{m}{\mu^{40}}$. The total increase in all edge budgets is then bounded by $\frac{m}{\mu^{40}} + \frac{24m}{\mu^{50}}$, and, since the total budgets of all vertices in $U$ decreases by at least $\frac{2m}{\mu^{40}}$, we get that the total budget $B$ decreases by at least $\frac{m}{4\mu^{40}} \leq \frac{1}{m}$.

Lastly, we assume that $|\delta_G(R)| > \frac{m}{3\mu^{40}}$. Consider some edge $e \in \delta_G(R)$. Since $|\delta_G(X)|, |\delta_G(Y)| \leq |\delta_G(R)| + |E'|$, the increase in the budget of $e$ is bounded by:

$$8\alpha_0 \cdot \beta_{\text{ARV}}(m) \cdot (\log_{3/2}(|\delta_G(R)| + |E'|) - \log_{3/2}(|\delta_G(R)|)) \leq 8\alpha_0 \cdot \beta_{\text{ARV}}(m) \cdot \log_{3/2}\left(1 + \frac{|E'|}{|\delta_G(R)|}\right).$$

Since we have assumed that $|\delta_G(R)| > \frac{m}{3\mu^{40}}$, while $|E'| \leq \frac{3m}{\mu^{50}}$, we get that $\frac{|E'|}{|\delta_G(R)|} < 1/2$. Since for all $\epsilon \in (0, 1/2)$, $\ln(1+\epsilon) \leq \epsilon$, we get that the increase in the budget of $e$ is bounded by $8\alpha_0 \cdot \beta_{\text{ARV}}(m) \cdot \frac{|E'|}{|\delta_G(R)| \cdot \ln(3/2)} \leq 24\alpha_0 \cdot \beta_{\text{ARV}}(m) \cdot \frac{|E'|}{|\delta_G(R)|}$. The increase in the budget of all edges of $\delta_G(R)$ is then bounded by $24\alpha_0 \cdot \beta_{\text{ARV}}(m) \cdot |E'| \leq \frac{576m\alpha_0\beta_{\text{ARV}}(m)}{\mu^{50}} \leq \frac{m}{\mu^{49}}$. Since the budget of all edges in $E'$ increases by at most $\frac{24m}{\mu^{50}}$, and the budget of the vertices of $U$ decreases by at least $\frac{2m}{\mu^{40}}$, the total budget in the system decreases by at least $\frac{m}{\mu^{40}} \geq \frac{1}{m}$.

Since the initial budget $B$ is bounded by $\frac{m}{\mu^{27}}$, and in every iteration, either a new cluster is added to set $\mathcal{R}^I$, or the budget $B$ decreases by at least $1/m$, the number of iterations is bounded by $\text{poly}(m)$, so the algorithm is efficient. Once the algorithm terminates, $\mathcal{R}^I = \mathcal{R}$ holds. We then return the set $\mathcal{C} = \mathcal{R}$ of clusters as the outcome of the algorithm. From our invarinats, we are guaranteed that $\bigcup_{C \in \mathcal{C}} V(C) = V(G)$, every cluster $C \in \mathcal{C}$ has the $\alpha_0$-bandwidth property, and for every cluster $C \in \mathcal{C}$, for every pair $u, v$ of distinct vertices of $C$ with $\deg_G(v), \deg_G(u) \geq \frac{m}{\mu^6}$, there is a collection of at least $\frac{8m}{\mu^{50}}$ edge-disjoint paths in $C$ connecting $u$ to $v$. Since the total budget $B$ remains bounded by $\frac{m}{\mu^{27}}$, and $\sum_{C \in \mathcal{C}} |\delta_G(C)| \leq B$, we get that $\sum_{C \in \mathcal{C}} |\delta_G(C)| \leq \frac{m}{\mu^{27}}$ holds. □

We are now ready to complete the proof of Theorem 3.12. We start by applying the algorithm from Lemma 8.1 to instance $I$, to obtain a collection $\mathcal{C}$ of clusters. We then apply Algorithm AlgAdvancedDisengagement from Theorem 7.1 to instance $I = (G, \Sigma)$, cluster set $\mathcal{C}$, parameter $\mu$

that remains unchanged, and parameter $m = |E(G)|$. Let $\mathcal{I}$ be the $2^{O((\log m)^{3/4}\log\log m)}$-decomposition of instance $I$ that the algorithm returns. Recall that every instance $I' \in \mathcal{I}$ is a subinstance of $I$. Consider any instance $I' = (G', \Sigma') \in \mathcal{I}$, and assume that $|E(G')| > m/\mu$, and that $I'$ is a wide instance. It is enough to prove that instance $I'$ is well-connected. Indeed, the algorithm from Theorem 7.1 ensures that there is at most one cluster $C \in \mathcal{C}$ with $E(C) \subseteq E(G')$. If no such cluster exists, then $E(G') \subseteq E^{\mathsf{out}}(\mathcal{C})$. Since $|E^{\mathsf{out}}(\mathcal{C})| \leq \frac{m}{\mu^{27}}$, $|E(G')| \leq \frac{m}{\mu^{27}}$ must hold in this case, contradicting our assumption that $|E(G')| > m/\mu$. Therefore, there must be a cluser $C \in \mathcal{C}$ with $C \subseteq G'$. The algorithm from Theorem 7.1 then guarantees that $E(G') \subseteq E(C) \cup E^{\mathsf{out}}(\mathcal{C})$. Consider some vertex $v \in V(G')$ with $\deg_{G'}(v) \geq \frac{|E(G')|}{\mu^5}$. Since $|E(G')| \geq \frac{m}{\mu}$, $\deg_{G'}(v) \geq \frac{|E(G')|}{\mu^5} \geq \frac{m}{\mu^6}$ must hold. In particular, since $I'$ is a subinstance of $I$, and since $|E^{\mathsf{out}}(\mathcal{C})| \leq \frac{m}{\mu^{27}}$, vertex $v$ must lie in cluster $C$ (as otherwise all edges incident to $v$ in $G'$ belong to $E^{\mathsf{out}}(\mathcal{C})$), and so $\deg_G(v) \geq \frac{m}{\mu^6}$ must hold as well. The algorithm from Lemma 8.1 ensures that, for every pair $u, v$ of vertices of $C$ with $\deg_G(u), \deg_G(v) \geq \frac{m}{\mu^6}$, there is a collection $\mathcal{P}$ of at least $\frac{8m}{\mu^{50}} \geq \frac{8|E(G')|}{\mu^{50}}$ edge-disjoint paths in $C$ connecting $u$ to $v$. Since $C \subseteq G'$, every path in $\mathcal{P}$ is also contained in $G'$. We conclude that instance $I'$ must be well-connected.

# 9 An Algorithm for Wide and Well-Connected Instances – Proof of Theorem 3.13

This section is dedicated to the proof of Theorem 3.13. Recall we are given a wide and a well-connected instance $I = (G, \Sigma)$ of the MCNwRS problem. For convenience, throughout this section, we refer to instance $I$ as $\check{I}^* = (\check{G}^*, \check{\Sigma}^*)$ and denote $\check{m} = |E(\check{G}^*)|$. We let $\check{G}$ be the graph that is obtained from graph $\check{G}^*$ by subdividing every edge of $\check{G}^*$ with a vertex. Since every vertex in $V(\check{G}) \setminus V(\check{G}^*)$ has degree 2, we can extend the rotation system $\check{\Sigma}^*$ for graph $\check{G}^*$ to a rotation system $\check{\Sigma}$ for graph $\check{G}$, in a natural way. We denote by $\check{I} = (\check{G}, \check{\Sigma})$ the resulting instance of MCNwRS. We sometimes refer to $\check{I}$ as the *subdivided instance corresponding to* $\check{I}^*$. Note that $|E(\check{G})| = 2\check{m}$ and $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) = \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}^*)$. Throughout this section, we will mostly be working with instance $\check{I}$. We will use notation $I$ and $m$ when discussing various subinstances of $\check{I}$. Recall that $\check{m} \geq \mu^{c'}$ must hold, where $c'$ is a large enough constant. Given a subgraph $G \subseteq \check{G}$, we let $\Sigma$ be the rotation system for $G$ induced by $\check{\Sigma}$, and we will refer to instance $I = (G, \Sigma)$ as the *subinstance of* $\check{I}$ *defined by graph* $G$.

We start with intuition. Fix an optimal solution $\varphi^*$ to instance $\check{I}$, where $\varphi^*$ is a drawing of graph $\check{G}$ on the sphere. In order to simplify the exposition, assume that $\mathsf{cr}(\varphi^*) = \mathsf{OPT}_{\mathsf{cnwrs}}(I) \leq \check{m}^2/\mu^{c'}$, where $c'$ is a large enough constant. Since instance $\check{I}^*$ is wide, there is a vertex $v \in V(\check{G})$, a partition $(E_1, E_2)$ of the edges of $\delta_{\check{G}}(v)$, with the edges of $E_1$ appearing consecutively in the rotation $\mathcal{O}_v \in \Sigma$, and a collection $\mathcal{P}$ of at least $\check{m}/\mu^{50}$ edge-disjoint cycles in graph $G$, where every cycle $P \in \mathcal{P}$ contains an edge of $E_1$ and an edge of $E_2$. Informally, we say that a vertex $u$ of $\check{G}$ is *heavy* if it lies on a large number of cycles of $\mathcal{P}$, otherwise we say it is *light*.

Let $P^* \in \mathcal{P}$ be a cycle that we select uniformly at random. Since $|\mathcal{P}| \geq \check{m}/\mu^{50}$, the expected number of crossings in which the edges of $E(P^*)$ participate in $\varphi^*$ is relatively small – at most $\mathsf{OPT}_{\mathsf{cnwrs}}(I) \cdot \mu^{50}/\check{m}$. We can use this fact in order to show that, with a high enough probability, there is a near-optimal solution $\varphi'$ to instance $\check{I}$, in which the images of the edges of $E(P^*)$ do not cross each other, and they participate in relatively few crossings. Let $E'$ denote the set of all edges $e \in E(\check{G})$, such that $e$ is incident to a light vertex $u \in V(P^*)$, and $e \notin E(P^*)$. From the definition of light vertices and the fact that $|\mathcal{P}|$ is large, we can show that with high enough probability, $|E'|$ is quite small. Additionally, using the cycles in $\mathcal{P}$, we can compute, for every heavy vertex $u \in V(P^*)$, an orientation $b_u \in \{-1, 1\}$. We show that with high probability, this orientation is consistent with the solution $\varphi'$ to instance $\check{I}$. In other words, if $b_u = -1$, then the images of the edges of $\delta_G(u)$ enter the image of $u$ in $\varphi'$ according to the ordering $\mathcal{O}_u \in \Sigma$ in clock-wise direction, and otherwise the direction is counter-clock-wise.

Note that the image of the cycle $P^*$ in the near-optimal solution $\varphi'$ to instance $\check{I}$ partitions the sphere into two internally disjoint discs $D$ and $D'$. Let $E''$ be the set of edges of $\check{G}$ whose images cross the images of the edges of $P^*$. From our construction, with reasonably high probability, $|E''|$ is relatively small. We can view the image of cycle $P^*$ in $\varphi'$ as splitting graph $G \setminus (E' \cup E'')$ into two subgraphs, $G_1$ and $G_2$, where $G_1$ contains all edges and vertices that are drawn inside $D$, while $G_2$ contains all edges and vertices that are drawn inside $D'$. The only vertices and edges that the two graphs share are the vertices and edges of $P^*$. We can further ensure that each of the subinstances contains a significant number of edges, so $|E(G_1)|, |E(G_2)|$ are both significantly smaller than $\check{m}$.

If we could compute the two graphs $G_1, G_2$ efficiently, then we could construct two subinstances $I_1 = (G_1', \Sigma_1'), I_2 = (G_2', \Sigma_2')$ of instance $\check{I}$, where graph $G_1'$ is obtained from $G_1$ by contracting the vertices of $V(P^*)$ into a supernode $v^*$, and letting the rotation of this supernode in $\Sigma_1'$ be determined by the rotations of the vertices of $V(P^*)$ in instance $\check{I}$, the orientations of the heavy vertices of $P^*$ that we have computed, and the order of the vertices of $P^*$ on the cycle; instance $I_2$ is computed similarly from $G_2$. Given solutions $\varphi_1, \varphi_2$ to the instances $I_1, I_2$, respectively, we can combine them to obtain a solution $\varphi$ to instance $\check{I}$, as follows. First, we un-contract the supernode $v^*$ in each of the two instances to obtain a cycle $P^*$, and then we "glue" the two drawings together via this cycle. Next, we insert the edges of $E' \cup E''$ into the resulting drawing, obtaining a drawing $\varphi$ of $G$. Since $|E'|, |E''|$ are relatively small, so is the increase in the number of crossings relatively to $\mathsf{cr}(\varphi_1) + \mathsf{cr}(\varphi_2)$. We could then apply the same decomposition process recursively to instances $I_1$ and $I_2$ (if these instances are wide). We refer to $I_1$ and $I_2$ as the *contracted instances* corresponding to subgraphs $G_1$ and $G_2$ of $G$, respectively.

The main difficulty with this approach is that we do not know the optimal solution $\varphi^*$ to instance $\check{I}$, or the near-optimal solution $\varphi'$, and so we cannot compute the two graphs $G_1, G_2$ with the required properties. We also do not know the set $E''$ of edges whose images cross the images of the edges of $E(P^*)$ in $\varphi'$. Instead, we compute a relatively small edge set $E^*$, and two subgraphs $\tilde{G}_1, \tilde{G}_2$ of $G \setminus (E' \cup E^*)$, for which the following hold. First, $E(\tilde{G}_1) \cup E(\tilde{G}_2) \cup E' \cup E^* = E(G)$. Second, the only vertices and edges that these two graphs share are the vertices and edges of $P^*$. Third, the number of edges in each of the graphs $\tilde{G}_1$, $\tilde{G}_2$ is significantly smaller than $|E(G)|$. Lastly, there is a near-optimal drawing $\varphi'$ of $G \setminus (E' \cup E^*)$, in which the edges of $P^*$ do not cross each other. Moreover, if we denote by $D$ and $D'$ the two discs of the sphere whose boundaries are the image of $P^*$ in $\varphi'$, then, by slightly modifying the drawing, we can ensure that all vertices of $\tilde{G}_1$ are drawn inside $D$, all vertices of $\tilde{G}_2$ are drawn inside $D'$, and the number of crossings in which the edges of $P^*$ participate is quite small. Unfortunately, we can no longer guarantee that every edge of $\tilde{G}_1$ is drawn inside $D$ and every edge of $\tilde{G}_2$ is drawn inside $D'$. We can then define the contracted instances $I_1$ and $I_2$ associated with the graphs $\tilde{G}_1$ and $\tilde{G}_2$ exactly as before. As before, given solutions to instances $I_1$ and $I_2$, we can efficiently combine them to obtain a solution to instance $\check{I}$. But unfortunately we can no longer claim that $\mathsf{OPT}_{\mathsf{cnwrs}}(I_1) + \mathsf{OPT}_{\mathsf{cnwrs}}(I_2)$ is small. This is since, in the near-optimal drawing of $G \setminus (E' \cup E^*)$, some edges of $\tilde{G}_1$ may be partially drawn inside the disc $D'$, and it is not clear how to "move" them to the interior of $D$ without significantly increasing the number of crossings. The same is true for edges of $\tilde{G}_2$ that may be partially drawn inside the disc $D$. This is the main difficulty in designing our algorithm using the framework outlined above.

Our algorithm consists of two phases. In the first phase, we follow the above framework to construct an initial collection $\mathcal{I}$ of subinstances of $\check{I}$, that have all required properties, except that we will not be able to ensure that $\sum_{I \in \mathcal{I}} \mathsf{OPT}_{\mathsf{cnwrs}}(I)$ is suitably bounded, even if $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})$ is small. In the second phase, we will try to "repair" each one of the instances $I = (G, \Sigma) \in \mathcal{I}$, by removing a small subset of edges from $G$. We will show that, after the removal of these edges, the expectation of $\sum_{I \in \mathcal{I}} \mathsf{OPT}_{\mathsf{cnwrs}}(I)$ is suitably bounded. In both phases, we rely on the same algorithm outlined above, that gradually decomposes an input instance $I = (G, \Sigma)$ into smaller and smaller subinstances. The

algorithms for both Phase 1 and Phase 2 follow this high-level framework, though the specifics are somewhat different. We start with the main definitions that we use throughout this section.

## 9.1 Main Definitions

Throughout, given a graph $H$ and a drawing $\varphi$ of $H$ on the sphere or in the plane, we denote by $\mathcal{F}(\varphi)$ the set of all faces in drawing $\varphi$.

We use the notion of a *subdivided graph*.

**Definition 9.1 (Subdivided graph)** *We say that a graph $G$ is a* subdivided graph*, if $G$ does not contain parallel edges, and additionally, for every edge $e = (u, v)$, either $\deg_G(u) \leq 2$ or $\deg_G(v) \leq 2$ holds.*

Note that, if $G$ is any graph, and $G'$ is a graph obtained by suvdividing every edge of $G$, then $G'$ is a subdivided graph, and so is every subgraph of $G'$. In particular, graph $\check{G}$ associated with instance $\check{I}$ of MCNwRS is subdivided, and so is every subgraph of $\check{G}$.

### 9.1.1 Cores and Core Structures

The first central notion that we use is that of a core, and its associated core structure.

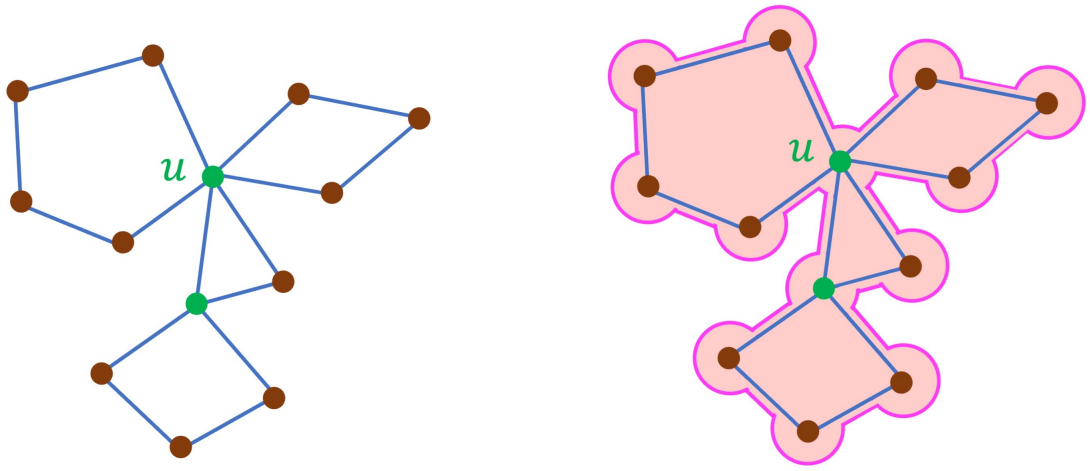
**Definition 9.2 (Core and Core Structure)** *Let $G$ be a subgraph of $\check{G}$, and let $I = (G, \Sigma)$ be the subinstance of $\check{I}$ defined by $G$. A* core structure *for instance $I$ consists of the following:*

- *a connected subgraph $J$ of $G$ called a* core*, such that, for every edge $e \in E(J)$, graph $J \setminus \{e\}$ is connected (but we also allow $J$ to consist of a single vertex);*
- *an orientation $b_u \in \{-1, 1\}$ for every vertex $u \in V(J)$;*
- *a drawing $\rho_J$ of $J$ on the sphere with no crossings, that is consistent with the rotation system $\Sigma$ and the orientations in $\{b_u\}_{u \in V(J)}$. In other words, for every vertex $u \in V(J)$, the images of the edges in $\delta_J(u)$ enter the image of $u$ in $\rho_J$ according to their order in the rotation $\mathcal{O}_u \in \Sigma$ and orientation $b_u$ (so, e.g. if $b_u = 1$ then the orientation is counter-clock-wise); and*
- *a distinguished face $F^*(\rho_J) \in \mathcal{F}(\rho_J)$, such that the image of every vertex $u \in V(J)$, and the image of every edge $e \in E(J)$ is contained in the boundary of face $F^*(\rho_J)$ in drawing $\rho_J$.*

We denote a core structure by $\mathcal{J} = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$, and we refer to graph $J$ as the *core associated with* $\mathcal{J}$. We denote $\mathcal{F}^{\times}(\rho_J) = \mathcal{F}(\rho_J) \setminus \{F^*(\rho_J)\}$, and we refer to the faces in $\mathcal{F}^{\times}(\rho_J)$ as the *forbidden faces* of the drawing $\rho_J$.

The last two requirements in the above definition impose a certain structure on the core graph $J$. Specifically, there must be a collection $\mathcal{W}$ of edge-disjoint cycles, with $\bigcup_{W \in \mathcal{W}} E(W) = E(J)$, such that every pair $W, W' \in \mathcal{W}$ of cycles share at most one vertex, which must be a separator vertex for $J$ (see Figure 28(a) for an illustration).

Note that, since graph $G$ is subdivided, for every edge $e \in E(G) \setminus E(J)$, at most one endpoint of $e$ may lie in $J$. Indeed, assume that $e = (u, v)$. From the definition of subdivided graphs, either $\deg_G(u) \leq 2$ or $\deg_G(v) \leq 2$ holds. Assume w.l.o.g. that it is the former. If $u \in V(J)$, then graph $J$ contains at most one edge incident to $u$, that we denote by $e'$. But then $J \setminus \{e'\}$ is not a connected graph, contradicting the definition of a core.




(a) A core $J$ and its drawing $\rho_J$, with the separator vertices of $J$ shown in green. The distinguished face $F^*(\rho_J)$ is the infinite face in this drawing.

(b) Disc $D(J)$ associated with core $J$, and its boundary (shown in pink).

Figure 28: An illustration of a core $J$ and its disc $D(J)$.

Consider now a core structure $\mathcal{J} = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$, and specifically the drawing $\rho_J$ of the graph $J$ on the sphere. We define a disc $D(J)$, that contains the drawing of $J$ in its interior, such that the boundary of disc $D(J)$ is contained in face $F^*(\rho_J)$, and it is a simple closed curve that closely follows the boundary of $F^*(\rho_J)$ (see Figure 28(b)).

Consider some vertex $u \in V(J)$, and the tiny $u$-disc $D(u) = D_{\rho_J}(u)$ in the drawing $\rho_J$. Since vertex $u$ lies on the boundary of face $F^*(\rho_J)$, we can define disc $D(u)$ so that, for every maximal segment $\sigma$ on the boundary of $D(u)$ that is contained in $F^*(\rho_J)$, there is a contiguous curve $\sigma' \subseteq \sigma$ of non-zero length, that is contained in the boundary of the disc $D(J)$ (see Figure 29).

Denote $\delta_G(u) = \left\{e_1^u, \ldots, e_{d_u}^u\right\}$, where $d_u = \deg_G(u)$, so that the edges are indexed according to their ordering in the rotation $\mathcal{O}_u \in \Sigma$. We can then define a collection $\left\{p_1^u, \ldots, p_{d_u}^u\right\}$ of distinct points on the boundary of the disc $D(u)$, such that the following hold:

- points $p_1^u, \ldots, p_{d_u}^u$ are encountered in this order when traversing the boundary of $D(u)$ in the direction of the orientation $b_u$;
- for every edge $e_i^u \in E(J)$, point $p_i^u$ is the unique point on the image of $e_i^u$ in $\rho_J$ lying on the boundary of $D(u)$; and
- if a point $p_i^u$ lies in the interior of face $F^*(\rho_J)$, then it lies on the boundary of the disc $D(J)$.

For all $1 \leq i \leq d_u$, we view point $p_i^u$ as representing the edge $e_i^u$. There is one more property that we require from a core structure.

**Definition 9.3 (Valid Core Structure)** *We say that a core structure $\mathcal{J} = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$ is* valid *if, for every vertex $u \in V(J)$, for every edge $e_i^u \in \delta_G(u) \setminus E(J)$, the corresponding point $p_i^u$ lies in the interior of face $F^*(\rho_J)$ (and hence on the boundary of the disc $D(J)$).*

In the remainder of this section, whenever we use the term "core structure", we assume that this core structure is valid.

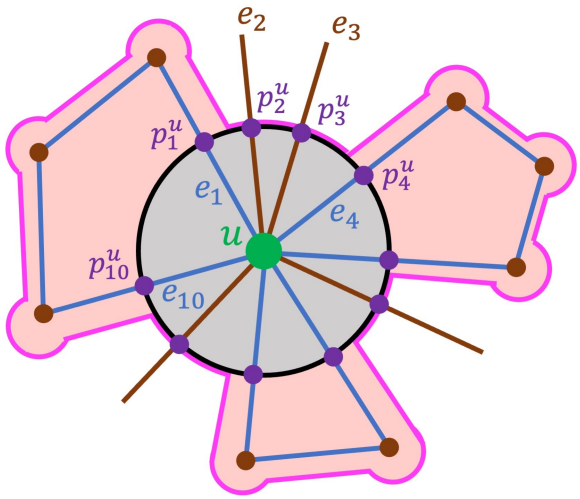



Figure 29: Illustration of disc $D(u)$ for a core that contains $u$. Disc $D(u)$ is shown in gray, and disc $D(J)$ is shown in pink. Note that both discs share portions of their boundaries that are contained in face $F^*(\rho_J)$ – the infinite face in the current drawing. Edges of $J$ incident to $u$ are shown in blue, and all other edges that are incident to $u$ are shown in brown.

**Ordering $\mathcal{O}(J)$ of the Edges of $\delta_G(J)$.** Consider a core structure $\mathcal{J} = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$, the drawing $\rho_J$ of $J$, and its corresponding disc $D(J)$. Recall that, for every vertex $u \in V(J)$ and every edge $e_i^u \in \delta_G(J)$, we have defined a point $p_i^u$ on the boundary of the disc $D(J)$ representing the edge $e_i^u$. Recall that each edge $e \in \delta_G(J)$ has exactly one endpoint in $J$. We define a circular oriented ordering $\mathcal{O}(J)$ of the edges of $\delta_G(J)$ to be the circular order in which the points $p_i^u$ corresponding to the edges of $\delta_G(J)$ are encountered, as we traverse the boundary of the disc $D(J)$ in the clock-wise direction.

### 9.1.2 Drawings of Graphs

Next, we define a valid drawing of a graph $G$ with respect to a core structure $\mathcal{J}$.

**Definition 9.4 ($\mathcal{J}$-Valid Solution)** *Let $G$ be a subgraph of $\check{G}$, let $I = (G, \Sigma)$ be the subinstance of $\check{I}$ defined by $G$, and let $\mathcal{J} = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$ be a core structure for $I$. A solution $\varphi$ of instance $I$ is* $\mathcal{J}$-valid *if we can define a disc $D'(J)$ that contains the images of all vertices and edges of the core $J$ in its interior, and the image of the core $J$ in $\varphi$ is identical to $\rho_J$ (including the orientation), with disc $D'(J)$ in $\varphi$ playing the role of the disc $D(J)$ in $\rho_J$. We sometimes refer to a $\mathcal{J}$-valid solution to instance $I$ as a* $\mathcal{J}$-valid drawing of graph $G$.

Abusing the notation, we will not distinguish between disc $D(J)$ in $\rho_J$ and disc $D'(J)$ in $\varphi$, denoting both discs by $D(J)$.

Consider now some solution $\varphi$ to instance $I$, that is $\mathcal{J}$-valid, with respect to some core structure $\mathcal{J}$. The image of graph $J$ in $\varphi$ partitions the sphere into regions, each of which corresponds to a unique face of $\mathcal{F}(\rho_J)$. We do not distinguish between these regions and faces of $\mathcal{F}(\rho_J)$, so we view $\mathcal{F}(\rho_J)$ as a collection of regions in the drawing $\varphi$ of $G$.

Note that the edges of $J$ may participate in crossings in $\varphi$, but no two edges of $J$ may cross each other. Consider now a crossing $(e, e')_p$ in drawing $\varphi$. We say that it is a *dirty* crossing if exactly one of the two edges $e, e'$ lies in $E(J)$. We denote by $\chi^{\mathsf{dirty}}(\varphi)$ the set of all dirty crossings of drawing $\varphi$. We say that an edge $e \in E(G) \setminus E(J)$ is *dirty* in $\varphi$ if it participates in some dirty crossing of $\varphi$.

Next, we define special types of $\mathcal{J}$-valid drawings, called clean and semi-clean drawings.

**Definition 9.5 (Clean and Semi-Clean Drawings)** *Let $G$ be a subgraph of $\check{G}$, let $I = (G, \Sigma)$ be the subinstance of $\check{I}$ defined by $G$, let $\mathcal{J} = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$ be a core structure for $I$, and let $\varphi$ be a solution to instance $I$. We say that $\varphi$ is a* semi-clean solution to instance $I$, *or a* semi-clean drawing of $G$, *with respect to $\mathcal{J}$, if it is a $\mathcal{J}$-valid drawing, and, additionally, the image of every vertex of $V(G) \setminus V(J)$ lies outside of the disc $D(J)$ (so in particular it must lie in the interior of the region $F^*(\rho_J) \in \mathcal{F}(\rho_J)$).*
*If, additionally, the image of every edge of $E(G) \setminus E(J)$ is entirely contained in region $F^*(\rho_J)$, then we say that $\varphi$ is a* clean *solution to $I$ with respect to $\mathcal{J}$, or that it is a $\mathcal{J}$-*clean *solution.*

Notice that, from the definition, if $\varphi$ is a clean solution to instance $I$ with respect to core structure $\mathcal{J}$, then the edges of $J$ may not participate in any crossings in $\varphi$.

**Drawings of Subgraphs and Compatible Drawings.** Notice that, if $G'$ is a subgraph of $G$ that contains $J$, then a core structure $\mathcal{J}$ for instance $I = (G, \Sigma)$ remains a valid core structure for the subinstance $I' = (G', \Sigma')$ of $\check{I}$ defined by $G'$. Therefore, $\mathcal{J}$-valid drawings are well-defined for every subgraph $G' \subseteq G$.
Assume now that we are given a $\mathcal{J}$-valid solution $\varphi$ to instance $I$, and a subinstance $I' = (G', \Sigma')$ of $I$ that is defined as above. Intuitively, we will often obtain a $\mathcal{J}$-valid solution $\varphi'$ to instance $I'$ by slightly modifying the solution $\varphi$ to instance $I$. We will, however, restrict the types of modifications that we allow. In particular we do now allow adding any new images of edges (or their segments), or new images of vertices to the forbidden regions in $\mathcal{F}^{\times}(\rho_J)$. We now define these restrictions formally.

**Definition 9.6 (Compatible Drawings.)** *Let $G$ be a subgraph of $\check{G}$, let $I = (G, \Sigma)$ be the subinstance of $\check{I}$ defined by $G$, let $\mathcal{J} = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$ be a core structure for $I$, and let $\varphi$ be a $\mathcal{J}$-valid solution to instance $I$. Let $G'$ be a subgraph of $G$ with $J \subseteq G'$, and let $I' = (G', \Sigma')$ be the subinstance of $\check{I}$ defined by $G'$. Finally, let $\varphi'$ be a $\mathcal{J}$-valid solution to instance $I'$. We say that drawing $\varphi'$ of $G'$ is* compatible *with drawing $\varphi$ of $G$ with respect to $\mathcal{J}$, if the following hold:*

- *the image of the core $J$ and the correpsonding disc $D(J)$ in $\varphi'$ are identical to those in $\varphi$;*
- *if a point $p$ is an inner point of an image of an edge in $\varphi'$, then it is an inner point of an image of an edge in $\varphi$;*
- *if a point $p$ is a crossing point between a pair of edges in $\varphi'$, then it is a crossing point between a pair of edges in $\varphi$;*
- *if a point $p$ is an image of a vertex $v$ in $\varphi'$, then either (i) point $p$ is an image of vertex $v$ in $\varphi$; or (ii) vertex $v$ has degree 2 in $G'$, and point $p$ is an inner point on an image of an edge in $\varphi$;*
- *if the image of a vertex $v \in V(G')$ lies outside the region $F^*(\rho_J)$ in $\varphi'$, then $\varphi'(v) = \varphi(v)$; and*
- *if $\sigma$ is a maximal segment of an image of an edge $e \in E(G')$ in $\varphi'$ that is internally disjoint from region $F^*(\rho_J)$, then $\sigma \subseteq \varphi(e)$.*

Note that, if we obtain drawing $\varphi'$ from drawing $\varphi$, then the only changes that are allowed outside of region $F^*(\rho_J)$ is the deletion of images of vertices or (segments of) images of edges. In other words, $\big(\varphi'(G') \setminus F^*(\rho_J)\big) \subseteq \big(\varphi(G) \setminus F^*(\rho_J)\big)$.

### 9.1.3 A $\mathcal{J}$-Contracted Instance

Suppose we are given a subgraph $G$ of $\check{G}$, together with a core structure $\mathcal{J} = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$ for the subinstance $I$ of $\check{I}$ defined by $G$. We now define a *$\mathcal{J}$-contracted* subinstance $\hat{I} = (\hat{G}, \hat{\Sigma})$ of instance $I$. Graph $\hat{G}$ is obtained from graph $G$, by contracting the vertices of the core $J$ into a supernode $v_J$. In order to define the rotation system $\hat{\Sigma}$, for every vertex $u \in V(\hat{G}) \setminus \{v_J\}$, we let the rotation $\mathcal{O}_u$ remain the same as in $\Sigma$, and for the supernode $v_J$, we set the corresponding rotation $\mathcal{O}_{v_J} \in \Sigma'$ to be the ordering $\mathcal{O}(J)$ of the edges of $\delta_{\hat{G}}(v_J) = \delta_G(J)$ that we have defined above (recall that this is the order in which points $p_i^u$ appear on the boundary of disc $D(J)$, for all $u \in V(J)$ and $1 \leq i \leq d_u$).

Throughout our algorithm, we will consider subgraphs $G \subseteq \check{G}$. Each such subgraph will always be associated with a core structure $\mathcal{J} = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$ for the subinstance $I$ of $\check{I}$ defined by $G$. The $\mathcal{J}$-contracted subgraph of $I$ will always be denoted by $\hat{I} = (\hat{G}, \hat{\Sigma})$. We denote by $\hat{m}(I) = |E(\hat{G})|$ – the number of edges in the $\mathcal{J}$-contracted subinstance of $I$. We need the following simple observation.

**Observation 9.7** *There is an efficient algorithm, whose input consists of a subgraph $G$ of $\check{G}$, a core structure $\mathcal{J} = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$ for the subinstance $I = (G, \Sigma)$ of $\check{I}$ defined by $G$, and a $\mathcal{J}$-clean solution $\varphi$ to instance $I$. The output of the algorithm is a solution $\hat{\varphi}$ to the corresponding $\mathcal{J}$-contracted instance $\hat{I} = (\hat{G}, \hat{\Sigma})$, with $\mathsf{cr}(\hat{\varphi}) = \mathsf{cr}(\varphi)$.*

**Proof:** Consider the solution $\varphi$ to instance $I$. From the definition of a clean solution, there is a disc $D(J)$ that contains the drawing of $J$ in $\varphi$, which is in turn identical to drawing $\rho_J$. For every vertex $u \in V(G) \setminus V(J)$, its image appears outside the disc $D(J)$ in $\varphi$. We are also guaranteed that, for every edge $e \in E(G) \setminus E(J)$, its drawing $\varphi(e)$ is contained in region $F^*(\rho_J)$.

By slightly manipulating the boundary of the disc $D(J)$, we can ensure that, for every edge $e \in E(G) \setminus E(J)$, if $e$ is not incident to any vertex of $J$, then $\varphi(e)$ does not intersect disc $D(J)$, and otherwise, $\varphi(e) \cap D(J)$ is a contiguous curve.

For every edge $e \in \delta_G(J)$, denote by $p_e$ the unique intersection point between the boundary of $D(J)$ and the curve $\varphi(e)$. Since the drawing $\varphi$ of $G$ is $\mathcal{J}$-valid, the circular ordering of the points of $\{p_e \mid e \in \delta_G(J)\}$ on the boundary of disc $D(J)$ is precisely $\mathcal{O}(J)$. For each edge $e \in \delta_G(J)$, we erase the segment of $\varphi(e)$ that is contained in $D(J)$. We then contract the disc $D(J)$ into a single point, that becomes the image of the supernode $v_J$. We have now obtained a valid solution $\hat{\varphi}$ to the $\mathcal{J}$-contracted instance $\hat{I}$, with $\mathsf{cr}(\hat{\varphi}) = \mathsf{cr}(\varphi)$. □

We note that the converse of Observation 9.7 is also true: given a solution $\hat{\varphi}$ to the $\mathcal{J}$-contracted instance $\hat{I}$, we can efficiently construct a clean solution $\varphi$ to instance $I$, with $\mathsf{cr}(\varphi) = \mathsf{cr}(\hat{\varphi})$, as follows. First, we expand the image of the supernode $v_J$ so it becomes a disc, that we denote by $D(J)$. We then plant the drawing $\rho_J$ of the core $J$ inside the disc. Note that the circular ordering in which the edges of $\delta_{\hat{G}}(v_J)$ enter the boundary of the disc $D(J)$ from the outside is identical to the circular ordering in which the edges of $\delta_G(J) = \delta_{\hat{G}}(v_J)$ enter the boundary of the disc $D(J)$ from the inside, and their orientations match. Therefore, we can "glue" the corresponding curves to obtain, for each edge $e \in \delta_G(J)$, a valid drawing connecting the images of its endpoints.

### 9.1.4 Core Enhancement and Promising Sets of Paths

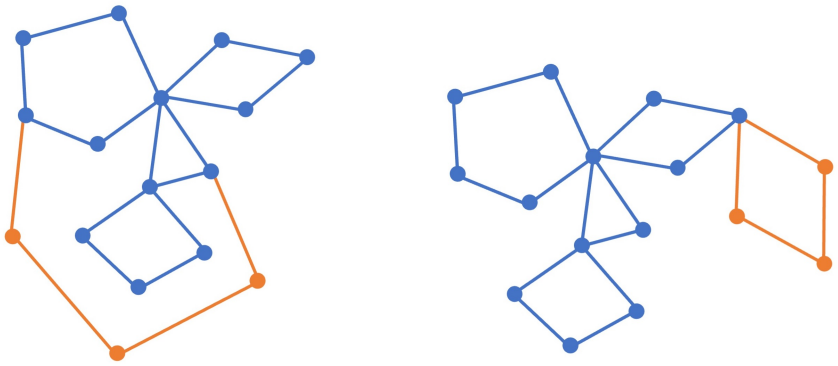
Our main subroutine, called ProcSplit, starts with a subinstance $I = (G, \Sigma)$ of $\check{I}$ that is defined by a subgraph $G$ of $\check{G}$, and a core structure $\mathcal{J} = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$ for it. We "enhance" the corresponding core $J$ by adding either one cycle, or one path to it, that we refer to as *core enhancenemt*. We also decompose instance $I = (G, \Sigma)$ into two subinstances, $I_1 = (G_1, \Sigma_1)$ and $I_2 = (G_2, \Sigma_2)$, where

$G_1, G_2 \subseteq G$. Using the enhancement for the core structure $\mathcal{J}$, we then construct two new core structures: core structure $\mathcal{J}_1$ for instance $I_1$, and core structure $\mathcal{J}_2$ for instance $I_2$.

In order to avoid cumbersome notation, we will sometimes refer to simple cycles as paths. Given a simple cycle $W$, we will designate one of the vertices $v \in V(W)$ to be the "endpoint" of the cycle. When referring to two endpoints of $W$, we will think of both endpoints as being $v$.

We start by defining the notions of *core enhancement* and *core enhancement structure*.

**Definition 9.8 (Core Enhancement)** *Given a subgraph $G$ of $\check{G}$, and a core structure $\mathcal{J} = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$ for the subinstance $I = (G, \Sigma)$ of $\check{I}$ defined by $G$, an* enhancement *of the core structure $\mathcal{J}$ is a simple path $P \subseteq G$ (that may be a simple cycle), whose both endpoints belong to $J$, such that $P$ is internally disjoint from $J$ (see Figure 30).*



(a) When $P$ is a simple path. (b) When $P$ is a simple cycle.

Figure 30: An illustration of an enhancement of a core structure $\mathcal{J}$. The core $J$ is shown in blue, and the enhancement in orange.

For simplicity of notation, we will sometimes refer to an enhancement of a core structure $\mathcal{J}$ as an enhancement of the corresponding core $J$, or as a $\mathcal{J}$-enhancement. Next, we define a core enhancement structure.

**Definition 9.9 (Core Enhancement Structure)** *Given a subgraph $G$ of $\check{G}$, and a core structure $\mathcal{J} = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$ for the subinstance $I = (G, \Sigma)$ of $\check{I}$ defined by $G$, a* $\mathcal{J}$-enhancement structure *consists of:*

- *a $\mathcal{J}$-enhancement $P$;*
- *an orientation $b_u \in \{-1, 1\}$ for every vertex $u \in V(P) \setminus V(J)$; and*
- *a drawing $\rho'$ of the graph $J' = J \cup P$ with no crossings, such that $\rho'$ is consistent with the rotation system $\Sigma$ and the orientations $b_u$ for all vertices $u \in V(J')$ (here, the orientations of vertices of $J$ are determined by $\mathcal{J}$), and moreover, $\rho'$ is a clean drawing of $J'$ with respect to $\mathcal{J}$.*

Intuitively, in the drawing $\rho'$ of graph $J'$, the drawing of graph $J$ should be identical to $\rho_J$, and the path $P$ should be drawn inside the region $F^*(\rho_J)$. For convenience of notation, we denote a $\mathcal{J}$-enhancement by $\mathcal{A} = \left(P, \{b_u\}_{u \in V(J')}, \rho'\right)$, where $J' = J \cup P$. We will always assume that, for every vertex $u \in V(J)$ its orientation $b_u$ in $\mathcal{A}$ is identical to its orientation in $\mathcal{J}$.

**Promising Set of Paths.** We now define promising sets of paths, that will be used in order to compute an enhancement of a given core structure.

**Definition 9.10 (Promising Set of Paths)** *Let $G$ be a subgraph of $\check{G}$, let $I = (G, \Sigma)$ be the subinstance of $\check{I}$ defined by $G$, let $\mathcal{J} = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$ be a core structure for $I$, and let $\mathcal{P}$ be a collection of simple edge-disjoint paths in $G$, that are internally disjoint from $J$. We say that $\mathcal{P}$ is a* promising set of paths*, if there is a partition $(E_1, E_2)$ of the edges of $\delta_G(J)$, such that the edges of $E_1$ appear consecutively in the ordering $\mathcal{O}(J)$, and every path in $\mathcal{P}$ has an edge of $E_1$ as its first edge, and an edge of $E_2$ as its last edge.*

We note that some paths in a promising path set may be cycles. We show an efficient algorithm to compute a large set of promising paths for an instance $I$ whose corresponding contracted instance is wide. The proof of the following claim is standard, and it is deferred to Section H.1 of Appendix.

**Claim 9.11** *There is an efficient algorithm that takes as input a subgraph $G \subseteq \check{G}$ and a core structure $\mathcal{J} = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$ for the subinstance $I = (G, \Sigma)$ of $\check{I}$ defined by graph $G$, such that the following properties hold:*

- *for every vertex $v \in V(G)$ with $\deg_G(v) \geq \frac{\hat{m}(I)}{\mu^4}$, there is a collection $\mathcal{Q}(v)$ of at least $\frac{2\hat{m}(I)}{\mu^{50}}$ edge-disjoint paths in $G$ connecting $v$ to the vertices of $J$; and*
- *the $\mathcal{J}$-contracted subinstance $\hat{I}$ of $I$ is wide.*

*The algorithm computes a promising set of paths for $I$ and $\mathcal{J}$, of cardinality $\left\lfloor \frac{\hat{m}(I)}{\mu^{50}} \right\rfloor$.*

### 9.1.5 Splitting a Core Structure and an Instance via an Enhancement Structure

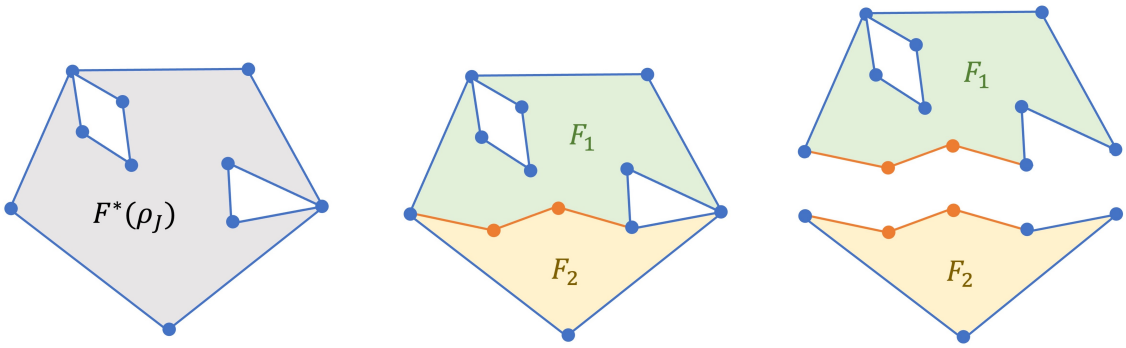
**Splitting the Core Structure.** Suppose we are given a subgraph $G$ of $\check{G}$, a core structure $\mathcal{J} = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$ for the subinstance $I = (G, \Sigma)$ of $\check{I}$ defined by $G$, and an enhancement structure $\mathcal{A} = \left(P, \{b_u\}_{u \in V(J')}, \rho'\right)$ for $\mathcal{J}$, where $J' = J \cup P$. We now show an efficient algorithm that, given $\mathcal{J}$ and $\mathcal{A}$, splits the core structure $\mathcal{J}$ into two new core structures, $\mathcal{J}_1$ and $\mathcal{J}_2$, using the enhancement structure $\mathcal{A}$. We refer to $(\mathcal{J}_1, \mathcal{J}_2)$ as a *split* of the core structure $\mathcal{J}$ via the enhancement structure $\mathcal{A}$.

We let $\rho'$ be the drawing of the graph $J'$ on the sphere given by the enhancement structure $\mathcal{A}$. Recall that there is a disc $D(J)$ that contains the image of $J$ in $\rho'$, whose drawing in $D(J)$ is identical to $\rho_J$. Additionally, all vertices and edges of $P$ must be drawn in region $F^*(\rho_J)$ of $\rho'$. Therefore, in the drawing $\rho'$ of $J'$, there are two faces incident to the image of the path $P$, that we denote by $F_1$ and $F_2$, respectively, and $F_1 \cup F_2 = F^*(\rho_J)$ holds. We let $J_1 \subseteq J'$ be the graph containing all vertices and edges, whose images lie on the boundary of face $F_1$ in $\rho'$, and we define graph $J_2 \subseteq J'$ similarly for face $F_2$.

We now define the core structure $\mathcal{J}_1$, whose corresponding core graph is $J_1$. For every vertex $u \in V(J_1)$, its orientation $b_u$ is the same as in $\mathcal{A}$. The drawing $\rho_{J_1}$ of $J_1$ is defined to be the drawing of $J_1$ induced by the drawing $\rho'$ of $J'$. Note that $F_1$ remains a face in the drawing $\rho_{J_1}$. We then let $F^*(\rho_{J_1}) = F_1$. The definition of the core structure $\mathcal{J}_2$ is symmetric, except that we use core $J_2$ instead of $J_1$ and face $F_2$ instead of $F_1$ (see Figure 31 for an illustration). This completes the description of the algorithm for computing a split $(\mathcal{J}_1, \mathcal{J}_2)$ of the core structure $\mathcal{J}$ via the enhancement structure $\mathcal{A}$.

Next, we define a split of an instance $I$ along a core enhancement structure $\mathcal{A}$.

**Definition 9.12 (Splitting an Instance along an Enhancement Structure)** *Let $G$ be a subgraph of $\check{G}$, let $\mathcal{J} = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$ be a core structure for the subinstance $I = (G, \Sigma)$ of $\check{I}$*




(a) Before the split. Core $J$ is shown in blue and face $F^*(\rho_J)$ is shown in gray.

(b) Face $F^*(\rho_J)$ is split into faces $F_1$ and $F_2$ by the image of the enhancement path $P$ (shown in orange).

(c) New cores $J_1$ (top) and $J_2$ (bottom).

Figure 31: Splitting a core structure via an enhancement structure.

*defined by $G$, and let $\mathcal{A} = \left(P, \{b_u\}_{u \in V(J')}, \rho'\right)$ be an enhancement structure for $\mathcal{J}$, where $J' = J \cup P$. Let $(\mathcal{J}_1, \mathcal{J}_2)$ be the split of $\mathcal{J}$ via the enhancement structure $\mathcal{A}$, and denote by $J_1, J_2$ the cores of $\mathcal{J}_1$ and $\mathcal{J}_2$, respectively. A* split *of instance $I$ along $\mathcal{A}$ is a pair $I_1 = (G_1, \Sigma_1), I_2 = (G_2, \Sigma_2)$ of instances of* MCNwRS*, for which the following hold.*

- *$V(G_1) \cup V(G_2) = V(G)$ and $E(G_1) \cup E(G_2) \subseteq E(G)$;*
- *every vertex $v \in V(G_1) \cap V(G_2)$ belongs to $V(J_1) \cap V(J_2)$;*
- *instance $I_1$ is the subinstance of $\check{I}$ defined by $G_1$, and instance $I_2$ is the subinstance of $\check{I}$ defined by $G_2$; and*
- *$\mathcal{J}_1$ is a valid core structure for $I_1$, and $\mathcal{J}_2$ is a valid core structure for $I_2$.*

Notice that some edges of graph $G$ may not lie in $E(G_1) \cup E(G_2)$. We informally refer to such edges as *deleted edges*, and we will sometimes denote the set of such deleted edges by $E^{\mathsf{del}}$. Typically we will ensure that $|E^{\mathsf{del}}|$ is quite small.

The following crucial observation shows that clean solutions to instances $I_1$ and $I_2$ can be combined to obtain a clean solution to instance $I$. The proof is deferred to Section H.2 of Appendix.

**Observation 9.13** *There is an efficient algorithm, whose input consists of a subgraph $G$ of $\check{G}$, a core structure $\mathcal{J}$ for the subinstance $I = (G, \Sigma)$ of $\check{I}$ defined by $G$, a $\mathcal{J}$-enhancement structure $\mathcal{A}$, and a split $(I_1, I_2)$ of $I$ along $\mathcal{A}$, together with a clean solution $\varphi_1$ to instance $I_1$ with respect to $\mathcal{J}_1$, and a clean solution $\varphi_2$ to instance $I_2$ with respect to $\mathcal{J}_2$, where $(\mathcal{J}_1, \mathcal{J}_2)$ is the split of $\mathcal{J}$ along $\mathcal{A}$. The algorithm computes a clean solution $\varphi$ to instance $I$ with respect to $\mathcal{J}$, with $\mathsf{cr}(\varphi) \leq \mathsf{cr}(\varphi_1) + \mathsf{cr}(\varphi_2) + |E^{\mathsf{del}}| \cdot |E(G)|$, where $E^{\mathsf{del}} = E(G) \setminus (E(G_1) \cup E(G_2))$.*

### 9.1.6 Auxiliary Claim

We will use the following simple auxiliary claim several times. The proof is similar to the proof of Claim 9.9 in [CMT20] and is deferred to Section H.3 of Appendix.

**Claim 9.14** *Let $I = (G, \Sigma)$ be an instance of* MCNwRS*, and let $\mathcal{P} = \{P_1, \ldots, P_{4k+2}\}$ be a collection of directed simple edge-disjoint paths in $G$, that are non-transversal with respect to $\Sigma$. For all $1 \leq i \leq 4k+2$, let $e_i$ be the first edge on path $P_i$. Assume that there are two distinct vertices $u, v \in V(G)$, such that all paths in $\mathcal{P}$ originate at $u$ and terminate at $v$, and assume further that edges $e_1, \ldots, e_{4k+2}$ appear in this order in the rotation $\mathcal{O}_u \in \Sigma$. Lastly, let $\varphi$ be any solution to instance $I$, such that the number of crossings $(e, e')_p$ in $\varphi$ with $e$ or $e'$ lying in $E(P_1)$ is at most $k$, and assume that the same is true for $E(P_{2k+1})$. Then $\varphi$ does not contain a crossing between an edge of $P_1$ and an edge of $P_{2k+1}$.*

## 9.2 Splitting a Subinstance: Procedure ProcSplit

In this subsection we describe the main subroutine that we use in our algorithm, called ProcSplit. The goal of the subroutine is to split a given subinstance of the input instance $\check{I}$ into two. In order to simplify the statement of the main result of this subsection, we start by defining a valid input and a valid output of the subroutine. Recall that $\check{I} = (\check{G}, \check{\Sigma})$ is the instance of MCNwRS that was obtained by subdividing the instance that serves as input to Theorem 3.13.

**Valid Input for ProcSplit.** A valid input to Procedure ProcSplit consists of a subgraph $G$ of $\check{G}$, a core structure $\mathcal{J} = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$ for the subinstance $I = (G, \Sigma)$ of $\check{I}$ defined by $G$, and a promising set $\mathcal{P}$ of paths for $I$ and $\mathcal{J}$ of cardinality $\left\lfloor \frac{|E(G)|}{\mu^b} \right\rfloor$ for some constant $b$, such that there exists a solution $\varphi$ to instance $I$ that is $\mathcal{J}$-valid, with $\mathsf{cr}(\varphi) \leq \frac{|E(G)|^2}{\mu^{60b}}$ and $|\chi^{\mathsf{dirty}}(\varphi)| \leq \frac{|E(G)|}{\mu^{60b}}$.
We emphasize that the solution $\varphi$ to instance $I$ is not given as part of input and it is not known to the algorithm.

**Valid Output for ProcSplit.** A valid output for ProcSplit consists of a $\mathcal{J}$-enhancement structure $\mathcal{A}$, and a split $(I_1 = (G_1, \Sigma_1), I_2 = (G_2, \Sigma_2))$ of $I$ along $\mathcal{A}$. Let $P^*$ be the enhancement path of $\mathcal{A}$, and let $(\mathcal{J}_1, \mathcal{J}_2)$ be the split of the core structure $\mathcal{J}$ along $\mathcal{A}$. Denote $E^{\mathsf{del}}(I) = E(G) \setminus (E(G_1) \cup E(G_2))$. Let $G' = G \setminus E^{\mathsf{del}}(I)$, and let $I' = (G', \Sigma')$ be the subinstance of $\check{I}$ defined by graph $G'$. We require that the following properties hold:

P1. $|E^{\mathsf{del}}(I)| \leq \frac{2\mathsf{cr}(\varphi) \cdot \mu^{38b}}{m} + |\chi^{\mathsf{dirty}}(\varphi)|$, where $m = |E(G)|$;

P2. $|E(G_1)|, |E(G_2)| \leq |E(G)| - \frac{|E(G)|}{32\mu^b}$; and

P3. there is a $\mathcal{J}$-valid solution $\varphi'$ for instance $I'$ that has the following properties:

- drawing $\varphi'$ is compatible with $\varphi$;
- the images of the edges of $E(J) \cup E(P^*)$ do not cross each other in $\varphi'$;
- $\mathsf{cr}(\varphi') \leq \mathsf{cr}(\varphi)$;
- the number of crossings in which the edges of $P^*$ participate in $\varphi'$ is at most $\frac{\mathsf{cr}(\varphi) \cdot \mu^{12b}}{m}$;
- if we let $\varphi_1$ be the solution to instance $I_1$ induced by $\varphi'$, then drawing $\varphi_1$ is $\mathcal{J}_1$-valid, and similarly, if we let $\varphi_2$ be the solution to instance $I_2$ induced by $\varphi'$, then drawing $\varphi_2$ is $\mathcal{J}_2$-valid.

The main theorem of this subsection summarizes the properties of Procedure ProcSplit.

**Theorem 9.15** *There is an efficient randomized algorithm, that, given a valid input $(G, \mathcal{J}, \mathcal{P})$ to procedure* ProcSplit*, with probability at least $1 - \frac{2^{20}}{\mu^{10b}}$ computes a valid output for the procedure.*

We will refer to the algorithm from Theorem 9.15 as ProcSplit. The remainder of this subsection is dedicated to the proof of Theorem 9.15. Throughout the proof, we denote by $I = (G, \Sigma)$ the subinstance of $\check{I}$ defined by graph $G$, by $\mathcal{J} = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$ the given core structure for $I$, and by $m = |E(G)|$. The algorithm consists of two steps. In the first step, we compute an enhancement $P^*$ of the core structure $\mathcal{J}$ and analyze its properties. In the second step, we complete the construction of the enhancement structure $\mathcal{A}$ and of the split $(I_1, I_2)$ of instance $I$ along $\mathcal{A}$. We now describe each of the steps in turn. In order to simplify the exposition, throughout this subsection, we use "enhancement" and "enhancement structure" when we refer to an enhancement of $\mathcal{J}$ and enhancement structure for $\mathcal{J}$. For simplicity of notation, we also denote the drawing $\rho_J$ of the core $J$ by $\rho$, and the face $F^*(\rho_J)$ of this drawing by $F^*$.

### 9.2.1 Step 1: Computing an Enhancement

We start by discarding from $\mathcal{P}$ all paths that contain more than $32\mu^b$ edges. Since the paths in $\mathcal{P}$ are edge-disjoint, the number of the discarded path is bounded by $\frac{m}{32\mu^b}$, and so $|\mathcal{P}| \geq \frac{15m}{16\mu^b}$ continues to hold.

In order to compute an enhancement, we slightly modify the promising set $\mathcal{P}$ of paths, to ensure that, for every pair $P, P' \in \mathcal{P}$ of distinct paths, for every vertex $v \in (V(P) \cap V(P')) \setminus V(J)$, the intersection of $P$ and $P'$ at vertex $v$ is non-transversal. Throughout, we denote by $(E_1, E_2)$ the partition of the edges of $\delta_G(J)$, with the edges of $E_1$ appearing consecutively in the ordering $\mathcal{O}(J)$, suh that every path $P \in \mathcal{P}$ has an edge of $E_1$ as its first edge and an edge of $E_2$ as its last edge.

In order to modify the set $\mathcal{P}$ of paths, we first subdivide every edge $e \in \delta_G(J)$ with a new vertex $t_e$, denoting $T_1 = \{t_e \mid e \in E_1\}$ and $T_2 = \{t_e \mid e \in E_2\}$. Let $H$ be a new graph obtained from $G$ after we delete the vertices and the edges of $J$ from it, contract all vertices of $T_1$ into a new vertex $s$, and contract all vertices of $T_2$ into a new vertex $t$. We define a rotation system $\tilde{\Sigma}$ for graph $H$ in a natural way: For every vertex $v \in V(H) \setminus \{s, t\}$, its rotation $\mathcal{O}_v$ in $\tilde{\Sigma}$ remains the same as in $\Sigma$. For vertex $s$ its rotation $\mathcal{O}_s \in \tilde{\Sigma}$ is the circular ordering of the edges of $E_1$ induced by the ordering $\mathcal{O}(J)$. Similarly, rotation $\mathcal{O}_t \in \tilde{\Sigma}$ is the circular ordering of the edges of $E_2$ induced by the ordering $\mathcal{O}(J)$.

The set $\mathcal{P}$ of paths in graph $G$ defines a set $\mathcal{Q}$ of $|\mathcal{P}|$ edge-disjoint paths in $H$ that connect $s$ to $t$, and are internally disjoint from $s$ and $t$. We apply the algorithm from Lemma 4.7 to graph $H$ and the set $\mathcal{Q}$ of paths to obtain another set $\mathcal{Q}'$ of $|\mathcal{P}|$ simple edge-disjoint paths in $H$, where every path connects $s$ to $t$ and is internally disjoint from both $s$ and $t$ as before, but now the paths are non-transversal with respect to $\tilde{\Sigma}$. Lastly, path set $\mathcal{Q}'$ naturally defines a collection $\mathcal{P}^*$ of $|\mathcal{P}|$ simple edge-disjoint paths in graph $G$, where every path in $\mathcal{P}^*$ contains an edge of $E_1$ as its first edge, and an edge of $E_2$ as its last edge, and is internally disjoint from $J$. Moreover, for every vertex $u \in V(G) \setminus V(J)$, for every pair $P, P' \in \mathcal{P}^*$ of paths that contain $u$, the intersection of $P$ and $P'$ at $u$ is non-transversal. Notice that $\mathcal{P}^*$ remains a promising set of paths of cardinality $|\mathcal{P}|$. We view the paths in $\mathcal{P}^*$ as being directed from the edges of $E_1$ towards the edges of $E_2$. We denote by $k = |\mathcal{P}^*|$, so $k = |\mathcal{P}| \geq \frac{15m}{16\mu^b}$.

Let $E_1^* \subseteq E_1$ be the subset of edges that belong to the paths of $\mathcal{P}^*$. We denote $E_1^* = \{e_1, \ldots, e_k\}$, where the edges are indexed so that $e_1, \ldots, e_k$ appear in the order of their indices in the ordering $\mathcal{O}(J)$. For all $1 \leq j \leq k$, we denote by $P_j \in \mathcal{P}^*$ the unique path originating at the edge $e_j$. We select an index $\lfloor k/3 \rfloor < j^* < \lceil 2k/3 \rceil$ uniformly at random, and we let $P^* = P_{j^*}$. Notice that $P^*$ is a valid enhancement for the core structure $\mathcal{J}$, and it is either a simple path or a simple cycle. We say that path $P^*$ is *chosen* from set $\mathcal{P}^*$. Notice that the probability that a path of $\mathcal{P}^*$ is chosen to be the enhancement path is at most $\frac{4}{k} \leq \frac{4 \cdot 16\mu^b}{15m} \leq \frac{16\mu^b}{m}$.

We will now define a number of bad events, and we will show that the probability that either of these events happens is low.

**Good Paths and Bad Event $\mathcal{E}_1$.** We need the following definition.

**Definition 9.16 (Good path)** *We say that a path $P \in \mathcal{P}^*$ is* good *if the following hold:*

- *the number of crossings in which the edges of $P$ participate in $\varphi$ is at most $\frac{\mathsf{cr}(\varphi)\cdot\mu^{12b}}{m}$; and*
- *there are no crossings in $\varphi$ between edges of $P$ and edges of $J$.*

*A path that is not good is called a* bad path.

We now bound the number of bad paths in $\mathcal{P}^*$.

**Observation 9.17** *The number of bad paths in $\mathcal{P}^*$ is at most $\frac{4m}{\mu^{12b}}$.*

**Proof:** Since the paths in $\mathcal{P}^*$ are edge-disjoint, and every crossing involves two edges, the number of paths $P \in \mathcal{P}^*$ such that there are more than $\frac{\mathsf{cr}(\varphi)\mu^{12b}}{m}$ crossings in $\varphi$ in which the edges of $P$ participate, is at most $\frac{2m}{\mu^{12b}}$. Additionally, we are guaranteed that $|\chi^{\mathsf{dirty}}(\varphi)| \leq \frac{m}{\mu^{60b}}$. Therefore, the number of paths $P \in \mathcal{P}^*$, for which there is a crossing between an edge of $P$ and an edge of $J$, is bounded by $\frac{m}{\mu^{60b}}$. Overall, the number of bad paths in $\mathcal{P}^*$ is bounded by $\frac{2m}{\mu^{12b}} + \frac{m}{\mu^{60b}} \leq \frac{4m}{\mu^{12b}}$. □

We say that bad event $\mathcal{E}_1$ happens if path $P^*$ is a bad path. Since the number of bad paths is bounded by $\frac{4m}{\mu^{12b}}$, and a path of $\mathcal{P}^*$ is chosen to be the enhancement path with probability at most $\frac{16\mu^b}{m}$, we immediately get the following observation.

**Claim 9.18** $\mathbf{Pr}[\mathcal{E}_1] \leq 64/\mu^{11b}$.

**Heavy and Light Vertices, and Bad Event $\mathcal{E}_2$.** We use a parameter $h = \frac{512\mathsf{cr}(\varphi)\cdot\mu^{26b}}{m}$. We say that a vertex $x \in V(G)$ is *heavy* if at least $h$ paths of $\mathcal{P}^*$ contain $x$; otherwise, we say that $x$ is *light*. Recall that in order to define an enhancement structure using the enhancement $P^*$, we need to define an orientation for every inner vertex of $P^*$. Intuitively, we would like this orientation to be consistent with that in the drawing $\varphi$ (which is not known to us). If $x$ is a heavy vertex lying on $P^*$, then computing such an orientation is not difficult, as we can exploit the paths of $\mathcal{P}^*$ containing $x$ to do so. But if $x$ is a light vertex, we do not have enough information in order to determine its orientation in $\varphi$. To get around this problem, we will simply delete all edges incident to the light vertices of $P^*$, except for the edges of $P^* \cup J$, and then let the orientation of each such vertex be arbitrary. We show below that with high probability, the number of edges that we delete is relatively small.

Specifically, we denote by $E'$ the set of all edges $e$, such that $e$ is incident to some light vertex $x \in V(P^*)$, and $e \notin E(J) \cup E(P^*)$. We say that bad event $\mathcal{E}_2$ happens if $|E'| > \frac{\mathsf{cr}(\varphi)\cdot\mu^{38b}}{m}$. We bound the probability of Event $\mathcal{E}_2$ in the next simple claim.

**Claim 9.19** $\mathbf{Pr}[\mathcal{E}_2] \leq 2^{14}/\mu^{11b}$.

**Proof:** Consider some light vertex $x \in V(G)$. Since $x$ lies on fewer than $h = \frac{512\mathsf{cr}(\varphi)\cdot\mu^{26b}}{m}$ paths of $\mathcal{P}^*$, and each such path is chosen to the enhancement with probability at most $\frac{16\mu^b}{m}$, the probability that $x$ lies in $V(P^*)$ is at most $\frac{512\mathsf{cr}(\varphi)\mu^{26b}}{m} \cdot \frac{16\mu^b}{m} \leq \frac{2^{13}\mathsf{cr}(\varphi)\mu^{27b}}{m^2}$.

Consider now some edge $e = (x, y) \in E(G)$. Edge $e$ may lie in $E'$ only if $x$ is a light vertex lying in $V(P^*)$, or the same is true for $y$. Therefore, the probability that $e \in E'$ is at most $\frac{2^{14}\mathsf{cr}(\varphi)\mu^{27b}}{m^2}$, and $\mathbf{E}[|E'|] \leq \frac{2^{14}\mathsf{cr}(\varphi)\mu^{27b}}{m}$. From Markov's inequality, $\mathbf{Pr}\left[|E'| > \frac{\mathsf{cr}(\varphi)\cdot\mu^{38b}}{m}\right] \leq \frac{2^{14}}{\mu^{11b}}$. □

**Unlucky Paths and Bad Event $\mathcal{E}_3$.** If Event $\mathcal{E}_1$ does not happen, then we are guaranteed that, in drawing $\varphi$, the edges lying on path $P^*$ do not cross the edges of $J$. However, it is still possible that there are crossings between edges that lie on path $P^*$ (that is, the image of path $P^*$ crosses itself). Intuitively, when the image of path $P^*$ crosses itself, then we obtain a loop. If we could show that all vertices lying on this loop are light vertices, then we can "repair" the drawing by straightening the loop. This is since we delete all edges incident to the light vertices lying on $P^*$, except for the edges of $E(P^*) \cup E(J)$. Unfortunately, it may happen that some of the vertices lying on these loops are heavy vertices. This may only happen in some limited circumstances, in which case we say that path $P^*$ is *unlucky*. We now define the notion of unlucky paths, and show that the probability that $P^*$ is unlucky is small.

**Definition 9.20 (Unlucky Paths)** *Let $x \in V(G) \setminus V(J)$ be a vertex, and let $P \in \mathcal{P}^*$ be a good path that contains $x$. Let $e, e'$ be the two edges of $P$ that are incident to $x$. Let $\hat{E}_1(x) \subseteq \delta_G(x)$ be the set of edges $\hat{e} \in \delta_G(x)$, such that $\hat{e}$ lies between $e$ and $e'$ in the rotation $\mathcal{O}_x \in \Sigma$ (in clock-wise orientation), and $\hat{e}$ lies on some good path of $\mathcal{P}^*$. Let $\hat{E}_2(x) \subseteq \delta_G(x)$ be the set of edges $\hat{e} \in \delta_G(x)$, such that $\hat{e}$ lies between $e'$ and $e$ in the rotation $\mathcal{O}_x \in \Sigma$ (in clock-wise orientation), and $\hat{e}$ lies on some good path of $\mathcal{P}^*$ (see Figure 32). We say that path $P$ is* unlucky with respect to vertex $x$ *if either $|\hat{E}_1(x)| < \frac{\mathsf{cr}(\varphi)\mu^{13b}}{m}$ or $|\hat{E}_2(x)| < \frac{\mathsf{cr}(\varphi)\mu^{13b}}{m}$ holds. We say that a path $P \in \mathcal{P}^*$ is an* unlucky path *if there is at least one heavy vertex $x \in V(G) \setminus V(J)$, such that $P$ is unlucky with respect to $x$.*

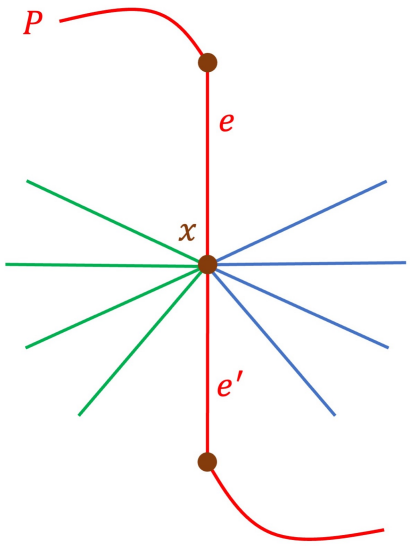



Figure 32: Definition of the sets $\hat{E}_1(x)$ and $\hat{E}_2(x)$ of edges. Path $P$ and its edges $e, e'$ are shown in red. Edges of $\delta(x)$ are depicted according to the circular order $\mathcal{O}_x \in \Sigma$, and the set $\delta(x) \setminus \{e, e'\}$ is split into two subsets (green and blue). Set $\hat{E}_1(x)$ contains every green edge that belongs to some good path of $\mathcal{P}^*$, and set $\hat{E}_2(x)$ contains every blue edge that belongs to some good path of $\mathcal{P}^*$.

We will now show that the total number of good paths in $\mathcal{P}^*$ that are unlucky is small, and we will conclude that the probability that an unlucky path was chosen to be the enhancement path $P^*$ is also small. The proof of the following claim is somewhat technical and is defered to Section H.4 of Appendix.

**Claim 9.21** *For every vertex $x \in V(G) \setminus V(J)$, the total number of good paths in $\mathcal{P}^*$ that are unlucky with respect to $x$ is at most $\frac{512\mathsf{cr}(\varphi) \cdot \mu^{13b}}{m}$.*

We say that bad event $\mathcal{E}_3$ happens if $P^*$ is an unlucky path.

**Claim 9.22** $\mathbf{Pr}[\mathcal{E}_3] \leq 64/\mu^{10b}$.

**Proof:** Recall that a heavy vertex must have degree at least $h$ in $G$. Therefore, the total number of heavy vertices is at most $\frac{2m}{h}$. From Claim 9.21, for every heavy vertex $x \in V(G) \setminus V(J)$, there are at most $\frac{512\mathsf{cr}(\varphi)\cdot\mu^{13b}}{m}$ paths in $\mathcal{P}^*$ that are good and unlucky for $x$. Since $h = \frac{512\mathsf{cr}(\varphi)\cdot\mu^{26b}}{m}$, the total number of good paths in $\mathcal{P}^*$ that are unlucky with respect to some heavy vertex is at most:

$$\frac{2m}{h} \cdot \frac{512\mathsf{cr}(\varphi)\cdot\mu^{13b}}{m} \leq \frac{2m}{\mu^{13b}}.$$

Since the probability that a given path $P \in \mathcal{P}^*$ is selected is at most $\frac{16\mu^b}{m}$, the probability that an unlucky path is selected is at most $\frac{32}{\mu^{12b}}$. □

Recall that we have denoted by $E'$ the set of all edges $e$, such that $e$ is incident to some light vertex $x \in V(P^*)$, and $e \notin E(J) \cup E(P^*)$. Denote $G' = G \setminus E'$, and let $\Sigma'$ be the rotation system for graph $G'$ induced by $\Sigma$. Denote $I' = (G', \Sigma')$, and denote $J' = J \cup P^*$. Solution $\varphi$ to instance $I$ naturally defines a soluton $\varphi'$ to instance $I'$, that is compatible with $\varphi$, with $\mathsf{cr}(\varphi) \leq \mathsf{cr}(\varphi')$. Moreover, if Event $\mathcal{E}_1$ did not happen, then there are no crossings in this drawing between the edges of $E(P^*)$ and the edges of $E(J)$. However, it is possible that this drawing contains crossings between pairs of edges in $E(P^*)$. In the next claim we show that, if events $\mathcal{E}_1$ and $\mathcal{E}_3$ did not happen, then drawing $\varphi$ can be modified to obtain a solution $\varphi'$ to instance $I'$ that is compatible with $\varphi$, in which the edges of $J'$ do not cross each other. The proof of the following claim is deferred to Section H.5 of Appendix.

**Claim 9.23** *Assume that neither of the events $\mathcal{E}_1$ and $\mathcal{E}_3$ happened. Then there is a solution $\varphi'$ to instance $I' = (G', \Sigma')$ that is compatible with $\varphi$, with $\mathsf{cr}(\varphi') \leq \mathsf{cr}(\varphi)$, such that the edges of $E(J')$ do not cross each other in $\varphi'$. Moreover, if $(e, e')_p$ is a crossing in drawing $\varphi'$, then there is a crossing between edges $e$ and $e'$ at point $p$ in drawing $\varphi$.*

We emphasize that drawing $\varphi'$ is derived from drawing $\varphi$ and neither are known to our algorithm. From now on, we fix the solution $\varphi'$ to instance $I' = (G', \Sigma')$ given by Claim 9.23.

**Terrible Vertices and Bad Event $\mathcal{E}_4$.** For a vertex $x \in V(G)$, let $N(x)$ denote the number of paths in $\mathcal{P}^*$ containing $x$. We also denote by $N^{\mathrm{bad}}(x)$ the number of bad paths in $\mathcal{P}^*$ containing $x$, and by $N^{\mathrm{good}}(x)$ the number of good paths in $\mathcal{P}^*$ containing $x$. Next, we define the notion of a terrible vertex.

**Definition 9.24 (Terrible Vertex)** *A vertex $x \in V(G)$ is* terrible *if it is a heavy vertex, and $N^{\mathrm{bad}}(x) \geq N^{\mathrm{good}}(x)/64$.*

We say that a bad event $\mathcal{E}_4$ happens if any vertex of $P^*$ is a terrible vertex. We bound the probability of Event $\mathcal{E}_4$ in the following claim.

**Claim 9.25** $\mathbf{Pr}[\mathcal{E}_4] \leq \frac{2^{18}}{\mu^{10b}}$.

**Proof:** Consider some terrible vertex $x \in V(G)$. Let $\mathcal{P}' \subseteq \mathcal{P}^*$ be the set of all bad paths in $\mathcal{P}^*$ containing $x$, and let $\mathcal{P}'' \subseteq \mathcal{P}^*$ be the set of all good paths in $\mathcal{P}^*$ containing $x$. From the definition of a terrible vertex, $|\mathcal{P}''| \leq 64|\mathcal{P}'|$. Therefore, we can define a mapping $f_x : \mathcal{P}'' \to \mathcal{P}'$, that maps every path in $\mathcal{P}''$ to some path in $\mathcal{P}'$, such that, for every path $P \in \mathcal{P}'$, at most 64 paths of $\mathcal{P}''$ are mapped

to $P$. If, for a pair $P'' \in \mathcal{P}''$, $P' \in \mathcal{P}'$ of paths, $f_x(P'') = P'$, then we say that path $P'$ *tags* path $P''$. Every bad path also tags itself.

Notice that, if path $P \in \mathcal{P}^*$ is a bad path, then the total number of paths that it may tag is bounded by $64|V(P)| \leq 2^{12}\mu^b$ (as every path in $\mathcal{P}^*$ contains at most $32\mu^b + 2$ vertices). Every path of $\mathcal{P}^*$ that contains a terrible vertex is now tagged. Since, from Observation 9.17, the number of bad paths in $\mathcal{P}^*$ is at most $\frac{4m}{\mu^{12b}}$, we conclude that the total number of paths in $\mathcal{P}^*$ that contain a terrible vertex is bounded by $\frac{2^{14}m}{\mu^{11b}}$. Lastly, since the probability that a given path $P \in \mathcal{P}^*$ is selected is at most $\frac{16\mu^b}{m}$, the probability that a path containing a terrible vertex is selected is bounded by:

$$\frac{2^{14}m}{\mu^{11b}} \cdot \frac{16\mu^b}{m} \leq \frac{2^{18}}{\mu^{10b}}.$$

□

**Bad Event $\mathcal{E}$.** Let $\mathcal{E}$ be the bad event that either of the events $\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3, \mathcal{E}_4$ happens. From the Union Bound and Claims 9.18, 9.19, 9.22 and 9.25, $\mathbf{Pr}[\mathcal{E}] \leq \frac{2^{20}}{\mu^{10b}}$.

### 9.2.2 Step 2: Computing the Enhancement Structure and the Split

In this step, we compute an orientation $b_u$ for every vertex $u \in V(P^*) \setminus V(J)$, that is identical to the orientation of $u$ in drawing $\varphi'$ (though drawing $\varphi'$ itself is not known to the algorithm). We will then complete the construction of the enhancement structure $\mathcal{A}$, and compute the split of instance $I$ along $\mathcal{A}$. Throughout, we denote $J' = J \cup P^*$. Let $\rho'$ be the drawing of graph $J'$ that is induced by drawing $\varphi'$ of $G'$. If Event $\mathcal{E}$ did not happen, then drawing $\rho'$ has no crossings, and the image of path $P^*$ is drawn in the region $F^*$. Let $\rho_{J'}$ be the unique drawing of graph $J'$ that has the following properties:

- drawing $\rho_{J'}$ contains no crossings;
- drawing $\rho_{J'}$ obeys the rotation system $\Sigma$, and, for every vertex $u \in V(J)$, the orientation of $u$ in $\rho_{J'}$ is the orientation $b_u$ given by $\mathcal{J}$;
- the drawing of graph $J$ induced by $\rho_{J'}$ is precisely $\rho_J$; and
- the image of path $P^*$ is contained in region $F^*$.

Note that there is a unique drawing $\rho_{J'}$ of $J'$ with the above properties, and it can be computed efficiently. Moreover, if event $\mathcal{E}$ did not happen, then $\rho_{J'} = \rho'$ must hold. The image of path $P^*$ partitions the region $F^*$ of $\rho_{J'}$ into two faces, that we denote by $F_1$ and $F_2$. These two faces define regions in drawing $\varphi'$ of $G'$, that we denote by $F_1$ and $F_2$, as well.

For every vertex $u \in V(J')$, we consider the tiny $u$-disc $D_{\varphi'}(u)$. For every edge $e \in \delta_{G'}(u)$, we denote by $\sigma(e)$ the segment of $\varphi'(e)$ that is drawn inside the disc $D_{\varphi'}(u)$. Let $\tilde{E} = \left(\bigcup_{u \in V(J')} \delta_{G'}(u)\right) \setminus E(J')$. Recall that, from the definiton of a valid core structure, and since the image of path $P^*$ is contained in region $F^*$ of $\varphi'$, for every edge $e \in \tilde{E}$, segment $\sigma(e)$ must be contained in region $F^*$. We partition edge set $\tilde{E}$ into a set $\tilde{E}^{\mathsf{in}}$ of *inner edges* and the set $\tilde{E}^{\mathsf{out}}$ of *outer edges*, as follows. Edge set $\tilde{E}^{\mathsf{in}}$ contains all edges $e \in \tilde{E}$ with $\sigma(e)$ contained in region $F_1$ of $\varphi'$, and $\tilde{E}^{\mathsf{out}}$ contains all remaining edges (so for every edge $e \in \tilde{E}^{\mathsf{out}}$, $\sigma(e)$ is contained in $F_2$). Let $e_1$ be the first edge of $E_1$ in the ordering $\mathcal{O}(J)$. We will assume without loss of generality that $e_1 \in \tilde{E}^{\mathsf{in}}$. We now show an algorithm that correctly computes the orientation of every vertex $u \in V(P^*) \setminus V(J)$ in the drawing $\varphi'$, and the partition $(\tilde{E}^{\mathsf{in}}, \tilde{E}^{\mathsf{out}})$ of the edges of $\tilde{E}$.

Before we describe the algorithm, we recall the definition of the oriented circular ordering $\mathcal{O}(J)$ of the edges of $\delta_G(J)$. In order to define the ordering, we considered the disc $D(J)$ in the drawing $\rho_J$ of $J$. In this drawing, the orientation of every vertex $u \in V(J)$ is the orientation $b_u$ given by the core structure $\mathcal{J}$. We have defined, for every edge $e \in \delta_G(J)$, a point $p(e)$ on the boundary of the disc $D(J)$, and we let $\mathcal{O}(J)$ be the circular ordering of the edges of $\delta_G(J)$, in which the points $p(e)$ corresponding to these edges appear on the boundary of the disc $D(J)$, as we traverse the boundary of the disc $D(J)$ in the clock-wise direction. From the definition of a $\mathcal{J}$-valid drawing, the drawing of the core $J$ induced by $\varphi'$ is identical to $\rho_J$, including its orientation. Additionally, for every vertex $u \in V(J)$, the orientation of $u$ in $\varphi'$ is the orientation $b_u$ given by $\mathcal{J}$.

**Computing Vertex Orientations and the Partition $(\tilde{E}^{\mathsf{in}}, \tilde{E}^{\mathsf{out}})$.** Consider any vertex $u \in V(P^*) \setminus V(J)$. Let $\hat{e}(u)$, $\hat{e}'(u)$ be the two edges of $P^*$ that are incident to $u$, where we assume that $\hat{e}(u)$ appears before $\hat{e}'(u)$ on $P^*$ (we assume that $P^*$ is directed from an edge of $E_1$ to an edge of $E_2$). Edges $\hat{e}(u), \hat{e}'(u)$ partition the edge set $\delta_{G'}(u) \setminus \{\hat{e}(u), \hat{e}'(u)\}$ into two subsets, that we denote by $\hat{E}_1(u)$ and $\hat{E}_2(u)$, each of which appears consecutively in the rotation $\mathcal{O}_u \in \Sigma'$. Note that either (i) $\hat{E}_1(u) \subseteq \tilde{E}^{\mathsf{in}}$ and $\hat{E}_2(u) \subseteq \tilde{E}^{\mathsf{out}}$ holds, or (ii) $\hat{E}_2(u) \subseteq \tilde{E}^{\mathsf{in}}$ and $\hat{E}_1(u) \subseteq \tilde{E}^{\mathsf{out}}$ holds. While we do not know the orientation of vertex $u$ in $\varphi'$, once we fix this orientation, we can efficiently determine which of the above two conditions holds. Therefore, we can assume w.l.o.g. that, if the orientation of $u$ in $\varphi'$ is 1 then $\hat{E}_1(u) \subseteq \tilde{E}^{\mathsf{in}}$ and $\hat{E}_2(u) \subseteq \tilde{E}^{\mathsf{out}}$ hold (as otherwise we can switch the names $\hat{E}_1(u)$ and $\hat{E}_2(u)$).

We now construct edge sets $\tilde{E}_1, \tilde{E}_2$, and fix an orientation $b_u$ for every vertex $u \in V(P^*) \setminus V(J)$. We then show that $\tilde{E}_1 = \tilde{E}^{\mathsf{in}}$, $\tilde{E}_2 = \tilde{E}^{\mathsf{out}}$, and that the orientations of all vertices of $V(P^*) \setminus V(J)$ that we compute are consistent with the drawing $\varphi'$.

Consider the drawing $\rho_{J'}$ of graph $J'$ that we have computed. Using this drawing, we can efficiently determine, for every edge $e \in \tilde{E}$ that is incident to a vertex of $J$, whether $e \in \tilde{E}^{\mathsf{in}}$ or $e \in \tilde{E}^{\mathsf{out}}$ holds. In the former case, we add $e$ to $\tilde{E}_1$, and in the latter case we add it to $\tilde{E}_2$. Notice that, for every path $P \in \mathcal{P}^*$, the first and the last edges of $P$ are already added to either $\tilde{E}_1$ or $\tilde{E}_2$, and so far $\tilde{E}_1 \subseteq \tilde{E}^{\mathsf{in}}$ and $\tilde{E}_2 \subseteq \tilde{E}^{\mathsf{out}}$ holds.

Next, we process every inner vertex $u$ on path $P^*$. Consider any such vertex $u$. If $u$ is a light vertex, then there are exactly two edges that are incident to $u$ in $G'$ – the edges of the path $P^*$. We can then set the orientation $b_u$ of $u$ to be arbitrary, and we can trivially assume that this orientaiton is identical to the orientation of $u$ in $\varphi'$.

Assume now that $u$ is a heavy vertex. In order to establish the orientation of $u$, we let $\mathcal{P}(u)$ contain all paths $P \in \mathcal{P}^* \setminus \{P^*\}$ with $u \in P$. We partition the set $\mathcal{P}(u)$ of paths into four subsets: set $\mathcal{P}_1(u)$ contains all paths $P$ whose first edge lies in $\tilde{E}_1$, and the first edge of $P$ that is incident to $u$ lies in $\hat{E}_1(u)$. Set $\mathcal{P}_2(u)$ contains all paths $P$ whose first edge lies in $\tilde{E}_2$, and the first edge of $P$ that is incident to $u$ lies in $\hat{E}_2(u)$. Similarly, set $\mathcal{P}'_1(u)$ contains all paths $P \in \mathcal{P}(u)$ whose first edge lies in $\tilde{E}_1$ and the first edge that is incident to $u$ lies in $\hat{E}_2(u)$, while set $\mathcal{P}'_2(u)$ contains all paths $P \in \mathcal{P}'(u)$, whose first edge lies in $\tilde{E}'_2$ and the first edge that is incident to $u$ lies in $\hat{E}_1(u)$. We let $w(u) = |\mathcal{P}_1| + |\mathcal{P}_2|$, and $w'(u) = |\mathcal{P}'_1| + |\mathcal{P}'_2|$. If $w(u) \geq w'(u)$, then we set $b_u = 1$, add the edges of $\hat{E}_1(u)$ to $\tilde{E}_1$, and add the edges of $\hat{E}_2(u)$ to $\tilde{E}_2$. Otherwise we set $b_u = -1$, add the edges of $\hat{E}_1(u)$ to $\tilde{E}_2$, and add the edges of $\hat{E}_2(u)$ to $\tilde{E}_1$.

This completes the algorithm for computing the orientations of the inner vertices of $P^*$, and of the partition $(\tilde{E}_1, \tilde{E}_2)$ of the edge set $\tilde{E}$. We use the following claim to show that both are computed correctly.

**Claim 9.26** *Assume that Event $\mathcal{E}$ did not happen. Then for every vertex $u \in V(P^*) \setminus V(J)$, the orientation of $u$ in $\varphi'$ is $b_u$.*

**Proof:** It is now enough to show that, if $u \in V(P^*) \setminus V(J)$ is a heavy vertex, then the orientation of $u$ in $\varphi'$ is $b_u$. We now consider any heavy vertex $u \in V(P^*) \setminus V(J)$.

Recall that we denoted $N(u) = |\mathcal{P}(u)|$, and we have denoted by $N^{\mathrm{bad}}(u)$ and $N^{\mathrm{good}}(u)$ the total number of bad and good paths in $\mathcal{P}(u)$, respectively. Since we have assumed that bad event $\mathcal{E}_4$ did not happen, vertex $u$ is not a terrible vertex, that is, $N^{\mathrm{bad}}(u) < N^{\mathrm{good}}(u)/64$. Since $N(u) = N^{\mathrm{bad}}(u) + N^{\mathrm{good}}(u)$, we get that $N^{\mathrm{bad}}(u) < N(u)/65$.

Assume first that the orientation of vertex $u$ in $\varphi'$ is 1, so $\hat{E}_1(u) \subseteq \tilde{E}^{\mathsf{in}}$ and $\hat{E}_2(u) \subseteq \tilde{E}^{\mathsf{out}}$. We claim that in this case $w(u) > w'(u)$ must hold, and so our algorithm sets $b_u = 1$ correctly. Indeed, assume otherwise. Then $w'(u) \geq N(u)/2$. Let $\mathcal{Q}$ denote the set of all good paths in $\mathcal{P}'_1(u) \cup \mathcal{P}'_2(u)$. Then $|\mathcal{Q}| \geq w'(u) - N^{\mathrm{bad}}(u) \geq N(u)/2 - N(u)/65 \geq N(u)/4 \geq h/4$, since $u$ is a heavy vertex. We now show that, for every path $Q \in \mathcal{Q}$, there must be a crossing between an edge of $Q$ and an edge of $P^*$ in $\varphi'$.

Indeed, consider any path $Q \in \mathcal{Q}$. Since $Q \in \mathcal{P}'_1(u) \cup \mathcal{P}'_2(u)$, either the first edge of $Q$ lies in $\tilde{E}^{\mathsf{in}}$ and the last edge of $Q$ lies in $\tilde{E}^{\mathsf{out}}$, or the opposite is true. Therefore, the image of the path $Q$ must cross the boundary of the region $F_1$. Since path $Q$ is a good path, and it does not contain vertices of $J$ as inner vertices, no inner point of the image of $Q$ in $\varphi'$ may belong to the image of $J$ in $\varphi'$. Since, for every pair $P, P' \in \mathcal{P}^*$ of paths, and for every vertex $v \in (V(P) \cap V(P')) \setminus V(J)$, the intersection of $P$ and $P'$ at $v$ is non-transversal, there must be a crossing between an edge of $Q$ and an edge of $P^*$ in $\varphi'$. But then the edges of $P^*$ participate in at least $\frac{h}{4} \geq \frac{128\mathsf{cr}(\varphi)\cdot\mu^{26b}}{m}$ crossings in $\varphi'$, and hence in $\varphi$. However, since we have assumed that bad event $\mathcal{E}_1$ did not happen, $P^*$ is a good path, and so its edges may participate in at most $\frac{\mathsf{cr}(\varphi)\cdot\mu^{12b}}{m}$ crossings in $\varphi$, a contradiction. Therefore, when the orientation of $u$ in $\varphi'$ is 1, our algorithm correctly sets $b_u = 1$.

In the case where the orientation of $u$ in $\varphi'$ is $-1$, the analysis is symmetric. In this case, we consider the set $\mathcal{Q}' \subseteq \mathcal{P}_1(u) \cup \mathcal{P}_2(u)$ containing all good paths. For each such path $P \in \mathcal{Q}'$, the image of $P$ in $\varphi$ must cross the image of $P^*$. If we assume that $w(u) \geq w'(u)$ in this case, then we reach a contradiction using the same argument as before. Therefore, $w(u) < w'(u)$ must hold, and our algorithm sets $b_u = 1$ correctly. □

We have now obtained an enhancement structure $\mathcal{A} = (P^*, \{b_u\}_{u \in V(J')}, \rho_{J'})$. For every vertex $u \in V(J')$, if $u \in V(J)$, then its orientaiton $b_u$ remains the same as in $\mathcal{J}$, and otherwise we let $b_u$ be the orientation that we have computed above. From the above discussion, if Event $\mathcal{E}$ did not happen, then for every vertex $u \in V(J')$ the orientation $b_u$ is identical to its orientation in $\varphi'$, and $\rho_{J'}$ is the drawing of $J'$ induced by $\varphi'$. We denote by $(\mathcal{J}_1, \mathcal{J}_2)$ the split of $\mathcal{J}$ via the enhancement structure $\mathcal{A}$, where $\mathcal{J}_1$ is the core structure associated with the face $F_1$. We denote $\mathcal{J}_1 = (J_1, \{b_u\}_{u \in V(J_1)}, \rho_{J_1}, F^*(\rho_{J_1}))$, and $\mathcal{J}_2 = (J_2, \{b_u\}_{u \in V(J_2)}, \rho_{J_2}, F^*(\rho_{J_2}))$, where $F^*(\rho_{J_1}) = F_1$ and $F^*(\rho_{J_2}) = F_2$.

**Computing the Split.** We now construct a split of instance $I$ along $\mathcal{A}$. In order to do so, we construct a flow network $H$ as follows. We start with $H = G'$, and then subdivide every edge $e \in \tilde{E}$ with a vertex $t_e$, denoting $T_1 = \left\{t_e \mid e \in \tilde{E}_1\right\}$ and $T_2 = \left\{t_e \mid e \in \tilde{E}_2\right\}$. We delete all vertices of $J'$ and their adjacent edges from the resulting graph, contract all vertices of $T_1$ into a source vertex $s$, and contract all vertices of $T_2$ into a destination vertex $t$. We then compute a minimum $s$-$t$ cut $(A, B)$ in the resulting flow network $H$, and we denote by $E'' = E_H(A, B)$. We use the following claim, whose proof is provided in Section H.6 of Appendix, in order to bound the cardinality of $E''$.

**Claim 9.27** *If Event $\mathcal{E}$ did not happen, then $|E''| \leq \frac{2\mathsf{cr}(\varphi)\cdot\mu^{12b}}{m} + |\chi^{\mathsf{dirty}}(\varphi)|$.*

Let $E^{\mathsf{del}} = E' \cup E''$. If bad event $\mathcal{E}$ did not happen, then $|E'| \leq \frac{\mathsf{cr}(\varphi)\cdot\mu^{38b}}{m}$, and, from Claim 9.27, $|E''| \leq \frac{2\mathsf{cr}(\varphi)\cdot\mu^{12b}}{m} + |\chi^{\mathsf{dirty}}(\varphi)|$. Therefore, overall, if bad event $\mathcal{E}$ did not happen, then $|E^{\mathsf{del}}| \leq \frac{2\mathsf{cr}(\varphi)\cdot\mu^{38b}}{m} + |\chi^{\mathsf{dirty}}(\varphi)|$. Let $G'' = G' \setminus E'' = G \setminus E^{\mathsf{del}}$, let $\Sigma''$ be the rotation system for graph $G''$

induced by $\Sigma$, and let $I'' = (G'', \Sigma'')$ be the resulting instance of MCNwRS. Solution $\varphi'$ to instance $I'$ then naturally induces a solution to instance $I''$, that we denote by $\varphi''$. From Claim 9.23, this solution is compatible with $\varphi$, and $\mathsf{cr}(\varphi'') \leq \mathsf{cr}(\varphi)$. Moreover, if Event $\mathcal{E}$ did not happen, then the number of crosings in which the edges of $P^*$ participate in $\varphi''$ is at most $\frac{\mathsf{cr}(\varphi)\cdot\mu^{12b}}{m}$, and the images of edges of $E(J) \cup E(P^*)$ do not cross each other in $\varphi'$.

We are now ready to define a split $(I_1 = (G_1, \Sigma_1), I_2 = (G_2, \Sigma_2))$ of instance $I$ along the enhacement structure $\mathcal{A}$. In order to do so, we define two sets $A', B'$ of vertices in graph $G'$, as follows. We start with $A' = A \setminus \{s\}$ and $B' = B \setminus \{t\}$, where $(A, B)$ is the cut that we have computed in graph $H$. We then add all vertices of the core $J_1$ to $A'$, and all vertices of the core $J_2$ to $B'$. We let $G_1 = G''[A']$ and $G_2 = G''[B']$. The rotation system $\Sigma_1$ for graph $G_1$ and the rotation system $\Sigma_2$ for graph $G_2$ are induced by $\Sigma$. Let $I_1 = (G_1, \Sigma_1)$ and $I_2 = (G_2, \Sigma_2)$ be the resulting two instances of MCNwRS. It is immediate to verify that $(I_1, I_2)$ is a valid split of instance $I$ along $\mathcal{A}$. Note that $E(G_1) \cup E(G_2) = E(G'')$.

Let $\varphi_1$ be the solution to instance $I_1$ induced by $\varphi''$, and let $\varphi_2$ be the solution to instance $I_2$ induced by $\varphi'$. From our construction and Claim 9.26, if bad event $\mathcal{E}$ did not happen, then drawing $\varphi_1$ of $G_1$ is $\mathcal{J}_1$-valid, and drawing $\varphi_2$ of $G_2$ is $\mathcal{J}_2$-valid.

Lastly, we need the following observation, whose proof appears in Section H.7 of Appendix.

**Observation 9.28** *If Event $\mathcal{E}$ did not happen, then $|E(G_1)|, |E(G_2)| \leq m - \frac{m}{32\mu^b}$.*

We conclude that, if bad event $\mathcal{E}$ did not happen, then our algorithm computes a valid output for Procedure ProcSplit. Since $\mathbf{Pr}[\mathcal{E}] \leq 2^{20}/\mu^{10b}$, this completes the proof of Theorem 9.15.

## 9.3 Phase 1 of the Algorithm

In Phase 1, we compute a collection $\mathcal{I}$ of subinstances of the subdivided instance $\check{I}$ that almost have all required properties, except that we will not be able to guarantee that the sum of the optimal solution costs of the resulting instances is suitably bounded. However, we will ensure that all resulting instances have a convenient structure, that will be utilized in Phase 2, in order to produce the final collection of subinstances of $\check{I}$. The algorithm that is used in Phase 1 is summarized in the following theorem.

**Theorem 9.29** *There is an efficient randomized algorithm, whose input consists of a wide and well-connected instance $\check{I}^* = (\check{G}^*, \check{\Sigma}^*)$, with $\check{m} = |E(\check{G}^*)| \geq \mu^{c'}$, for some large enough constant $c'$. Let $\check{I} = (\check{G}, \check{\Sigma})$ be the coresponding subdivided instance. The algorithm either returns FAIL, or computes a non-empty collection $\mathcal{I}$ of subinstances of $\check{I}$, such that, for every instance $I = (G, \Sigma) \in \mathcal{I}$, $G \subseteq \check{G}$, and $I$ is the subinstance of $\check{I}$ defined by $G$. Additionally, the algorithm computes, for every instance $I \in \mathcal{I}$, a core structure $\mathcal{J}(I)$ for $I$, such that, if we denote, for every instance $I \in \mathcal{I}$, the $\mathcal{J}(I)$-contracted subinstance of $I$ by $\hat{I}$, and let $\hat{\mathcal{I}} = \left\{\hat{I} \mid I \in \mathcal{I}\right\}$, then the following hold:*

- *for every instance $I \in \mathcal{I}$, if the corresponding contracted instance $\hat{I} = (\hat{G}, \hat{\Sigma})$ is a wide instance, then $|E(\hat{G})| \leq \check{m}/\mu$;*
- *$\sum_{\hat{I}=(\hat{G},\hat{\Sigma})\in\hat{\mathcal{I}}} |E(\hat{G})| \leq 2\check{m}$; and*
- *there is an efficient algorithm, called $\mathsf{AlgCombineDrawings}_1$, that, given a solution $\varphi(\hat{I})$ to every instance $\hat{I} \in \hat{\mathcal{I}}$, computes a solution $\check{\varphi}$ to instance $\check{I}$.*

*Moreover, if $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \leq \check{m}^2/\mu^{c'}$, then with probability at least $(1 - 1/\mu^{200})$, all of the following hold:*

1. *the algorithm does not return FAIL;*
2. *for every instance $I \in \mathcal{I}$, there is a solution $\psi(I)$ to $I$, that is $\mathcal{J}(I)$-valid, with $\sum_{I\in\mathcal{I}} \mathsf{cr}(\psi(I)) \leq \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})$, and $\sum_{I\in\mathcal{I}} |\chi^{\mathsf{dirty}}(\psi(I))| \leq \frac{\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})\cdot\mu^{900}}{\check{m}}$; and*
3. *if algorithm $\mathsf{AlgCombineDrawings}_1$ is given as input a solution $\varphi(\hat{I})$ to every instance $\hat{I}\in\hat{\mathcal{I}}$, then the solution $\check{\varphi}$ to instance $\check{I}$ that it computes has cost at most: $\sum_{\hat{I}\in\hat{\mathcal{I}}} \mathsf{cr}(\varphi(\hat{I}))+\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})\cdot\mu^{8000}$.*

If the algorithm from Theorem 9.29 returns a collection $\mathcal{I}$ of subinstances of $\check{I}$ together with a core structure $\mathcal{J}(I)$ for each such subinstance, such that properties (1) – (3) hold, then we say that the algorithm is successful, and otherwise we say that it is unsuccessful. Notice that, if $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \leq \check{m}^2/\mu^{c'}$, then the probability that the algorithm is unsuccessful is bounded by $1/\mu^{200}$.

The remainder of this subsection is dedicated to the proof of Theorem 9.29. The algorithm repeatedly applies Procedure $\mathsf{ProcSplit}$ to subinstances of the input instance $\check{I}$. Throughout, we set the constant $b$ used in Procedure $\mathsf{ProcSplit}$ to $b = 52$. We may not always be able to ensure that the input to Procedure $\mathsf{ProcSplit}$ is valid. We will always ensure that the input consists of a graph $G \subseteq \check{G}$, a core structure $\mathcal{J} = (J, \{b_u\}_{u\in V(J)}, \rho_J, F^*(\rho_J))$ for the subinstance $I$ of $\check{I}$ defined by $G$, and a promising set $\mathcal{P}$ of paths for $I$ and $\mathcal{J}$ of cardinality $\left\lfloor \frac{|E(G)|}{\mu^{52}} \right\rfloor$. But unfortunately we may not be able to ensure that there exists a solution $\varphi$ to instance $I$ that is $\mathcal{J}$-valid, with $\mathsf{cr}(\varphi) \leq \frac{|E(G)|^2}{\mu^{60b}}$ and $|\chi^{\mathsf{dirty}}(\varphi)| \leq \frac{|E(G)|}{\mu^{60b}}$, since drawing $\varphi$ is not given explicitly as part of input. If the input to Procedure $\mathsf{ProcSplit}$ is not valid, then the procedure may fail during its execution. In this case, we will assume that the procedure returned FAIL (we will also say that the procedure fails). It is also possible that the output $(\mathcal{A}, I_1, I_2)$ of the procedure is not a valid output. We can verify efficiently that $\mathcal{A}$ is a valid enhancement structure for $\mathcal{J}$, and that $(I_1, I_2)$ is a valid split of $I$ along $\mathcal{A}$. We can also efficiently verify that Property P2 holds for the resulting output. If we establish that either of these properties does not hold, then we will also assume that the procedure returned FAIL, or that it failed. However, it is possible that all above properties hold for the procedure's output, but properties P1 or P3 do not. As we are unable to efficiently verify these latter two properties, we will say in such a case that the procedure did not fail, but that it was unsuccessful. If the input to procedure $\mathsf{ProcSplit}$ is valid, it is still possible that, with small probability (up to $2^{10}/\mu^{520}$), its output is not valid. As before, if $\mathcal{A}$ is not a valid enhancement structure for $\mathcal{J}$, or $(I_1, I_2)$ is not a valid split of $I$ along $\mathcal{A}$, or Property P2 does not hold (which we can verify efficiently), we will say that the procedure returned FAIL, or that the procedure failed. Otherwise, if all these properties hold but the output of the procedure is not valid, we will say that the application of the procedure was unsuccessful. If the procedure returns a valid output, then we say that its application was successful.

As before, we denote $|E(\check{G}^*)|$ by $\check{m}$. The algorithm for Phase 1 consists of a number of iterations. The input to iteration $j \geq 1$ consists of a collection $\mathcal{I}_j$ of subinstances of instance $\check{I}$, where for every instance $I = (G, \Sigma) \in \mathcal{I}_j$, $G \subseteq \check{G}$, and $I$ is the subinstance of $\check{I}$ defined by $G$. Additionally, for every instance $I \in \mathcal{I}_j$, we are given a core structure $\mathcal{J}(I)$ for $I$. We will ensure that, with high probability, the subinstances in $\mathcal{I}_j$ satisfy the following properties:

A1. for every instance $I = (G, \Sigma) \in \mathcal{I}_j$, either the $\mathcal{J}(I)$-contracted subinstance $\hat{I} = (\hat{G}, \hat{\Sigma})$ of $I$ is narrow, or $|E(\hat{G})| \leq \max\left\{\frac{\check{m}}{\mu}, 2\check{m} - (j-1)\cdot\frac{\check{m}}{32\mu^{53}}\right\}$;

A2. if we denote by $E_j^{\mathsf{del}} = E(\check{G}) \setminus \left(\bigcup_{I=(G,\Sigma)\in\mathcal{I}_j} E(G)\right)$, then $|E_j^{\mathsf{del}}| \leq \frac{j\cdot\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})\cdot\mu^{6000}}{\check{m}}$;

A3. for every instance $I \in \mathcal{I}_j$, there exists a solution $\psi(I)$ to instance $I$ that is $\mathcal{J}(I)$-valid, such that $\sum_{I\in\mathcal{I}_j} \mathsf{cr}(\psi(I)) \leq \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})$, and $\sum_{I\in\mathcal{I}_j} |\chi^{\mathsf{dirty}}(\psi(I))| \leq \frac{j\cdot\mu^{800}\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})}{\check{m}}$;

A4. for every instance $I = (G, \Sigma) \in \mathcal{I}_j$, whose corresponding $\mathcal{J}(I)$-contracted instance $\hat{I} = (\hat{G}, \hat{\Sigma})$ is wide with $|E(\hat{G})| > \check{m}/\mu$, for every vertex $v \in V(G)$ with $\deg_G(v) \geq \frac{\check{m}}{\mu^5}$, there is a collection $\mathcal{Q}(v)$ of at least $\frac{8\check{m}}{\mu^{50}} - |E_j^{\mathsf{del}}|$ edge-disjoint paths in $G$ connecting $v$ to the vertices of the core $J(I)$ associated with the core structure $\mathcal{J}(I)$;

A5. if we denote, for every instance $I \in \mathcal{I}_j$, by $\hat{m}(I)$ the number of edges in the corresponding $\mathcal{J}(I)$-contracted instance $\hat{I}$, then $\sum_{I \in \mathcal{I}_j} \hat{m}(I) \leq 2\check{m}$; and

A6. there is an efficient algorithm, that, given, for every instance $I \in \mathcal{I}_j$, a solution $\varphi(I)$ that is clean with respect to $\mathcal{J}(I)$, constructs a solution $\varphi(\check{I})$ to instance $\check{I}$, of cost at most $\sum_{I \in \mathcal{I}_j} \mathsf{cr}(\varphi(I)) + j \cdot \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \cdot \mu^{6000}$.

Specifically, for all $j \geq 1$, the input to iteration $j$ consists of a collection $\mathcal{I}_j$ of subinstances of $\check{I}$, where for every instance $I = (G, \Sigma)$, $G \subseteq \check{G}$, and $I$ is the subgraph of $\check{I}$ defined by $G$. Additionally, for every instance $I \in \mathcal{I}_j$, we are given a core structure $\mathcal{J}(I)$ for $I$. We denote, for each instance $I = (G, \Sigma) \in \mathcal{I}_j$, the corresponding $\mathcal{J}(I)$-contracted instance by $\hat{I} = (\hat{G}, \hat{\Sigma})$, and we denote by $\hat{m}(I) = |E(\hat{G})|$. We will guarantee that, if Properties A1–A6 hold for input $\mathcal{I}_j$ to iteration $j$, then, with probability at least $1 - \frac{1}{\mu^{400}}$, at the end of the iteration, we obtain a collection $\mathcal{I}_{j+1}$ of subinstances of $\check{I}$, each of which is defined by a subgraph of $\check{G}$, and, for every instance $I \in \mathcal{I}_{j+1}$, a core structure $\mathcal{J}(I)$, for which Properties A1–A6 hold. With the remaining probability, the algorithm may either return FAIL, or produce an output for which some of the properties A1–A6 do not hold. In each of these two cases, we say that the iteration was unsuccessful. If the input $\mathcal{I}_j$ to iteration $j$ does not have properties A1–A6, then it is possible that the algorithm returns FAIL, or it returns output $\mathcal{I}_{j+1}$ for which some of the invariants A1–A6 do not hold. In both of these cases, we say that the iteration was unsuccessful. If the iteration produces output $\mathcal{I}_{j+1}$ for which properties A1–A6 hold, then we say that it was successful. For all $j \geq 1$, we denote by $\tilde{\mathcal{E}}_j$ the bad event that iteration $j$ was unsuccessful. The number of iterations in our algorithm is at most $z = 128 \left\lceil \mu^{53} \right\rceil$. Note that, from Invariant A1, for every instance $I = (G, \Sigma) \in \mathcal{I}_z$, either the corresponding $\mathcal{J}(I)$-contracted instance $\hat{I} = (\hat{G}, \hat{\Sigma})$ is narrow, or $|E(\hat{G})| \leq \check{m}/\mu$. We will ensure that, for all $1 \leq j \leq z$, if $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \leq \check{m}^2/\mu^{c'}$, then $\mathbf{Pr}\left[\tilde{\mathcal{E}}_j \mid \neg\tilde{\mathcal{E}}_1 \wedge \cdots \wedge \neg\tilde{\mathcal{E}}_{j-1}\right] \leq \frac{1}{\mu^{400}}$. This will guarantee that, if $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \leq \check{m}^2/\mu^{c'}$, then with probability at least $1 - 1/\mu^{200}$, Properties A1–A6 hold for $\mathcal{I}_z$.

The input to the first iteration is a set $\mathcal{I}_1$ of instances, consisting of a single instance $\check{I}$. Since instance $\check{I}^*$ is wide, there is at least one vertex $v^*$ with $\deg_{\check{G}}(v^*) \geq \check{m}/\mu^4$. We define a core structure $\mathcal{J}(\check{I}) = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$ associated with instance $\check{I}$ as follows. The core $J$ consists of a single vertex $v^*$, and its orientation $b_v$ is set to be arbitrary (say 1). Drawing $\rho_J$ is the unique trivial drawing of $J$, and face $F^*(\rho_J)$ is the unique face of this drawing. It is easy to verify that Invariants A1–A6 hold for $\mathcal{I}_1$. The only invariant that is not immediate is A4. This invariant follows from the fact that the input instance $\check{I}^*$ is wide and well-connected. Therefore, for every vertex $u$ of $\check{G}$ with $\deg_{\check{G}}(u) \geq \check{m}/\mu^5$, $\deg_{\check{G}^*}(u) \geq \check{m}/\mu^5$ also holds, and there is a collection of at least $\frac{8\check{m}}{\mu^{50}}$ edge-disjoint paths connecting $u$ to $v^*$ in $\check{G}^*$ and hence in $\check{G}$.

We now describe the execution of iteration $j$. Consider an instance $I = (G, \Sigma) \in \mathcal{I}_j$, and its corresponding core structure $\mathcal{J}(I)$. We say that instance $I$ is *inactive* if either the $\mathcal{J}(I)$-contracted subinstance $\hat{I} = (\hat{G}, \hat{\Sigma})$ of $I$ is narrow, or $|E(\hat{G})| \leq \check{m}/\mu$. Otherwise, we say that instance $I$ is *active*. We denote by $\mathcal{I}_j^A$ the set of all active instances in $\mathcal{I}_j$, and by $\mathcal{I}_j^I$ the set of all inactive instances. We start with the set $\mathcal{I}_{j+1}$ containing every instance in $\mathcal{I}_j^I$. We also maintain the set $E_{j+1}^{\mathsf{del}}$ of deleted edges, that is initialized to $E_j^{\mathsf{del}}$. We then process every active instance $I \in \mathcal{I}_j^A$ one by one. We now describe the algorithm for processing one such instance $I = (G, \Sigma)$.

**Processing instance $I = (G, \Sigma) \in \mathcal{I}_j^A$.** Assume that Invariants A1, A3 and A4 hold for $\mathcal{I}_j$. Denote $\mathcal{J}(I) = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$, $|E(G)| = m$, and $|E(\hat{G})| = \hat{m}(I)$. Since instance $I$ is active, $\hat{m}(I) > \check{m}/\mu$. Since $G \subseteq \check{G}$, $m \leq 2\check{m}$. Therefore, $\frac{m}{2\mu} \leq \hat{m}(I) \leq m$.

Consider any vertex $v \in V(G)$ with $\deg_G(v) \geq \frac{\hat{m}(I)}{\mu^4}$. Since $\hat{m}(I) \geq \frac{\check{m}}{\mu}$, we get that $\deg_G(v) = \deg_{\hat{G}}(v) \geq \frac{\check{m}}{\mu^5}$. From Invariant A4, there is a collection $\mathcal{Q}(v)$ of at least $\frac{8\check{m}}{\mu^{50}} - |E_j^{\mathsf{del}}| \geq \frac{8\check{m}}{\mu^{50}} - \frac{j \cdot \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \cdot \mu^{6000}}{\check{m}}$ edge-disjoint paths in $G$ connecting $v$ to vertices of $J$. Since $j \leq z = 128\left\lceil \mu^{53} \right\rceil$, if we assume that $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \leq \check{m}^2/\mu^{c'}$ for a large enough constant $c'$, we get that $|\mathcal{Q}(v)| \geq \frac{4\check{m}}{\mu^{50}} \geq \frac{2\hat{m}(I)}{\mu^{50}}$. We apply the algorithm from Claim 9.11 to instance $I$ and core structure $\mathcal{J}(I)$, to obtain a promising set of paths $\mathcal{P}$, of cardinality $\left\lfloor \frac{\hat{m}(I)}{\mu^{50}} \right\rfloor \geq \left\lfloor \frac{m}{\mu^{52}} \right\rfloor$, since $\hat{m}(I) \geq \frac{m}{2\mu}$. We use the following claim.

**Claim 9.30** *If $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \leq \check{m}^2/\mu^{c'}$ for some large enough constant $c'$, and Invariants A1–A6 hold for $\mathcal{I}_j$, then $(G, \mathcal{J}(I), \mathcal{P})$ is a valid input to Procedure* ProcSplit.

**Proof:** From the invariants it is immediate to verify that $\mathcal{J}(I)$ is a valid core structure for the subinstance $I$ of $\check{I}$ defined by $G$.

Consider the solution $\psi(I)$ to instance $I$, that is given by Invariant A3. This solution is guaranteed to be $\mathcal{J}(I)$-valid. It is enough to verify that $\mathsf{cr}(\psi(I)) \leq \frac{m^2}{\mu^{3120}}$ and $|\chi^{\mathsf{dirty}}(\psi(I))| \leq \frac{m}{\mu^{3120}}$.

Recall that Invariant A3 guarantees that $\sum_{I' \in \mathcal{I}_j} \mathsf{cr}(\psi(I')) \leq \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})$. In particular, $\mathsf{cr}(\psi(I)) \leq \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \leq \frac{\check{m}^2}{\mu^{c'}}$ must hold for a large enough constant $c'$. Since $m \geq \hat{m}(I) \geq \frac{\check{m}}{\mu}$, we get that $\mathsf{cr}(\psi(I)) \leq \frac{m^2}{\mu^{3120}}$ as required. Similarly, Invariant A3 guarantees that $\sum_{I' \in \mathcal{I}_j} |\chi^{\mathsf{dirty}}(\psi(I'))| \leq \frac{j \cdot \mu^{800} \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})}{\check{m}}$. In particular, $|\chi^{\mathsf{dirty}}(\psi(I))| \leq \frac{j \cdot \mu^{800} \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})}{\check{m}}$. Since $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \leq \frac{\check{m}^2}{\mu^{c'}}$ for a large enough constant $c'$, while $j \leq z = 128\left\lceil \mu^{53} \right\rceil$, we get that $|\chi^{\mathsf{dirty}}(\psi(I))| \leq \frac{\check{m}}{\mu^{c'-852}}$. Since $m \geq \hat{m}(I) \geq \frac{\check{m}}{\mu}$, we get that $|\chi^{\mathsf{dirty}}(\psi(I))| \leq \frac{m}{\mu^{3120}}$. □

In order to process instance $I \in \mathcal{I}_j^A$, we apply Procedure ProcSplit to input $(G, \mathcal{J}(I), \mathcal{P})$. If the procedure returns FAIL, then we terminate the algorithm and return FAIL as well. In this case we say that the current iteration failed. Otherwise, the procedure returns a $\mathcal{J}(I)$-enhancement structure $\mathcal{A}$, and a split $(I_1 = (G_1, \Sigma_1), I_2 = (G_2, \Sigma_2))$ of $I$ along $\mathcal{A}$. Let $P^*$ be the enhancement path of $\mathcal{A}$, and let $(\mathcal{J}_1, \mathcal{J}_2)$ be the split of the core structure $\mathcal{J}$ along $\mathcal{A}$. Denote $E^{\mathsf{del}}(I) = E(G) \setminus (E(G_1) \cup E(G_2))$. Let $G' = G \setminus E^{\mathsf{del}}(I)$, let $\Sigma'$ be the rotation system for $G'$ induced by $\Sigma$, and let $I' = (G', \Sigma')$ be the resulting instance of MCNwRS. We add the edges of $E^{\mathsf{del}}(I)$ to set $E_{j+1}^{\mathsf{del}}$, and we add instances $I_1, I_2$ to the collection $\mathcal{I}_{j+1}$ of instances, letting $\mathcal{J}(I_1) = \mathcal{J}_1$ and $\mathcal{J}(I_2) = \mathcal{J}_2$. From the definition of a split of an instance along an enhancement structure, $G_1, G_2 \subseteq G$, $\mathcal{J}_1$ is a valid core structure for $I_1$, and $\mathcal{J}_2$ is a valid core structure for $I_2$. This completes the description of the algorithm for processing an instance $I \in \mathcal{I}_j^A$, and of the $j$th iteration. We now analyze its properties.

We say that iteration $j$ is *good* if, for every instance $I \in \mathcal{I}_j^A$, the algorithm from Claim 9.11, when applied to instance $I$ and core structure $\mathcal{J}(I)$ returned a promising set of paths $\mathcal{P}$ of cardinality $\left\lfloor \frac{\hat{m}(I)}{\mu^{50}} \right\rfloor$, and additionally, the application of Procedure ProcSplit to input $(G, \mathcal{J}(I), \mathcal{P})$ was successful. We use the following claim to show that iteration $j$ is good with high probability.

**Claim 9.31** *If Invariants A1–A6 hold for $\mathcal{I}_j$ and $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \leq \check{m}^2/\mu^{c'}$, then the probability that iteration $j$ is good is at least $1 - 1/\mu^{498}$.*

**Proof:** From the discussion above, if Invariants A1–A6 hold for $\mathcal{I}_j$, and $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \leq \check{m}^2/\mu^{c'}$, then for every instance $I \in \mathcal{I}_j^A$, the algorithm from Claim 9.11, when applied to instance $I$ and

core structure $\mathcal{J}(I)$ returns a promising set of paths $\mathcal{P}(I)$ of cardinality $\left\lfloor \frac{\hat{m}(I)}{\mu^{50}} \right\rfloor$. Additionally, from Claim 9.30, if Invariants A1–A6 hold for $\mathcal{I}_j$, and $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \leq \check{m}^2/\mu^{c'}$, then for every instance $I \in \mathcal{I}_j^A$, $(G, \mathcal{J}(I), \mathcal{P}(I))$ is a valid input to Procedure ProcSplit. In this case, from Theorem 9.15, the probability that Procedure ProcSplit is either unsuccessful or fails, when applied to $(I, \mathcal{J}(I), \mathcal{P}(I))$, is at most $2^{20}/\mu^{520}$. Since, from Invariant A5, $\sum_{I \in \mathcal{I}_j} \hat{m}(I) \leq 2\check{m}$, while for every active instance $I \in \mathcal{I}_j^A$, $\hat{m}(I) \geq \check{m}/\mu$, we get that $|\mathcal{I}_j^A| \leq 2\mu$. From the Union Bound, we conclude that, if Invariants A1–A6 hold, and $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \leq \check{m}^2/\mu^{c'}$, then the probability that iteration $j$ is good is at least $1 - 1/\mu^{498}$. □

Lastly, the next claim allows us to bound the probability of the bad event $\tilde{\mathcal{E}}_z$.

**Claim 9.32** *Assume that Invariants A1–A6 hold for $\mathcal{I}_j$, $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \leq \check{m}^2/\mu^{c'}$, and that iteration $j$ is good. Then the bad event $\tilde{\mathcal{E}}_j$ does not happen.*

**Proof:** Throughout the proof, we assume that Invariants A1–A6 hold for $\mathcal{I}_j$, $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \leq \check{m}^2/\mu^{c'}$, and iteration $j$ is good. From the definition of a split of an instance (see Definition 9.12), for every instance $I = (G, \Sigma) \in \mathcal{I}_{j+1}$, $G \subseteq \check{G}$, and $I$ is the subinstance of $\check{I}$ defined by graph $G$. It is now enough to show that Invariants A1–A6 continue to hold for the collection $\mathcal{I}_{j+1}$ of instances.

We first observe that, for each inactive instance $I \in \mathcal{I}_j^I$, invariant A1 continue to hold for $I$, and $\hat{m}(I)$ does not change.

Consider now some active instance $I = (G, \Sigma) \in \mathcal{I}_j^A$, and let $(I_1 = (G_1, \Sigma_1), I_2 = (G_2, \Sigma_2))$ be the split of $I$ that was computed by Procedure ProcSplit. We also let $\mathcal{A}$ be the core enhancement structure computed by the procedure, and we let $(\mathcal{J}_1, \mathcal{J}_2)$ be the split of the core structure $\mathcal{J}(I)$ via $\mathcal{A}$. Note that Property P2 of a valid output for Procedure ProcSplit ensures that $|E(G_1)|, |E(G_2)| \leq |E(G)| - \frac{|E(G)|}{32\mu^{52}}$. Since $|E(G)| \geq \hat{m}(I) \geq \frac{\check{m}}{\mu}$, we get that $|E(G_1)|, |E(G_2)| \leq |E(G)| - \frac{\check{m}}{32\mu^{53}} \leq 2\check{m} - j \cdot \frac{\check{m}}{32\mu^{53}}$ (from the fact that Property A1 holds for $\mathcal{I}_j$). This establishes Property A1 for $\mathcal{I}_{j+1}$.

Recall that Property P1 of a valid output for Procedure ProcSplit ensures that $|E^{\mathsf{del}}(I)| \leq \frac{2\mathsf{cr}(\psi(I)) \cdot \mu^{2000}}{|E(G)|} + |\chi^{\mathsf{dirty}}(\psi(I))|$. Therefore, we get that:

$$\begin{aligned}
|E^{\mathsf{del}}_{j+1}| &\leq |E^{\mathsf{del}}_j| + \sum_{I=(G,\Sigma) \in \mathcal{I}_j^A} \left( \frac{2\mathsf{cr}(\psi(I)) \cdot \mu^{2000}}{|E(G)|} + |\chi^{\mathsf{dirty}}(\psi(I))| \right) \\
&\leq \frac{j \cdot \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \cdot \mu^{6000}}{\check{m}} + \sum_{I \in \mathcal{I}_j^A} \frac{\mathsf{cr}(\psi(I)) \cdot \mu^{2002}}{\check{m}} + \sum_{I \in \mathcal{I}_j^A} |\chi^{\mathsf{dirty}}(\psi(I))|.
\end{aligned}$$

(we have used the fact that Invariant A2 holds for $\mathcal{I}_j$, and that, for every instance $I = (G, \Sigma) \in \mathcal{I}_j^A$, $|E(G)| \geq \hat{m}(I) \geq \frac{\check{m}}{\mu}$). Recall that, from Invariant A3, $\sum_{I \in \mathcal{I}_j} \mathsf{cr}(\psi(I)) \leq \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})$, and $\sum_{I \in \mathcal{I}_j} |\chi^{\mathsf{dirty}}(\psi(I))| \leq \frac{j \cdot \mu^{800} \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})}{\check{m}} \leq \frac{\mu^{854} \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})}{\check{m}}$ (since $j \leq z = 128 \left\lceil \mu^{53} \right\rceil$). Altogether, we get that $|E^{\mathsf{del}}_{j+1}| \leq \frac{(j+1) \cdot \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \cdot \mu^{6000}}{\check{m}}$, establising Invariant A2 for $\mathcal{I}_{j+1}$.

Next, we establish Invariant A3 for $\mathcal{I}_{j+1}$. For every instance $I = (G, \Sigma) \in \mathcal{I}_j^I$, its solution $\psi(I)$ remains unchanged. Consider now some instance $I = (G, \Sigma) \in \mathcal{I}_j^A$, and the two subinstances $(I_1, I_2)$ of $I$ that Procedure ProcSplit produced. Let $G' = G \setminus E^{\mathsf{del}}(I)$, let $\Sigma'$ be the rotation system for $G'$ induced by $\Sigma$, and let $I' = (G', \Sigma')$ be the resulting instance of MCNwRS. Consider the solution $\varphi'$ for instance $I'$ that is guaranteed by Property P3 of valid output of Procedure ProcSplit. Let $\varphi_1$ be the solution to instance $I_1$ induced by $\varphi'$, and let $\varphi_2$ be the solution to instance $I_2$ induced by $\varphi'$. Property P3 guarantees that $\varphi_1$ is a $\mathcal{J}_1$-valid solution to $I_1$, and $\varphi_2$ is a $\mathcal{J}_2$-valid solution to $I_2$. We implicitly set $\psi(I_1) = \varphi_1$ and $\psi(I_2) = \varphi_2$. From the definition of an instance split, (see Definition 9.12), the only edges that may be shared by graphs $G_1$ and $G_2$ are edges of $E(J) \cup E(P^*)$. Since no pair of

edges in $E(J) \cup E(P^*)$ may cross each other in $\varphi'$, we get that $\mathsf{cr}(\varphi_1) + \mathsf{cr}(\varphi_2) \leq \mathsf{cr}(\varphi')$. Moreover, if $(e, e')_p \in \chi^{\mathsf{dirty}}(\varphi_1)$, then either $(e, e')_p \in \chi^{\mathsf{dirty}}(\varphi')$, or one of the edges $e, e'$ lies on $P^*$. Since the edges of $P^*$ participate in at most $\frac{\mathsf{cr}(\psi(I)) \cdot \mu^{624}}{|E(G)|} \leq \frac{\mathsf{cr}(\psi(I)) \cdot \mu^{625}}{\check{m}}$ crossings (as $|E(G)| \geq \hat{m}(I) \geq \check{m}/\mu$), we get that $|\chi^{\mathsf{dirty}}(\varphi_1)| + |\chi^{\mathsf{dirty}}(\varphi_2)| \leq |\chi^{\mathsf{dirty}}(\psi(I))| + \frac{\mathsf{cr}(\psi(I)) \cdot \mu^{625}}{\check{m}}$.

Overall, we get that:

$$\sum_{I \in \mathcal{I}_{j+1}} \mathsf{cr}(\psi(I)) \leq \sum_{I \in \mathcal{I}_j} \mathsf{cr}(\psi(I)) \leq \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I});$$

and:

$$\begin{aligned}
\sum_{I \in \mathcal{I}_{j+1}} |\chi^{\mathsf{dirty}}(\psi(I))| &\leq \sum_{I \in \mathcal{I}_j} |\chi^{\mathsf{dirty}}(\psi(I))| + \sum_{I \in \mathcal{I}_j^A} \frac{\mathsf{cr}(\psi(I)) \cdot \mu^{625}}{\check{m}} \\
&\leq \frac{j \cdot \mu^{800} \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})}{\check{m}} + \frac{\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \cdot \mu^{625}}{\check{m}} \\
&\leq \frac{(j+1) \cdot \mu^{800} \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})}{\check{m}},
\end{aligned}$$

establishing Invariant A3 for $\mathcal{I}_{j+1}$.

Next, we establish Invariant A4. Consider some instance $\tilde{I} = (\tilde{G}, \tilde{\Sigma}) \in \mathcal{I}_{j+1}$ with $\hat{m}(\tilde{I}) > \check{m}/\mu$, whose corresponding $\mathcal{J}(\tilde{I})$-contracted graph is wide. If $\tilde{I} \in \mathcal{I}_j$, then $\tilde{I}$ is an inactive instance, and so Invariant A4 holds for it. Otherwise, there is some instance $I \in \mathcal{I}_j^A$, such that, if $(I_1, I_2)$ is the split of instance $I$ that we have computed, $\tilde{I} = I_1$ or $\tilde{I} = I_2$ holds. We assume w.l.o.g. that it is the former. We denote $I = (G, \Sigma)$, $I_1 = (G_1, \Sigma_1)$, and we let $J$, $J_1$, and $J_2$ be the cores associated with the core structures $\mathcal{J}(I)$, $\mathcal{J}(I_1)$, and $\mathcal{J}(I_2)$, respectively. Consider now any vertex $v \in V(G_1)$, whose degree in $G_1$ is at least $\frac{\check{m}}{\mu^5}$. Then $\deg_G(v) \geq \frac{\check{m}}{\mu^5}$ must hold as well. From Invariant A4, there is a collection $\mathcal{Q}(v)$ of at least $\frac{8\check{m}}{\mu^{50}} - |E_j^{\mathsf{del}}|$ edge-disjoint paths in $G$ connecting $v$ to the vertices of $J$. We assume w.l.o.g. that the paths in $\mathcal{Q}(v)$ are internally disjoint from $V(J)$. Let $\mathcal{Q}'(v) \subseteq \mathcal{Q}(v)$ be the set of paths that do not contain edges of $E^{\mathsf{del}}(I)$. Clearly, $|\mathcal{Q}'(v)| \geq \frac{8\check{m}}{\mu^{50}} - |E_j^{\mathsf{del}}| - |E^{\mathsf{del}}(I)| \geq \frac{8\check{m}}{\mu^{50}} - |E_{j+1}^{\mathsf{del}}|$. We direct the paths in $\mathcal{Q}'(v)$ from $v$ to the vertices of $V(J)$. Notice that every path $Q \in \mathcal{Q}'(v)$ is contained in graph $G'$. Consider now any such path $Q \in \mathcal{Q}'(v)$. If path $Q$ contains a vertex of $P^*$ as an inner vertex, then we truncate it so it connects $v$ to a vertex of $P^*$, and is internally disjoint from $V(J) \cup V(P^*)$. We claim that the resulting path $Q$ must be contained in graph $G_1$. This is since, from the definition of a split of an instance, $V(G_1) \cup V(G_2) = V(G)$, and every vertex $u \in V(G_1) \cap V(G_2)$ belongs to $V(J_1) \cap V(J_2)$, while $J_1, J_2 \subseteq J \cup P^*$. Since $E(G') = E(G_1) \cup E(G_2)$, we get that every path $Q$ in the resulting set $\mathcal{Q}'(v)$ is contained in graph $G_1$, and it connects $v$ to a vertex of $J_1$. This establishes Invariant A4 for $\mathcal{I}_{j+1}$.

Invariant A5 follows from the fact that, for every instance $I \in \mathcal{I}_j^A$, if $(I_1 = (G_1, \Sigma_1), I_2 = (G_2, \Sigma_2))$ is the split of instance $I$ that we have computed, then $E(G_1) \cap E(G_2) \subseteq E(J_1) \cap E(J_2)$ (since, from definition of a split, every vertex $u \in V(G_1) \cap V(G_2)$ belongs to $V(J_1) \cap V(J_2)$, and since a subdivided instance may not have parallel edges).

It now remains to establish Invariant A6. Assume we are given, for every instance $I' \in \mathcal{I}_{j+1}$, a solution $\varphi(I')$ that is clean with respect to $\mathcal{J}(I')$. Consider any active instance $I = (G, \Sigma) \in \mathcal{I}_j^A$, and let $(I_1 = (G_1, \Sigma_2), I_2 = (G_2, \Sigma_2))$ be the split of $I$ that we have constructed. We apply the algorithm from Observation 9.13 in order to obtain a solution $\varphi(I)$ to instance $I$ that is clean with respect to $\mathcal{J}(I)$, and $\mathsf{cr}(\varphi(I)) \leq \mathsf{cr}(\varphi(I_1)) + \mathsf{cr}(\varphi(I_2)) + |E^{\mathsf{del}}(I)| \cdot |E(G)|$. From Property P1 of a valid output for $\mathsf{ProcSplit}$, $|E^{\mathsf{del}}(I)| \leq \frac{2\mathsf{cr}(\varphi) \cdot \mu^{2000}}{|E(G)|} + |\chi^{\mathsf{dirty}}(\psi(I))|$. Overall, we have now obtained a solution $\varphi(I)$ for

every instance $I \in \mathcal{I}_j$, that is clean with respect to $\mathcal{J}(I)$, such that:

$$\begin{aligned}\sum_{I\in\mathcal{I}_j}\mathsf{cr}(\varphi(I)) &\leq \sum_{I'\in\mathcal{I}_{j+1}}\mathsf{cr}(\varphi(I')) + \sum_{I=(G,\Sigma)\in\mathcal{I}_j^A}|E(G)|\cdot\left(\frac{\mathsf{cr}(\psi(I))\cdot\mu^{2000}}{|E(G)|}+|\chi^{\mathsf{dirty}}(\psi(I))|\right)\\ &\leq \sum_{I'\in\mathcal{I}_{j+1}}\mathsf{cr}(\varphi(I')) + \sum_{I=(G,\Sigma)\in\mathcal{I}_j^A}\mathsf{cr}(\psi(I))\cdot\mu^{2000}+2\check{m}\cdot\sum_{I=(G,\Sigma)\in\mathcal{I}_j^A}|\chi^{\mathsf{dirty}}(\psi(I))|.\end{aligned}$$

From Invariant A3, $\sum_{I\in\mathcal{I}_j}\mathsf{cr}(\psi(I))\leq\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})$, and $\sum_{I\in\mathcal{I}_j}|\chi^{\mathsf{dirty}}(\psi(I))|\leq\frac{j\cdot\mu^{800}\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})}{\check{m}}$. Therefore, altogether:

$$\begin{aligned}\sum_{I\in\mathcal{I}_j}\mathsf{cr}(\varphi(I)) &\leq \sum_{I'\in\mathcal{I}_{j+1}}\mathsf{cr}(\varphi(I'))+\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})\cdot\mu^{2000}+2z\cdot\mu^{800}\cdot\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})\\ &\leq \sum_{I'\in\mathcal{I}_{j+1}}\mathsf{cr}(\varphi(I'))+2\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})\cdot\mu^{2000},\end{aligned}$$

since $z=128\left\lceil\mu^{53}\right\rceil$. We then apply the algorithm that is guaranteed by Invariant A6 to the collection $\mathcal{I}_j$ of instances, to compute a solution $\varphi(\check{I})$ to instance $\check{I}$, whose is at most:

$$\begin{aligned}\sum_{I\in\mathcal{I}_j}\mathsf{cr}(\varphi(I))+j\cdot\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})\cdot\mu^{6000} &\leq \sum_{I'\in\mathcal{I}_{j+1}}\mathsf{cr}(\varphi(I'))+j\cdot\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})\cdot\mu^{6000}+2\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})\cdot\mu^{2000}\\ &\leq \sum_{I'\in\mathcal{I}_{j+1}}\mathsf{cr}(\varphi(I'))+(j+1)\cdot\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})\cdot\mu^{6000}.\end{aligned}$$

□

From Claims 9.31 and 9.32, for all $1\leq j\leq z$, if $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})\leq\check{m}^2/\mu^{c'}$, then $\mathbf{Pr}\left[\tilde{\mathcal{E}}_j\mid\neg\tilde{\mathcal{E}}_1\wedge\cdots\wedge\neg\tilde{\mathcal{E}}_{j-1}\right]\leq 1/\mu^{498}$. Therefore,

$$\mathbf{Pr}\left[\tilde{\mathcal{E}}_z\right]\leq\mathbf{Pr}\left[\tilde{\mathcal{E}}_z\mid\neg\tilde{\mathcal{E}}_1\wedge\cdots\wedge\neg\tilde{\mathcal{E}}_{z-1}\right]+\mathbf{Pr}\left[\tilde{\mathcal{E}}_{z-1}\mid\neg\tilde{\mathcal{E}}_1\wedge\cdots\wedge\neg\tilde{\mathcal{E}}_{z-2}\right]+\ldots+\mathbf{Pr}\left[\tilde{\mathcal{E}}_1\right]\leq z/\mu^{498}\leq 1/\mu^{400},$$

since $z=128\left\lceil\mu^{53}\right\rceil$. If the algorithm did not return FAIL, then we return the set $\mathcal{I}_z$ of subinstances of $\check{I}$, and, for every instance $I\in\mathcal{I}_z$, the corresponding core structure $\mathcal{I}(I)$. Assume that Event $\tilde{\mathcal{E}}_z$ did not happen.

From Invariant A1, we are guaranteed that, for every instance $I\in\mathcal{I}$, if the corresponding contracted instance $\hat{I}=(\hat{G},\hat{\Sigma})$ is a wide instance, then $|E(\hat{G})|\leq\check{m}/\mu$. Invariant A5 ensures that $\sum_{\hat{I}=(\hat{G},\hat{\Sigma})\in\hat{\mathcal{I}}}|E(\hat{G})|\leq 2\check{m}$, and Invariant A6 provides an efficient algorithm for combining clean solutions $\varphi(I)$ to instances in $I\in\mathcal{I}_z$ to obtain a solution $\check{\varphi}$ to instance $\check{I}$. If Event $\tilde{\mathcal{E}}_z$ does not happen, then we are guaranteed that:

$$\mathsf{cr}(\check{\varphi})\leq\sum_{I\in\mathcal{I}_z}\mathsf{cr}(\varphi(I))+z\cdot\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})\cdot\mu^{6000}\leq\sum_{I\in\mathcal{I}_z}\mathsf{cr}(\varphi(I))+\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})\cdot\mu^{6054},$$

(since $z=128\left\lceil\mu^{53}\right\rceil$). Since there is an efficient algorithm that, given, for every instance $I\in\mathcal{I}_z$, a solution $\hat{\varphi}(I)$ to the corresponding $\mathcal{J}(I)$-contracted instance $\hat{I}$, computes a solution $\varphi(I)$ to instance $I$ that is clean with respect to $\mathcal{J}(I)$ with $\mathsf{cr}(\varphi(I))\leq\mathsf{cr}(\hat{\varphi}(I))$, we obtain the desired efficient algorithm for combining solutions to instances in $\hat{\mathcal{I}}$ to obtain a solution to instance $\check{I}$.

Lastly, Invariant A3 ensures that, if Event $\tilde{\mathcal{E}}_z$ did not happen, then, for every instance $I\in\mathcal{I}_z$, there exists a solution $\psi(I)$ to $I$ that is $\mathcal{J}(I)$-valid, such that $\sum_{I\in\mathcal{I}_z}\mathsf{cr}(\psi(I))\leq\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})$, and $\sum_{I\in\mathcal{I}_z}|\chi^{\mathsf{dirty}}(\psi(I))|\leq\frac{\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})\cdot z\mu^{800}}{\check{m}}\leq\frac{\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})\cdot\mu^{900}}{\check{m}}$ (since $z=128\left\lceil\mu^{53}\right\rceil$). Notice also that, if Event $\tilde{\mathcal{E}}_z$ does not happen, then the algorithm does not return FAIL. This completes the proof of Theorem 9.29.

## 9.4 Phase 2 of the Algorithm

The goal of this phase is to "repair" each one of the instances $I \in \mathcal{I}$ computed in the first phase in order to ensure that each such instance has a cheap solution that is clean with respect to the core structure $\mathcal{J}(I)$. This, in turn, will ensure that the corresponding contracted graph has a cheap solution as well. In order to repair an instance $I = (G, \Sigma) \in \mathcal{I}$, we will compute a collection $E^{\mathsf{del}}(I)$ of edges of $G$. We will ensure that no edge of the core $J(I)$ corresponding to the core structure $\mathcal{J}(I)$ lies in $E^{\mathsf{del}}(I)$. We can then define a new instance $I' = (G', \Sigma')$, where $G' = G \setminus E^{\mathsf{del}}(I)$, and $\Sigma'$ is the rotation system for $G'$ that is induced by $\Sigma$. Note that $\mathcal{J}(I)$ remains a valid core structure for $I'$. Our goal is to ensure that, on the one hand, $|E^{\mathsf{del}}(I)|$ is not too large, and, on the other hand, there is a solution $\psi(I')$ to instance $I'$ that is clean with respect to $\mathcal{J}(I')$, and $\mathsf{cr}(\psi(I'))$ is not too large compared to $\mathsf{cr}(\psi(I)) + |\chi^{\mathsf{dirty}}(\psi(I))|^2$, where $\psi(I)$ is the $\mathcal{J}(I)$-valid solution for instance $I$ from Theorem 9.29. We now state the main result of this subsection, summarizing the algorithm for Phase 2.

**Theorem 9.33** *There is an efficient randomized algorithm, whose input consists of a large enough constant $b$, a subinstance $I = (G, \Sigma)$ of $\check{I}$ with $|E(G)| = m$ and $G \subseteq \check{G}$, and a core structure $\mathcal{J}(I)$ for $I$, whose corresponding core is denoted by $J(I)$. The algorithm computes a set $E^{\mathsf{del}}(I) \subseteq E(G) \setminus E(J(I))$ of edges, for which the following hold. Let $G' = G \setminus E^{\mathsf{del}}(I)$, and let $I' = (G', \Sigma')$ be the subinstance of $\check{I}$ defined by $G'$. The algorithm ensures that, if there is a solution $\psi(I)$ to instance $I$ that is $\mathcal{J}(I)$-valid, with $\mathsf{cr}(\psi(I)) \leq m^2/\mu^{240b}$, and $|\chi^{\mathsf{dirty}}(\psi(I))| \leq m/\mu^{240b}$, then with probability at least $1 - 1/\mu^{2b}$, $|E^{\mathsf{del}}(I)| \leq \left(\frac{\mathsf{cr}(\psi(I))}{m} + |\chi^{\mathsf{dirty}}(\psi(I))|\right) \cdot \mu^{O(b)}$, and there is a solution $\psi(I')$ to instance $I'$ that is clean with respect to $\mathcal{J}(I)$, with $\mathsf{cr}(\psi(I')) \leq \left(\mathsf{cr}(\psi(I)) + |\chi^{\mathsf{dirty}}(\psi(I))|^2 + \frac{|\chi^{\mathsf{dirty}}(\psi(I))| \cdot |E(G)|}{\mu^b}\right) \cdot (\log m)^{O(1)}$.*

If there is a solution $\psi(I)$ to instance $I$ that is $\mathcal{J}(I)$-valid, with $\mathsf{cr}(\psi(I)) \leq m^2/\mu^{240b}$, and $|\chi^{\mathsf{dirty}}(\psi(I))| \leq m/\mu^{240b}$, then we let $\psi(I)$ be this solution, and we say that $\psi(I)$ is a good solution to instance $I$. Otherwise, we let $\psi(I)$ be any solution to instance $I$ that is $\mathcal{J}(I)$-valid, and we say that $\psi(I)$ is a bad solution to instance $I$. We say that an application of the algorithm from Theorem 9.33 is *successful*, if (i) $|E^{\mathsf{del}}(I)| \leq \left(\frac{\mathsf{cr}(\psi(I))}{m} + |\chi^{\mathsf{dirty}}(\psi(I))|\right) \cdot \mu^{O(b)}$, and (ii) there is a solution $\psi(I')$ to the resulting instance $I' = (G', \Sigma')$, that is clean with respect to $\mathcal{J}(I)$, with:

$$\mathsf{cr}(\psi(I')) \leq \left(\mathsf{cr}(\psi(I)) + |\chi^{\mathsf{dirty}}(\psi(I))|^2 + \frac{|\chi^{\mathsf{dirty}}(\psi(I))| \cdot |E(G)|}{\mu^b}\right) \cdot (\log m)^{O(1)}.$$

From Theorem 9.33, if there is a good solution $\psi(I)$ to instance $I$, then the algorithm is successful with probability at least $1 - 1/\mu^{2b}$. We provide the proof of the theorem below, after we complete the proof of Theorem 3.13 using it.

### 9.4.1 Completing the Proof of Theorem 3.13

Given an input instance $\check{I}^* = (\check{G}^*, \check{\Sigma}^*)$, we first apply the algorithm from Theorem 9.29 to this input. If the algorithm from Theorem 9.29 fails, then we terminate the algorithm and return FAIL as well. We denote by $\tilde{\mathcal{E}}'_1$ the bad event that the application of this algorithm is unsuccessful. Recall that, from Theorem 9.29, if $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}^*) \leq \check{m}^2/\mu^{c'}$, then $\mathbf{Pr}\left[\tilde{\mathcal{E}}'_1\right] \leq 1/\mu^{200}$, and, if bad event $\mathcal{E}'_1$ does not happen, then the algorithm does not fail. We assume from now on that the algorithm from Theorem 9.29 did not fail. Let $\mathcal{I}$ be the collection of subinstances of $\check{I}$ computed by the algorithm from Theorem 9.29. Recall that, if Event $\tilde{\mathcal{E}}'_1$ did not happen, then for every instance $I \in \mathcal{I}$, there is a solution $\psi(I)$ to $I$, that is $\mathcal{J}(I)$-valid, such that: $\sum_{I \in \mathcal{I}} \mathsf{cr}(\psi(I)) \leq \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})$, and $\sum_{I \in \mathcal{I}} |\chi^{\mathsf{dirty}}(\psi(I))| \leq \frac{\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \cdot \mu^{900}}{\check{m}}$. We let $b$ be a large enough constant, so that $b \geq 4000$. We assume that the parameter $c'$ from the statement of Theorem 3.13 is sufficiently large compared to $b$, for example, $c' \geq 400b$.

Recall that, for every instance $I = (G, \Sigma) \in \mathcal{I}$, we have denoted by $\hat{I} = (\hat{G}, \hat{\Sigma})$ the corresponding $\mathcal{J}(I)$-contracted instance, and by $\hat{m}(I) = E(\hat{G})$. We say that instance $I$ is *small* if $\hat{m}(I) \leq \frac{\check{m}}{\mu^{1000}}$, and otherwise it is *large*. We partition the set $\mathcal{I}$ of instances into two subsets: set $\mathcal{I}_1$ containing all small instances, and set $\mathcal{I}_2$ containing all large instances. We let $\hat{\mathcal{I}}_1 = \left\{\hat{I} \mid I \in \mathcal{I}_1\right\}$ contain the set of all contracted instances corresponding to the instances of $\mathcal{I}_1$, and we define set $\hat{\mathcal{I}}_2$ of instances corresponding to the instances of $\mathcal{I}_2$ similarly. We need the following obsevation.

**Observation 9.34** *Assume that* $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}^*) \leq \check{m}^2/\mu^{c'}$, *and that bad event* $\tilde{\mathcal{E}}'$ *did not happen. Then for every instance* $I = (G, \Sigma) \in \mathcal{I}_2$, *there is a solution* $\psi(I)$ *to instance* $I$ *that is* $\mathcal{J}(I)$*-valid, with* $\mathsf{cr}(\psi(I)) \leq m^2/\mu^{240b}$, *and* $|\chi^{\mathsf{dirty}}(\psi(I))| \leq m/\mu^{240b}$, *where* $m = |E(G)|$.

**Proof:** Assume that $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) = \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}^*) \leq \check{m}^2/\mu^{c'}$, and that bad event $\tilde{\mathcal{E}}'$ did not happen. Recall that the algorithm from Theorem 9.29 ensures that, for every instance $I \in \mathcal{I}$, there is a solution $\psi(I)$ to $I$, that is $\mathcal{J}(I)$-valid, with $\sum_{I \in \mathcal{I}} \mathsf{cr}(\psi(I)) \leq \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})$, and $\sum_{I \in \mathcal{I}} |\chi^{\mathsf{dirty}}(\psi(I))| \leq \frac{\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \cdot \mu^{900}}{\check{m}}$.

Consider now some instance $I = (G, \Sigma) \in \mathcal{I}_2$, and denote $|E(G)| = m$. From the above discussion, $\mathsf{cr}(\psi(I)) \leq \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \leq \frac{\check{m}^2}{\mu^{c'}}$ must hold. Since, from definition of set $\mathcal{I}_2$ of instances, $m \geq \hat{m}(I) > \frac{\check{m}}{\mu^{1000}}$ holds, we get that $\mathsf{cr}(\psi(I)) \leq \frac{m^2}{\mu^{c'-2000}} \leq \frac{m^2}{\mu^{240b}}$, since we can set $c'$ to be a large enough constant. Similarly, $|\chi^{\mathsf{dirty}}(\psi(I))| \leq \frac{\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \cdot \mu^{900}}{\check{m}} \leq \frac{\check{m}}{\mu^{c'-900}}$, since $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \leq \check{m}^2/\mu^{c'}$. Using the fact that $m \geq \frac{\check{m}}{\mu^{1000}}$, we get that:

$$|\chi^{\mathsf{dirty}}(\psi(I))| \leq \frac{m}{\mu^{c'-1900}} \leq \frac{m}{\mu^{240b}}.$$

□

We process every instance $I \in \mathcal{I}_2$ one by one. For each such instance $I$, we apply the algorithm from Theorem 9.33 to instance $I$, core structure $\mathcal{J}(I)$, and the constant parameter $b$ defined above. Let $\tilde{\mathcal{E}}'_2(I)$ be the bad event that this application of the algorithm was unsuccessful.

From Observation 9.34 and Theorem 9.33, if $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}^*) \leq \check{m}^2/\mu^{c'}$, then $\mathbf{Pr}\left[\tilde{\mathcal{E}}'_2(I) \mid \neg\tilde{\mathcal{E}}'_1\right] \leq 1/\mu^{2b}$. We denote by $I' = (G', \Sigma')$ the resulting instance of MCNwRS, and by $\hat{I}'$ the corresponding $\mathcal{J}(I)$-contracted instance. We then denote $\hat{\mathcal{I}}'_2 = \left\{\hat{I}' \mid I \in \mathcal{I}_2\right\}$. The final output of our algorithm is the collection $\mathcal{I}' = \hat{\mathcal{I}}_1 \cup \hat{\mathcal{I}}'_2$ of subinstances of $\check{I}$.

We now verify that the collection $\mathcal{I}'$ of instances has all required properties. First, the algorithm from Theorem 9.29 ensures that, for every instance $I \in \mathcal{I}$, if the corresponding contracted instance $\hat{I} = (\hat{G}, \hat{\Sigma})$ is a wide instance, then $|E(\hat{G})| \leq \check{m}/\mu$. If instance $I$ lies in $\mathcal{I}_2$, and $\hat{I}' = (\hat{G}', \hat{\Sigma}')$ is the $\mathcal{J}(I)$-contracted instance corresponding to $I'$, then $|E(\hat{G}')| \leq |E(\hat{G})|$, and, if $\hat{I}$ is not a wide instance, then neither is $\hat{I}'$. This is since graph $\hat{G}'$ can be obtained from graph $\hat{G}$ by deleting the edges of $E^{\mathsf{del}}(I)$ from it[5]. Therefore, we are guaranteed that, for every instance $I' = (G', \Sigma') \in \mathcal{I}'$, if $I'$ is a wide instance, then $|E(G')| \leq \check{m}/\mu$.

The algorithm from Theorem 9.29 also guarantees that $\sum_{I \in \mathcal{I}} \hat{m}(I) \leq 2\check{m}$. Since, for every instance $I \in \mathcal{I}_2$, the corresponding instance $\hat{I}' = (\hat{G}', \Sigma') \in \hat{\mathcal{I}}'_2$ has $|E(\hat{G}')| \leq \hat{m}(I)$, we get that

[5]This is slightly imprecise, since it is possible that $|E(\hat{G}')| < \hat{m}(I)$. Therefore, a vertex $v$ may be a high-degree vertex for $\hat{G}'$ but not for graph $\hat{G}$. It is therefore possible that $\hat{I}$ is narrow but $\hat{I}'$ is not, due to difference in the parameters $|E(\hat{G}')|$ and $|E(\hat{G})|$. However, we can easily fix this issue by adding $\hat{m}(I) - |E(\hat{G}')|$ new vertices to graph $\hat{G}'$, and connecting each of these vertices to a vertex whose degree in $\hat{G}'$ is smaller than in $\hat{G}$, so that for every vertex $v \in V(\hat{G})$, $\deg_{\hat{G}'}(v) \leq \deg_{\hat{G}}(v)$ and $|E(\hat{G}')| = |E(\hat{G})|$ holds. This ensures that, if $\hat{I}$ is a narrow instance, then so is $\hat{I}'$. Adding degree-1 vertices to an instance of MCNwRS does not increase its optimal solution value.

$\sum_{I'=(G',\Sigma')\in\mathcal{I}'}|E(G')|\leq 2\check{m}$.

Let $\tilde{\mathcal{E}}_2'$ be the bad event that any of the events in $\left\{\tilde{\mathcal{E}}_2'(I)\mid I\in\mathcal{I}_2\right\}$ happened. From the definition of the set $\mathcal{I}_2$ of instances, for all $I\in\mathcal{I}_2$, $\hat{m}(I)\geq\frac{\check{m}}{\mu^{1000}}$. Since, from Theorem 9.29, $\sum_{I\in\mathcal{I}}\hat{m}(I)\leq 2\check{m}$, we get that $|\mathcal{I}_2|\leq 2\mu^{1000}$. Therefore, if $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}^*)\leq\check{m}^2/\mu^{c'}$, then:

$$\mathbf{Pr}\left[\tilde{\mathcal{E}}_2'\mid\neg\tilde{\mathcal{E}}_1'\right]\leq\sum_{I\in\mathcal{I}_2}\mathbf{Pr}\left[\tilde{\mathcal{E}}_2'(I)\mid\neg\tilde{\mathcal{E}}_1'\right]\leq\frac{2\mu^{1000}}{\mu^{2b}}\leq\frac{1}{\mu^4},$$

since $b\geq 4000$. Lastly, we let $\tilde{\mathcal{E}}'$ be the bad event that either of the events $\tilde{\mathcal{E}}_1'$ or $\tilde{\mathcal{E}}_2'$ happened. Then $\mathbf{Pr}\left[\tilde{\mathcal{E}}'\right]\leq\mathbf{Pr}\left[\tilde{\mathcal{E}}_1'\right]+\mathbf{Pr}\left[\tilde{\mathcal{E}}_2'\mid\neg\tilde{\mathcal{E}}_1'\right]$. From the above discussion, if $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}^*)\leq\check{m}^2/\mu^{c'}$, then $\mathbf{Pr}\left[\tilde{\mathcal{E}}'\right]\leq\frac{1}{\mu^{200}}+\frac{1}{\mu^4}\leq\frac{1}{\mu^3}$. We use the following two observations in order to complete the proof of Theorem 3.13.

**Observation 9.35** *Assume that $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}^*)\leq\check{m}^2/\mu^{c'}$, and that event $\tilde{\mathcal{E}}'$ did not happen. Then there is an efficient algorithm, that, given a solution $\varphi(I')$ to every instance $I'\in\mathcal{I}'$, computes a solution $\check{\varphi}$ to instance $\check{I}^*$, with $\mathsf{cr}(\check{\varphi})\leq\sum_{I'\in\mathcal{I}'}\mathsf{cr}(\varphi(I'))+\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}^*)\cdot\mu^{O(1)}$.*

**Proof:** We assume that $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}^*)\leq\check{m}^2/\mu^{c'}$, that Event $\tilde{\mathcal{E}}'$ did not happen, and that we are given a solution $\varphi(I')$ to every instance $I'\in\mathcal{I}'$. We show an efficient algorithm to compute a solution $\check{\varphi}$ to instance $\check{I}$. In order to do so, we consider every instance $I\in\mathcal{I}_2$ one by one, and compute a solution $\varphi(\hat{I})$ to instance $\hat{I}$, from the solution $\varphi(\hat{I}')$ to instance $\hat{I}'$.

Consider now some instance $I=(G,\Sigma)\in\mathcal{I}_2$. Let $\hat{I}=(\hat{G},\hat{\Sigma})$ be the corresponding $\mathcal{J}(I)$-contracted instance, and let $\hat{I}'=(\hat{G}',\hat{\Sigma}')$ be the $\mathcal{J}(I)$-contracted instance corresponding to the instance $I'$. Note that $V(\hat{G})=V(\hat{G}')$ and $E(\hat{G}')=E(\hat{G})\setminus E^{\mathsf{del}}(I)$. We use the algorithm from Lemma 2.9 in order to insert the edges of $E^{\mathsf{del}}(I)$ into the solution $\varphi(\hat{I}')$ to instance $\hat{I}'$, obtaining a solution $\varphi(\hat{I})$ to instance $\hat{I}$, whose cost is at most $\mathsf{cr}(\varphi(\hat{I}'))+|E^{\mathsf{del}}(I)|\cdot|E(\hat{G})|$. Recall that, from Theorem 9.33:

$$|E^{\mathsf{del}}(I)|\leq\left(\frac{\mathsf{cr}(\psi(I))}{|E(G)|}+|\chi^{\mathsf{dirty}}(\psi(I))|\right)\cdot\mu^{O(b)}.$$

Therefore:

$$\begin{aligned}\mathsf{cr}(\varphi(\hat{I}))&\leq\mathsf{cr}(\varphi(\hat{I}'))+\left(\mathsf{cr}(\psi(I))+|\chi^{\mathsf{dirty}}(\psi(I))|\cdot|E(\hat{G})|\right)\cdot\mu^{O(b)}\\&\leq\mathsf{cr}(\varphi(\hat{I}'))+\left(\mathsf{cr}(\psi(I))+|\chi^{\mathsf{dirty}}(\psi(I))|\cdot\check{m}\right)\cdot\mu^{O(b)}.\end{aligned}$$

Lastly, using the algorithm from Theorem 9.29, we obtain a solution $\check{\varphi}$ to instance $\check{I}$, whose cost is bounded by:

$$\begin{aligned}\mathsf{cr}(\check{\varphi})&\leq\sum_{\hat{I}\in\hat{\mathcal{I}}}\mathsf{cr}(\varphi(\hat{I}))+\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})\cdot\mu^{8000}\\&\leq\sum_{I\in\mathcal{I}_1}\mathsf{cr}(\varphi(\hat{I}))+\sum_{I\in\mathcal{I}_2}\left(\mathsf{cr}(\varphi(\hat{I}'))+\mathsf{cr}(\psi(I))\cdot\mu^{O(b)}+|\chi^{\mathsf{dirty}}(\psi(I))|\cdot\check{m}\cdot\mu^{O(b)}\right)+\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})\cdot\mu^{O(1)}\\&=\sum_{I'\in\mathcal{I}'}\mathsf{cr}(\varphi(I'))+\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})\cdot\mu^{O(1)}+\sum_{I\in\mathcal{I}_2}\mathsf{cr}(\psi(I))\cdot\mu^{O(1)}+\sum_{I\in\mathcal{I}_2}|\chi^{\mathsf{dirty}}(\psi(I))|\cdot\check{m}\cdot\mu^{O(1)}.\end{aligned}$$

From Theorem 9.29, if $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}^*)\leq\check{m}^2/\mu^{c'}$, and Event $\tilde{\mathcal{E}}'$ did not happen, then $\sum_{I\in\mathcal{I}}\mathsf{cr}(\psi(I))\leq\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})$, and $\sum_{I\in\mathcal{I}}|\chi^{\mathsf{dirty}}(\psi(I))|\leq\frac{\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})\cdot\mu^{900}}{\check{m}}$. Therefore, we get that $\mathsf{cr}(\check{\varphi})\leq\sum_{I'\in\hat{\mathcal{I}}'}\mathsf{cr}(\varphi(I'))+$

$\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \cdot \mu^{O(1)}$. By suppressing the vertices that were used to subdivide the edges of graph $\check{G}^*$ to obtain graph $\check{G}$, we obtain a solution to the original instanc $\check{I}^*$ of the same cost. □

Lastly, the following observation will complete the proof of Theorem 3.13.

**Observation 9.36** *Assume that* $\mathsf{OPT}_{\mathsf{cnwrs}}(I^*) \leq \check{m}^2/\mu^{c'}$, *and that event* $\tilde{\mathcal{E}}'$ *did not happen. Then* $\sum_{I' \in \mathcal{I}'} \mathsf{OPT}_{\mathsf{cnwrs}}(I') \leq \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}^*) \cdot (\log \check{m})^{O(1)}$.

**Proof:** We bound $\sum_{I \in \mathcal{I}_2} \mathsf{OPT}_{\mathsf{cnwrs}}(\hat{I})$ and $\sum_{I \in \mathcal{I}_1} \mathsf{OPT}_{\mathsf{cnwrs}}(\hat{I}')$ separately.

From Theorem 9.33, if Event $\tilde{\mathcal{E}}'$ did not happen, then, for every instance $I = (G, \Sigma) \in \mathcal{I}_2$, there is a solution $\psi(I')$ to the corresponding instance $I'$, that is clean with respect to $\mathcal{J}(I)$, with $\mathsf{cr}(\psi(I')) \leq \left(\mathsf{cr}(\psi(I)) + |\chi^{\mathsf{dirty}}(\psi(I))|^2 + \frac{|\chi^{\mathsf{dirty}}(\psi(I))| \cdot |E(G)|}{\mu^b}\right) \cdot (\log \check{m})^{O(1)}$. From Observation 9.7, there is a solution to the corresponding contracted instance $\hat{I}'$, of cost at most $\mathsf{cr}(\psi(I'))$. Altogether, we get that:

$$\sum_{I \in \mathcal{I}_2} \mathsf{OPT}_{\mathsf{cnwrs}}(\hat{I}') \leq \sum_{I \in \mathcal{I}_2} \left(\mathsf{cr}(\psi(I)) + |\chi^{\mathsf{dirty}}(\psi(I))|^2 + \frac{|\chi^{\mathsf{dirty}}(\psi(I))| \cdot \check{m}}{\mu^b}\right) \cdot (\log \check{m})^{O(1)}.$$

$$\leq \sum_{I \in \mathcal{I}_2} \mathsf{cr}(\psi(I)) \cdot (\log \check{m})^{O(1)} + \left(\sum_{I \in \mathcal{I}_2} |\chi^{\mathsf{dirty}}(\psi(I))|\right)^2 \cdot (\log \check{m})^{O(1)} + \frac{\check{m} \cdot (\log \check{m})^{O(1)}}{\mu^b} \cdot \left(\sum_{I \in \mathcal{I}_2} |\chi^{\mathsf{dirty}}(\psi(I))|\right).$$

From Theorem 9.29, $\sum_{I \in \mathcal{I}} \mathsf{cr}(\psi(I)) \leq \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})$ and $\sum_{I \in \mathcal{I}} |\chi^{\mathsf{dirty}}(\psi(I))| \leq \frac{\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \cdot \mu^{900}}{\check{m}}$. Additionally, since we have assumed that $\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}^*) = \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \leq \check{m}^2/\mu^{c'}$ for a large enough constant $c'$, $\left(\sum_{I \in \mathcal{I}_2} |\chi^{\mathsf{dirty}}(\psi(I))|\right)^2 \leq \frac{(\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}))^2 \cdot \mu^{1800}}{\check{m}^2} \leq \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})$. Altogether, we get that:

$$\begin{aligned}\sum_{I \in \mathcal{I}_2} \mathsf{OPT}_{\mathsf{cnwrs}}(\hat{I}') &\leq \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \cdot (\log \check{m})^{O(1)} + \frac{\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \cdot \mu^{900} \cdot (\log \check{m})^{O(1)}}{\mu^b} \\ &\leq \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \cdot (\log \check{m})^{O(1)},\end{aligned}$$

since $b \geq 4000$.

Next, we bound $\sum_{I \in \mathcal{I}_1} \mathsf{OPT}_{\mathsf{cnwrs}}(\hat{I})$. Consider some instance $I = (G, \Sigma) \in \mathcal{I}_1$ and the solution $\psi(I)$ that is $\mathcal{J}(I)$-valid. Let $J(I)$ be the core associated with the core structure $\mathcal{J}(I)$. Let $E^{\mathsf{dirty}}(I) \subseteq E(G) \setminus E(J(I))$ be the set of all edges $e$, such that the image of $e$ in $\psi(I)$ crosses the image of some edge of $J(I)$. Let $\hat{I} = (\hat{G}, \hat{\Sigma})$ be the $\mathcal{J}(I)$-contracted instance corresponding to instance $I$.

Denote $G' = G \setminus E^{\mathsf{dirty}}(I)$, and let $\Sigma'$ be the rotation system for graph $G'$ that is induced by $\Sigma$. Observe that $\mathcal{J}(I)$ is a valid core structure for the resulting instance $I' = (G', \Sigma')$. Let $\hat{I}' = (\hat{G}', \hat{\Sigma}')$ be the $\mathcal{J}(I)$-contracted instance associated with $I'$.

Observe that we can easily modify the solution $\psi(I)$ to instance $I$ to obtain a solution $\psi(I')$ to instance $I'$ that is clean with respect to $\mathcal{J}(I)$, with $\mathsf{cr}(\psi(I')) \leq \mathsf{cr}(\psi(I))$. Indeed, denote $\mathcal{J}(I) = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*)$. Let $\psi'(I')$ be the solution to instance $I'$ induced by $\psi(I)$. Since $G' = G \setminus E^{\mathsf{dirty}}(I)$, for every connected component $C$ of $G'$, either the images of all edges and vertices of $C$ in $\psi'(I')$ are contained in the region $F^*$ of the drawing, or the images of all edges and vertices of $C$ in $\psi'(I')$ are disjoint from $F^*$ (note that, if $E(C) \cap \delta_G(J) \neq \emptyset$, then the image of $C$ must be contained in $F^*$, since the image of $C$ must intersect the interior of $F^*$, from the definition of a valid core structure (see Definition 9.3)). If the images of all edges and vertices of $C$ in $\psi'(I')$ are disjoint from $F^*$, then $C \cap J = \emptyset$ must hold, and so we can simply move the image of $C$ to lie in the interior of the region $F^*$ without changing the drawing of $C$ itself, and without introducing any new crossings. Once we move the image of each such connected component to lie inside region $F^*$, we obtain a solution $\psi(I')$ to instance $I'$ that is clean with respect to $\mathcal{J}(I)$, and $\mathsf{cr}(\psi(I')) \leq \mathsf{cr}(\psi(I))$.

From Observation 9.7, there is a solution $\psi(\hat{I}')$ to the contracted instance $\hat{I}'$ with $\mathsf{cr}(\psi(\hat{I}')) \leq \mathsf{cr}(\psi(I')) \leq \mathsf{cr}(\psi(I))$. We use the algorithm from Lemma 2.9 in order to insert the edges of $E^{\mathsf{dirty}}(I)$ into the drawing $\psi(\hat{I}')$ to obtain a solution $\psi(\hat{I})$ of instance $\hat{I}$, with the number of crossings bounded by $\mathsf{cr}(\psi(\hat{I}')) + |E^{\mathsf{dirty}}(I)| \cdot |E(\hat{G})| \leq \mathsf{cr}(\psi(I)) + |E^{\mathsf{dirty}}(I)| \cdot |E(\hat{G})|$. Since $I \in \mathcal{I}_1$, $|E(\hat{G})| \leq \frac{\check{m}}{\mu^{1000}}$, so $\mathsf{cr}(\psi(\hat{I})) \leq \mathsf{cr}(\psi(I)) + |\chi^{\mathsf{dirty}}(I)| \cdot \frac{\check{m}}{\mu^{1000}}$. We then get that:

$$\sum_{I \in \mathcal{I}_1} \mathsf{OPT}_{\mathsf{cnwrs}}(\hat{I}) \leq \sum_{I \in \mathcal{I}_1} \mathsf{cr}(\psi(I)) + \sum_{I \in \mathcal{I}_1} |\chi^{\mathsf{dirty}}(I)| \cdot \frac{\check{m}}{\mu^{1000}}.$$

From Theorem 9.29, $\sum_{I \in \mathcal{I}} \mathsf{cr}(\psi(I)) \leq \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})$ and $\sum_{I \in \mathcal{I}} |\chi^{\mathsf{dirty}}(\psi(I))| \leq \frac{\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \cdot \mu^{900}}{\check{m}}$. Therefore:

$$\sum_{I \in \mathcal{I}_1} \mathsf{OPT}_{\mathsf{cnwrs}}(\hat{I}) \leq \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) + \frac{\check{m}}{\mu^{1000}} \cdot \frac{\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \cdot \mu^{900}}{\check{m}} \leq O(\mathsf{OPT}_{\mathsf{cnwrs}}(\check{I})).$$

Overall, we get that $\sum_{I' \in \mathcal{I}'} \mathsf{OPT}_{\mathsf{cnwrs}}(I') = \sum_{I \in \mathcal{I}_1} \mathsf{OPT}_{\mathsf{cnwrs}}(\hat{I}) + \sum_{I \in \mathcal{I}_2} \mathsf{OPT}_{\mathsf{cnwrs}}(\hat{I}') \leq \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}) \cdot (\log \check{m})^{O(1)} = \mathsf{OPT}_{\mathsf{cnwrs}}(\check{I}^*) \cdot (\log \check{m})^{O(1)}$. □

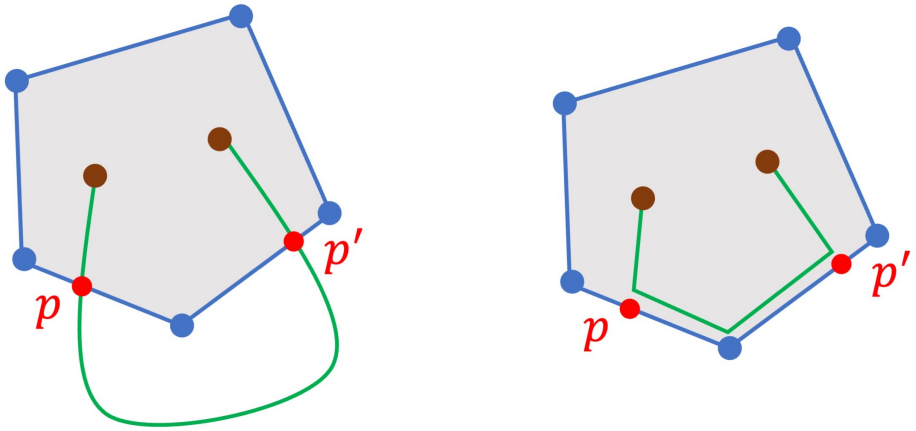
In the remainder of this section we focus on the proof of Theorem 9.33. Throughout the proof, we denote the instance $I = (G, \Sigma)$ that serves as the input to the algorithm by $\check{I}' = (\check{G}', \check{\Sigma}')$, with $|E(\check{G}')|$ denoted by $\check{m}'$. We denote the core structure $\mathcal{J}(I)$ by $\check{\mathcal{J}} = (\check{J}, \{b_u\}_{u \in V(\check{J})}, \rho_{\check{J}}, F^*)$. We can assume that there is a $\check{\mathcal{J}}$-valid solution $\check{\psi}$ to instance $\check{I}'$ with $\mathsf{cr}(\check{\psi}) \leq (\check{m}')^2/\mu^{240b}$, and $|\chi^{\mathsf{dirty}}(\check{\psi})| \leq \check{m}'/\mu^{240b}$, since otherwise we can set $E^{\mathsf{del}}(\check{I}') = E(\check{G}') \setminus E(\check{J})$, which trivially satisfies the requirements of the theorem. From now on we fix a $\check{\mathcal{J}}$-valid solution $\check{\psi}$ to instance $\check{I}'$, with $\mathsf{cr}(\check{\psi}) \leq (\check{m}')^2/\mu^{240b}$ and $|\chi^{\mathsf{dirty}}(\check{\psi})| \leq \check{m}'/\mu^{240b}$. We emphasize that solution $\check{\psi}$ is not known to the algorithm.

### 9.4.2 Proof of Theorem 9.33 – Intuition

For simplicity of exposition, assume that the core $\check{J}$ corresponding to the core structure $\check{\mathcal{J}}$ is a simple cycle. Generally it is not difficult to modify the solution $\check{\psi}$ to instance $\check{I}'$ so that it becomes semi-clean, while only increasing the number of crossings by at most $|\chi^{\mathsf{dirty}}(\check{\psi})|^2$. In order to do so, we let $E^{\mathsf{dirty}}$ be the set of all dirty edges – that is, edges whose image in $\check{\psi}$ crosses the image of some edge of $\check{J}$. Let $\mathcal{C}$ be the set of all connected components of $\check{G}' \setminus E^{\mathsf{dirty}}$. It is easy to verify that for each component $C \in \mathcal{C}$, either the images of all vertices and edges of $C$ in $\check{\psi}$ lie in the region $F^*$; or the images of all vertices and edges of $C$ in $\check{\psi}$ are disjoint from $F^*$. In the latter case, we move the image of $C$ to lie in the interior of the face $F^*$, without changing the image itself. We then need to modify the images of the edges in set $E^{\mathsf{dirty}}$, so that they connect the new images of their endpoints. This can be easily done while introducing at most $|\chi^{\mathsf{dirty}}(\check{\psi})|^2$ new crossings. We do not provide the details here, since we do not use this algorithm eventually.

Let $\check{\psi}'$ denote this semi-clean solution to instance $\check{I}'$ with respect to $\check{\mathcal{J}}$. Denote by $\gamma$ the image of the cycle $\check{J}$ in $\check{\psi}$, which must be a simple closed curve. For convenience, we will now denote by $E^{\mathsf{dirty}}$ the set of all dirty edges of $\check{G}'$ – edges whose image in $\check{\psi}'$ crosses the image of some edge of $\check{J}$. Consider now some dirty edge $e \in E^{\mathsf{dirty}}$. For simplicity of exposition, assume that $e$ is not incident to any vertex of $\check{J}$. Since $\check{\psi}'$ is a semi-clean drawing of $\check{G}'$ with respect to $\check{\mathcal{J}}$, the images of the endpoints of $e$ must lie in region $F^*$. Therefore, there must be at least two points on $\check{\psi}'(e)$ that lie on $\gamma$. We assign the curve $\check{\psi}(e)$ an arbitrary direction, denote by $p$ the first point on $\check{\psi}'(e)$ that lies on $\gamma$, and by $p'$ the last point on $\check{\psi}'(e)$ that lies on $\gamma$. Points $p$ and $p'$ partition the curve $\gamma$ into two disjoint simple open curves, that we denote by $\gamma'$ and $\gamma''$, respectively. A simple way to "repair" the drawing of the edge $e$ so that it no longer crosses the edges of $\check{J}$ would be to replace the segment of $\check{\psi}(e)$ between $p$ and $p'$ with a new segment $\sigma(e)$, that follows the curve $\gamma'$ closely, in the interior of region $F^*$ (see Figure 33).

A problem with this approach is that this may greatly increase the number of crossings, as the segment $\sigma(e)$ may cross many edges in drawing $\check{\psi}'$. Intuitively, the requirements of Theorem 9.33 allow us to add up to $\frac{\check{m}'(\log \check{m}')^{O(1)}}{\mu^b}$ new crossings to the drawing $\check{\psi}$ for each dirty edge whose image we modify, but unfortunately it is possible that, after the modification, $\sigma(e)$ crosses the images of many more edges.




(a) Before: the image of $e$ (green) and its intersections $p, p'$ (red) with the image of $\check{J}$ (blue). Region $F^*$ is shown in gray.

(b) After: the new image of $e$ is shown in green.

Figure 33: Repairing the image of an edge $e \in E^{\mathsf{dirty}}$.

Let $S$ denote the set of all vertices of $\check{J}$ whose images lie on $\gamma'$, and let $T$ be defined similary for $\gamma''$. Consider the minimun cut $(X, Y)$ separating vertices of $S$ from vertices of $T$ in graph $\check{G}' \setminus E^{\mathsf{dirty}}$, so $S \subseteq X$ and $T \subseteq Y$. Assume first that $|E_{\check{G}'}(X, Y)| < \check{m}'/\mu^b$. In this case, we can rearrange the drawing $\check{\psi}'$, so that all vertices and edges of $\check{G}[X] \setminus E^{\mathsf{dirty}}$ are drawn very close to the segment $\gamma'$ (but in the interior of region $F^*$), and similarly all vertices and edges of $\check{G}[Y] \setminus E^{\mathsf{dirty}}$ are drawn very close to the segment $\gamma''$. We can then define a curve $\sigma(e)$ connecting points $p$ and $p'$, so that $\sigma(e)$ is contained in $F^*$, and it only crosses the images of the edges in $E_{\check{G}'}(X, Y)$. Therefore, we can ensure that $\sigma(e)$ participates in few crossings. We can then modify the image of edge $e$ to follow the segment $\sigma(e)$ as before, without increasing the number of crossings by too much.

Note that each dirty edge $e \in E^{\mathsf{dirty}}$ may define a different partition $(S, T)$ of the vertices of $\check{J}$, and a different cut $(X, Y)$. However, if we can ensure that the number of edges crossing each such cut is sufficiently low, then we can still rearrange the drawing $\check{\psi}'$, and modify the drawings of all edges in $E^{\mathsf{dirty}}$, so that they become contained in region $F^*$, while ensuring that the total number of crossings only increases moderately.

It is however possible that, for some edge $e \in E^{\mathsf{dirty}}$, and its corresponding partition $(S, T)$ of $V(\check{J})$, the minimum cut separating $S$ from $T$ in $\check{G}'$ contains more than $\check{m}'/\mu^b$ edges. In this case, there must be at least $\lceil \check{m}'/\mu^b \rceil$ edge-disjoint paths in $\check{G}'$ connecting vertices of $S$ to vertices of $T$. We can treat this set of paths as a promising set of paths, that can be used in order to define an enhancement $P$ of the core structure $\check{\mathcal{J}}$, using Procedure ProcSplit. We can also use the procedure in order to compute an enhancement structure $\mathcal{A}$, and a split $(I_1 = (G_1, \Sigma_1), I_2 = (G_2, \Sigma_2))$ of instance $\check{I}'$ along $\mathcal{A}$. Unlike the algorithm from Phase 1, we will not view the resulting two instances $I_1, I_2$ as separate instances. Instead, we will initialize the set $E^{\mathsf{del}}(\check{I}')$ of deleted edges to edge set $E(\check{G}') \setminus (E(G_1) \cup E(G_2))$. We then consider the graph $K = \check{J} \cup P$, that we call a *skeleton* , and fix a planar drawing $\rho_K$ of it (which is uniquely defined). From Property P3 of valid output to Procedure ProcSplit, there is a drawing $\psi$

of graph $\check{G}' \setminus E^{\mathsf{del}}(\check{I}')$, that obeys the rotation system $\check{\Sigma}'$, such that drawing $\psi$ is $\check{\mathcal{J}}$-valid, and the edges of $\check{J} \cup P$ do not cross each other in $\psi$, with $\mathsf{cr}(\psi) \leq \mathsf{cr}(\check{\psi}')$. If we consider the drawing $\rho_{\check{J}}$ of the core $\check{J}$, then the image of path $P$ partitions face $F^*$ into two new faces, that we denote by $F_1$ and $F_2$. Consider the split $(\mathcal{J}_1, \mathcal{J}_2)$ of the core structure $\check{\mathcal{J}}$ along $\mathcal{A}$. The cores $J_1, J_2$ associated with the core structures $\mathcal{J}_1$ and $\mathcal{J}_2$, respectively, serve as the boundaries of the faces $F_1$, $F_2$, respectively, in the drawing $\rho_K$ of graph $K$.

We note that the drawing of instance $I_1$ induced by $\check{\psi}$ is not necessarily semi-clean with respect to $\mathcal{J}_1$, and the same is true regarding instance $I_2$ and core structure $\mathcal{J}_2$. But we could modify $\check{\psi}$ to ensure this property, obtaining a new drawing $\psi'$ of graph $\check{G}' \setminus E^{\mathsf{del}}(\check{I}')$ (though this process would increase the number of crossings by factor $\mathrm{poly}\log \check{m}'$; we ignore this technicality for now). If we now consider some edge $e$, whose image in $\psi'$ crosses the image of some edge in $\check{J}$, then edge $e$ must either lie in graph $G_1$, or in graph $G_2$. Assume w.l.o.g. that it is the former. In the modified drawing $\psi'$ of $\check{G}' \setminus E^{\mathsf{del}}(\check{I}')$, both endpoints of edge $e$ are drawn inside the region $F_1$. The image of $e$ must then cross the boundary of region $F_1$ in at least two points (recall that the boundary of region $F_1$ in $\psi'$ is the image of the core $J_1$). We can again use these two points to define a partition $(S,T)$ of the vertices of $J_1$, and compute a minimum cut $(X,Y)$ separating $S$ from $T$ in $G_1$. As before, if the value of this minimum cut is small, then we can modify the current drawing $\psi'$ locally inside region $F_1$ and modify the image of the edge $e$, so that it is contained in $F_1$, and no longer crosses the edges of $\check{J}$. If the value of this minimum cut is large, then we can again define a promising set of paths for instance $I_1$ and core structure $\mathcal{J}_1$, and then invoke Procedure $\mathsf{ProcSplit}$ in order to further split core structure $\mathcal{J}_1$ and instance $I_1$, thereby adding new edges to set $E^{\mathsf{del}}(\check{I}')$.

At a high level, our algorithm can be thought of as maintaining a single *skeleton* graph $K$ – a planar subgraph of $\check{G}'$ with $\check{J} \subseteq K$, such that, for every edge $e \in E(K)$, graph $K \setminus \{e\}$ is connected. We also maintain a *skeleton structure* $\mathcal{K}$, that, in addition to the skeleton $K$, specifies the orientation $b_u$ of every vertex $u \in V(K)$, and a planar drawing $\rho_K$ of graph $K$ on the sphere. We require that, for every vertex $u \in V(\check{J})$, its orientations in $\mathcal{K}$ and $\check{\mathcal{J}}$ are identical, and that drawing $\rho_K$ of $K$ is clean with respect to core structure $\check{\mathcal{J}}$, and is consistent with rotation system $\Sigma$ and orientations $b_u$ of the vertices $u \in V(K)$. Let $\mathcal{F}(\rho_K)$ be the set of all faces of the drawing $\rho_K$. Since drawing $\rho_K$ is clean with respect to $\check{\mathcal{J}}$, every forbidden face $F \in \mathcal{F}^{\times}(\rho_{\check{J}})$ is also a face of $\mathcal{F}(\rho_K)$. We denote by $\mathcal{F}^{\times}(\rho_K) = \mathcal{F}^{\times}(\rho_{\check{J}})$ the set of all such faces of $\mathcal{F}(\rho_K)$, that we refer to as *forbidden faces* of drawing $\rho_K$. For every face $F \in \mathcal{F}(\rho_K)$, the set of vertices and edges of $K$ lying on its boundary define a core $J_F$. Using the skeleton structure $\mathcal{K}$, we can define a core structure $\mathcal{J}_F$ associated with the core $J_F$. We also maintain, for every face $F \in \mathcal{F}(\rho_K)$, a subgraph $G_F$ of $\check{G}'$. We let $\Sigma_F$ be the rotation system for $G_F$ induced by $\Sigma$, and we let $\mathcal{I}_F = (G_F, \Sigma_F)$ be the resulting instance of $\mathsf{MCNwRS}$. We require that $\mathcal{J}_F$ is a valid core structure for instance $I_F$, and, if face $F$ is a forbidden face, then $G_F = J_F$ holds. We will ensure that, for every pair $F, F' \in \mathcal{F}(\rho_K)$ of distinct faces, every vertex and every edge of $G_F \cap G_{F'}$ belong to $J_F \cap J_{F'}$, and that $E^{\mathsf{del}}(\check{I}') = E(\check{G}') \setminus \left(\bigcup_{F \in \mathcal{F}(\rho_K)} E(G_F)\right)$.

Consider now some face $F \in \mathcal{F}(\rho_K) \setminus \mathcal{F}^{\times}(\rho_K)$. Intuitively (though somewhat imprecisely), we say that the corresponding instance $I_F$ is *acceptable* if, for every parition $(S,T)$ of the vertices of the core $J_F$, where the vertices of $S$ appear consecutively on the cycle $J_F$, there is a small cut in $G_F$ separating $S$ from $T$ (the definition is slightly more involved when $I_F$ is not a simple cycle). If instance $I_F$ is unacceptable, then we will use Procedure $\mathsf{ProcSplit}$ in order to further augment the skeleton and partition instance $I_F$ into two subinstances. Once we reach a state where, for every face $F \in \mathcal{F}(\rho_K)$ of the current skeleton $K$, the corresponding instance $I_F$ is acceptable, we terminate the algorithm. We show that the resulting drawing $\psi$ of graph $\check{G}' \setminus E^{\mathsf{del}}(\check{I}')$ can be modified so that all crossings with the edges of the original core $\check{J}$ are eliminated, and the number of crossings only increases moderately.

### 9.4.3 Proof of Theorem 9.33 – Main Definitions and Notation

In this subsection we define the main notions that we use in the proof of Theorem 9.33, and also state the main lemmas from which the proof of Theorem 9.33 follows.

Suppose $G'$ is any subgraph of $\check{G}'$. Let $\Sigma'$ be the rotation system for $G'$ induced by $\check{\Sigma}'$, and let $I' = (G', \Sigma')$ be the resulting instance of MCNwRS. For brevity, we will say that $I' = (G', \Sigma')$ is the *subinstance of $\check{I}'$ defined by $G'$*. Recal that $\check{G}' \subseteq \check{G}$, so $I'$ is also the subinstance of $\check{I}$ defined by $G'$.

**Planar Drawings and Face Boundaries.** Suppose we are given a planar graph $H$, and a drawing $\varphi$ of $H$ on the sphere with no crossings. As before, we denote by $\mathcal{F}(\varphi)$ the set of all faces of the drawing $\varphi$. For a face $F \in \mathcal{F}(\varphi)$, we denote by $\partial_\varphi(H, F)$ the subgraph of $H$ containing all vertices and edges whose images are contained in the boundary of the face $F$. We omit the subscript $\varphi$ when clear of context. We sometimes say that the vertices and edges of $\partial_\varphi(H, F)$ serve as the boundary of face $F$ in $\varphi$.

**Skeleton and Skeleton Structure.** We now define the central notions that the proof of Theorem 9.33 uses, namely a skeleton and a skeleton structure, that can be thought of as extending the notions of a core and a core structure.

**Definition 9.37 (Skeleton)** *Let $K$ be a subgraph of $\check{G}'$. We say that $K$ is a* skeleton graph*, if $\check{J}' \subseteq K$, and, for every edge $e \in E(K)$, graph $K \setminus \{e\}$ is connected.*

While the definition of the skeleton $K$ is quite general, we will construct the skeleton using a specific procedure. At the beginning of the algorithm, we let $K = \check{J}'$. In every iteration of the algorithm, we augment $K$ by adding to it a simple path (or a cycle) $P$, whose both endpoints belong to $K$, and all inner vertices are disjoint from $K$. Next, we define a skeleton structure.

**Definition 9.38 (Skeleton Structure)** *A skeleton structure $\mathcal{K}$ consists of the following three ingredients:*

- *a skeleton graph $K$;*
- *for every vertex $u \in V(K)$, an orientation $b_u \in \{-1, 1\}$, such that, for every vertex $u \in V(\check{J}')$, its orientation $b_u$ is identical to that given by the core structure $\check{\mathcal{J}}$; and*
- *a drawing $\rho_K$ of graph $K$ on the sphere with no crossings, such that $\rho_K$ obeys the rotation system $\check{\Sigma}'$, the orientation of every vertex $u \in V(K)$ in $\rho_K$ is $b_u$, and drawing $\rho_K$ is clean with respect to $\check{\mathcal{J}}'$.*

Consider now some skeleton structure $\mathcal{K} = (K, \{b_u\}_{u \in V(K)}, \rho_K)$. For brevity of notation, we denote by $\tilde{\mathcal{F}}(\mathcal{K}) = \mathcal{F}(\rho_K)$ the set of all faces in the drawing $\rho_K$. Since drawing $\rho_K$ is clean with respect to $\check{\mathcal{J}}'$, the image of every vertex and every edge of $K$ appears in the region $F^* = F^*(\rho_{\check{J}'})$ of this drawing (including its boundary). Therefore, every forbidden face $F \in \mathcal{F}^\times(\rho_{\check{J}'})$ is also a face of $\tilde{\mathcal{F}}(\mathcal{K})$. We denote by $\tilde{\mathcal{F}}^\times(\mathcal{K}) = \mathcal{F}^\times(\rho_{\check{J}'})$ the set of all such faces, that we refer to as *forbidden faces of drawing* $\rho_K$, and we denote by $\tilde{\mathcal{F}}'(\mathcal{K}) = \tilde{\mathcal{F}}(\mathcal{K}) \setminus \tilde{\mathcal{F}}^\times(\mathcal{K})$ the set of all remaining faces.

Consider some face $F \in \tilde{\mathcal{F}}(\mathcal{K})$, and let $J_F = \partial_{\rho_K}(K, F)$ be the graph consisting of all vertices and edges of $K$ whose images appear on the boundary of face $F$ in drawing $\rho_K$. From the definition of a skeleton graph $K$, graph $J_F$ is a core. We can then define a core structure $\mathcal{J}_F = (J_F, \{b_u\}_{u \in V(J_F)}, \rho_{J_F}, F^*(\rho_{J_F}))$ as follows: for every vertex $u \in V(J_F)$, its orientation $b_u$ remains the same as in $\mathcal{K}$. Drawing $\rho_{J_F}$ of graph $J_F$ is the drawing induced by $\rho_K$. Notice that face $F \in \tilde{\mathcal{F}}(\mathcal{K})$ remains a face in the drawing

$\rho_{J_F}$. Face $F^*(\rho_{J_F})$ is then defined to be the face $F$. (We note that a core structure is generally defined for some specific graph $G'$, for which the properties specified in Definition 9.3 must hold. Therefore, for now we view the core structure $\mathcal{J}_F$ as simply a tuple $(J_F, \{b_u\}_{u\in V(J_F)}, \rho_{J_F}, F^*(\rho_{J_F}))$, and we will later define a subgraph $G_F$ of $\check{G}'$, for which $\mathcal{J}_F$ will be a valid core structure.)

**$\mathcal{K}$-Valid Drawings.** Next, we consider a skeleton structure $\mathcal{K} = (K, \{b_u\}_{u\in V(K)}, \rho_K)$ and a subgraph $G' \subseteq G$ with $K \subseteq G'$. We then define $\mathcal{K}$-valid drawings of graph $G'$, in a natural way.

**Definition 9.39 ($\mathcal{K}$-valid drawings)** *Let $\mathcal{K} = (K, \{b_u\}_{u\in V(K)}, \rho_K)$ be a skeleton structure, let $G'$ be a subgraph of $\check{G}'$ with $K \subseteq G'$, and let $I' = (G', \Sigma')$ be the subinstance of $\check{I}'$ defined by $G'$. We say that a solution $\varphi$ to instance $I'$ is $\mathcal{K}$-valid, if the drawing of the skeleton $K$ induced by $\varphi$ is identical to $\rho_K$, and the orientation of every vertex $u \in V(K)$ in drawing $\varphi$ is identical to the orientation $b_u$ given by $\mathcal{K}$. We also say that a $\mathcal{K}$-valid solution $\varphi$ to instance $I'$ is a* $\mathcal{K}$-valid drawing *of graph $G'$.*

Note that, if $\varphi$ is a $\mathcal{K}$-valid solution to instance $I'$, for any skeleton structure $\mathcal{K}$, then it is also a $\check{\mathcal{J}}$-valid solution to $I'$.

Since we will be considering various core structures $\mathcal{J}_F$ associated with faces $F \in \tilde{\mathcal{F}}(\mathcal{K})$ of a given skeleton structure $\mathcal{K}$, the definition of dirty edges and dirty crossings will change depending on which core structure we consider. As the algorithm pogresses, the drawing $\psi$ that we maintain for the current graph $G' = \check{G}' \setminus E^{\mathsf{del}}(\check{I}')$ will evolve. We need to keep track of the crossings in which the edges of the original core $\check{J}$ participate, and of the edges involved in these crossings.

Therefore, given a subgraph $G' \subseteq \check{G}'$, a skeleton structure $\mathcal{K} = (K, \{b_u\}_{u\in V(K)}, \rho_K)$, and a $\mathcal{K}$-valid solution to subinstance $I' = (G', \Sigma')$ defined by graph $G'$, we denote by $\chi^*(\varphi)$ the set of all crossings $(e, e')_p$ in $\varphi$ where $e$ or $e'$ belong to the core $\check{J}$.

**A $\mathcal{K}$-Decomposition of $\check{G}'$ and Face-Based Instances.** Over the course of the algorithm, we will maintain a skeleton structure $\mathcal{K}$, and an associated decomposition of graph $\check{G}'$ into subgraphs. Intuitively, for every face $F \in \tilde{\mathcal{F}}(\mathcal{K})$, we will define a subgraph $G_F$ of $\check{G}'$ that we associate with face $F$. We now define a $\mathcal{K}$-decomposition of $\check{G}'$.

**Definition 9.40 (A $\mathcal{K}$-decomposition of $\check{G}'$)** *Let $\mathcal{K} = (K, \{b_u\}_{u\in V(K)}, \rho_K)$ be a skeleton structure, and, for every face $F \in \tilde{\mathcal{F}}(\mathcal{K})$, let $\mathcal{J}_F = (J_F, \{b_u\}_{u\in V(J_F)}, \rho_{J_F}, F^*(\rho_{J_F}))$ be the core structure that $\mathcal{K}$ defines for face $F$. A $\mathcal{K}$-decomposition of the input graph $\check{G}'$ is a collection $\mathcal{G} = \left\{G_F \mid F \in \tilde{\mathcal{F}}(\mathcal{K})\right\}$ of subgraphs of $\check{G}'$, for which the following hold:*

- *for every face $F \in \tilde{\mathcal{F}}(\mathcal{K})$, the core structure $\mathcal{J}_F$ associated with face $F$ is a valid core structure for instance $I_F = (G_F, \Sigma_F)$ defined by graph $G_F$;*
- *for every forbidden face $F \in \tilde{\mathcal{F}}^{\times}(\mathcal{K})$, $G_F = J_F$;*
- *for every face $F \in \tilde{\mathcal{F}}(\mathcal{K})$, $G_F \cap K = J_F$; and*
- *for every pair $F, F' \in \tilde{\mathcal{F}}(\mathcal{K})$ of distinct faces, $V(G_F) \cap V(G_{F'}) \subseteq V(J_F) \cap V(J_{F'})$, and $E(G_F) \cap E(G_{F'}) \subseteq E(J_F) \cap E(J_{F'})$.*

*We will sometimes refer to the subinstance $I_F = (G_F, \Sigma_F)$ of $\check{I}'$ defined by graph $G_F$ associated with a face $F \in \tilde{\mathcal{F}}(\mathcal{K})$ as a* face-based subinstance *associated with face $F$.*

In order to prove Theorem 9.33, we will gradually construct a skeleton $K$ and its associated skeleton structure $\mathcal{K}$, starting with $K = \check{J}$. We will also maintain a $\mathcal{K}$-decomposition $\mathcal{G}$ of the graph $\check{G}'$. We will denote by $E^{\mathsf{del}} = E(\check{G}') \setminus \left(\bigcup_{F \in \tilde{\mathcal{F}}(\mathcal{K})} E(G_F)\right)$, and we will view $E^{\mathsf{del}}$ as the set of *deleted edges*, that will eventually be added to set $E^{\mathsf{del}}(\check{I}')$. We will ensure that $|E^{\mathsf{del}}|$ will remain small over the course of the algorithm. Consider the graph $G' = \check{G}' \setminus E^{\mathsf{del}}$ and its associated subinstance $I' = (G', \Sigma')$ of $\check{I}'$. We will ensure that throughout the algorithm, there is a solution $\varphi$ to instance $I'$, that is compatible with the solution $\psi(\check{I}')$ to instance $\check{I}'$ (with respect to the core structure $\check{\mathcal{J}}'$), with $\mathsf{cr}(\varphi) \leq \mathsf{cr}(\psi(\check{I}'))$ and $|\chi^*(\varphi)| \leq |\chi^*(\psi(\check{I}'))|$. The algorithm terminates once every instance $G_F$ in the resulting decomposition $\mathcal{G}$ becomes "acceptable" – a notion that we define next.

Given a skeleton structure $\mathcal{K} = (K, \{b_u\}_{u \in V(K)}, \rho_K)$, a $\mathcal{K}$-decomposition $\mathcal{G} = \left\{G_F \mid F \in \tilde{\mathcal{F}}(\mathcal{K})\right\}$ of graph $\check{G}'$, and a face $F \in \tilde{\mathcal{F}}(\mathcal{K})$, we denote by $\tilde{E}_F$ the set of all edges $e \in E(G_F)$, such that exactly one endpoint of $e$ lies in $J_F$; in other words, $\tilde{E}_F = \delta_{G_F}(J_F)$ (recall that, since $G_F$ is a subgraph of the subdivided graph $\check{G}$, no edge of $E(G_F) \setminus E(J_F)$ may have both its endpoints in the core $J_F$). Recall that, when we defined a core structure, we have also defined an ordering $\mathcal{O}(J_F)$ of the edges of $\tilde{E}_F$. Intuitively, this is the order in which the edges of $\tilde{E}_F$ are encountered in any $\mathcal{J}_F$-valid solution to instance $I_F$, as we follow along the boudary of face $F^*(\rho_{J_F})$ inside the face, in the counter-clock-wise direction.

**Definition 9.41 (Acceptable Instances)** *Let $\mathcal{K} = (K, \{b_u\}_{u \in V(K)}, \rho_K)$ be skeleton structure, let $\mathcal{G} = \left\{G_F \mid F \in \tilde{\mathcal{F}}(\mathcal{K})\right\}$ be a $\mathcal{K}$-decomposition of graph $\check{G}'$, and let $F \in \tilde{\mathcal{F}}(\mathcal{K})$ be a face. Consider a graph $H_F$, that is obtained from graph $G_F$ by first subdividing every edge $e \in \tilde{E}_F$ with a vertex $t_e$, and then deleting all vertices and edges of $J_F$ from it. We say that instance $G_F$ is* acceptable *if, for every partition $(E_1, E_2)$ of the edges of $\tilde{E}_F$, such that the edges of $E_1$ appear consecutively in the ordering $\mathcal{O}(J_F)$, there is a cut $(X, Y)$ in graph $H_F$ with vertex set $\{t_e \mid e \in E_1\}$ contained in $X$, vertex set $\{t_e \mid e \in E_2\}$ contained in $Y$, and $|E_{H_F}(X, Y)| \leq \check{m}'/\mu^{2b}$.*

The main ingredients in the proof of Theorem 9.33 are the following two lemmas.

**Lemma 9.42** *There is an efficient randomized algorithm, whose input consists of a large enough constant $b \geq 1$, a subinstance $\check{I}' = (\check{G}', \check{\Sigma}')$ of $\check{I}$ with $|E(\check{G}')| = \check{m}'$ and $\check{G}' \subseteq \check{G}$, and a core structure $\check{\mathcal{J}}$ for $\check{I}'$, whose corresponding core is denoted by $\check{J}$. The algorithm either returns FAIL, or computes a skeleton structure $\mathcal{K} = (K, \{b_u\}_{u \in V(K)}, \rho_K)$, and a $\mathcal{K}$-decomposition $\mathcal{G}$ of $\check{G}'$, such that, for every face $F \in \tilde{\mathcal{F}}(\mathcal{K})$, the corresponding subinstance $I_F = (G_F, \Sigma_F)$ of $\check{I}'$ defined by the graph $G_F \in \mathcal{G}$ is acceptable. Moreover, if there is a solution $\psi(\check{I}')$ to instance $\check{I}'$ that is $\check{\mathcal{J}}$-valid, with $\mathsf{cr}(\psi(\check{I}')) \leq (\check{m}')^2/\mu^{240b}$, and $|\chi^*(\psi(\check{I}'))| \leq \check{m}'/\mu^{240b}$, then with probability at least $1 - 1/\mu^{2b}$, the following hold:*

- *the algorithm does not return FAIL;*
- *the cardinality of edge set $E^{\mathsf{del}}(\check{I}') = E(\check{G}') \setminus \left(\bigcup_{F \in \tilde{\mathcal{F}}(\mathcal{K})} E(G_F)\right)$ is bounded by:*

$$\left(\frac{\mathsf{cr}(\psi(\check{I}'))}{\check{m}'} + |\chi^*(\psi(\check{I}'))|\right) \cdot \mu^{O(b)}; \text{ and}$$

- *if we denote $G' = \check{G}' \setminus E^{\mathsf{del}}(\check{I}')$, and let $I' = (G', \Sigma')$ be the subinstance of $\check{I}'$ defined by graph $G'$, then there is a $\mathcal{K}$-valid solution $\varphi$ to instance $I'$, with $\mathsf{cr}(\varphi) \leq \mathsf{cr}(\psi(\check{I}'))$, $|\chi^*(\varphi)| \leq |\chi^*(\psi(\check{I}'))|$, and the total number of crossings in which the edges of $E(K) \setminus E(\check{J})$ participate is at most $\frac{\mathsf{cr}(\psi(\check{I}')) \cdot \mu^{50b}}{\check{m}'}$.*

**Lemma 9.43** *Suppose we are given a subinstance $\check{I}' = (\check{G}', \check{\Sigma}')$ of $\check{I}$ with $\check{G}' \subseteq \check{G}$ and $|E(\check{G}')| = \check{m}'$, a core structure $\check{\mathcal{J}}$ for $\check{I}'$, a skeleton structure $\mathcal{K} = (K, \{b_u\}_{u \in V(K)}, \rho_K)$, and a $\mathcal{K}$-decomposition $\mathcal{G}$ of $\check{G}'$, such that, for every face $F \in \tilde{\mathcal{F}}(\mathcal{K})$, the corresponding subinstance $I_F = (G_F, \Sigma_F)$ of $\check{I}'$ defined by the graph $G_F \in \mathcal{G}$ is acceptable. Let $E^{\mathsf{del}}(\check{I}') = E(\check{G}') \setminus \left(\bigcup_{F \in \tilde{\mathcal{F}}(\mathcal{K})} E(G_F)\right)$, $G' = \check{G}' \setminus E^{\mathsf{del}}(\check{I}')$, and let $I' = (G', \Sigma')$ be the subinstance of $\check{I}'$ defined by graph $G'$. Assume further that there is a $\mathcal{K}$-valid solution $\varphi$ to instance $I'$, so that the total number of crossings in which the edges of $K$ participate is $N \leq \frac{\check{m}'}{\mu^{3b}}$ and $|\chi^*(\varphi)| \leq \check{m}'/\mu^{240b}$. Then there is a solution $\psi(I')$ to instance $I'$ that is clean with respect to $\check{\mathcal{J}}$, with $\mathsf{cr}(\psi(I')) \leq \left(\mathsf{cr}(\varphi) + N^2 + |\chi^*(\varphi)|^2 + \frac{|\chi^*(\varphi)| \cdot \check{m}'}{\mu^b}\right) \cdot (\log \check{m}')^{O(1)}$.*

We provide the proofs of Lemma 9.42 and Lemma 9.43 in Sections 9.4.4 and 9.4.5, respectively. The proof of Theorem 9.33 easily follows from the above two lemmas. Indeed, consider the input instance $\check{I}' = (\check{G}', \check{\Sigma}')$ of $\mathsf{MCNwRS}$ that is a subinstance of $\check{I}$, with $\check{G}' \subseteq \check{G}$ and $|E(\check{G}')| = \check{m}'$, and a core structure $\check{\mathcal{J}}$ for $\check{I}'$, whose corresponding core is denoted by $\check{J}$. We apply the algorithm from Lemma 9.42 to this input. If the algorithm returns FAIL, then we set $E^{\mathsf{del}}(\check{I}') = E(\check{G}') \setminus E(\check{J})$, and return this edge set as the output. We say that the algorithm from Lemma 9.42 *fails* in this case. Otherwise, the algorithm from Lemma 9.42 computes a skeleton structure $\mathcal{K} = (K, \{b_u\}_{u \in V(K)}, \rho_K)$, and a $\mathcal{K}$-decomposition $\mathcal{G}$ of $\check{G}'$, such that, for every face $F \in \tilde{\mathcal{F}}(\mathcal{K})$, the corresponding subinstance $I_F = (G_F, \Sigma_F)$ of $\check{I}'$ defined by the graph $G_F \in \mathcal{G}$ is acceptable. Let $E^{\mathsf{del}}(\check{I}') = E(\check{G}') \setminus \left(\bigcup_{F \in \tilde{\mathcal{F}}(\mathcal{K})} E(G_F)\right)$, let $G' = \check{G}' \setminus E^{\mathsf{del}}(\check{I}')$, and let $I' = (G', \Sigma')$ be the subinstance of $\check{I}'$ defined by graph $G'$. We say that the algorithm from Lemma 9.42 is *successful* if (i) it does not fail; (ii) $|E^{\mathsf{del}}(\check{I}')| \leq \left(\frac{\mathsf{cr}(\psi(\check{I}'))}{\check{m}'} + |\chi^*(\psi(\check{I}'))|\right) \cdot \mu^{O(b)}$; and (iii) there is a $\mathcal{K}$-valid solution $\varphi$ to instance $I'$, with $\mathsf{cr}(\varphi) \leq \mathsf{cr}(\psi(\check{I}'))$, $|\chi^*(\varphi)| \leq |\chi^*(\psi(\check{I}'))|$, and the total number of crossings in which the edges of $E(K) \setminus E(\check{J})$ participate is at most $\frac{\mathsf{cr}(\psi(\check{I}')) \cdot \mu^{50b}}{\check{m}'}$. If the algorithm is not successful, then we say that it is *unsuccessful*. Theorem 9.33 guarantees that, if there is a solution $\psi(\check{I}')$ to instance $\check{I}'$ that is $\check{\mathcal{J}}$-valid, with $\mathsf{cr}(\psi(\check{I}')) \leq (\check{m}')^2/\mu^{240b}$, and $|\chi^*(\psi(\check{I}'))| \leq \check{m}'/\mu^{240b}$, then the algorithm from Lemma 9.42 is successful with probability at least $1 - 1/\mu^{2b}$. If the algorithm from Lemma 9.42 does not fail, then we return edge set $E^{\mathsf{del}}(\check{I}')$ as the outcome of the algorithm.

In order to complete the proof of Theorem 9.33, it is enough to show that, if there is a solution $\psi(\check{I}')$ to instance $\check{I}'$ that is $\check{\mathcal{J}}$-valid, with $\mathsf{cr}(\psi(\check{I}')) \leq (\check{m}')^2/\mu^{240b}$ and $|\chi^*(\psi(\check{I}'))| \leq \check{m}'/\mu^{240b}$, and the algorithm from Lemma 9.42 is successful, then there is a solution $\psi(I')$ to instance $I'$ that is clean with respect to $\mathcal{J}(I)$, with $\mathsf{cr}(\psi(I')) \leq \left(\mathsf{cr}(\psi(\check{I}')) + |\chi^{\mathsf{dirty}}(\psi(\check{I}'))|^2 + \frac{|\chi^{\mathsf{dirty}}(\psi(\check{I}'))| \cdot \check{m}'}{\mu^b}\right) \cdot (\log \check{m}')^{O(1)}$.

Assume that there is a solution $\psi(\check{I}')$ to instance $\check{I}'$ that is $\check{\mathcal{J}}$-valid, with $\mathsf{cr}(\psi(\check{I}')) \leq (\check{m}')^2/\mu^{240b}$ and $|\chi^*(\psi(\check{I}'))| \leq \check{m}'/\mu^{240b}$, and that the algorithm from Lemma 9.42 is successful. Then there is a $\mathcal{K}$-valid solution $\varphi$ to instance $I'$, with $\mathsf{cr}(\varphi) \leq \mathsf{cr}(\psi(\check{I}'))$, $|\chi^*(\varphi)| \leq |\chi^*(\psi(\check{I}'))| \leq \check{m}'/\mu^{240b}$, and the total number of crossings in which the edges of $K$ participate is at most $N = \frac{\mathsf{cr}(\psi(\check{I}')) \cdot \mu^{50b}}{\check{m}'} + |\chi^*(\psi(\check{I}'))| \leq \frac{\check{m}'}{\mu^{3b}}$ (since $\mathsf{cr}(\psi(\check{I}')) \leq \frac{(\check{m}')^2}{\mu^{240b}}$ and $|\chi^*(\psi(\check{I}'))| \leq \frac{\check{m}'}{\mu^{240b}}$). From Lemma 9.43, there is a solution $\psi(I')$ to instance $I'$ that is clean with respect to $\check{\mathcal{J}}$, with $\mathsf{cr}(\psi(I')) \leq \left(\mathsf{cr}(\varphi) + N^2 + |\chi^*(\varphi)|^2 + \frac{|\chi^*(\varphi)| \cdot \check{m}'}{\mu^b}\right) \cdot (\log \check{m}')^{O(1)}$. Note that $\mathsf{cr}(\varphi) \leq \mathsf{cr}(\psi(\check{I}'))$ and $|\chi^*(\varphi)| \leq |\chi^*(\psi(\check{I}'))| = |\chi^{\mathsf{dirty}}(\psi(\check{I}'))|$. Additionally, $N \leq \frac{\mathsf{cr}(\psi(\check{I}')) \cdot \mu^{50b}}{\check{m}'} + |\chi^{\mathsf{dirty}}(\psi(\check{I}'))|$, so $N^2 \leq \frac{2(\mathsf{cr}(\psi(\check{I}')))^2 \cdot \mu^{100b}}{(\check{m}')^2} + 2|\chi^{\mathsf{dirty}}(\psi(\check{I}'))|^2 \leq \mathsf{cr}(\psi(\check{I}'))$, since we have assumed that $\mathsf{cr}(\psi(\check{I}')) \leq (\check{m}')^2/\mu^{240b}$. Therefore, we get that:

$$\mathsf{cr}(\psi(I')) \leq \left(\mathsf{cr}(\psi(\check{I}')) + |\chi^{\mathsf{dirty}}(\psi(\check{I}'))|^2 + \frac{|\chi^{\mathsf{dirty}}(\psi(\check{I}'))| \cdot \check{m}'}{\mu^b}\right) \cdot (\log \check{m}')^{O(1)}$$

as required.

In order to complete the proof of Theorem 9.33, it is now enough to prove Lemma 9.42 and Lemma 9.43, which we do next.

### 9.4.4 Proof of Lemma 9.42

The proof of Lemma 9.42 is somewhat similar to the proof of Theorem 9.29, in that we repeatedly apply Procedure ProcSplit to obtain the desired decomposition, though the details are different. For simplicity of notation, we say that a solution $\psi$ to instance $\check{I}'$ is *good*, if it is $\check{\mathcal{J}}$-valid, with $\mathsf{cr}(\psi) \leq (\check{m}')^2/\mu^{240b}$, and $|\chi^*(\psi)| \leq \check{m}'/\mu^{240b}$. If such a solution exists, then we denote it by $\psi$ throughout the algorithm (though we note that the solution is not known to the algorithm). If such a solution does not exist, then we let $\psi$ be any $\check{\mathcal{J}}$-valid solution to instance $\check{I}'$.

Given a skeleton structure $\mathcal{K}$, a $\mathcal{K}$-decomposition $\mathcal{G}$ of instance $\check{I}'$, and some face $F \in \tilde{\mathcal{F}}(\mathcal{K})$, we denote by $m_F = |E(G_F)|$ and by $\hat{m}_F = |E(G_F) \setminus E(K)|$, where $G_F \in \mathcal{G}$ is the graph associated with face $F$. Our algorithm consists of at most $\lceil 32\mu^{6b} \rceil$ stages. For $1 \leq i \leq \lceil 32\mu^{6b} \rceil$, the input to stage $i$ is a skeleton structure $\mathcal{K}_i$, whose corresponding skeleton is denoted by $K_i$, and a $\mathcal{K}_i$-decomposition $\mathcal{G}_i$ of graph $\check{G}'$. We will ensure that, with high enough probability, the following properties hold.

- A'1. for every face $F \in \tilde{\mathcal{F}}(\mathcal{K}_i)$, either $m_F \leq \check{m}' \cdot \left(1 - \frac{i-1}{32\mu^{6b}}\right)$, or the corresponding instance $I_F = (G_F, \Sigma_F)$ with $G_F \in \mathcal{G}_i$ is acceptable;
- A'2. if we denote by $E_i^{\mathsf{del}} = E(\check{G}') \setminus \left(\bigcup_{G_F \in \mathcal{G}_i} E(G_F)\right)$, then $|E_i^{\mathsf{del}}| \leq (i-1) \cdot \left(\frac{\mathsf{cr}(\psi)\mu^{100b}}{\check{m}'} + |\chi^*(\psi)| \cdot \mu^{3b}\right)$; and
- A'3. let $G_i = \check{G}' \setminus E_i^{\mathsf{del}}$, and let $\Sigma_i$ be the rotation system for $G_i$ induced by $\check{\Sigma}'$. Then there exists a solution $\varphi_i$ to instance $(G_i, \Sigma_i)$, that is $\mathcal{K}_i$-valid, and has the following additional properties:
  - $\mathsf{cr}(\varphi_i) \leq \mathsf{cr}(\psi)$;
  - $|\chi^*(\varphi_i)| \leq |\chi^*(\psi)|$; and
  - if we denote by $N(\varphi_i)$ the total number of crossings of $\varphi_i$ in which the edges of $E(K_i) \setminus E(\check{J})$ participate, then $N(\varphi_i) \leq \frac{\mathsf{cr}(\psi) \cdot i \cdot \mu^{40b}}{\check{m}'}$.

We say that the execution of stage $i$ is *successful* if the output of the stage satisfies the invariants A'1–A'3. We denote by $\tilde{\mathcal{E}}_i'$ the bad event that the execution of stage $i$ is unsuccessful.

We start by constructing the input to the first iteration. The skeleton structure $\mathcal{K}_1$ is defined by the core structure $\check{\mathcal{J}} = (\check{J}, \{b_u\}_{u \in V(\check{J})}, \rho_{\check{J}}, F^*(\rho_{\check{J}}))$ in a natural way: we let the skeleton $K_1$ be $\check{J}$, the orientations $b_u$ for vertices $u \in V(\check{J})$ remain unchanged, and the drawing $\rho_{\check{K}_1}$ of skeleton $\check{K}_1$ is $\rho_{\check{J}}$. Notice that the collection $\tilde{\mathcal{F}}(\mathcal{K}_1)$ of faces has a single non-forbidden face $F^* = F^*(\rho_{\check{J}})$, and we let $G_{F^*} = \check{G}'$ be the graph associated with that face. For every other face $F \in \tilde{\mathcal{F}}(\mathcal{K}_1)$, skeleton structure $\mathcal{K}_1$ defines a corresponding core structure $\mathcal{J}_F$, whose core graph is denoted by $J_F$. We then let $G_F = J_F$ be the graph associated with face $F$. We let $\mathcal{G}_1 = \left\{G_F \mid F \in \tilde{\mathcal{F}}(\mathcal{K})\right\}$ be the resulting $\mathcal{K}_1$-decomposition of $\check{G}'$, so $E_1^{\mathsf{del}} = \emptyset$, and we let $\varphi_1 = \psi$ be the solution to instance $\check{I}' = (G_1, \Sigma_1)$ that we defined above. Note that $N(\varphi_1) = 0$. We have now obtained an input to Stage 1 of the algorithm. If $\psi$ is a good solution to $\check{I}'$, then Invariants A'1–A'3 hold for this input.

**Stage Execution.** We now describe an execution of Stage $i$, for $1 \leq i \leq \lceil 32\mu^{6b} \rceil$. At the beginning of Stage $i$, we set $\mathcal{K}_{i+1} = \mathcal{K}_i$, denoting the corresponding skeleton by $K_{i+1}$, $\mathcal{G}_{i+1} = \mathcal{G}_i$, and $E_{i+1}^{\mathsf{del}} = E_i^{\mathsf{del}}$. Let $\mathcal{F}_i' \subseteq \tilde{\mathcal{F}}(\mathcal{K}_i)$ be the collection of all faces $F$, for which the corresponding instance $I_F = (G_F, \Sigma_F)$

is not acceptable, where $G_F \in \mathcal{G}_i$ is the graph associated with face $F$. Denote $\mathcal{F}'_i = \{F_1, \ldots, F_q\}$. Consider any face $F_j \in \mathcal{F}'_i$. Since instance $I_{F_j} = (G_{F_j}, \Sigma_{F_j})$ is not acceptable, from the definition of acceptable instances (see Definition 9.41), $|E(G_{F_j})| \geq \check{m}'/\mu^{2b}$ must hold. From the definition of a $\mathcal{K}_i$-decomposition, for every pair $G_F, G_{F'}$ of graphs, if an edge $e$ lies in both graphs, then $e \in E(J_F) \cap E(J_{F'})$. Since an edge $e \in E(K_i)$ may lie on the boundary of at most two faces of the drawing $\rho_{K_i}$, we get that $\sum_{j=1}^{q} |E(G_{F_j})| \leq 2\check{m}'$. Therefore, $q \leq 2\mu^{2b}$ must hold.

The algorithm for Stage $i$ consists of $q$ iterations. In iteration $j$, we apply Procedure ProcSplit from Theorem 9.15 to instance $I_{F_j}$ that is defined by graph $G_{F_j}$, and the corresponding core structure $\mathcal{J}_{F_j}$. We will use parameter $b' = 2b$ instead of parameter $b$ in Procedure ProcSplit. The set $\mathcal{P}_j$ of promising paths of cardinality $\left\lfloor \frac{|E(G_{F_j})|}{\mu^{2b}} \right\rfloor$ is computed as follows. From the definition of acceptable instances (see Definition 9.41), if instance $I_{F_j}$ is not acceptable, then there is a partition $(E_1, E_2)$ of the edges of $\tilde{E}_{F_j}$ (the edges of $G_{F_j}$ with exactly one endpoint in the core $J_{F_j}$), such that the edges of $E_1$ appear consecutively in the ordering $\mathcal{O}(J_{F_j})$, and the minimum cut in the correpsonding graph $H_{F_j}$ separating separating these two sets of edges contains more than $\check{m}'/\mu^{2b}$ edges. From the Maximum Flow / Minimum Cut theorem, there is a collection $\mathcal{P}_j$ of $\left\lfloor \frac{\check{m}'}{\mu^{2b}} \right\rfloor \geq \left\lfloor \frac{|E(G_{F_j})|}{\mu^{2b}} \right\rfloor$ edge-disjoint paths in graph $G_{F_j}$, where each path $P \in \mathcal{P}_j$ has an edge of $E_1$ as its first edge, an edge of $E_2$ as its last edge, and is internally disjoint from $J_{F_j}$. Moreover, a set of paths with these properties can be computed efficiently.

Let $\mathcal{A}_j = \left\{P_j, \{b_u\}_{u \in V(J_{F_j}) \cup P_j}, \rho'\right\}$ be the enhancement structure computed by Procedure ProcSplit, and let $(I_1^j = (G_1^j, \Sigma_1^j), I_2^j = (G_2^j, \Sigma_2^j))$ be the split of instance $I_{F_j}$ along $\mathcal{A}_j$ that the algorithm returns. We denote by $E^{\mathsf{del}}(I_{F_j}) = E(G_j) \setminus (E(G_1^j) \cup E(G_2^j))$ the set of the deleted edges. We add the edges of $E^{\mathsf{del}}(I_{F_j})$ to $E^{\mathsf{del}}_{i+1}$, and we define a new skeleton $K'_{i+1}$ and skeleton structure $\mathcal{K}'_{i+1}$ using the enhancement $\mathcal{A}_j$ in a natural way: we let $K'_{i+1} = K_{i+1} \cup P_j$. The orientations of vertices $u$ that belong to $K_{i+1}$ remain unchanged in $\mathcal{K}'_{i+1}$, and the orientations of inner vertices on path $P_j$ are set to be identical to those given by $\mathcal{A}_j$. Consider now drawing $\rho'$ of graph $J_{F_j} \cup P_j$. In this drawing, the image of the core $J_{F_j}$ is identical to that in the drawing $\rho_{K_{i+1}}$ of skeleton $K_{i+1}$, and the image of path $P_j$ is drawn inside face $F_j$, partitioning it into two faces, $F_1^j$ and $F_2^j$. We obtain a drawing $\rho_{K'_{i+1}}$ of the new skeleton $K'_{i+1}$ by starting with the drawing $\rho_{K_{i+1}}$ of skeleton $K_{i+1}$, and then adding the drawing of path $P_j$ inside face $F_j$ exactly like in the drawing $\rho'$. This completes the definition of the new skeleton structure $\mathcal{K}'_{i+1}$. Notice that $\tilde{\mathcal{F}}(\mathcal{K}'_{i+1}) = (\tilde{\mathcal{F}}(\mathcal{K}_{i+1}) \setminus \{F_j\}) \cup \left\{F_1^j \cup F_2^j\right\}$. We modify the decomposition $\mathcal{G}_{i+1}$ by replacing graph $G_{F_j}$ with the graphs $G_{F_1^j}$ (that becomes associated with face $F_1^j$) and $G_{F_2^j}$ (that becomes associated with face $F_2^j$). We then replace skeleton structure $\mathcal{K}_{i+1}$ with the new skeleton structure $\mathcal{K}'_{i+1}$ and continue to the next iteration. We denote by $\mathcal{E}_j^i$ the bad event that the output of Procedure ProcSplit computed in iteration $j$ is not a valid output. This completes the definition of the algorithm for stage $i$.

We start with the following observation.

**Observation 9.44** *Assume that there is a good solution $\psi$ to instance $\check{I}'$, that the Invariants A'1–A'3 hold at the beginning of stage $i$. Then for all $1 \leq j \leq q$, the input to Procedure* ProcSplit *in iteration $j$ of stage $i$ is a valid input.*

**Proof:** Since we have assumed that there is a good solution $\psi$ to instance $\check{I}'$, and that the Invariants A'1–A'3 hold at the beginning of stage $i$, from Invariant A'3, at the beginning of stage $i$, there exists a solution $\varphi_i$ to instance $(G_i, \Sigma_i)$, that is $\mathcal{K}_i$-valid, with $\mathsf{cr}(\varphi_i) \leq \mathsf{cr}(\psi)$, $|\chi^*(\varphi_i)| \leq |\chi^*(\psi)|$, and $N(\varphi_i) \leq \frac{\mathsf{cr}(\psi) \cdot i \cdot \mu^{40b}}{\check{m}'}$.

Consider now some index $1 \leq j \leq q$. Let $\varphi_{i,j}$ be the solution to instance $I_{F_j}$ defined by the graph $G_{F_j} \in \mathcal{G}_i$ that is induced by $\varphi_i$. This solution must be $\mathcal{J}_{F_j}$-valid. From the above discussion,

$$\mathsf{cr}(\varphi_{i,j}) \leq \mathsf{cr}(\varphi_i) \leq \mathsf{cr}(\psi) \leq \frac{(\check{m}')^2}{\mu^{240b}} \leq \frac{|E(G_{F_j})|^2}{\mu^{120b}},$$

since $|E(G_{F_j})| \geq \frac{\check{m}'}{\mu^{2b}}$. Next, we bound the number of dirty crossings of drawing $\varphi_{i,j}$ with respect to the core structure $\mathcal{J}_{F_j}$. The set of such dirty crossings may include all crossings of $\chi^*(\varphi_i)$ (whose number is bounded by $|\chi^*(\psi)| \leq \frac{\check{m}'}{\mu^{240b}} \leq \frac{|E(G_{F_j})|}{\mu^{238b}}$), and the crossings in which the edges of $J_{F_j} \setminus \check{J}'$ participate, whose number is bounded by:

$$N(\varphi_i) \leq \frac{\mathsf{cr}(\psi) \cdot i \cdot \mu^{40b}}{\check{m}'} \leq \frac{i\check{m}}{\mu^{200b}} \leq \frac{|E(G_{F_j})|}{\mu^{150b}},$$

since $\mathsf{cr}(\psi) \leq \frac{(\check{m}')^2}{\mu^{240b}}$, $i \leq \left\lceil 32\mu^{6b} \right\rceil$ and $|E(G_{F_1})| \geq \check{m}'/\mu^{2b}$. Therefore, the total number of dirty crossings in drawing $\varphi_{i,j}$ with respect to the core structure $\mathcal{J}_{F_j}$ is bounded by $\frac{|E(G_{F_1})|}{\mu^{120b}}$, as required. We conclude that there exists there exists a solution $\varphi_{i,j}$ to instance $I_{G_{F_j}}$ that is $\mathcal{J}_{F_j}$-valid, with $\mathsf{cr}(\varphi_{i,j}) \leq \frac{|E(G_{F_j})|^2}{\mu^{60b'}}$ and $|\chi^{\mathsf{dirty}}(\varphi_{i,j})| \leq \frac{|E(G_{F_j})|}{\mu^{60b'}}$, and so the input to Procedure $\mathsf{ProcSplit}$ in iteration $j$ of stage $i$ is valid. □

The following claim is central in the analysis of the algorithm.

**Claim 9.45** *Assume that there is a good solution $\psi$ to instance $\check{I}'$, that Invariants A'1–A'3 hold at the beginning of stage $i$, and that neither of the bad events $\mathcal{E}^i_1, \ldots, \mathcal{E}^i_q$ happened over the course of the $i$th stage. Then Event $\tilde{\mathcal{E}}'_{i+1}$ does not happen either, and so Invariants A'1–A'3 hold at the end of stage $i+1$.*

**Proof:** Throughout the proof, we assume that there is a good solution $\psi$ to instance $\check{I}'$, that Invariants A'1–A'3 hold at the beginning of stage $i$, and that neither of the bad events $\mathcal{E}^i_1, \ldots, \mathcal{E}^i_q$ happens.

We start by establishing Invariant A'1. Consider some face $F \in \tilde{\mathcal{F}}(\mathcal{K}_{i+1})$. If $F \in \tilde{\mathcal{F}}(\mathcal{K}_i)$, then, since $F$ was not added to the set $\{F_1, \ldots, F_q\}$ of faces to be processed in stage $i$, the corresponding instance $I_F = (G_F, \Sigma_F)$ with $G_F \in \mathcal{G}_i$ is acceptable. Since graph $G_F$ remains unchagnged in $\mathcal{G}_{i+1}$, instance $I_F$ remains an acceptable instance. Assume now that $F \notin \tilde{\mathcal{F}}(\mathcal{K}_i)$. Then there must be an index $1 \leq j \leq q$, for which $F = F^j_1$ or $F = F^j_2$. In other words, face $F$ was created in iteration $j$ of Stage $i$. Since Event $\mathcal{E}^i_j$ did not happen, the output produced by Procedure $\mathsf{ProcSplit}$ in iteration $j$ is a valid output. From Property P2, $|E(G_{F^j_1})|, |E(G_{F^j_2})| \leq |E(G_F)| - \frac{|E(G_{F_j})|}{32\mu^{b'}} \leq |E(G_{F_j})| - \frac{\check{m}'}{32\mu^{4b}}$, since $b' = 2b$, and since $|E(G_{F_j})| \geq \check{m}/\mu^{2b}$ must hold, as instance $I_{F_j}$ is not acceptable. Since, from Invariant A'1, $|E(G_{F_j})| \leq \check{m}' \cdot \left(1 - \frac{i-1}{32\mu^{6b}}\right)$, we get that $|E(G_{F^j_1})|, |E(G_{F^j_2})| \leq \check{m}' \cdot \left(1 - \frac{i}{32\mu^{6b}}\right)$. Therefore, invariant A'1 continues to hold.

Next, we establish Invariant A'2. Fix an index $1 \leq j \leq q$. Using the arguments from the proof of Observation 9.44, there is a soluton $\varphi_{i,j}$ to instance $I_{F_j}$ defined by the graph $G_{F_j} \in \mathcal{G}_i$, that is $\mathcal{J}_{F_j}$-valid, with $\mathsf{cr}(\varphi_{i,j}) \leq \mathsf{cr}(\psi) \leq \frac{|E(G_{F_j})|^2}{\mu^{60b'}}$, and $|\chi^{\mathsf{dirty}}(\varphi_{i,j})| \leq |\chi^*(\psi)| + N(\varphi_i) \leq \frac{|E(G_{F_j})|}{\mu^{60b'}}$. From Property P1:

$$
\begin{aligned}
|E^{\mathsf{del}}(I_{F_j})| &\leq \frac{2\mathsf{cr}(\varphi_{i,j})\cdot\mu^{38b'}}{|E(G_{F_j})|}+|\chi^{\mathsf{dirty}}(\varphi_{i,j})|\\
&\leq \frac{2\mathsf{cr}(\psi)\cdot\mu^{78b}}{\check{m}'}+|\chi^*(\psi)|+N(\varphi_i)\\
&\leq \frac{2\mathsf{cr}(\psi)\cdot\mu^{78b}}{\check{m}'}+|\chi^*(\psi)|+\frac{\mathsf{cr}(\psi)\cdot i\cdot\mu^{40b}}{\check{m}'}\\
&\leq \frac{2\mathsf{cr}(\psi)\cdot\mu^{78b}}{\check{m}'}+|\chi^*(\psi)|+\frac{32\mathsf{cr}(\psi)\cdot\mu^{46b}}{\check{m}'}\\
&\leq \frac{\mathsf{cr}(\psi)\cdot\mu^{80b}}{\check{m}'}+|\chi^*(\psi)|.
\end{aligned}
$$

(we have used the fact that $|E(G_{F_j})|\geq \check{m}/\mu^{2b}$, $b'=2b$, and $i\leq\lceil 32\mu^{6b}\rceil$). Since $q\leq 2\mu^{2b}$, we get that $|E^{\mathsf{del}}_{i+1}|\leq |E^{\mathsf{del}}_i|+\frac{\mathsf{cr}(\psi)\cdot\mu^{100b}}{\check{m}'}+|\chi^*(\psi)|\cdot\mu^{3b}$, establishing invariant A'2.

It now remains to establish Invariant A'3. For convenience, we denote by $G^0=G_i$, and, for $1\leq j\leq q$, we denote by $G^j$ the graph obtained after the $j$th iteration of the $i$th stage, that is, $G^j=G^{j-1}\setminus E^{\mathsf{del}}(I_{F_j})$. We denote by $I^j$ the subinstance of $\check{I}'$ defined by graph $G^j$. We also denote by $K^0=K_i$ the initial skeleton at the beginning of phase $i$, and, for $1\leq j\leq q$, we denote by $K^j$ the skeleton obtained at the end of iteration $j$, so $K^j=K^{j-1}\cup P_j$. We denote by $\mathcal{K}^j$ the skeleton structure associated with skeleton $K^j$, that can be obtained from $\mathcal{K}^{j-1}$ and the enhancement structure $\mathcal{A}_j$ as described above. Lastly, we will define, for all $0\leq j\leq q$, a solution $\varphi^j$ to instance $I^j$ that is $\mathcal{K}^j$-valid. We will ensure that for all $0\leq j<j'\leq q$, the drawing of graph $G_{F_{j'}}$ in $\varphi^j$ is identical to that in $\varphi^0$. Additionally, we will ensure that $\mathsf{cr}(\varphi^j)\leq\mathsf{cr}(\psi)$ and $|\chi^*(\varphi^j)|\leq|\chi^*(\psi)|$.

Initially, we let $\varphi^0=\varphi_i$ be the solution for instance $I^0$ that is guaranteed to exist by Invariant A'3. Recall that this solution is $\mathcal{K}^0$-valid; $\mathsf{cr}(\varphi^0)\leq\mathsf{cr}(\psi)$; $|\chi^*(\varphi^0)|\leq|\chi^*(\psi)|$; and the total number of crossings of $\varphi^0$ in which the edges of $E(K^0)\setminus E(\check{J})$ participate is at most $\frac{\mathsf{cr}(\psi)\cdot i\cdot\mu^{40b}}{\check{m}'}$. For $0\leq j\leq q$, we denote by $N^j$ the total number of crossings in which the edges of $E(K^j)\setminus E(\check{J})$ participate in $\varphi^j$.

Consider now some index $1\leq j\leq q$, and assume that we are given a solution $\varphi^{j-1}$ to instance $I^{j-1}$ that is $\mathcal{K}^{j-1}$-valid. Recall that the drawing of graph $G_{F_j}$ induced by $\varphi^{j-1}$ is identical to that in $\varphi^0$. Therefore, the drawing of $G_{F_j}$ induced by $\varphi^{j-1}$ is precisely $\varphi_{i,j}$. From Property P3 of the valid output to procedure $\mathsf{ProcSplit}$, there is a $\mathcal{J}_{F_j}$-valid solution $\varphi'_j$ to the subinstance of $\check{I}'$ that is defined by graph $G'_{F_j}=G_{F_j}\setminus E^{\mathsf{del}}(I_{F_j})$, that is compatible with $\varphi_{i,j}$, in which the edges of $E(J_{F_j})\cup E(P_j)$ do not cross each other, and the number of crosings in which the edges of $P_j$ participate is at most $\frac{\mathsf{cr}(\varphi^{j-1})\cdot\mu^{12b'}}{|E(G_{F_j})|}\leq\frac{\mathsf{cr}(\psi)\cdot\mu^{26b}}{\check{m}'}$, since $\mathsf{cr}(\varphi^{j-1})\leq\mathsf{cr}(\psi)$, $b'=2b$, and $|E(G_{F_j})|\geq\check{m}'/\mu^{2b}$. Note that, from the definition of compatible drawings (see Definition 9.6), the image of the core $J_{F_j}$ in $\varphi'_j$ is identical to that in $\varphi_{i,j}$. The only difference between drawing $\varphi'_j$ and $\varphi_{i,j}$ is that the images of the edges of $E^{\mathsf{del}}(I_{F_j})$ were deleted, and some additional local changes were made within the face $F_j$. We are guaranteed that, if a point $p$ is an inner point on the image of some edge $\varphi'_j$, then is an inner point on the image of some edge in $\varphi_{i,j}$. Moreover, if $p$ is an image of some vertex $v$ in $\varphi'_j$, then either (i) $p$ is the image of $v$ in $\varphi_{i,j}$; or (ii) the degree of $p$ in $G_{F_j}\setminus E^{\mathsf{del}}(I_{F_j})$ is 2, and $p$ is an inner point on the image of some edge in $\varphi_{i,j}$. In order to obtain drawing $\varphi^j$ of graph $G^j$, we first delete, from drawing $\varphi^{j-1}$, the images of all edges in $E^{\mathsf{del}}(I_{F_j})$. Next, we delete the image of the graph $G_{F_j}$ from the current drawing, and copy instead the image of the graph $G_{F_j}\setminus E^{\mathsf{del}}(I_{F_j})$ from drawing $\varphi'_j$. Note that the two images of graph $G_{F_j}\setminus E^{\mathsf{del}}(I_{F_j})$ are identical except for some small local changes inside face $F_j$. While it is possible that edges and vertices of $G^j\setminus G_{F_j}$ are drawn inside face $F_j$ in $\varphi^{j-1}$, it is easy to see that no new crossings between edges of $G_{F_j}$ and edges of $G^j\setminus G_{F_j}$ are introduced. Since $\mathsf{cr}(\varphi'_j)\leq\mathsf{cr}(\varphi_{i,j})$,

we get that $\mathsf{cr}(\varphi^j) \leq \mathsf{cr}(\varphi^{j-1}) \leq \mathsf{cr}(\psi)$. Since the only changes to the drawing outside face $F_j$ is the deletion of the segments of the images of some edges, $|\chi^*(\varphi^j)| \leq |\chi^*(\varphi^{j-1})| \leq |\chi^*(\psi)|$. Since we are guaranteed from Property P3 that the number of crosings in which the edges of $P_j$ participate in $\varphi'_j$ is at most $\frac{\mathsf{cr}(\psi)\cdot\mu^{26b}}{\check{m}'}$, we get that $N^j \leq N^{j-1} + \frac{\mathsf{cr}(\psi)\cdot\mu^{26b}}{\check{m}'}$.

We define the solution $\varphi_{i+1}$ to instance $(G_{i+1}, \Sigma_{i+1})$ to be $\varphi^q$. From the above discussion, drawing $\varphi^q$ is $\mathcal{K}_{i+1}$-valid, $\mathsf{cr}(\varphi_{i+1}) \leq \mathsf{cr}(\psi)$, and $|\chi^*(\varphi_i)| \leq |\chi^*(\psi)|$. Since the number of iterations $q \leq 2\mu^{2b}$, and in every iteration we introduce at most $\frac{\mathsf{cr}(\psi)\cdot\mu^{26b}}{\check{m}'}$ new crossings with the edges of the new skeleton $K_{i+1}$, we get that $N(\varphi_{i+1}) \leq N(\varphi_i) + 2\mu^{2b}\cdot\frac{\mathsf{cr}(\psi)\cdot\mu^{26b}}{\check{m}'} \leq \frac{\mathsf{cr}(\psi)\cdot i\cdot\mu^{40b}}{\check{m}'} + \frac{2\mathsf{cr}(\psi)\cdot\mu^{28b}}{\check{m}'} \leq \frac{\mathsf{cr}(\psi)\cdot(i+1)\cdot\mu^{40b}}{\check{m}'}$. □

From Observation 9.44 and Theorem 9.15, for all $1 \leq j \leq q$, $\mathbf{Pr}\left[\mathcal{E}^i_j \mid \neg\tilde{\mathcal{E}}'_1 \wedge \cdots \wedge \neg\mathcal{E}'_i\right] \leq \frac{2^{20}}{\mu^{10b'}} \leq \frac{2^{20}}{\mu^{20b}}$. Since $q \leq 2\mu^{2b}$, we get that $\mathbf{Pr}\left[\mathcal{E}'_{i+1} \mid \neg\tilde{\mathcal{E}}'_1 \wedge \cdots \wedge \neg\mathcal{E}'_i\right] \leq \frac{2^{21}}{\mu^{18b}}$. Let $\tilde{\mathcal{E}}'$ be the bad event that either of the events $\tilde{\mathcal{E}}'_1, \ldots, \tilde{\mathcal{E}}'_z$ happened. Since $z = \left\lceil 32\mu^{6b}\right\rceil$, we get that $\mathbf{Pr}\left[\tilde{\mathcal{E}}'_z\right] \geq \frac{z}{\mu^{20b}} \leq \frac{1}{\mu^{10b}}$.

We return the skeleton structure $\mathcal{K}_z$, and the $\mathcal{K}_z$-decomposition $\mathcal{G}_z$ of $\check{G}'$. From Invariant A'1, for every face $F \in \tilde{\mathcal{F}}(\mathcal{K}_i)$, the corresponding instance $I_F = (G_F, \Sigma_F)$ with $G_F \in \mathcal{G}_i$ must be acceptable (this is since the graph associated with an unacceptable instance must have at least $\check{m}'/\mu^{2b}$ edges). Assume now that there is a good solution $\psi$ to instance $\check{I}'$, and that bad event $\tilde{\mathcal{E}}'$ did not happen. Then we are guaranteed that the algorithm does not return FAIL, and, from Invariant A'2, $|E^{\mathsf{del}}_z| \leq z\cdot\left(\frac{\mathsf{cr}(\psi)\mu^{100b}}{\check{m}'} + |\chi^*(\psi)|\cdot\mu^{3b}\right) \leq \frac{\mathsf{cr}(\psi)\mu^{108b}}{\check{m}'} + |\chi^{\mathsf{dirty}}(\psi)|\cdot\mu^{10b}$, since $z = \left\lceil 32\mu^{6b}\right\rceil$. Lastly, Invariant A'3 guarantees that there is a solution $\varphi$ to instance $(G_z, \Sigma_z)$, where $G_z = \check{G}' \setminus E^{\mathsf{del}}_z$, and $\Sigma_z$ is the rotation system for $G_z$ induced by $\check{\Sigma}'$, with the following properties. First, drawing $\varphi$ is $\mathcal{K}_z$-valid. Additionally, $\mathsf{cr}(\varphi) \leq \mathsf{cr}(\psi)$, $|\chi^*(\varphi)| \leq |\chi^*(\psi)|$, and the total number of crossings in which the edges of $E(K_z)\setminus E(\check{J})$ participate is at most $\frac{\mathsf{cr}(\psi)\cdot z\cdot\mu^{40b}}{\check{m}'} \leq \frac{\mathsf{cr}(\psi)\cdot\mu^{47b}}{\check{m}'}$. Since $\mathbf{Pr}\left[\tilde{\mathcal{E}}'\right] \leq 1/\mu^{10b}$, this completes the proof of Lemma 9.42.

### 9.4.5 Proof of Lemma 9.43

We assume that we are given a subinstance $\check{I}' = (\check{G}', \check{\Sigma}')$ of $\check{I}$ with $|E(\check{G}')| = \check{m}'$, a core structure $\check{\mathcal{J}}$ for $\check{I}'$, whose corresponding core graph is denoted by $\check{J}$, a skeleton structure $\mathcal{K} = (K, \{b_u\}_{u\in V(K)}, \rho_K)$, and a $\mathcal{K}$-decomposition $\mathcal{G}$ of $\check{G}'$, such that, for every face $F \in \tilde{\mathcal{F}}(\mathcal{K})$, the corresponding subinstance $I_F = (G_F, \Sigma_F)$ of $\check{I}'$ defined by the graph $G_F \in \mathcal{G}$ is acceptable. Let $E^{\mathsf{del}}(\check{I}') = E(\check{G}') \setminus \left(\bigcup_{F\in\tilde{\mathcal{F}}(\mathcal{K})} E(G_F)\right)$, $G = \check{G}' \setminus E^{\mathsf{del}}(\check{I}')$, and let $I = (G, \Sigma)$ be the subinstance of $\check{I}'$ defined by graph $G$. We also assume that there is a $\mathcal{K}$-valid solution $\varphi$ to instance $I$, so that the total number of crossings in which the edges of $K$ participate is at most $N \leq \frac{\check{m}'}{\mu^{3b}}$. Note that $\check{\mathcal{J}}$ remains a valid core structure, and $\mathcal{K}$ remains a valid skeleton structure for instance $I$. From this point onward we will only work with instance $I$, and we will not need the initial subinstance $\check{I}'$ of instance $\check{I}$, or the set $E^{\mathsf{del}}(\check{I}')$ of edges, but we will use the parameter $\check{m}' = |E(\check{G}')|$. Our goal is to prove that there is a solution $\psi$ to instance $I$ that is clean with respect to $\check{\mathcal{J}}$, with $\mathsf{cr}(\psi) \leq \left(\mathsf{cr}(\varphi) + N^2 + |\chi^*(\varphi)|^2 + \frac{|\chi^*(\varphi)|\cdot\check{m}'}{\mu^b}\right)\cdot(\log\check{m}')^{O(1)}$.

Since the statement of the lemma is existential, it is sufficient to show an algorithm that transforms the solution $\varphi$ to instance $I$ into another solution $\psi$ that is clean with respect to $\check{\mathcal{J}}$, with the number of crossings bounded as above. In order to do so, we need to "repair" the drawing, so that for every edge $e \in E(G)$, the image of $e$ is disjoint from the interior of the forbidden faces in $\tilde{\mathcal{F}}^{\times}(\mathcal{K})$. We do so in two steps. In the first step, we modify the drawing $\varphi$ so that, for every non-forbidden face $F \in \tilde{\mathcal{F}}(\mathcal{K})\setminus\tilde{\mathcal{F}}^{\times}(\mathcal{K})$, the images of all vertices of graph $G_F \in \mathcal{G}$ associated with face $F$ lie in the region $F$ of the drawing. We say that an edge $e \in E(G)\setminus E(K)$ is *bad* if the image of $e$ in the resulting drawing intersects the interior of at least one forbidden face in $\tilde{\mathcal{F}}^{\times}(\mathcal{K})$. Notice that for each such bad

edge $e$, there must be some face $F \in \tilde{\mathcal{F}}(\mathcal{K}) \setminus \tilde{\mathcal{F}}^{\times}(\mathcal{K})$ with $e \in G_F$, so the images of the endpoints of $e$ lie in region $F$ of the current drawing. In the second step, we further modify the drawing to obtain a $\mathcal{K}$-clean solution to instance $I$. In order to do so, for every bad edge $e$, if $e \in E(G_F)$ for some face $F \in \tilde{\mathcal{F}}(\mathcal{K}) \setminus \tilde{\mathcal{F}}^{\times}(\mathcal{K})$, we "move" the image of $e$ so it is drawn completely inside face $F$. In order to ensure that the new image of $e$ crosses few edges, we may need to rearrange the current drawing inside the face $F$. This step exploits the fact that instance $I_F$ corresponding to face $F$ is acceptable.

We now proceed to describe each of the steps in turn. For convenience, we denote $\tilde{\mathcal{F}}' = \tilde{\mathcal{F}}(\mathcal{K}) \setminus \tilde{\mathcal{F}}^{\times}(\mathcal{K})$.

**Step 1: Moving the Vertices**

The goal of the first step is to prove the following claim.

**Claim 9.46** *There is a solution $\psi_1$ to instance $I$ that is $\mathcal{K}$-valid, such that, for every face $F \in \tilde{\mathcal{F}}'$, the images of all vertices of $G_F$ lie in region $F$ of the drawing. Additionally, $\mathsf{cr}(\psi_1) \leq (\mathsf{cr}(\varphi) + N^2) \cdot (\log \check{m}')^{O(1)}$, $|\chi^*(\psi_1)| \leq |\chi^*(\varphi)| \cdot (\log \check{m}')^{O(1)}$, and the total number of crossings in which the edges of $K$ participate in $\psi_1$ is at most $N \cdot (\log \check{m}')^{O(1)}$.*

**Proof:** Consider some face $F \in \tilde{\mathcal{F}}'$. Recall that we have defined a core structure $\mathcal{J}_F$ associated with face $F$; we denote by $J_F$ the corresponding core graph, so $J_F \subseteq K$. We also denote by $I_F = (G_F, \Sigma_F)$ the instance associated with face $F$, where $G_F \in \mathcal{G}$.

Let $E'_F \subseteq E(G_F)$ be the set of edges $e = (x, y) \in E(G_F)$, such that either (i) the image of one of the vertices $x, y$ lies in region $F$ in $\varphi$, and the image of the other vertex lies outside of $F$; or (ii) the images of both vertices lie outside region $F$ in $\varphi$, and the image of $e$ crosses the boundary of $F$. Since $\mathcal{J}_F$ is a valid core structure for instance $I_F$ (see Definition 9.3), for every edge $e \in E(G_F)$ that is incident to a vertex $x \in V(J_F)$, a segment of $\varphi(e)$ that contains $\varphi(x)$ must be contained in region $F$ in $\varphi$. Therefore, for every edge $e \in E'_F$, its image $\varphi(e)$ intersects the interior of $F$, and it must cross the image of some edge of $J_F$. Since the total number of crossings in which the edges of $K$ participate in $\varphi$ is bounded by $N$, we get the following immediate observation:

**Observation 9.47** $\sum_{F \in \tilde{\mathcal{F}}'} |E'_F| \leq N$.

Consider again some face $F \in \tilde{\mathcal{F}}'$. It will be convenient for us to subdivide every edge of $E'_F$ with one or two vertices, and to adjust the drawing $\varphi$ of $G$ to include these new vertices, as follows. Consider an edge $e = (x, y) \in E'_F$. Assume first that for both $x$ and $y$, their images in $\varphi$ lie outside the region $F$. In this case, we replace edge $e$ in both $G$ and $G_F$ with a path $(x, t_e, t'_e, y)$. Consider now the image $\varphi(e)$ of edge $e$ in drawing $\varphi$, and direct it from $\varphi(x)$ to $\varphi(y)$. Let $p$ be the first point on $\varphi(e)$ that belongs to the boundary of face $F$, and let $p'$ be the last point on $\varphi(e)$ lying on the boundary of $F$. We place the image of the new vertex $t_e$ on curve $\varphi(e)$ immediately next to point $p$, in the interior of face $F$. Similarly, we place the image of the new vertex $t'_e$ on curve $\varphi(e)$ immediately next to point $p'$, in the interior of face $F$. Assume now that the image of one of the vertices $x, y$ lies in $F$ (for example, vertex $y$), and the image of the other vertex (vertex $x$) lies outside of $F$. We direct $\varphi(e)$ from $\varphi(x)$ to $\varphi(y)$, and we let $p$ be the first point on $\varphi(e)$ that lies on the boundary of $F$. We replace edge $e$ with a path $(x, t_e, y)$ in graph $G$ and in graph $G_F$, and we place the image of vertex $t_e$ on $\varphi(e)$, next to point $p$, in the interior of face $F$. For convenience, the graph that is obtained from $G$ after these modifications is still denoted by $G$, and for every face $F \in \tilde{\mathcal{F}}'$, the resulting graph associated with the face is still denoted by $G_F$. Since the newly added vertices all have degree 2 in $G$, it is easy to extend the rotation system $\Sigma$ for graph $G$ to include these vertices, and we can similarly extend the rotation system $\Sigma_F$ for each graph $G_F \in \mathcal{G}$.

For a face $F \in \tilde{\mathcal{F}}'$, let $H_F \subseteq G_F$ be the graph whose vertex set contains every vertex $x \in V(G_F)$ with $\varphi(x) \notin F$, and every edge $e \in E(G_F)$ whose image in $\varphi$ is disjoint from $F$. Let $E''_F \subseteq E(G_F)$ be

the set of all edges $e \in E(G_F)$ with one endpoint in $H_F$ and another in $G_F \setminus H_F$. Observe that each such edge $e \in E''_F$ was obtained by subdividing some edge of $E'_F$, so $|E''_F| \leq 2|E'_F|$. For every edge $e = (x, y) \in E''_F$ with $x \in V(H_F)$, the intersection of $\varphi(e)$ with the region $F$ is a very short segment of $\varphi(e)$ that is incident to $\varphi(x)$ (recall that $\varphi(x)$ lies inside $F$ very close to its boundary). We denote by $Z_F$ the set of all endpoints of edges $e \in E''_F$ whose image lies in region $F$. We also note that graph $H_F$ is a subgraph of $G_F$ induced by $V(H_F)$. Indeed, if $e = (x, y)$ is an edge of $G_F$ with $x, y \in V(H_F)$, then the image of edge $e$ must be entirely disjoint from region $F$ (otherwise it would have been subdivided).

We need a slight modification of the algorithm for computing a well-linked decomposition from Theorem 4.19, that is summarized in the following theorem. The proof is deferred to Section H.8 of Appendix.

**Theorem 9.48** *There is an efficient algorithm, whose input is a graph $G$, a vertex-induced subgraph $S$ of $G$, and parameters $m$ and $\alpha$, for which $|E(G)| \leq m$ and $0 < \alpha < \frac{1}{c \log^2 m}$ hold, for a large enough constant $c$. The algorithm computes a collection $\mathcal{R}$ of vertex-disjoint clusters of $S$, and, for every cluster $R \in \mathcal{R}$, two sets $\mathcal{P}_1(R)$, $\mathcal{P}_2(R)$ of paths, such that the following hold:*

- $\bigcup_{R \in \mathcal{R}} V(R) = V(S)$;
- *for every cluster $R \in \mathcal{R}$, $|\delta_G(R)| \leq |\delta_G(S)|$;*
- *every cluster $R \in \mathcal{R}$ has the $\alpha$-bandwidth property in graph $G$;*
- $\sum_{R \in \mathcal{R}} |\delta_G(R)| \leq 4|\delta_G(S)|$;
- *for every cluster $R \in \mathcal{R}$, $\mathcal{P}_1(R) = \{P_1(e) \mid e \in \delta_G(R)\}$, where for every edge $e \in \delta_G(R)$, path $P_1(e)$ has $e$ as its first edge and some edge of $\delta_G(S)$ as its last edge, and all inner vertices of $P_1(e)$ lie in $V(S) \setminus V(R)$. Additionally, $\mathrm{cong}_G(\mathcal{P}_1(R)) \leq 400/\alpha$; and*
- *for every cluster $R \in \mathcal{R}$, there is a subset $\hat{E}_R \subseteq \delta_G(R)$ of at least $\lfloor |\delta_G(R)|/64 \rfloor$ edges, such that $\mathcal{P}_2(R) = \left\{P_2(e) \mid e \in \hat{E}_R\right\}$, where for every edge $e \in \hat{E}_R$, path $P_2(e)$ has $e$ as its first edge and some edge of $\delta_G(S)$ as its last edge, and all inner vertices of $P_2(e)$ lie in $V(S) \setminus V(R)$. Moreover, $\mathrm{cong}_G\left(\bigcup_{R \in \mathcal{R}} \mathcal{P}_2(R)\right) \leq O\left(\frac{\log m}{\alpha}\right)$.*

We apply the algorithm from Theorem 9.48 to graph $G_F$, subgraph $S = H_F$ of $G_F$, parameter $\breve{m}'$ and $\alpha = \frac{1}{\log^4 \breve{m}'}$, to compute a collection $\mathcal{R}_F$ of vertex-disjoint clusters of $H_F$, such that $\bigcup_{R \in \mathcal{R}_F} V(R) = V(H_F)$, $\sum_{R \in \mathcal{R}_F} |\delta_{G_F}(R)| \leq 4|E''_F| \leq 8|E'_F|$, and every cluster $R \in \mathcal{R}_F$ has the $\alpha$-bandwidth property in graph $G_F$. Additionally, the algorithm computes, for every cluster $R \in \mathcal{R}_F$, a set $\mathcal{P}_1(R) = \{P_1(e) \mid e \in \delta_{G_F}(R)\}$ of paths in graph $G_F$, with $\mathrm{cong}_{G_F}(\mathcal{P}_1(R)) \leq 400/\alpha \leq O(\log^4 \breve{m}')$, such that, for every edge $e \in \delta_{G_F}(R)$, path $P_1(e)$ has $e$ as its first edge and some edge of $E''_F$ as its last edge, and all inner vertices of $P_1(e)$ lie in $V(H_F) \setminus V(R)$. It also computes, for every cluster $R \in \mathcal{R}_F$, a subset $\hat{E}_R \subseteq \delta_G(R)$ of at least $\lfloor |\delta_G(R)|/64 \rfloor$ edges, and another set $\mathcal{P}_2(R) = \left\{P_2(e) \mid e \in \hat{E}_R\right\}$ of paths, where for every edge $e \in \hat{E}_R$, path $P_2(e)$ has $e$ as its first edge and some edge of $E''_F$ as its last edge, such that all inner vertices of $P_2(e)$ lie in $V(H_F) \setminus V(R)$, and the total congestion caused by the paths in $\bigcup_{R \in \mathcal{R}} \mathcal{P}_2(R)$ is at most $O\left(\frac{\log \breve{m}'}{\alpha}\right) \leq O(\log^5 \breve{m}')$. We let $E'''_F = \bigcup_{R \in \mathcal{R}_F} \delta_{G_F}(R)$, so $E''_F \subseteq E'''_F$.

It will be convenient for us to further slightly modify graph $G$, by subdividing some of its edges, as follows. Consider a face $F \in \tilde{\mathcal{F}}'$ and an edge $e = (x, y) \in E'''_F$. If there are two distinct clusters $R, R' \in \mathcal{R}_F$ with $x \in R$ and $y \in R'$, then we replace edge $e$ with a path $(x, t_e^R, t_e^{R'}, y)$ in $G$, and we denote edge $\tilde{e} = (t_e^R, t_e^{R'})$. We also modify the current drawing $\varphi$ by placing the images of the newly added vertices $t_e^R, t_e^{R'}$ on $\varphi(e)$. Otherwise, there must be a cluster $R \in \mathcal{R}_F$, such that one endpoint of

$e$ (say $x$) lies in $R$, and the other endpoint (vertex $y$) lies in $Z_F$. In this case, we replace edge $e$ with a path $(x, t_e^R, y)$ in graph $G$, and we denote edge $\tilde{e} = (t_e^R, y)$. We place the image of vertex $t_e^R$ on $\varphi(e)$, outside the region $F$.

Let $G'$ denote the final graph that is obtained from $G$ once every face $F \in \tilde{\mathcal{F}}'$ and every edge $e \in E_F'''$ is processed. For each face $F \in \tilde{\mathcal{F}}'$ we define a subgraph $G'_F \subseteq G'$ similarly, by subdividing the edges of $E_F'''$ as before, and we denote $\tilde{E}_F = \{\tilde{e} \mid e \in E_F'''\}$. We similarly update graph $H_F$, to obtain a new graph $H'_F \subseteq G'_F$, as follows. First, for every edge $e \in E(H_F)$, whose endpoints lie in different clusters of $\mathcal{R}_F$, we subdivide edge $e$ with two vertices as before. Additionally, for every edge $e = (x, y) \in E(G_F)$ with $x \in V(H_F)$ and $y \in Z_F$, we add the new edge $(x, t_e^R)$ that was obtained by subdividing $e$ to graph $H'_F$ (here, $R \in \mathcal{R}_F$ is the cluster containing $x$).

For every cluster $R \in \mathcal{R}_F$, the set $\mathcal{P}_1(R)$ of paths naturally defines a set $\mathcal{P}'_1(R)$ of paths in graph $G'_F$, where $\mathcal{P}'_1(R) = \left\{P'_1(e) \mid e \in \delta_{G'_F}(R)\right\}$, with $\mathrm{cong}_{G'_F}(\mathcal{P}'_1(R)) \leq O(\log^4 \check{m}')$, such that, for every edge $e \in \delta_{G'_F}(R)$, path $P'_1(e)$ has $e$ as its first edge, and it terminates at some vertex of $Z_F$. Furthermore, all inner vertices of $P'_1(e)$ lie in $V(H'_F) \setminus V(R)$. Similarly, set $\mathcal{P}_2(R)$ of paths naturally defines a set $\mathcal{P}'_2(R)$ of paths in graph $G'_F$, where each path in $\mathcal{P}'_2(R)$ starts at a distinct edge of $\delta_{G'_F}(R)$, terminates at some vertex of $Z_F$, and has all its inner vertices contained in $V(H'_F) \setminus V(R)$. As before, $|\mathcal{P}'_2(R)| \geq \left\lfloor |\delta_{G'_F}(R)|/64 \right\rfloor$, and all paths in $\bigcup_{R \in \mathcal{R}_F} \mathcal{P}'_2(R)$ cause congestion at most $O(\log^5 \check{m}')$.

Notice that drawing $\varphi$ of $G$ naturally defines a drawing $\varphi'$ of graph $G'$. We denote $\tilde{E} = \bigcup_{F \in \tilde{\mathcal{F}}'} \tilde{E}_F$. Observe that we have never subdivided the edges of the skeleton $K$, so $K \subseteq G'$ still holds. We can naturally extend the rotation system $\Sigma$ to graph $G'$, to obtain a rotation system $\Sigma'$, and we denote by $I' = (G', \Sigma')$ the resulting instance of $\mathsf{MCNwRS}$.

Let $\varphi''$ be any drawing of graph $G'$, and let $(e, e')_p$ be a crossing of $\varphi''$. We say that crossing $(e, e')_p$ is *uninteresting* if both $e, e' \in \tilde{E}$, and we say that this crossing is *interesting* otherwise. We prove the following weaker analogue of Claim 9.46.

**Claim 9.49** *There is a solution $\psi_2$ to instance $I' = (G', \Sigma')$ that is $\mathcal{K}$-valid, such that, for every face $F \in \tilde{\mathcal{F}}'$, the images of all vertices of $G'_F$ lie in region $F$ of the drawing. Additionally, the number of interesting crossings in $\psi_2$ is bounded by $\mathsf{cr}(\varphi) \cdot (\log \check{m}')^{O(1)}$; $|\chi^*(\psi_2))| \leq |\chi^*(\varphi))| \cdot (\log \check{m}')^{O(1)}$; and the total number of crossings in which the edges of $K$ participate in $\psi_2$ is at most $N \cdot (\log \check{m}')^{O(1)}$.*

The proof of Claim 9.46 easily follows from Claim 9.49. Indeed, consider the solution $\psi_2$ to instance $I'$. We apply type-1 uncrossing operation to the images of the edges in $\tilde{E}$ (see Section 4.4.2 and Theorem 4.33 for a formal description). The operation repeatedly selects pairs $e, e' \in \tilde{E}$ of edges that cross more than once, and then eliminates at least one of the crossings between these edges by a local uncrossing operation that "swaps" segments of images of these two edges without affecting the rest of the drawing. Therefore, if $\psi_3$ is the drawing of graph $G'$ obtained at the end of this procedure, then $\psi_3$ is a valid solution to instance $I'$ that remains $\mathcal{K}$-valid. Since the edges of the core $\check{J}$ may not belong to $\tilde{E}$, $|\chi^*(\psi_3))| \leq |\chi^*(\psi_2))| \leq |\chi^*(\varphi))| \cdot (\log \check{m}')^{O(1)}$. The number of interesting crossings in $\psi_3$ is bounded by the number of interesting crossings in $\psi_2$, which, in turn, is bounded by $\mathsf{cr}(\varphi) \cdot (\log \check{m}')^{O(1)}$. Since the edges of the skeleton $K$ may not lie in $\tilde{E}$, the total number of crossings in which the edges of $K$ participate in $\psi_3$ remains at most $N \cdot (\log \check{m}')^{O(1)}$. As before, for every face $F \in \tilde{\mathcal{F}}'$, for every vertex $x \in V(G'_F)$, $\psi_3(x) \in F$. The number of uninteresting crossings in $\psi_3$ is now bounded by $|\tilde{E}|^2$, as every pair of edges in $\tilde{E}$ may now cross at most once. For every face $F \in \tilde{F}'$, $|\tilde{E}_F| = |E_F'''| = \sum_{R \in \mathcal{R}_F} |\delta_{G_F}(R)| \leq 8|E'_F|$. From Observation 9.47, $\sum_{F \in \tilde{\mathcal{F}}'} |E'_F| \leq N$. Therefore, $|\tilde{E}| \leq \sum_{F \in \tilde{\mathcal{F}}'} |\tilde{E}_F| \leq 8N$. The number of uninteresting crossings in $\psi_3$ is then bounded by $|\tilde{E}|^2 \leq 64N^2$. We conclude that the total number of crossings in $\psi_3$ is bounded by $(\mathsf{cr}(\varphi) + N^2) \cdot (\log \check{m}')^{O(1)}$. Lastly, we can modify solution $\psi_3$ to instance $I'$ to obtain a solution $\psi_1$ to instance $I$ by suppressing the degree-2 vertices that we used to subdivide some of the edges of graph $G$. It is immediate to verify that this

drawing has all required properties. In order to complete the proof of Claim 9.46, it is now enough to prove Claim 9.49, which we do next.

Consider a face $F \in \tilde{\mathcal{F}}'$. We partition the edges of $\tilde{E}_F$ into two subsets: set $\tilde{E}'_F$ containing all edges $(x,y)$ with $x \in V(H'_F)$ and $y \in Z_F$, and set $\tilde{E}''_F$ containing all remaining edges. For a cluster $R \in \mathcal{R}_F$, we denote $R^+ = R \cup \delta_{G'_F}(R)$ the augmentation of cluster $R$ with respect to graph $G'_F$. We also denote by $T_R = \left\{t_e^R \mid e \in \delta_{G_F}(R)\right\}$ the set of vertices that serve as endpoints of the edges of $\tilde{E}_F$ and lie in $R^+$. Note that every edge $e \in \tilde{E}''_F$ connects a vertex of $R_1^+$ to a vertex of $R_2^+$ for some pair $R_1, R_2 \in \mathcal{R}_F$ of distinct clusters.

For each face $F \in \tilde{\mathcal{F}}'$ and cluster $R \in \mathcal{R}_F$, we denote by $\chi(R)$ the set of all crossings in the drawing $\varphi'$ of $G'$ in which the edges of $R^+$ participate.

We first use the drawing $\varphi'$ of $G'$ to compute, for each cluster $R \in \mathcal{R}_F$, a drawing $\psi_{R^+}$ of graph $R^+$ inside a disc $D(R)$, with the images of the vertices of $T_R$ lying on the boundary of the disc. We then select a location inside the region $F$, next to its boundary, into which we plant the disc $D(R)$ together with the drawing $\psi_{R^+}$ that is contained in it. Lastly, we modify the images of the edges of $\tilde{E}$ so that they connect the new images of their endpoints. All these modifications exploit the sets $\mathcal{P}'_1(R)$ and $\mathcal{P}'_2(R)$ of paths that we have defined for every cluster $R \in \mathcal{R}_F$ and face $F \in \tilde{\mathcal{F}}'$. For a face $F \in \tilde{\mathcal{F}}'$ and a cluster $R \in \mathcal{R}_F$, we can now think of the paths in set $\mathcal{P}'_1(R)$ as routing the vertices of $T_R$ to vertices of $Z_F$ in graph $G'_F$ (after we discard the first edge from each such path), and similarly we can think of paths in $\mathcal{P}'_2(R)$ as routing a subset of at least $\lfloor |T_R|/64 \rfloor$ vertices of $T_R$ to vertices of $Z_F$ in graph $G'_F$. Recall that the paths in $\mathcal{P}'_1(R) \cup \mathcal{P}'_2(R)$ are internally disjoint from $V(R)$, and we can ensure that they are internally disjoint from $Z_F$. The paths in each set $\mathcal{P}'_1(R)$ cause congestion at most $O(\log^4 \check{m}')$, and the paths in $\bigcup_{R \in \mathcal{R}(F)} \mathcal{P}'_2(R)$ cause congestion at most $O(\log^5 \check{m}')$ in graph $G'_F$. Recall also that our transformation of the graph $G$ and the initial drawing $\varphi$ ensures that the image of every vertex $t \in Z_F$ in $\varphi'$ appears in the interior of the region $F$, very close to its boundary. If $e$ is the unique edge of $\tilde{E}'$ incident to $t$, then only a small segment of $\varphi'(e)$ that is incident to $\varphi'(t)$ is contained in $F$, and that segment does not participate in any crossings.

**Computing the Drawings $\psi_{R^+}$.** Consider a face $F \in \tilde{\mathcal{F}}'$ and a cluster $R \in \mathcal{R}_F$. We view the drawing $\varphi'$ of $G'$ as a drawing on the sphere. Recall that, from the definition of graph $H_F$, for every edge $e \in E(H_F)$, the image of $e$ in $\varphi'$ is disjoint from $F$. Consider the disc $D(J_F)$, that is associated with the core $J_F$. Recall that disc $D(J_F)$ is a disc that contains the image of the core $J_F$ in its interior, and the boundary of $D(J_F)$ closely follows the boundary of the region $F$ inside the region (see Figure 28(b)). We can assume w.l.o.g. that the images of all vertices of $Z_F$ lie on the boundary of the disc $D(J_F)$. We let $D(R)$ be the disc that is the complement of disc $D(J_F)$, that is, the boundaries of both discs are identical, but their interiors are disjoint. Note that, from the definition of the graph $H_F$, the images of all vertices and edges of graph $R^+$ lie in disc $D(R)$ in drawing $\varphi'$.

Consider a graph $\tilde{H}_R$, containing all edges and vertices of $R^+$, and all edges and vertices that lie on the paths of $\mathcal{P}'_1(R)$. Let $\varphi'(R)$ be the drawing of $\tilde{H}_R$ that is induced by the drawing $\varphi'$ of $G'$. We apply the algorithm from Corollary 4.38 to perform a type-2 uncrossing of the images of the paths in $\mathcal{P}'_1(R)$. The input to the algorithm is graph $\tilde{H}_R$, its subgraph $C = R^+$, and a set $\mathcal{Q} = \mathcal{P}'_1(R)$ of paths, together with the drawing $\varphi'(R)$ of $\tilde{H}_R$. We direct all paths in $\mathcal{P}'_1(R)$ away from the vertices of $T_R$. The algorithm computes a collection $\Gamma = \{\gamma(t) \mid t \in T_R\}$ of curves, where, for every vertex $t \in T_R$, curve $\gamma(t)$ originates at the image of $t$ in $\varphi'(R)$, and terminates at the image of some vertex of $Z_F$, which lies on the boundary of disc $D(R)$. For every pair $e, e' \in E(\tilde{H}_R)$ of distinct edges, let $N(e,e')$ denote the number of crossings in the drawing $\varphi'(R)$ between edges $e$ and $e'$. The algorithm from Corollary 4.38 also ensures that the curves in $\Gamma$ do not cross each other, and, for every edge $e \in E(R^+)$, the number of crossings between the image of $e$ in $\varphi'(R)$ and the curves in $\Gamma$ is bounded by $\sum_{e' \in E(G_R) \setminus E(R^+)} N(e,e') \cdot \mathrm{cong}_{G_F}(\mathcal{P}'_1(R), e') \leq (\log \check{m}')^{O(1)} \cdot \sum_{e' \in E(G_R) \setminus E(R^+)} N(e,e')$.

We are now ready to define the drawing $\psi_{R^+}$ of graph $R^+$. Recall that for every vertex $t \in T_R$, there is exactly one edge in $R^+$ that is incident to $t$, and we denote this edge by $e_t = (x_t, t)$, where $x_t \in V(R)$. The images of all vertices and edges of $R$ in $\psi_{R^+}$ remain the same as in $\varphi'(R)$ (which are in turn identical to those in $\varphi'$). For every vertex $t \in T_R$, the image of edge $e_t$ is obtained by concatenating the image of edge $e_t$ in $\varphi'(R)$ and the curve $\gamma_t \in \Gamma$. The resulting curve connects the image of vertex $x_t$ in the current drawing, to some point $p$ on the boundary of disc $D(R)$. The image of vertex $t$ becomes that point $p$. We note that the image of $e_t$ is contained in disc $D(R)$. This completes the definition of the drawing $\psi_{R^+}$ of graph $R^+$. This drawing obeys the rotation system $\Sigma'$, is contained in disc $D(R)$, with the vertices of $T_R$ lying on the boundary of the disc. From the above discussion, the total number of crossings in this drawing is bounded by $(\log \check{m}')^{O(1)} \cdot |\chi(R)|$. For convenience, we define another disc $D'(R)$, that contains $D(R)$, so that the boundaries of both discs are disjoint.

**Modifying the Drawing $\varphi'$ of $G'$** In this step we select, for every face $F \in \tilde{\mathcal{F}}'$ and every cluster $R \in \mathcal{R}_F$, a small disc $D''(R)$ in the interior of the region $F$ in $\varphi'$. We will then copy the contents of disc $D'(R)$ (including the drawing $\psi_{R^+}$) into the disc $D''(R)$, and extend the images of all edges of $\tilde{E}_F$ that are incident to the vertices of $T_R$, so that they terminate at the boundary of the disc $D''(R)$. We will then "stitch" the images of these edges inside the region $D''(R) \setminus D(R)$, so that the image of each edge terminates at the image of its endpoint.

Consider a face $F \in \tilde{\mathcal{F}}'$, and a cluster $R \in \mathcal{R}_F$. Since cluster $R$ has the $\alpha$-bandwidth property, for $\alpha = \frac{1}{\log^4 \check{m}'}$, from Observation 4.16, the set $T_R$ of vertices is $\alpha$-well-linked in $R^+$. We apply the algorithm from Lemma 4.27 to compute, for every vertex $t \in T$, a set $\mathcal{Q}_t = \{Q_t(t') \mid t' \in T_R \setminus \{t\}\}$ of paths, where, for all $t' \in T_R \setminus \{t\}$, path $Q_t(t')$ connects $t'$ to $t$. Let $\hat{T}_R \subseteq T_R$ be the set containing all vertices $t \in T_R$, such that some path of $\mathcal{P}'_2(R)$ originates from $t$. We then select a vertex $t_R \in \hat{T}_R$ uniformly at random, and we let $\mathcal{Q}_R = \{Q_t(t') \mid t' \in T_R \setminus \{t_R\}\}$ be a collection of path connecting every vertex in $T_R \setminus \{t_R\}$ to $t_R$. We need the following observation.

**Observation 9.50** *For every edge $e \in E(R^+)$, $\mathbf{E}\left[\mathrm{cong}(\mathcal{Q}_R, e)\right] \leq O(\log^8 \check{m}')$.*

**Proof:** Fix an edge $e \in E(R^+)$. From Lemma 4.27, if we were to select a vertex $t \in T_R$ uniformly at random, then $\mathbf{E}\left[\mathrm{cong}(\mathcal{Q}_t, e)\right] \leq O\left(\frac{\log^4 \check{m}'}{\alpha}\right) \leq O(\log^8 \check{m}')$. Clearly, in the above process, a vertex $t \in T_R$ is selected with probability $1/|T_R|$. Our algorithm instead selects a vertex $t_R \in \hat{T}_R$ uniformly at random, so a vertex $t \in \hat{T}_R$ is selected with probability $\frac{1}{|\hat{T}_R|} \leq \frac{128}{|T_R|}$, since $|\hat{T}_R| \geq \lfloor |T_R| \rfloor 64$. Therefore, $\mathbf{E}\left[\mathrm{cong}(\mathcal{Q}_R, e)\right] \leq 128 \mathbf{E}_{t \sim T_R}\left[\mathrm{cong}(\mathcal{Q}_t, e)\right] \leq O(\log^8 \check{m}')$. □

We construct another set $\mathcal{Q}'_R = \{Q'(t') \mid t' \in T_R\}$ of paths in graph $G'_F$, as follows. Consider the unique path $P'_2(t_R) \in \mathcal{P}'_2(R)$ that originates at vertex $t_R$. We denote by $z_R \in Z_F$ the other endpoint of path $P'_2(t_R)$; recall that the image of $z_R$ lies in region $F$, very close to its boundary. For every vertex $t' \in T_R \setminus \{t_R\}$, we let $Q'(t')$ be the path obtained by concatenating path $Q(t') \in \mathcal{Q}_R$ with path $P'_2(t_R)$. For vertex $t_R$, we simply set $Q'(t_R) = P'_2(t_R)$. The resulting set $\mathcal{Q}'_R = \{Q'(t') \mid t' \in T_R\}$ of paths is contained in graph $G'_F$, and connects every vertex of $T_R$ to vertex $z_R \in Z_F$. Moreover, the only vertex of $G'_F \setminus H'_F$ that lies on the paths of $\mathcal{Q}'_R$ is vertex $z_R$. We assume w.l.o.g. that the paths in $\mathcal{Q}'_R$ are simple. For every face $F \in \tilde{\mathcal{F}}'$, we denote $\mathcal{Q}(F) = \bigcup_{R \in \mathcal{R}_F} \mathcal{Q}'_R$. We need the following observation.

**Observation 9.51** *For every face $F \in \tilde{\mathcal{F}}'$, for every edge $e \in E(H'_F) \cup \tilde{E}'_F$, $\mathbf{E}\left[\mathrm{cong}_{G'_F}(\mathcal{Q}(F), e)\right] \leq O(\log^8 \check{m}')$.*

**Proof:** Consider some edge $e \in E(H'_F) \cup \tilde{E}'_F$. If there is some cluster $R \in \mathcal{R}_F$ with $e \in E(R)$, then denote $R_e = R$; otherwise we let $R_e$ be undefined. Recall that there are at most $O(\log^5 \check{m}')$

paths in $\bigcup_{R\in\mathcal{R}_F}\mathcal{P}'_2(R)$ that contain the edge $e$. Let $S(e)$ be the collection of pairs $(R,t)$, where $R\in\mathcal{R}_F\setminus\{R_e\}$, $t\in\hat{T}_R$, and the uniue path of $\mathcal{P}'_2(R)$ that originates at $t$ contains the edge $e$. From the above discussion, $|S(e)|\leq O(\log^5\check{m}')$. Consider now a pair $(R,t)\in S(e)$. The probability that vertex $t$ is selected as vertex $t_R$ is bounded by $\frac{1}{|\hat{T}_R|}\leq\frac{128}{|T_R|}$, since $|\hat{T}_R|\geq\left\lfloor\frac{|T_R|}{64}\right\rfloor$. If vertex $t$ is selected as $t_R$, then every path in set $\mathcal{Q}'_R$ may contain the edge $e$, and the number of such paths is $|T_R|$. Therefore, the expected number of paths in $\bigcup_{R\in\mathcal{R}_F\setminus\{R_e\}}\mathcal{Q}'_R$ that contain edge $e$ is at most $O(\log^5\check{m}')$. If cluster $R_e$ is defined, then $\mathbf{E}\left[\mathrm{cong}(\mathcal{Q}'_{R_e},e)\right]\leq\mathbf{E}\left[\mathrm{cong}(\mathcal{Q}_{R_e},e)\right]\leq O(\log^8\check{m}')$. Therefore, overall, $\mathbf{E}\left[\mathrm{cong}_{G'_F}(\mathcal{Q}(F),e)\right]\leq O(\log^8\check{m}')$. □

For every face $F\in\tilde{\mathcal{F}}'$ and cluster $R\in\mathcal{R}_F$, we let $D''(R)$ be a very small disc lying in the interior of region $F$ of $\varphi'$, right next to the image of vertex $z_R$. Notice that it is possible that for several distinct clusters $R,R'\in\mathcal{R}_F$, $z_R=z_{R'}$ holds. We ensure that all such discs in $\{D''(R)\}_{R\in\mathcal{R}_F}$ are mutually disjoint. Eventually, for every component $R\in\mathcal{R}_F$, we will plant the drawing $\psi_{R^+}$ inside disc $D''(R)$, so that the discs $D'(R)$ and $D''(R)$ will coincide. In order to modify the images of the edges of $\tilde{E}_F$, we define, for every cluster $R\in\mathcal{R}_F$, for every vertex $t\in T_R$, a curve $\gamma(t)$ connecting the image of $t$ in drawing $\varphi'$ to the boundary of disc $D''(R)$. Consider now some edge $e\in\tilde{E}_F$. Assume first that $e\in\tilde{E}''_F$, and that $e=(t,t')$, with $t\in T_R$ and $t'\in T_{R'}$, for some clusters $R,R'\in\mathcal{R}_F$. In order to define a new image of edge $e$, we start by concatenating the curve $\gamma(t)$ with the image of edge $e$ in $\varphi'$, and curve $\gamma(t')$, thereby obtaining a curve connecting a point on the boundary of disc $D''(R)$ to a point on the boundary of disc $D''(R')$. We then extend the curve within $D''(R)\setminus D(R)$ so that it originates at the new image of vertex $t$, and we similarly extend the curve within $D''(R')\setminus D(R')$ so that it terminates at the new image of vertex $t'$. Notice that all crossings between the images of the edges of $\tilde{E}$ are uninteresting crossings, and Claim 9.49 allows us to introduce arbitrary number of such crossings. However, we need to ensure that the number of new crossings between the new images of the edges of $\tilde{E}$ and the remaining edges of $G'$ is small. In order to do so, we need to ensure that the curves in sets $\{\gamma(t)\mid t\in T_R\}$, for all $F\in\tilde{\mathcal{F}}'$ and $R\in\mathcal{R}_F$ have few crossings with the images of edges of $G'$.

We now proceed to define the curves $\gamma(t)$, wich is the main component in the remainder of the proof. Intuitively, these curves will follow the images of the paths in $\mathcal{Q}(F)$, for all $F\in\tilde{\mathcal{F}}'$. In order to do so, for every face $F\in\tilde{\mathcal{F}}'$, for every edge $e\in E(H'_F)\cup\tilde{E}'_F$, let $N_e$ denote the number of paths in set $\mathcal{Q}(F)$ containing the edge $e$.

Let $G''$ be a new graph that is obtained from $G'$ as follows. For every face $F\in\tilde{\mathcal{F}}'$, for every edge $e\in E(H'_F)\cup\tilde{E}'_F$, we add a collection $J(e)$ of $N_e+1$ parallel copies of edge $e$ to the graph. We then let $\varphi''$ be a drawing of graph $G''$ that is obtained from the drawing $\varphi'$ of graph $G'$ in a natural way: for every face $F\in\tilde{\mathcal{F}}'$ and edge $e\in E(H'_F)\cup\tilde{E}'_F$, we add images of $N_e+1$ copies of edge $e$ in parallel to the original drawing of edge $e$, very close to it.

Consider a crossing $(e,e')_p$ in this new drawing $\varphi''$. We say that the crossing is of *type 1*, if there is some face $F\in\tilde{\mathcal{F}}'$, such that both $e$ and $e'$ are copies of edges that lie in $E(H'_F)\cup\tilde{E}'_F$. Otherwise, we say that the crossing is of *type 2*. If a crossing $(e_1,e_2)_p$ in $\varphi''$ is of type 2, then there is a crossing $(e'_1,e'_2)_{p'}$ in $\varphi'$ in the vicinity of point $p$, such that either $e'_1=e_1$, or $e_1$ is a copy of edge $e'_1$, and similarly, either $e'_2=e_2$, or $e_2$ is a copy of edge $e'_2$. We say that crossing $(e'_1,e'_2)_{p'}$ in $\varphi'$ is *responsible* for crossing $(e_1,e_2)_p$ in $\varphi''$. Since, from Observation 9.51, for every face $F\in\tilde{\mathcal{F}}'$ and edge $e\in E(H'_F)\cup\tilde{E}'_F$, $\mathbf{E}\left[\mathrm{cong}_{G'_F}(\mathcal{Q}(F),e)\right]\leq O(\log^8\check{m}')$, and since the random choices made when computing sets $\mathcal{Q}(F)$ of paths for different faces $F\in\tilde{\mathcal{F}}'$ are independent, the expected number of type-2 crossings in $\varphi''$ for which a single crossing in $\varphi'$ is responsible is bounded by $O(\log^{16}\check{m}')$. Therefore, the expected number of type-2 crossings in $\varphi''$ is at most $\mathsf{cr}(\varphi')\cdot O(\log^{16}\check{m}')$.

Consider a face $F\in\tilde{\mathcal{F}}'$. We can use the set $\mathcal{Q}(F)$ of paths in graph $G'$ in a natural way, in order

to define a set $\tilde{\mathcal{Q}}(F) = \left\{\tilde{Q}(t) \mid t \in \bigcup_{R\in\mathcal{R}_F} T_R\right\}$ of edge-disjoint paths in $G''$, where for every cluster $R \in \mathcal{R}_F$, for every vertex $t \in T_R$, path $\tilde{Q}(t)$ connects the image of $t$ in $\varphi'$ to vertex $z_R$. Since, for every edge $e \in E(H'_F) \cup \tilde{E}'_F$, we have added $N_e + 1$ copies of edge $e$ to $G''$, for every edge $e \in \tilde{E}_F$, there is a copy $e^* \in J(e)$ of edge $e$ (that we call *distinguished copy*), that does not belong to any path in $\tilde{\mathcal{Q}}(F)$. For every component $R \in \mathcal{R}_F$, for every vertex $t \in T_R$, we let $\gamma(t)$ be the image of path $\tilde{Q}(t)$ in $\varphi''$. Notice that curve $\gamma(t)$ connects the image of $t$ in $\varphi'$ to the image of vertex $z_R$. We slightly modify the final segment of $\gamma(t)$ so that it terminates at the boundary of disc $D''(R)$ (while ensuring that each such curve terminates at a different point on the boundary of the disc). We note that we are allowed to introduce arbitrary number of crossings between the curves in set $\left\{\gamma(t) \mid t \in \bigcup_{R\in\mathcal{R}_F} T_R\right\}$, as all such crossings will become unimportant crossings in the final drawing of graph $G'$ that we construct.

Consider now some edge $e \in \tilde{E}''_F$. Assume that $e = (t, t')$, with $t \in T_R$, $t' \in T_{R'}$, where $R, R' \in \mathcal{R}_F$ are two distinct clusters. We define a curve $\gamma'(e)$ representing the edge $e$ by concatenating the curve $\gamma(t)$, the image of the distinguished copy $e^*$ of $e$ in $\varphi''$; and curve $\gamma(t')$. Notice that the resulting curve $\gamma'(e)$ connects a point on the boundary of disc $D''(t)$ to a point on the boundary of disc $D''(t')$.

Next, we consider some edge $e \in \tilde{E}'_F$. Assume that $e = (t, z)$, with $z \in Z_F$, and $t \in T_R$, for some cluster $R \in \mathcal{R}_F$. We let $\gamma'(e)$ be the curve obtained by concatenating the image of the distinguished copy $e^*$ of $e$ in $\varphi''$ with the curve $\gamma(t)$. Therefore, curve $\gamma'(e)$ connects the image of vertex $z$ to a point on the boundary of disc $D''(t)$.

Consider the resulting set $\Gamma_F = \left\{\gamma'(e) \mid e \in \tilde{E}_F\right\}$ of curves. Notice that these curves may not be in general position, since it is possible that for some vertex $v \in V(H'_F)$, the point $\varphi''(v)$ lies on more than two such curves (for example, this can happen when $v$ lies on several paths in $\mathcal{Q}(F)$). In order to overcome this difficulty, for every vertex $v \in V(H'_F)$ with point $\varphi''(v)$ lying on more than two curves of $\Gamma_F$, we modify the curves of $\Gamma_F$ containing point $\varphi'(v)$ within the tiny $v$-disc $D_{\varphi'}(v)$, for example, as described in the nudging procedure (see Section 4.4.3). This may introduce new crossings between the curves in $\Gamma_F$, but since these crossings will eventually become unimportant crossings of drawing $\psi_2$, we can afford to introduce an arbitrary number of such crossings.

We are now ready to define the solution $\psi_2$ to instance $I'$. Let $\tilde{G} = G' \setminus \bigcup_{F\in\tilde{\mathcal{F}}'}(H'_F \cup \tilde{E}'_F)$. The drawing of graph $\tilde{G}$ in $\psi_2$ remains the same as in $\varphi'$. Consider now some face $F \in \tilde{\mathcal{F}}'$ and cluster $R \in \mathcal{R}_F$. We plant drawing $\psi_{R^+}$ inside disc $D''(R)$, so that the boundaries of discs $D'(R)$ and $D''(R)$ coincide. Recall that, in drawing $\psi_{R^+}$, the image of every vertex $t \in T_R$ appears on the boundary of the disc $D(R)$, and the images of all other vertices, and of all edges, are disjoint from $D'(R) \setminus D(R)$. It now remains to add the images of the edges in $\tilde{E}$ to this drawing. Consider again a face $F \in \tilde{\mathcal{F}}'$, and an edge $e \in \tilde{E}$. Initially, we let the image of $e$ be the curve $\gamma'(e) \in \Gamma_F$. We now need to modify this curve slightly so it connects the images of the endpoints of edge $e$. Assume first that $e \in \tilde{E}'_F$, and denote $e = (t, z)$, with $z \in Z_F$, and $t \in T_R$, for some cluster $R \in \mathcal{R}_F$. Then curve $\gamma'(e)$ connects the image of $z$ to a point on the boundary of disc $D''(R)$. Recall that the image of $t$ appears on the boundary of disc $D(R)$. We extend curve $\gamma'(e)$ within the region $D''(R) \setminus D(R)$, so that it terminates at the image of vertex $t$. Assume now that $e \in \tilde{E}''_F$, and denote $e = (t, t')$, where $t \in T_R$, $t' \in T_{R'}$, for some distinct clusters $R, R' \in \mathcal{R}_F$. Initially, we let the image of edge $e$ be the curve $\gamma'(e)$ that connects a point on the boundary of $D''(R)$ to a point on the boundary of $D''(R')$. We extend the curve inside the regions $D''(R) \setminus D(R)$ and $D''(R') \setminus D(R')$, so that it connects the image of $t$ to the image of $t'$. The extensions to the curves in $\Gamma_F$ can be performed so that they remain in general position; we may introduce an arbitrary number of new crossings between these curves, but we will not introduce any crossings between these curves and the images of the edges of $\tilde{G}$. This completes the definition of the solution $\psi_2$ to instance $I'$. We now ensure that this drawing has the required properties. Observe first that every interesting crossing in $\psi_2$ is either between a pair of edges in $\tilde{G}$ (and so it must be a type-2 crossing in drawing $\varphi''$ of $G''$); or it is a crossing between a pair of edges of graph $R^+$ for some

cluster $R \in \bigcup_{F \in \tilde{\mathcal{F}}'} \mathcal{R}_F$ (in which case it exists in drawing $\psi_{R^+}$); or it is a crossing between a curve in $\bigcup_{F \in \tilde{\mathcal{F}}'} \Gamma_F$ and an image of an edge of $\tilde{G}$ (in which case it corresponds to some type-2 crossing in drawing $\varphi''$ of $G''$). Therefore, if we denote by $\chi_2(\varphi'')$ the set of all type-2 crossings in drawing $\varphi''$, then we get that the number of interesting crossings in $\psi_2$ is bounded by:

$$|\chi_2(\varphi'')| + \sum_{F \in \tilde{\mathcal{F}}'} \sum_{R \in \mathcal{R}_F} \mathsf{cr}(\psi_{R^+}).$$

Recall that for every cluster $R \in \bigcup_{F \in \tilde{\mathcal{F}}'} \mathcal{R}_F$, $\mathsf{cr}(\psi_{R^+}) \leq |\chi(R)|$, where $\chi(R)$ the set of all crossings in the drawing $\varphi'$ of $G'$ in which the edges of $R^+$ participate. Therefore, $\sum_{F \in \tilde{\mathcal{F}}'} \sum_{R \in \mathcal{R}_F} \mathsf{cr}(\psi_{R^+}) \leq 2\mathsf{cr}(\varphi') \leq 2\mathsf{cr}(\varphi)$. Since, from the above discussion, the expected number of type-2 crossings in $\varphi''$ is bounded by $c \cdot \mathsf{cr}(\varphi') \cdot \log^{16} \check{m}' \leq c \cdot \mathsf{cr}(\varphi) \cdot \log^{16} \check{m}'$ for some large enough constant $c$, we get that the expected number of type-2 crossings in $\varphi''$ is at most $4c \cdot \mathsf{cr}(\varphi) \cdot \log^{16} \check{m}'$. We say that a bad event $\mathcal{E}_1$ happens if the number of type-2 crossings in $\varphi''$ is greater than $16c \cdot \mathsf{cr}(\varphi) \cdot \log^{16} \check{m}'$. From Markov's inequality, $\mathbf{Pr}[\mathcal{E}_1] \leq 1/4$.

Next, we bound $|\chi^*(\psi_2)|$ – the number of crossings in which the edges of the core $\check{J}$ participate. Notice that the edges of the core $\check{J}$ may not lie in $\bigcup_{F \in \tilde{\mathcal{F}}'} (H'_F \cup \tilde{E}'_F)$. Consider any crossing $(e, e')_p \in \chi^*(\psi_2)$, and assume that $e \in E(\check{J})$. Then either $e' \in E(\tilde{G})$ must hold (in which case crossing $(e, e')_p$ is a type-2 crossing in drawing $\varphi''$ of $G''$), or $e' \in \tilde{E}$ (in which case curve $\gamma'(e')$ crosses the image of $e$). Since curve $\gamma'(e')$ was constructed by following the images of one or two paths in $\bigcup_{F \in \tilde{\mathcal{F}}'} \tilde{Q}(F)$ in $\varphi''$, and using the image of edge $e'$ in $\varphi''$, there is a crossing $(e, e'')_p$ in drawing $\varphi''$, such that the image of edge $e''$ has a non-zero length intersection with curve $\gamma'(e')$.

Therefore, $|\chi^*(\psi_2)| \leq |\chi^*(\varphi'')|$. It now remains to bound the number of crossings in which the edges of $\check{J}$ participate in $\varphi''$. Consider any such crossing $(e, e'')_p$, with $e \in E(\check{J})$. Then either $e'' \in E(\tilde{G})$, and crossing $(e, e'')_p$ also exists in drawing $\varphi'$ of $G'$; or $e''$ is a copy of some edge $e' \in \bigcup_{F \in \tilde{\mathcal{F}}'} (H_F \cup \tilde{E}_F)$, and crossing $(e, e')_p$ also exists in drawing $\varphi'$ of $G'$. In the former case, we say that crossing $(e, e'')_p$ in $\varphi'$ is responsible for crossing $(e, e'')_p$ in $\varphi''$, while in the latter case we say that crossing $(e, e')_p$ in $\varphi'$ is responsible for crossing $(e, e'')_p$ in $\varphi''$. Consider now some crossing $(e, e')_p$ in drawing $\varphi'$. If $e' \in E(\tilde{G})$, then this crossing may be responsible for at most one crossing in $\varphi''$. Otherwise, there must be a face $F \in \tilde{\mathcal{F}}'$, with $e' \in E(H_F) \cup \tilde{E}_F$. In this case, crossing $(e, e')_p$ may be responsible for at most $N_{e'} + 1$ crossings in $\varphi''$. Since, from Observation 9.51, for every face $F \in \tilde{\mathcal{F}}'$, for every edge $e' \in E(H'_F) \cup \tilde{E}'_F$, $\mathbf{E}[N_{e'} + 1] = \mathbf{E}\left[\mathrm{cong}_{G'_F}(\mathcal{Q}(F), e') + 1\right] \leq O(\log^8 \check{m}')$, we get that $\mathbf{E}[|\chi^*(\psi_2)|] \leq c|\chi^*(\varphi')| \log^8 \check{m}' \leq c|\chi^*(\varphi)| \log^8 \check{m}'$ for some large enough constant $c$. We say that bad event $\mathcal{E}_2$ happens if $|\chi^*(\psi_2)| > 4c|\chi^*(\varphi)| \log^8 \check{m}'$. As before, from Markov inequality, $\mathbf{Pr}[\mathcal{E}_2] \leq 1/4$.

Lastly, we need to bound the number of crossings in which the edges of $E(K)$ participate in the drawing $\psi_2$. We denote by $N(\varphi')$ and $N(\psi_2)$ the number of crossings in which the edges of $E(K)$ participate in drawings $\varphi'$ and $\psi_2$, respectively. Recall that $N(\varphi')$ is bounded by the number of crossings in which the edges of $K$ participate in $\varphi$, which was denoted by $N$. From the definition, the edges of $E(K)$ may not lie in $\bigcup_{F \in \tilde{\mathcal{F}}'} (H'_F \cup \tilde{E}'_F)$. We can use the same argument that we used in bounding $|\chi^*(\psi_2)|$ to show that every crossing $(e, e'')_p$ in $\varphi''$ in which $e \in E(K)$ can be mapped to some crossing $(e, e')_p$ in $\varphi'$, such that the total number of crossings mapped to a single crossing $(e, e')_p$ is 1 if $e' \in E(\tilde{G})$ and $N_{e'} + 1$ otherwise. Using the same reasoning as above, we get that $\mathbf{E}[N(\varphi'')] \leq c \cdot N(\varphi') \cdot \log^8 \check{m}' \leq c \cdot N \cdot \log^8 \check{m}'$, where $c$ is a large enough constant. We say that bad event $\mathcal{E}_3$ happens if $N(\psi_2) > 4c \cdot N \cdot \log^8 \check{m}'$. As before, from Markov inequality, $\mathbf{Pr}[\mathcal{E}_3] \leq 1/4$.

Using the Union Bound, with probability at least $1/4$ neither of the bad events $\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3$ happens. Therefore, there must exist a solution $\psi_2$ to instance $I' = (G', \Sigma')$ that is $\mathcal{K}$-valid, such that, for every face $F \in \tilde{\mathcal{F}}'$, the images of all vertices of $G'_F$ lie in region $F$ of the drawing, the number of interesting crossings in $\psi_2$ is bounded by $4c \cdot \mathsf{cr}(\varphi) \cdot \log^{16} \check{m}'$, $|\chi^*(\psi_2)| \leq 4c \cdot |\chi^*(\varphi)| \cdot \log^8 \check{m}'$, and $N(\psi_2) \leq 4c \cdot N \cdot \log^8 \check{m}'$. This completes the proof of Claim 9.49 and of Claim 9.46. □

### Step 2: Moving the Bad Edges

In this step we consider the $\mathcal{K}$-valid solution $\psi_1$ to instance $I$ that is guaranteed to exist from Claim 9.46. Recall that for every face $F \in \tilde{\mathcal{F}}'$, the images of all vertices of $G_F$ lie in region $F$ of $\psi_1$. Additionally, $\mathsf{cr}(\psi_1) \leq (\mathsf{cr}(\varphi) + N^2) \cdot (\log \check{m}')^{O(1)}$, $|\chi^*(\psi_1))| \leq |\chi^*(\varphi)| \cdot (\log \check{m}')^{O(1)}$, and the total number of crossings in which the edges of $K$ participate in $\psi_1$ is at most $N \cdot (\log \check{m}')^{O(1)}$.

For every face $F \in \tilde{\mathcal{F}}'$, we define a disc $D(F)$, that is contained in region $F$ of $\psi_1$, so that the boundary of disc $D(F)$ closely follows the boundary of the region $F$. Equivalently, we can think of disc $D(F)$ as the complement of the disc $D(J_F)$ associated with the core $J_F$ (see e.g. Definition 9.4); the boundaries of both discs coincide, and their interiors are disjoint. Note that, if we traverse the boundary of the disc $D(F)$ in counter-clock direction, we encounter the images of the edges $\delta_{G_F}(J_F)$ in the oriented circular ordering $\mathcal{O}(J_F)$. Notice however that images of additional edges of $G$ may cross the boundary of disc $D(F)$ – edges whose images cross the image of some edge in $E(J_F)$. As before, we say that an edge of $G$ is *bad* if its image in $\psi_1$ crosses the image of some edge of the original core $\check{J}$. Note that for every bad edge $e = (u, v)$, there must be a face $F \in \tilde{\mathcal{F}}'$ with $e \in E(G_F)$. Intuitively, in this step, for each face $F \in \tilde{\mathcal{F}}'$, we will modify the drawing that is contained in disc $D(F)$ in $\psi_1$, and add the images of the bad edges $e \in E(G_F)$ to this drawing. In order to do so in a modular fashion, so that the resulting drawings for different faces $F$ can be "glued" together, we will define a new graph that is associated with the face $F$, that, intuitively, will contain all vertices and (segments of) edges that are drawn inside disc $D(F)$ in $\psi_1$. We will exclude all bad edges that do not lie in $G_F$, and include all bad edges that lie in $G_F$. We will also subdivide the image of every edge that crosses the boundary of disc $D(F)$ with a new vertex, whose image will be placed on the boundary of disc $D(F)$. We view these new vertices as "anchors". Intuitively, we will allow the image of the graph associated with face $F$ to be modified arbitrarily, as long as it is contained in disc $D(F)$, and as long as the images of the anchors remain unchanged. This will allow us to replace the part of the drawing of graph $G$ that is contained in $D(F)$ with a new drawing, to which the bad edges that lie in $G_F$ are added.

We start by constructing a new graph $\tilde{G}$, that is obtained from graph $G$, by subdividing some edges of $G$. Notice that, if graph $\tilde{G}$ is obtained from graph $G$ in this way, then the rotation system $\Sigma$ for $G$ naturally defines a rotation system $\tilde{\Sigma}$ for $\tilde{G}$. We then obtain an instance $\tilde{I} = (\tilde{G}, \tilde{\Sigma})$ of $\mathsf{MCNwRS}$ problem that we call *instance defined by graph* $\tilde{G}$. We will also define a solution $\tilde{\varphi}$ to instance $\tilde{I}$, that will be obtained from solution $\psi_1$ to instance $I$ in a natural way. Initially, we start with $\tilde{G} = G$ and $\tilde{\varphi} = \psi_1$. For every face $F \in \tilde{\mathcal{F}}'$, we also construct a set $A_F$ of *anchor vertices*, whose image in $\tilde{\varphi}$ appears on the boundary of the disc $D(F)$. Initially, we set $A_F = \emptyset$ for all $F \in \tilde{\mathcal{F}}'$.

Consider some face $F \in \tilde{\mathcal{F}}'$, an an edge $e = (x, y)$ that is incident to some vertex of the core $J_F$. Recall that exactly one endpoint of $e$ (say $x$) belongs to the core $J_F$. We subdivide edge $e$ with a new vertex $t_e$, so that edge $e$ is replaced with a path $(x, t_e, y)$. We also subdivide the image of edge $e$ in $\tilde{\varphi}$ by placing the image of vertex $t_e$ on the first intersection point of $\tilde{\varphi}(e)$ with the boundary of the disc $D(F)$, as we traverse $\tilde{\varphi}(e)$ from $x$ to $y$. We add vertex $t_e$ to the set $A_F$ of anchor vertices. In our final graph $\tilde{G}$, we will delete the edge $(x, t_e)$, and we will view edge $(t_e, y)$ as representing the original edge $e$.

Consider the current graph $\tilde{G}$, and its current drawing $\tilde{\varphi}$. We say that an edge $e \in E(\tilde{G})$ is *good* if its image in $\tilde{\varphi}$ does not cross the image of any edge in $K$. Since we have subdivided the edges incident to the vertices of $K$ that do not lie in $E(K)$, for every face $F \in \tilde{\mathcal{F}}'$, every edge of $\tilde{G}$ that is incident to a vertex of $J_F$ is a good edge. We say that an edge $e \in E(\tilde{G})$ is *bad* if its image in the current drawing $\tilde{\varphi}$ crosses the image of some edge of the original core $\check{J}$. Notice that the total number of bad edges is bounded by $|\chi^*(\tilde{\varphi})| \leq |\chi^*(\psi_1)| \leq |\chi^*(\varphi)| \cdot (\log \check{m}')^{O(1)}$ from Claim 9.46. Notice also that for every bad edge $e \in E(\tilde{G})$, there must be a face $F \in \tilde{\mathcal{F}}'$, such that the images of both endpoints of $e$ lie in the disc $D(F)$. Lastly, we say that an edge $e \in E(\tilde{G})$ is a *migrating edge* if it is neither good nor bad.

In this case, the image of $e$ must cross the image of some edge in $E(K) \setminus E(\check{J})$. Next, we process all bad edges and all migrating edges.

Consider first a bad edge $e = (x, y)$, and assume that the images of both $x$ and $y$ lie inside disc $D(F)$ for some face $F \in \tilde{\mathcal{F}}'$. Then the image of edge $e$ in $\tilde{\varphi}$ must intersect the boundary of the disc $D(F)$ in at least two points. We direct the image of $e$ in $\tilde{\varphi}$ from $x$ to $y$, denote by $p(e)$ the first point on the boundary of disc $D(F)$ that lies on $\tilde{\varphi}(e)$, and by $p'(e)$ the last point on the boundary of disc $D(F)$ that lies on $\tilde{\varphi}(e)$. If $p(e)$ is the image of the vertex $x$, then we denote $t_e = x$ (in this case, $x$ is already added to the set $A_F$ of anchor vertices for face $F$). Otherwise, we subdivide the edge $e$ with a new vertex $t_e$, whose image is placed at point $p(e)$, and we add $t_e$ to the set $A_F$ of anchor vertices. Similarly, if $p'(e)$ is the image of the vertex $y$ in $\tilde{\varphi}$, then we denote $t'_e = y$. Otherwise, we subdivide the edge $e$ with a new vertex $t'_e$, whose image is placed at point $p'(e)$, and we add $t'_e$ to the set $A_F$ of anchor vertices. If edge $e$ has been subdivided twice, then we have replaced it with path $(x, t_e, t'_e, y)$. If $x \neq t_e$, then edge $(x, t_e)$ now becomes a good edge. This edge does not represent any edge in the original graph $G_F$, so we call it an *extra edge*. Similarly, if $y \neq t'_e$, then edge $(y, t'_e)$ now becomes a good edge, and we also call it an extra edge. The edge $(t_e, t'_e)$ is a bad edge, and we view it as representing the bad edge $e$. Note that the images of both endpoints of this edge now appear on the boundary of disc $D(F)$. We say that face $F$ *owns* this bad edge.

Lastly, we consider a migrating edge $e = (x, y)$. Note that there must be a face $F \in \tilde{\mathcal{F}}'$, such that the images of both $x$ and $y$ lie in disc $D(F)$ in $\tilde{\varphi}$. We direct the image of the edge $e$ in $\tilde{\varphi}$ from $x$ to $y$. Denote by $F_1, F_2, \ldots, F_r$ the sequence of the regions of $\tilde{\mathcal{F}}'$ that the image of the edge $e$ visits, in the order in which it visits them, so $F_1 = F$ and $F_r = F$. For all $1 < i < r$, we let $\sigma_i$ be the maximal segment on the image of $e$ that is contained in disc $D(F_i)$, and we denote by $p_i(e)$ and $p'_i(e)$ the first and the last endpoints of $\sigma_i$, respectively, that must lie on the boundary of disc $D(F_i)$. We also let $\sigma_1$ the segment of $\tilde{\varphi}(e)$ from the image of $x$ to the first point that lies on the boundary of disc $D(F)$, denoting by $p'_1(e)$ the endpoint of $\sigma_1$ that is different from the image of $x$. Similarly, we let $\sigma_r$ be the segment of $\tilde{\varphi}(e)$ from the last point that lies on the boundary of disc $D(F)$ to the image of $y$, denoting by $p_r(e)$ the endpoint of $\sigma_r$ that is different from the image of $y$. Note that $\tilde{\varphi}(e) \setminus (\bigcup_{i=1}^{r} \sigma_i)$ is a collection of short segments, each of which lies outside of $\bigcup_{F \in \tilde{\mathcal{F}}'} D(F')$, and crosses some edge of $K$. We subdivide edge $e$, replacing it with a path $(x, t'_1, t_2, t'_2, \ldots, t_r, y)$. For all $1 < i \leq r$, we place the image of the new vertex $t_i$ at point $p_i(e)$, and for all $1 \leq i < r$, we place the image of the new vertex $t'_i$ at point $p'_i(e)$. Denote $t_1 = x$ and $t'_r = y$. For $1 \leq i \leq r$, denote by $e_i$ the new edge $(t_i, t'_i)$, and for $1 \leq i < r$, denote by $e'_i$ the new edge $(t'_i, t_{i+1})$. Notice that, for all $1 \leq i \leq r$, the image of edge $e_i$ is precisely the segment $\sigma_i$, which is contained in disc $D(F_i)$. Both endpoints of the edge are drawn on the boundary of the disc $D(F_i)$ (except that, for $i = 1$, the first endpoint of $\sigma_1$ is the image of $x$ that may lie in the interior of disc $D(F)$, and, for $i = r$, the last endpoint of $\sigma_r$ is the image of vertex $y$ that may lie in the interior of disc $D(F)$). For all $1 \leq i \leq r$, edge $e_i$ now becomes a good edge, and we also call it an *extra edge*. As discussed above, the images of edges $e'_1, \ldots, e'_{r-1}$ are short segments that lie outside of $\bigcup_{F' \in \tilde{\mathcal{F}}'} D(F')$ (except for their endpoints that lie on the boundaries of the discs), and each such segment crosses some edge of $K$. For all $1 < i < r$, we add the new vertices $t_i$ and $t'_i$ to the set $A_{F_i}$ of anchor vertices for face $F_i$. Additionally, we add $t'_1$ and $t_r$ to $A_F$.

Consider the final graph $\tilde{G}$ obtained after all bad and migrating edges have been procesed, and the resulting solution $\tilde{\varphi}$ to the instance $\tilde{I}$ defined by $\tilde{G}$. From the above discussion, the total number of extra edges in $\tilde{G}$ is bounded by $2N(\psi_1) + 2|\chi^*(\psi_1)|$, where $N(\psi_1)$ is the number of crossings in which the edges of the skeleton $K$ participate in $\psi_1$. Recall that $N(\psi_1) \leq N \cdot (\log \check{m}')^{O(1)} \leq \frac{\check{m}'}{\mu^{3b}} \cdot (\log \check{m}')^{O(1)}$, and $|\chi^*(\psi_1)| \leq |\chi^*(\psi)| \cdot (\log \check{m}')^{O(1)} \leq \frac{\check{m}'}{\mu^{240b}} \cdot (\log \check{m}')^{O(1)}$. Therefore, the total number of exta edges in graph $\tilde{G}$ is bounded by $\frac{\check{m}'}{\mu^{2b}}$.

For every face $F \in \tilde{\mathcal{F}}'$, we now define a subgraph $\tilde{G}_F$ of graph $\tilde{G}$ associated with face $F$. The set

of vertices of $\tilde{G}_F$ contains all vertices whose images appear in disc $D(F)$ in the current drawing $\tilde{\varphi}$. Notice that this includes all vertices in the original graph $G_F$ (except for vertices that belong to the core $J_F$), and, additionally, new vertices whose images were added to the boundary of $D(F)$, and were added to the set $A_F$ of anchor vertices. Therefore, $V(\tilde{G}_F) = (V(G_F) \setminus V(J_F)) \cup A_F$. The set of edges consists of all edges of $\tilde{G}$ whose image in $\tilde{\varphi}$ is contained in the disc $D(F)$, and all bad edges that belong to face $F$ (that is, all bad edges whose both endpoints lie in $A_F$). Notice that, if $e$ is an edge of $\tilde{G}_F$, then either $e$ is an edge of $G_F$, or it was obtained by subdividing some edge of $G_F$, or $e$ was obtained by subdividing some migrating edge $e'$ that lied in some graph $G_{F'}$ for $F' \neq F$. In the latter case, the endpoints of $e'$ belong to the set $A_F$ of anchors, and edge $e'$ with its endpoints forms a separate connected component in graph $\tilde{G}_F$. We need the following observation.

**Observation 9.52** *Let $F \in \tilde{\mathcal{F}}'$ be a face, and let $(S,T)$ be a partition of the set $A_F$ of the anchor vertices, so that the images of the vertices of $S$ in $\tilde{\varphi}$ appear consecutively on the boundary of disc $D(F)$. Then there is a collection $E'$ of at most $4\check{m}'/\mu^{2b}$ edges in graph $\tilde{G}_F$, so that there is no path connecting a vertex of $S$ to a vertex of $T$ in $\tilde{G}_F \setminus E'$.*

**Proof:** Assume otherwise. From the max-flow min-cut theorem, there is a collection $\mathcal{P}$ of $\left\lceil 4\check{m}'/\mu^{2b} \right\rceil$ edge-disjoint paths in graph $\tilde{G}_F$ connecting vertices of $S$ to vertices of $T$. Since there are at most $\check{m}'/\mu^{2b}$ extra edges in graph $\tilde{G}$, there is a subset $\mathcal{P}' \subseteq \mathcal{P}$ of at least $\left\lceil 2\check{m}'/\mu^{2b} \right\rceil$ paths that do not contain extra edges. Therefore, every path in $\mathcal{P}'$ only contains edges that were obtained by subdividing the edges of $G_F$. Observe that the anchor vertices representing the edges in $\delta_{G_F}(J_F)$ appear on the boundary of disc $D(F)$ in drawing $\tilde{\varphi}$ according to the ordering $\mathcal{O}(J_F)$. Therefore, partition $(S,T)$ of vertices of $A_F$ naturally induces a partition $(E_1, E_2)$ of the set $\delta_{G_F}(J_F)$ of edges, where the edges of $E_1$ appear consecutively in the ordering $\mathcal{O}(J_F)$. But then the existence of the set $\mathcal{P}'$ of paths contradicts the fact that $I_F$ is an acceptable instance (see Definition 9.41). □

For a face $F \in \tilde{\mathcal{F}}'$, we denote by $\chi(F)$ the set of all crossings in the drawing $\tilde{\varphi}$ in which the edges of $\tilde{G}_F$ participate, and we denote by $E^{\mathrm{bad}}(F)$ the set of all bad edges that face $F$ owns. The proof of Lemma 9.43 follows from the following claim.

**Claim 9.53** *For every face $F \in \tilde{\mathcal{F}}'$, there is a solution $\psi_F$ to instance $\tilde{I}_F$, with:*

$$\mathsf{cr}(\psi_F) \leq \left( |\chi(F)| + |E^{\mathrm{bad}}(F)|^2 + |E^{\mathrm{bad}}(F)| \cdot \frac{\check{m}'}{\mu^{2b}} \right) \cdot (\log \check{m}')^{O(1)},$$

*such that the drawing of the graph $\tilde{G}_F$ is contained in disc $D(F)$, and, for every anchor vertex $t \in A_F$, the image of $t$ in $\psi_F$ is identical to its image in $\tilde{\varphi}$.*

We provide the proof of Claim 9.53 below, after we complete the proof of Lemma 9.43 using it. We start from the solution $\tilde{\varphi}$ to instance $\tilde{I}$. For every face $F \in \tilde{\mathcal{F}}'$, we delete the contents of the disc $D(F)$, and replace them with the contents of disc $D(F)$ in drawing $\psi_F$ of graph $\tilde{G}_F$. Since the images of the anchor vertices of $A_F$ remain unchanged, once every face $F \in \tilde{\mathcal{F}}'$ is processed, we obtain a valid solution to instance $\tilde{I}$, that we denote by $\tilde{\psi}'$. Since, for every face $F \in \tilde{\mathcal{F}}'$, for every bad edge $e \in E^{\mathrm{bad}}(F)$, the image of $e$ now lies in disc $D(F)$, for every bad face $F' \in \tilde{\mathcal{F}}^{\times}$, the image of every edge of $\tilde{G}$ in $\tilde{\psi}'$ is disjoint from the interior of $F'$. Since graph $\tilde{G}$ was obtained from graph $G$ by subdividing some of its edges, solution $\tilde{\psi}'$ to instance $\tilde{I}$ naturally defines a solution $\psi$ to instance $I$, by suppressing the images of vertices that were used to subdivided edges. From the above discussion, the resulting

solution $\psi$ to instance $I$ is clean with respect to $\check{\mathcal{J}}$. Moreover:

$$\begin{aligned}
\mathsf{cr}(\psi) &\leq \mathsf{cr}(\tilde{\varphi}') \\
&\leq \mathsf{cr}(\tilde{\varphi}) + \sum_{F\in\tilde{\mathcal{F}}'} \mathsf{cr}(\psi_F) \\
&\leq \mathsf{cr}(\tilde{\varphi}) + \sum_{F\in\tilde{\mathcal{F}}'} \left(|\chi(F)| + |E^{\mathrm{bad}}(F)|^2 + |E^{\mathrm{bad}}(F)|\cdot\frac{\check{m}'}{\mu^{2b}}\right)\cdot(\log\check{m}')^{O(1)} \\
&\leq \mathsf{cr}(\tilde{\varphi})\cdot(\log\check{m}')^{O(1)} + \sum_{F\in\tilde{\mathcal{F}}'} |E^{\mathrm{bad}}(F)|^2\cdot(\log\check{m}')^{O(1)} + |\chi^*(\psi_1)|\cdot\frac{\check{m}'\cdot(\log\check{m}')^{O(1)}}{\mu^{2b}} \\
&\leq (\mathsf{cr}(\varphi)+N^2)\cdot(\log\check{m}')^{O(1)} + |\chi^*(\psi_1)|^2\cdot(\log\check{m}')^{O(1)} + |\chi^*(\varphi)|\cdot\frac{\check{m}'\cdot(\log\check{m}')^{O(1)}}{\mu^{2b}} \\
&\leq \left(\mathsf{cr}(\varphi)+N^2+|\chi^*(\varphi)|^2+\frac{\check{m}'\cdot|\chi^*(\varphi)|}{\mu^{2b}}\right)(\log\check{m}')^{O(1)}.
\end{aligned}$$

In order to complete the proof of Lemma 9.43, it is now enough to prove Claim 9.53, which we do next.

### 9.4.6 Proof of Claim 9.53

We fix a face $F\in\tilde{\mathcal{F}}'$. For convenience, we denote graph $\tilde{G}_F$ by $G$, and the corresponding instance $\tilde{I}_F=(\tilde{G}_F,\tilde{\Sigma}_F)$ of MCNwRS by $I=(G,\Sigma)$. We also denote the solution to instance $I$ induced by drawing $\tilde{\varphi}$ of $\tilde{G}$ by $\varphi$, so $|\chi(F)|\geq\mathsf{cr}(\varphi)$. Recall that we are given a disc $D(F)$, that we denote by $D$, and a collection $A_F$ of anchor vertices (that we denote by $A$). The images of all vertices of $A$ in $\varphi$ lie on the boundary of the disc $D$, and the images of all other vertices of $G$ lie in the interior of the disc. The set of edges of $G$ is partitioned into two subsets: set $E^{\mathrm{bad}}$ of bad edges, and set $E^{\mathrm{good}}$ of all remaining edges, that we refer to as good edges. The images of all good edges in $\varphi$ are contained in disc $D$. For every bad edge $e\in E^{\mathrm{bad}}$, the endpoints of $e$ (which must be anchor vertices) lie on the boundary of disc $D$.

Our goal is to show that there exists another solution $\psi$ to instance $I$, in which the images of the anchor vertices remain unchanged from $\varphi$, the images of all vertices and edges of $G$ are contained in disc $D$, and $\mathsf{cr}(\psi)\leq\left(\mathsf{cr}(\varphi)+|E^{\mathrm{bad}}|^2+|E^{\mathrm{bad}}|\cdot\frac{\check{m}'}{\mu^{2b}}\right)\cdot(\log\check{m}')^{O(1)}$. Recall that, from Observation 9.52, for any partition $(S,T)$ of the vertices of $A$, so that the vertices of $S$ appear consecutively on the boundary of $D$ in $\varphi$, there is a set $E'$ of at most $4\check{m}'/\mu^{2b}$ edges in $G$, so that there is no path connecting a vertex of $S$ to a vertex of $T$ in $G\setminus E'$.

We let $A'\subseteq A$ be the set of vertices that serve as endpoints of the bad edges. Note that the edges of $E^{\mathrm{bad}}$ define a perfect matching between the vertices of $A'$. We then let $\Pi=\{\varphi(v)\mid v\in A'\}$ be the set of points that serve as images of the vertices of $A'$. Let $r$ be the smallest integer, so that $|\Pi|\leq 2^r$. Clearly, $2^r\leq 4|E^{\mathrm{bad}}|\leq 4\check{m}'$, and $r\leq\log(4\check{m}')$. We add additional arbitrary points on the boundary of the disc $D$ to set $\Pi$ until $\Pi$ contains $2^r+1$ distinct points. We denote $\Pi=\{p_0,p_1,\ldots,p_{2^r}\}$, and we assume that the points appear on the boundary of disc $D$ in this order, as we traverse the boundary in counter-clock-wise direction.

Next, we define a number of guiding curves, that we call *corridors*. We will ensure that all these curves are disjoint, except for possibly sharing their endpoints. The curves are partitioned into $r$ levels.

The set $\Lambda_0$ of level-0 curves contains, for all $0\leq i<2^r$, a curve $\lambda_{0,i}$, that connects point $p_i$ to point $p_{i+1}$, and is contained in the interior of disc $D$ (except for its two endpoints that lie on the disc boundary). We ensure that all curves in set $\Lambda_0$ are disjoint from each other. Note that for all

$0 \leq i < 2^r$, points $p_i$ and $p_{i+1}$ partition the boundary of the disc $D$ into two segments. Let $\sigma_{0,i}$ be the segment that is disjoint from point $p_{i+2}$. Then we can define a disc $D_{0,i} \subseteq D$ that corresponds to curve $\lambda_{0,i}$, whose boundary is the concatentation of curves $\sigma_{0,i}$ and $\lambda_{0,i}$.

For a level $0 < j < r$, we consider the points in $\left\{p_{i\cdot 2^j} \mid 0 \leq i \leq 2^{r-j}\right\}$, and we connect every consecutive pair of such points with a curve. Specifically, the set $\Lambda_j$ of level-$j$ curves contains, for all $0 \leq i < 2^{r-j}$, a curve $\lambda_{j,i}$, that connects point $p_{i\cdot 2^j}$ to point $p_{(i+1)\cdot 2^j}$. We draw these curves so that they are internally disjoint from each other and from the curves in $\Lambda_0 \cup \cdots \cup \Lambda_{j-1}$, and every curve is contained in the interior of the disc $D$ (except for its endpoints that lie on the disc's boundary).

As before, for every index $0 \leq i < 2^{r-j}$, points $p_{i\cdot 2^j}$, $p_{(i+1)\cdot 2^j}$ partition the boundary of the disc $D$ into two segments. We denote by $\sigma_{j,i}$ the segment that does not contain the point $p_{(i+1)\cdot 2^j+1}$. We let $D_{j,i}$ be the disc that is contained in $D$, whose boundary is the concatenation of curves $\sigma_{j,i}$ and $\lambda_{j,i}$ (see Figure 34).

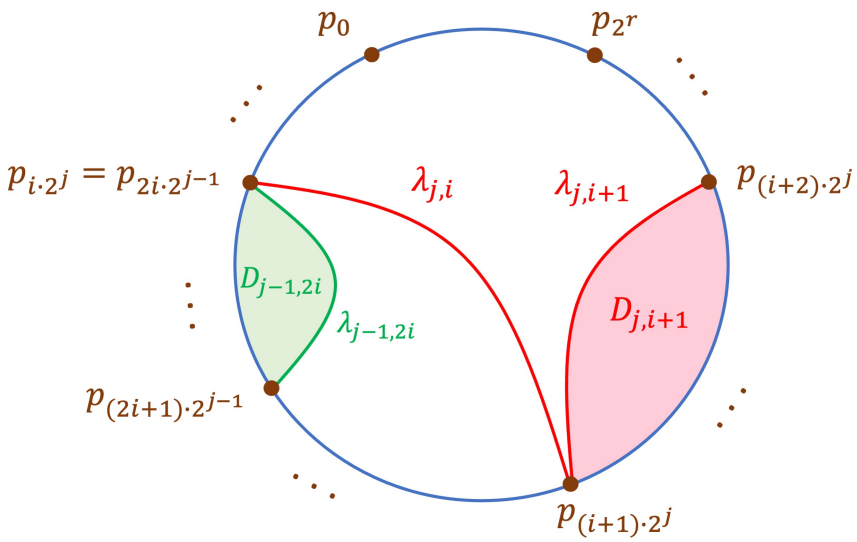



Figure 34: Level-$j$ curves $\lambda_{j,i}$ and $\lambda_{j,i+1}$, with disc $D_{j,i+1}$ shown in red, and a level-$(j-1)$ curve $\lambda_{j-1,2i}$, with disc $D_{j-1,2_i}$ shown in green.

Lastly, the set $\Lambda_r$ of level-$r$ curves contains a single curve $\lambda_{r,0}$ that connects points $p_0$ and $p_{2^r}$. We ensure that this curve is internally disjoint from all curves in $\Lambda_0 \cup \cdots \cup \Lambda_{r-1}$, and is contained in the interior of $D$, except for its endpoints that lie on the boundary of the disc. We define a disc $D_{r,0}$ associated with this curve exactly like before, so points $p_0, \ldots, p_r$ lie on the boundary of $D_{r,0}$.

We denote by $\Lambda = \bigcup_{j=0}^{r} \Lambda_j$. Let $G' = G \setminus E^{\mathrm{bad}}$, and let $I' = (G', \Sigma')$ be the subinstance of $I$ that is defined by graph $G'$. The crux of the proof of Claim 9.53 is the following claim, that allows us to "rearrange" the image of graph $G'$, so that each guiding curve only crosses a small number of edges.

**Claim 9.54** *There is a solution $\psi'$ to instance $I'$, with $\mathsf{cr}(\psi') \leq \mathsf{cr}(\varphi) \cdot (\log \check{m}')^{O(1)}$, so that the images of all vertices and edges of $G'$ lie in disc $D$, and the images of the anchor vertices in set $A$ remain the same as in $\varphi$. Moreover, for every curve $\lambda \in \Lambda$, for every vertex $v \in V(G')$, the image of $v$ in $\psi'$ does not lie on an inner point of $\lambda$; for every edge $e \in E(G')$, the image of $e$ in $\psi'$ may intersect $\lambda$ in at most one point; and the total number of edges in $E(G')$ whose images intersect $\lambda$ is at most $4\check{m}'/\mu^{2b}$.*

We provide the proof of Claim 9.54 in Section H.9 of Appendix. Given two points $p_i, p_{i'} \in \Pi$, a *tunnel* connecting $p_i$ to $p_{i'}$ is a sequence $L = (\lambda^1, \ldots, \lambda^z)$ of curves of $\Lambda$, such that the concatenation of the

curves in $L$ is a simple curve connecting points $p_i$ and $p_{i'}$. The *length* of the tunnel is $z$ – the number of curves in the sequence. We also need the following simple observation, whose proof appears in Section H.11 of Appendix.

**Observation 9.55** *For every pair $p_i, p_{i'}$ of distinct points of $\Pi$, there is a tunnel of length $O(\log \check{m}')$ connecting $p_i$ to $p_{i'}$.*

The proof of Claim 9.53 easily follows from Claim 9.54 and Observation 9.55. We start with the solution $\psi'$ to instance $I'$, that is given by Claim 9.54. Recall that $\mathsf{cr}(\psi') \leq \mathsf{cr}(\varphi) \cdot (\log \check{m}')^{O(1)}$, the image of $G'$ is contained in disc $D$, and the images of the anchor vertices in set $A$ in $\psi'$ are identical to those in $\varphi$. Next, we consider the bad edges one by one, and insert them into the drawing $\psi'$. Consider any such bad edge $e = (x, y) \in E^{\mathrm{bad}}$. Recall that there are two points $p_i, p_{i'} \in \Pi$, such that $\psi'(x) = \varphi(x) = p_i$, and $\psi'(y) = \varphi(x) = p_{i'}$. From our construction of graph $\tilde{G}$, vertices $x$ and $y$ each have degree 2 in $G$. Let $L = (\lambda^1, \ldots, \lambda^z)$ be a tunnel connecting $p_i$ to $p_{i'}$, with $z \leq O(\log \check{m}')$, that is given by Observation 9.55. We let the image of the edge $e$ to be a simple curve that connect $p_i$ to $p_{i'}$, and closely follows the image of the curve $\gamma(L)$, obtained by concatenating all curves in $L$, next to this curve. From Claim 9.54, this new image of edge $e$ crosses the images of at most $O\left(\frac{\check{m}' \cdot \log \check{m}'}{\mu^{2b}}\right)$ edges of $G'$. We allow the images of the edges of $E^{\mathrm{bad}}$ to cross arbitrarily.

Once all bad edges are processed, we obtain a solution $\psi$ to instance $G$, where the images of all vertices and edges are contained in $D$, and the images of the anchor vertices in $A$ are identical to those in $\varphi$. The number of crossings between pairs of edges in $E(G')$ is at most $\mathsf{cr}(\varphi) \cdot (\log \check{m}')^{O(1)}$; the number of crossings between edges of $E^{\mathrm{bad}}$ and edges of $E(G')$ is at most $O\left(\frac{|E^{\mathrm{bad}}| \cdot \check{m}' \cdot \log \check{m}'}{\mu^{2b}}\right)$; and the number of crossings between the edges of $E^{\mathrm{bad}}$ may be arbitrary. As our last step, we perform a type-1 uncrossing of the images of the bad edges (see Theorem 4.33). This procedure locally modifies the images of the bad edges by swapping segments between pairs of images of these edges (see Figure 7). At the end of this procedure, we are guaranteed that every pair of edges in $E^{\mathrm{bad}}$ cross at most once, and the number of crossings between the new images of the edges in $E^{\mathrm{bad}}$ and the images of the edges in $E(G')$ does not grow. Therefore, the number of crossings in this final solution to instance $I$ is bounded by $\mathsf{cr}(\varphi) \cdot (\log \check{m}')^{O(1)} + O\left(\frac{|E^{\mathrm{bad}}| \cdot \check{m}' \cdot \log \check{m}'}{\mu^{2b}}\right) + |E^{\mathrm{bad}}|^2$, as required.

# 10 An Algorithm for Narrow Instances – Proof of Theorem 3.14

We assume that we are given a narrow instance $I = (G, \Sigma)$ of the MCNwRS problem. Throughout this section, we denote $|E(G)| = m$. We fix some optimal solution $\varphi^*$ to instance $I$. We will gradually construct the desired family $\mathcal{I}$ of instances, over the course of three phases. We will employ partitions of graphs into clusters, that are defined as follows.

**Definition 10.1 (Partition into Clusters)** *Let $H$ be a graph, and let $\mathcal{C}$ be a collection of subgraphs of $H$. We say that $\mathcal{C}$ is a* partition of $H$ into clusters*, if each subgraph $C \in \mathcal{C}$ is a connected vertex-induced subgraph (cluster) of $H$, $\bigcup_{C \in \mathcal{C}} V(C) = V(H)$, and for every pair $C, C' \in \mathcal{C}$ of distinct subgraphs, $V(C) \cap V(C') = \emptyset$.*

Recall that a vertex $v \in V(G)$ is a high-degree vertex, if $\deg_G(v) \geq m/\mu^4$. It will be convenient for us to assume that, if $u$ is a neighbor vertex of a high-degree vertex, then the degree of $u$ is 2, and that no vertex is a neighbor of two high-degree vertices. In order to achieve this, we simply subdivide every edge that is incident to a high-degree vertex with a single vertex; if, for an edge $e = (u, u')$, both its endpoints are high-degree vertices, then we subdivide this edge with two new vertices. Let $G'$ denote the resulting graph, and let $\Sigma'$ be the rotation system associated with $G'$, that is naturally defined

from rotation system $\Sigma$ for $G$: for every vertex $v \in V(G) \cap V(G')$, the circular ordering of the edges of $\delta_G(v) = \delta_{G'}(v)$ remains the same, and for every vertex $v \in V(G') \setminus V(G)$, $|\delta_{G'}(v)| = 2$, so its rotation is trivial. We denote by $I' = (G', \Sigma')$ the resulting instance of MCNwRS. Assume that we compute an $\eta$-decomposition $\mathcal{I}'$ of instance $I'$. It is easy to verify that $\mathcal{I}'$ is also an $O(\eta)$-decomposition of instance $I$. This is since $|E(G')| \leq O(|E(G)|)$, $\mathsf{OPT}_{\mathsf{cnwrs}}(I') = \mathsf{OPT}_{\mathsf{cnwrs}}(I)$, and, if we are given, for every instance $\tilde{I} \in \mathcal{I}$, a solution $\varphi(\tilde{I})$, then we can efficiently compute a solution $\varphi(\tilde{I}')$ of the same cost to the corresponding instance $\tilde{I}' \in \mathcal{I}'$. Lastly, a solution to instance $I'$ can be efficiently transformed into a solution to instance $I$ of the same cost. Therefore, from now on we will focus on decomposing instance $I'$ into a collection $\mathcal{I}'$ of subinstances with required properties. To simplify the notation, we denote $G'$ by $G$, $\Sigma'$ by $\Sigma$, and $I'$ by $I$. Note that $|E(G)| \leq 3m$ now holds; every neighbor of a high-degree vertex in $G$ has degree at most 2; and no vertex is a neighbor of two high-degree vertices.

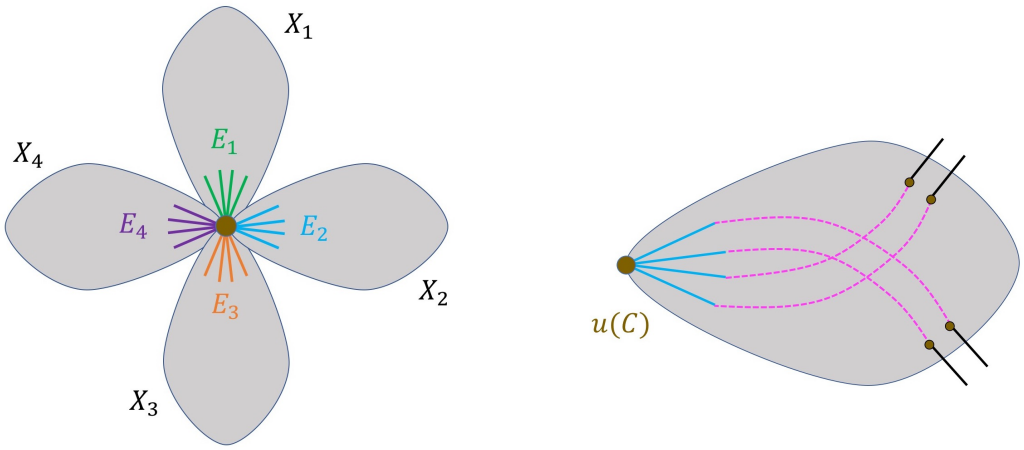
Intuitively, in order to prove Theorem 3.14, it is enough to compute a partition $\mathcal{C}$ of the input graph $G$ into clusters, such that, for every cluster $C \in \mathcal{C}$, $|E(C)| \leq m/\mu^2$ holds; we refer to such clusters as *small* clusters. Additionally, we require that the total number of edges with endpoints in different clusters is small, namely, $|E^{\mathsf{out}}(\mathcal{C})| \leq m/\mu^2$. Once such a collection $\mathcal{C}$ of clusters is computed, we can use the algorithm from Theorem 7.1 in order to compute the desired decomposition $\mathcal{I}$ of instance $I$. Unfortunately, we are unable to compute such a decomposition $\mathcal{C}$ of graph $G$ into small clusters directly. The main obstacle to computing such a decomposition via standard techniques is that graph $G$ may contain high-degree vertices. In order to overcome this difficulty, we define a new type of clusters, called *flower clusters*. Flower clusters will be used in order to isolate high-degree vertices in graph $G$. Eventually, we will compute a decomposition $\mathcal{C}$ of $G$ into clusters, where each cluster in $\mathcal{C}$ is either a small cluster or a flower cluster, and the number of edges whose endpoints lie in different clusters of $\mathcal{C}$ is sufficiently small. We now formally define flower clusters (see Figure 35 for an illustration).

**Definition 10.2 (Flower Cluster)** *We say that a subgraph $C \subseteq G$ is a* flower cluster *with a center vertex $u(C) \in V(C)$ and a set $\mathcal{X}(C) = \{X_1, \ldots X_k\}$ of petals, if the following hold:*

- *F1. $C$ is a connected vertex-induced subgraph of $G$, and for all $1 \leq i \leq k$, $X_i$ is a vertex-induced subgraph of $C$;*
- *F2. $\bigcup_{i=1}^{k} V(X_i) = V(C)$, and for every pair $X_i, X_j \in \mathcal{X}$ of clusters with $i \neq j$, $X_i \cap X_j = \{u(C)\}$;*
- *F3. the degree of $u(C)$ in $G$ is at least $m/\mu^4$, and all other vertices of $C$ have degrees below $m/\mu^4$ in $G$;*
- *F4. $|\delta_G(C)| \leq 96m/\mu^{42}$, and the total number of edges $e = (u, v)$ with $u \in V(X_i) \setminus \{u(C)\}$ and $v \in V(X_j) \setminus \{u(C)\}$ for all $1 \leq i < j \leq k$ is at most $96m/\mu^{42}$;*
- *F5. there is a partition $E_1, \ldots, E_k$ of all edges of $\delta_C(u(C))$ into subsets, such that, for all $1 \leq i \leq k$, edges in $E_i$ are consecutive in the ordering $\mathcal{O}_{u(C)} \in \Sigma$, $|E_i| \leq m/(2\mu^4)$, and $E_i \subseteq E(X_i)$ (so in particular $\delta_C(u(C)) = \delta_G(u(C))$; and*
- *F6. for every cluster $X_i \in \mathcal{X}$, there is a set $\mathcal{Q}_i$ of edge-disjoint paths, routing all edges of $\delta_G(X_i) \setminus \delta_G(u(C))$ to $u(C)$, such that all inner vertices on all paths of $\mathcal{Q}_i$ are contained in $X_i$.*

Notice that $\mathcal{Q} = \bigcup_{i=1}^{k} \mathcal{Q}_i$ is a set of edge-disjoint paths, routing all edges of $\delta_G(C)$ to vertex $u(C)$ inside $C$, and its existence certifies that a flower cluster must have 1-bandwidth property.

The remainder of the proof of Theorem 3.14 consists of three phases. In the first phase, we compute a decomposition $\mathcal{C}$ of the graph $G$ into clusters, such that the total number of edges with endpoints lying in different clusters is small, and every cluster in $\mathcal{C}$ is either a small cluster or a flower cluster. We then




(a) A schematic view of the petals $X_1, X_2, X_3, X_4$.

(b) Paths in set $\mathcal{Q}_2$ for petal $X_2$ are shown in pink.

Figure 35: An illustration of a 4-petal flower cluster.

use the algorithm from Theorem 7.1 in order to compute an initial collection $\mathcal{I}_1$ of subinstances of instance $I$. This collection will have all required properties, except that for some instances $\tilde{I} = (\tilde{G}, \tilde{\Sigma}) \in \mathcal{I}_1$, $|E(\tilde{G})| \leq m/(2\mu)$ may not hold. We call such instances *problematic*. Each such problematic instance consists of a single flower cluster $C \in \mathcal{C}$, and possibly some additional edges that lie in $E^{\mathsf{out}}(\mathcal{C})$ (recall that $E^{\mathsf{out}}(\mathcal{C})$ is the set of all edges whose endpoints belong to different clusters of $\mathcal{C}$). In the subsequent two phases, we consider each of the problematic instances in $\mathcal{I}_1$ separately, and further decompose it into smaller subinstances. Specifically, suppose we are given some problematic instance $\tilde{I} = (\tilde{G}, \tilde{\Sigma}) \in \mathcal{I}_1$ with corresponding flower cluster $C \subseteq \tilde{G}$, whose center is vertex $u$ and the set of petals is $\mathcal{X}$. Consider any petal $X \in \mathcal{X}$. We say that petal $X$ is *routable* in graph $\tilde{G}$ if there is a collection $\mathcal{P}(X) = \left\{P(e) \mid e \in \delta_{\tilde{G}}(X) \setminus \delta_{\tilde{G}}(u)\right\}$ of paths in $\tilde{G}$ that cause congestion at most 4000, such that for each edge $e \in \delta_{\tilde{G}}(X) \setminus \delta_{\tilde{G}}(u)$, path $P(e)$ has $e$ as its first edge and $u$ as its last vertex, and all inner vertices of $P(e)$ are disjoint from $X$. We show that, if every petal of a flower cluster $C$ is routable in $\tilde{G}$, then we can further decompose instance $\tilde{I}$ into a collection $\mathcal{I}'(\tilde{I})$ smaller subinstances, using reasoning similar to that in the basic disengagement procedure (see Section 5.3). It is still however possible that for some resulting instance $\tilde{I}' = (\tilde{G}', \tilde{\Sigma}') \in \mathcal{I}'(\tilde{I})$, $|E(\tilde{G}')| > m/(2\mu)$ holds. However, the disengagement procedure ensures that such an instance may not contain high-degree vertices. Therefore, applying the algorithm from Phase 1 to each such instance $\tilde{I}'$ will yield a decomposition of the initial instance $I$ into subinstances with all required properties. One difficulty with the approach we have just outlined is that, if $\tilde{I} = (\tilde{G}, \tilde{\Sigma}) \in \mathcal{I}_1$ is a problematic instance with corresponding flower cluster $C \subseteq \tilde{G}$, then we are not guaranteed that the petals of $C$ are routable in $\tilde{G}$. In order to overcome this difficulty, in Phase 2, we consider each such problematic instance $\tilde{I} \in \mathcal{I}$ separately. By performing a layered well-linked decomposition and an additional disengagement step, we decompose each such instance into a collection of subinstances, such that at most one resulting subinstance contains the flower cluster, and we further modify this flower cluster to ensure that each of its petals is routable in the resulting graph. In the third phase, we perform further disengagement on instances containing flower clusters, that exploits the fact that now every petal of the flower cluster is routable. Some of the resulting instances may still contain too many edges, but, as we show, they may not contain high-degree vertices. We then perform one final disengagement on such instances in order to obtain the final decomposition. We now describe each of the three phases in turn.

## 10.1 Phase 1: Flower Clusters, Small Clusters, and Initial Disengagement

The first phase consists of three steps. In the first step, we carve flower clusters out of the graph $G$. In the second step, we decompose the remainder of the graph $G$ into small clusters. Lastly, we perform a disengagement step for all resulting clusters in the third step.

### 10.1.1 Step 1: Carving out Flower Clusters

Let $S = \{s_1, \ldots, s_k\}$ be the set of all vertices of $G$ that have degree at least $m/\mu^4$. Notice that, since we have assumed that no edge connects a pair of high-degree vertices, and $|E(G)| \leq 3m$, $k \leq 3\mu^4$ must hold. In this step, we use with the following lemma, in order to carve flower clusters out of graph $G$.

**Lemma 10.3** *There is an efficient algorithm that computes a collection $\mathcal{C}^f = \left\{C_1^f, \ldots, C_k^f\right\}$ of disjoint clusters of $G$, such that for all $1 \leq i \leq k$, $C_i^f$ is a flower cluster with center $s_i$. The algorithm also computes, for each resulting flower cluster $C_i^f$, the corresponding set $\mathcal{X}(C_i^f)$ of petals.*

**Proof:** We start with the following simple claim that allows us to compute an initial collection $\{C_1, \ldots, C_k\}$ of clusters with some useful properties. Eventually, for all $1 \leq i \leq k$, we will define a flower cluster $C_i^f \subseteq C_i$.

**Claim 10.4** *There is an efficient algorithm to compute $k$ disjoint clusters $C_1, \ldots, C_k$ of $G$, such that for all $1 \leq i \leq k$, $s_i \in V(C_i)$, $\delta_G(s_i) \subseteq E(C_i)$, and $|\delta_G(C_i)| \leq 9m/\mu^{46}$. Additionally, the algorithm computes, for all $1 \leq i \leq k$, a set $\mathcal{Q}_i$ of edge-disjoint paths routing the edges of $\delta_G(C_i)$ to $s_i$, such that all inner vertices of every path lie in $C_i$.*

**Proof:** We use the algorithm from Lemma 4.8 to compute, for all $1 \leq i \leq k$, a set $A_i$ of vertices of $G$, such that $S \cap A_i = \{s_i\}$, and $(A_i, V(G) \setminus A_i)$ is a minimum cut separating $s_i$ from the vertices of $S \setminus \{s_i\}$ in $G$, and the vertex sets $A_1, \ldots, A_k$ are mutually disjoint. Recall that the algorithm also computes, for all $1 \leq i \leq k$, a set $\mathcal{Q}_i$ of edge-disjoint paths, routing the edges of $\delta_G(A_i)$ to vertex $s_i$, with all inner vertices on every path of $\mathcal{Q}_i$ lying in $A_i$.

Recall that we have assumed that, for all $1 \leq i \leq r$, for every edge $e = (s_i, v_e) \in \delta_G(s_i)$, the degree of vertex $v_e$ in $G$ is at most 2, and $v_e$ is the neighbor of at most one vertex in $S$ – vertex $s_i$. If such a vertex $v_e$ does not lie in $A_i$, then we move it to $A_i$ (if $v_e$ lies in some other vertex set $A_{i'}$, we remove it from that vertex set). Since the degree of $v_e$ in $G$ is 2, this does not increase the cardinalities of edge sets $\delta_G(A_{i'})$ for any $1 \leq i' \leq k$. Moreover, we can adjust the sets of paths $\{\mathcal{Q}_i\}_{i=1}^k$, such that, for all $1 \leq i \leq k$, set $\mathcal{Q}_i$ remains a set of edge-disjoint paths routing the edges of $\delta_G(A_i)$ to $s_i$, with all inner vertices on every path lying in $A_i$.

For all $1 \leq i \leq k$, we let $C_i = G[A_i]$. Note that, from the max-flow / min-cut theorem, for each $i$, there is a collection $\mathcal{P}_i$ of at least $|\delta_G(C_i)|$ edge-disjoint paths connecting $s_i$ to vertices of $S \setminus \{s_i\}$. Since $|S| = k \leq 3\mu^4$, there is some vertex $s_j \in S \setminus \{s_i\}$, such that at least $|\delta_G(C_i)|/k \geq |\delta_G(C_i)|/(3\mu^4)$ of the paths in $\mathcal{P}_i$ connect $s_i$ to $s_j$. Since the original instance $I$ that served as input to Theorem 3.14 is narrow, from Observation 3.10, the number of such paths must be bounded by $2\left\lceil m/\mu^{50}\right\rceil \leq 3m/\mu^{50}$, and so for all $1 \leq i \leq k$, $|\delta_G(C_i)| \leq 9m/\mu^{46}$. □

The following claim will complete the proof of Lemma 10.3.

**Claim 10.5** *There is an efficient algorithm, that, for all $1 \leq i \leq k$, computes a flower cluster $C_i^f \subseteq C_i$ with center $s_i$, together with a set $\mathcal{X}_i$ of petals.*

**Proof:** Fix an index $1 \leq i \leq k$. For simplicity of notation, in the remainder of the proof, we denote $C_i$ by $C$, $s_i$ by $s$, and the set $\mathcal{Q}_i$ of paths by $\mathcal{Q}$. Recall that $|\delta_G(C)| \leq 9m/\mu^{46}$.

Since $\deg_G(s) \geq m/\mu^4$, and $\delta_G(s) = \delta_C(s)$, we get that $\deg_C(s) \geq m/\mu^4$. Let $r = \left\lfloor \deg_C(s) \cdot \frac{6\mu^4}{m} \right\rfloor$. Since $m/\mu^4 \leq \deg_C(s) \leq m$, we get that $6 \leq r \leq 6\mu^4$. We compute a partition $E_1, \ldots, E_r$ of the edges of $\delta_C(s)$ into disjoint subsets, each of which contains at most $\lfloor m/(2\mu^4) \rfloor$ edges that appear in the ordering $\mathcal{O}_s \in \Sigma$ consecutively. From our choice of $r$, we get that $r \cdot \lfloor m/(2\mu^4) \rfloor \geq \deg_G(s)$, so such a partition must exist.

Consider now a graph $H$ that is defined as follows. We start with the graph $C \cup \delta_G(C)$. Let $R$ be the set of all endpoints of the edges of $\delta_G(C)$ that do not lie in $C$. We unify all vertices of $R$ into a vertex $a_0$. Next, we subdivide every edge $e \in \delta_C(s)$ with a new vertex $v_e$, and delete vertex $s$ from the resulting graph. Finally, for all $1 \leq j \leq r$, we unify the vertices in set $V_j = \{v_e \mid e \in E_j\}$ into a single vertex $a_j$, obtaining the graph $H$. Denote $Z = \{a_0, a_1, \ldots, a_r\}$. Note that $\deg_H(a_0) = |\delta_G(C)| \leq 9m/\mu^{46}$, so for all $1 \leq j \leq r$, there are at most $9m/\mu^{46}$ edge-disjoint paths connecting $a_0$ to $a_j$. Since the original instance $I$ that served as input to Theorem 3.14 is a narrow instance, from Observation 3.10, for all $1 \leq j < j' \leq r$, the maximum number of edge-disjoint paths connecting $a_j$ to $a_{j'}$ is at most $2m/\mu^{50}$. Therefore, for all $1 \leq j \leq r$, there is a cut separating $a_j$ from all vertices of $Z \setminus a_j$, containing at most $9m/\mu^{46} + 2rm/\mu^{50} \leq 16m/\mu^{46}$ edges.

We apply the algorithm from Lemma 4.8 to graph $H$ and vertex set $Z$. For all $1 \leq j \leq r$, we denote by $A'_j$ the set of vertices that the algorithm returns for vertex $a_j$. Recall that, if $e = (s, x)$ is an edge of $E_j$, then vertex $x$ must have degree at most 2 in graph $G$. We can then assume without loss of generality that $x$ lies in $A'_j$; if it lies in another set $A'_{j'}$, or it does not lie in any such set, we can simply move it to $A'_j$; this will not increase the values of the cuts $|E(A'_{j'}, V(C) \setminus A'_{j'})|$ for any $j'$. We can also adjust the sets $\left\{\mathcal{Q}'_j\right\}$ of paths that the algorithm from Lemma 4.8 returns for each $1 \leq j \leq r$, so that the paths in $\mathcal{Q}'_j$ are edge-disjoint and route edges of $\delta_H(A'_j)$ to vertex $a_j$, with all inner vertices on every path lying in $A'_j$.

For all $1 \leq j \leq k$, let $X_j$ be the subgraph of $C$ induced by the vertex set $(A'_j \setminus \{a_j\}) \cup \{s\}$. We then let $C^f$ be the subgraph of $G$ induced by vertex set $\bigcup_{j=1}^{k} V(X_j)$. We claim that $C^f$ is a flower cluster with center $s$ and set $\mathcal{X} = \{X_1, \ldots, X_r\}$ of petals. We now verify that $C^f$ and $\mathcal{X}$ have all required properties.

First, it is immediate to verify that $C^f$ is a vertex-induced subgraph of $G$, and that for all $1 \leq j \leq r$, $X_j$ is a vertex-induced subgraph of $C^f$ with $\bigcup_{j=1}^{r} V(X_j) = V(C^f)$, and, for all $1 \leq j < j' \leq r$, $X_j \cap X_{j'} = \{u\}$. Additionally, we can assume that $C$ is connected: if this is not the case, then connected components of $C$ that are disjoint from $u$ can be discarded. This establishes Properties F1 and F2.

Consider now some index $1 \leq j \leq r$. Since we have assumed that, for every edge $e = (u, v) \in E_j$, $v \in A'_j$ holds, we get that $E_j \subseteq X_j$. Since $\delta_G(s) \subseteq \delta_C(s)$, and edge sets $E_1, \ldots, E_r$ partition $\delta_C(s)$, this establishes Property F5. $\delta_G(s) \subseteq E(C^f)$.

Recall that for all $1 \leq j \leq r$, $|\delta_C(X_j) \setminus \delta_C(s)| \leq 16m/\mu^{46}$ (since $(A'_j, V(H) \setminus A'_j)$ is a minimum cut in $H$ separating $a_j$ from all vertices of $Z \setminus a_j$, whose value we have bounded above). Therefore, $|\delta_G(C^f)| \leq r \cdot 16m/\mu^{46} \leq (6\mu^4) \cdot (16m/\mu^{46}) \leq 96m/\mu^{42}$, and moreover the total number of edges $e = (u, v)$ with $u, v$ lying in different sets $X_1, \ldots, X_r$ is also bounded by $96m/\mu^{42}$. This establishes Property F4. Since no vertex of $S$ may lie in $C^f$ (as $C^f \subseteq C$), all vertices of $V(C) \setminus \{s\}$ have degrees at most $m/\mu^4$ in $G$. This establishes Property F3.

Lastly, we need to establish Property F6. Consider any petal $X_j \in \mathcal{X}$. From the definition of graph $H$, and since vertex $a_0$ may not belong to set $A'_j$, we get that $\delta_H(A'_j) = \delta_G(X_j) \setminus \delta_G(s)$. Recall that the algorithm from Lemma 4.8 provided a set $\mathcal{Q}'_j$ of edge-disjoint paths that route the edges of $\delta_H(A'_j)$ to vertex $a_j$, with all inner vertices on the paths lying in $A'_j$. By replacing vertex $a_j$ with vertex $s$ on each such path, we obtain a collection $\mathcal{Q}''_j$ of edge-disjoint paths in graph $C^f$, that route the edges of

$\delta_G(X_j)$ to $s$, with all inner vertices on all paths lying in $X_j$. We conclude that $C^f$ is a valid flower cluster with center $s$ and set $\mathcal{X}$ of petals. □ □

### 10.1.2 Step 2: Small Clusters

Let $C_0$ be the cluster that is obtained from graph $G$ after we delete all vertices lying in $\bigcup_{C\in\mathcal{C}^f} V(C)$ from it, that is, $C_0 = G \setminus (\bigcup_{C\in\mathcal{C}^f} V(C))$. Note that, since $|\mathcal{C}^f| \leq 3\mu^4$, and, for all $C \in \mathcal{C}^f$, $|\delta_G(C)| \leq 96m/\mu^{42}$, we get that $|\bigcup_{C\in\mathcal{C}^f} \delta_G(C)| \leq (3\mu^4)\cdot(96m/\mu^{42}) \leq 288m/\mu^{38}$, and so $|\delta_G(C_0)| \leq 288m/\mu^{38}$. In this step, our goal is to further decompose cluster $C_0$ into a collection $\mathcal{C}^s$ of clusters, that we refer to as *small clusters*. We will require that each cluster $C \in \mathcal{C}^s$ has the $\alpha_0$-bandwidth property, for an appropriately chosen parameter $\alpha_0$, and that $|E(C)| \leq |E(G)|/\mu^2$. Moreover, we will require that the total number of edges whose endpoints lie in different clusters of $\mathcal{C}^f \cup \mathcal{C}^s$ is relatively small. We show an algorithm that either computes such a decomposition of $C_0$, or establishes that $\mathsf{OPT}_{\mathsf{cnwrs}}(I)$ is sufficiently large, by utilizing the following lemma; the lemma will also be used later in this section in a slightly different setting, so it is stated for a more general setting than what is needed here.

**Lemma 10.6** *There is an efficient algorithm, whose input consists of a graph $H$, a set $T \subseteq V(H)$ of $k$ vertices called terminals (where possibly $T = \emptyset$), and parameters $m, \tau \geq 0$, such that $|E(H)| \leq m$, $k \leq m/(64\tau\log m)$; every vertex in $T$ has degree 1 in $H$; and maximum vertex degree in $H$ is at most $\frac{m}{\check{c}\tau^3\log^5 m}$, for a large enough constant $\check{c}$. The algorithm either correctly certifies that $\mathsf{OPT}_{\mathsf{cr}}(H) \geq \Omega\left(\frac{m^2}{\tau^4\log^5 m}\right)$, or computes a collection $\mathcal{C}$ of disjoint clusters of $H \setminus T$ with the following properties:*

- *every cluster $C \in \mathcal{C}$ has the $\alpha'$-bandwidth property, where $\alpha' = \frac{1}{16\beta_{\mathrm{ARV}}(m)\cdot\log m} = \Omega\left(\frac{1}{\log^{1.5} m}\right)$;*
- *$\bigcup_{C\in\mathcal{C}} V(C) = V(H) \setminus T$;*
- *for every cluster $C \in \mathcal{C}$, $|E(C)| \leq m/\tau$; and*
- *$|\bigcup_{C\in\mathcal{C}} \delta_H(C)| \leq m/\tau$.*

The proof of the lemma is very similar to the proof of Theorem 6.3 and is deferred to Section I.1 of Appendix.

We consider the graph $C_0^+$, that is an augmentation of cluster $C_0$. Recall that $C_0^+$ is obtained from graph $G$, by first subdividing every edge $e \in \delta_G(C_0)$ with a vertex $t_e$, setting $T = \{t_e \mid e \in \delta_G(C_0)\}$, and letting $C_0^+$ be the subgraph of the resulting graph induced by $T \cup V(C_0)$. We apply the algorithm from Lemma 10.6 to graph $H = C_0^+$, the set $T$ of terminals, and parameter $\tau = 160\mu^{1.1}$. Recall that, since $C_0$ contains no high-degree vertices, maximum vertex degree in $C_0$ is bounded by $\frac{m}{\mu^4} \leq \frac{m}{\check{c}\tau^3\log^5 m}$. Recall also that $|T| = |\delta_G(C_0)| \leq 288m/\mu^{38} \leq m/(64\tau\log m)$. If the algorithm from Lemma 10.6 certifies that $\mathsf{OPT}_{\mathsf{cr}}(C_0^+) \geq \Omega\left(\frac{m^2}{\tau^4\log^5 m}\right)$, then we terminate the algorithm and return FAIL. Notice that we are guaranteed that $\mathsf{OPT}_{\mathsf{cnwrs}}(I) \geq \Omega\left(\frac{m^2}{\mu^5}\right)$.

Therefore, we assume from now on that the algorithm from Lemma 10.6 computed a collection $\mathcal{C}^s$ of disjoint clusters, such that $\bigcup_{C\in\mathcal{C}^s} V(C) = V(C_0)$, every cluster $C \in \mathcal{C}^s$ has the $\alpha'$-bandwidth property in $G$, where $\alpha' = \frac{1}{16\beta_{\mathrm{ARV}}(m)\cdot\log m} = \Omega\left(\frac{1}{\log^{1.5} m}\right)$, for every cluster $C \in \mathcal{C}^s$, $|E(C)| \leq m/(160\mu^{1.1})$; and $|\bigcup_{C\in\mathcal{C}^s} \delta_G(C)| \leq m/(160\mu^{1.1})$. We refer to clusters in set $\mathcal{C}^s$ as *small clusters*. Let $\mathcal{C} = \mathcal{C}^s \cup \mathcal{C}^f$. Recall that $E^{\mathsf{out}}(\mathcal{C})$ is the set of all edges whose endpoints lie in different clusters of $\mathcal{C}$. Since $|\bigcup_{C\in\mathcal{C}^s} \delta_G(C)| \leq m/(160\mu^{1.1})$, $|\mathcal{C}^f| \leq 3\mu^4$, and, for all $C \in \mathcal{C}^f$, $|\delta_G(C)| \leq 96m/\mu^{42}$, we get that:

$$|E^{\mathsf{out}}(\mathcal{C})| \leq \frac{m}{160\mu^{1.1}} + (3\mu^4)\cdot\frac{96m}{\mu^{42}} \leq \frac{m}{80\mu^{1.1}}. \tag{15}$$

### 10.1.3 Step 3: Initial Disengagement

In this step, we consider the set $\mathcal{C} = \mathcal{C}^s \cup \mathcal{C}^f$ of clusters. Recall that all clusters in $\mathcal{C}$ are disjoint and $\bigcup_{C\in\mathcal{C}} V(C) = V(G)$. Moreover, every cluster in $\mathcal{C}^s$ has the $\alpha'$-bandwidth property, for $\alpha' = \frac{1}{16\beta_{\mathrm{ARV}}(m)\cdot\log m}$, while every cluster in $\mathcal{C}^f$ has 1-bandwidth property (which follows from the definition of flower clusters). Since $m \geq \mu^4$, we then get that every cluster in $\mathcal{C}$ has the $\alpha_0$-bandwidth property, for $\alpha_0 = 1/\log^3 m'$, where $m' = |E(G)| \leq 3m$.

We apply the algorithm from Theorem 7.1 to instance $I = (G, \Sigma)$ of $\mathsf{MCNwRS}$, with parameter $m'$ replacing $m$, and the set $\mathcal{C}$ of clusters; parameter $\mu$ remains unchanged. Let $\mathcal{I}_1$ be the resulting collection of instances that the algorithm computes, that is a $2^{O((\log m)^{3/4}\log\log m)}$-decomposition of instance $I$. Recall that we are guaranteed that, for each instance $I' = (G', \Sigma') \in \mathcal{I}_1$, $I'$ is a subinstance of $I$, and there is at most one cluster $C \in \mathcal{C}$ with $E(C) \subseteq E(G')$, and all other edges of $G'$ lie in set $E^{\mathsf{out}}(\mathcal{C})$. We have therefore shown an efficient randomized algorithm that computes a $\nu_0$-decomposition $\mathcal{I}_1$ of instance $I$, for $\nu_0 = 2^{O((\log m)^{3/4}\log\log m)}$.

We partition instances in $\mathcal{I}_1$ into two subsets, set $\mathcal{I}'_1$ containing all instances $I' = (G', \Sigma')$ with $|E(G')| \leq m/(2\mu)$, and set $\mathcal{I}''_1$ containing all remaining instances. We refer to instances in $\mathcal{I}''_1$ as *problematic instances*. Consider now any problematic instance $I' = (G', \Sigma') \in \mathcal{I}''_1$. Recall that, from the guarantees of Theorem 7.1, there is at most one cluster $C' \in \mathcal{C}$ with $C' \subseteq G'$, and all edges of $G'$ lie in $E(C') \cup E^{\mathsf{out}}(\mathcal{C})$. Since we are guaranteed that

$$\left|\bigcup_{C\in\mathcal{C}}\delta(C)\right| \leq \left|\bigcup_{C\in\mathcal{C}^s}\delta(C)\right| + \left|\bigcup_{C\in\mathcal{C}^f}\delta(C)\right| \leq m/(160\mu) + 288m/\mu^{38} \leq m/(80\mu),$$

we get that $|E(G') \setminus E(C')| \leq m/(80\mu)$, and $|E(C')| \geq m/(2\mu) - m/(80\mu)$. In particular, cluster $C'$ may not be a small cluster, so it must be a flower cluster. We say that $C'$ is the flower cluster associated with the problematic instance $I' \in \mathcal{I}''_1$. In the remaining phases, we will further decompose each problematic instance into subinstances, proving the following theorem.

**Theorem 10.7** *There is an efficient randomized algorithm, that, given a problematic instance $I' = (G', \Sigma') \in \mathcal{I}''_1$, either returns FAIL, or computes a $\nu_1$-decomposition $\tilde{\mathcal{I}}(I')$ of $I'$, where $\nu_1 = 2^{O((\log m)^{3/4}\log\log m)}$, such that, for each instance $\tilde{I} = (\tilde{G}, \tilde{\Sigma}) \in \tilde{\mathcal{I}}(I')$, $|E(\tilde{G})| \leq m/(2\mu)$. Moreover, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I') < m^2/\left(\mu^{18}\cdot 2^{c'(\log m)^{3/4}\log\log m}\right)$ for some large enough constant $c'$, then the probability that the algorithm returns FAIL is at most $1/\mu^4$. (Here, $m = |E(G)|$, where $G$ is associated with the original instance $I$, that serves as input to Theorem 3.14).*

Before providing the proof of Theorem 10.7, we show that the proof of Theorem 3.14 follows from it. We apply the algorithm from Theorem 10.7 to every problematic instance $I' \in \mathcal{I}''_1$. Assume first that, for each such instance $I'$, the algortihm returns a $\nu_1$-decomposition $\tilde{\mathcal{I}}(I')$ of $I'$, such that, for each instance $\tilde{I} = (\tilde{G}, \tilde{\Sigma}) \in \tilde{\mathcal{I}}(I')$, $|E(\tilde{G})| \leq m/(2\mu)$. In this case, we return the collection $\mathcal{I} = \mathcal{I}'_1 \cup \left(\bigcup_{I'\in\mathcal{I}''_1}\tilde{\mathcal{I}}(I')\right)$ of instances. From Claim 2.11, since $\nu_1\cdot\nu_2 = 2^{O((\log m)^{3/4}\log\log m)}$, $\mathcal{I}$ is indeed a $\nu$-decomposition of $I$, and, from the above discussion, we are guaranteed that, for every instance $\tilde{I} = (\tilde{G}, \tilde{\Sigma}) \in \mathcal{I}$, $|E(\tilde{G})| \leq m/(2\mu)$.

If, for any problematic instance $I' \in \mathcal{I}''_1$, the algorithm from Theorem 10.7 returned FAIL, then our algorithm returns FAIL as well.

Recall that $\mu = 2^{\check{c}(\log m^*)^{7/8}\log\log m^*}$, where $m^* \geq m$. Assume now that $\mathsf{OPT}_{\mathsf{cnwrs}}(I) < m^2/\mu^{21}$. We will show that in this case, the probability that our algorithm returns FAIL is at most $O(1/\mu^2)$. Since we have assumed that $m > \mu^{50}$ (from the statement of Theorem 3.14), $|E(G)| < m^2/\mu^{20}$.

Recall that $\mathcal{I}_1$ is a $2^{O((\log m)^{3/4}\log\log m)}$-decomposition of instance $I$, and so:

$$\mathbf{E}\left[\sum_{I'\in\mathcal{I}_1}\mathsf{OPT}_{\mathsf{cnwrs}}(I')\right]\leq 2^{O((\log m)^{3/4}\log\log m)}\cdot\left(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+|E(G)|\right).$$

In particular, there is some constant $c$, such that $\mathbf{E}\left[\sum_{I'\in\mathcal{I}''_1}\mathsf{OPT}_{\mathsf{cnwrs}}(I')\right]\leq 2^{c(\log m)^{3/4}\log\log m)}\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+|E(G)|)$. Let $\mathcal{E}$ be the bad event that $\sum_{I'\in\mathcal{I}''_1}\mathsf{OPT}_{\mathsf{cnwrs}}(I')>8\mu^2\cdot 2^{c(\log m)^{3/4}\log\log m)}\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+|E(G)|)$. From Markov's inequality, the probability that $\mathcal{E}$ happens is at most $1/(8\mu^2)$. Assume now that $\mathsf{OPT}_{\mathsf{cnwrs}}(I)<m^2/\mu^{21}$, and that the bad event $\mathcal{E}$ does not happen. Then for every problematic instance $I'\in\mathcal{I}''_1$:

$$\begin{aligned}\mathsf{OPT}_{\mathsf{cnwrs}}(I')&\leq 8\mu^2\cdot 2^{c(\log m)^{3/4}\log\log m)}\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+|E(G)|)\\&\leq 8\mu^2\cdot 2^{c(\log m)^{3/4}\log\log m)}\cdot\frac{2m^2}{\mu^{21}}\\&\leq\frac{m^2}{\mu^{18}\cdot 2^{c'(\log m)^{3/4}\log\log m}},\end{aligned}$$

where $c'$ is the constant from Theorem 10.7. Therefore, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I)<m^2/\mu^{21}$ and Event $\mathcal{E}$ does not happen, then, for every problematic instance $I'\in\mathcal{I}''_1$, the probability that the algorithm from Theorem 10.7 returns FAIL is at most $1/\mu^4$. Since $\sum_{I'=(G',\Sigma')\in\mathcal{I}_1}|E(G')|\leq|E(G)|\cdot(\log m)^{O(1)}$, and, for every problematic instance $I''=(G',\Sigma')\in\mathcal{I}''_1$, $|E(G')|\geq m/(2\mu)$, we get that $|\mathcal{I}''_1|\leq\mu\cdot(\log m)^{O(1)}$. Therefore, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I)<m^2/\mu^{21}$ and $\mathcal{E}$ does not happen, then the probability that the algorithm from Theorem 10.7 returns FAIL for any problematic instance $I'\in\mathcal{I}''_2$ is at most $O(1/\mu^2)$. Since the probability that event $\mathcal{E}$ happens is at most $1/(8\mu^2)$, we get that, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I)<m^2/\mu^{21}$, then the probability that our algorithm returns FAIL is at most $O(1/\mu^2)$.

In order to complete the proof of Theorem 3.14, it is now enough to prove Theorem 10.7. From now on, we fix a single a problematic instance $I'=(G',\Sigma')\in\mathcal{I}''_1$, and its corresponding flower cluster $C'\in\mathcal{C}^f$, and provide an algorithm to compute a decomposition of $I'$ into subclusters with required properties.

## 10.2 Phase 2: Layered Well-Linked Decomposition, Further Disengagement, and Fixing the Flower Cluster

From now on we focus on the proof of Theorem 10.7. In order to simplify the notation, we denote the input problematic instance by $I=(G,\Sigma)$. Our goal is to design an efficient randomized algorithm that computes a collection $\mathcal{I}'$ of instances of $\mathsf{MCNwRS}$, such that, for each resulting instance $\tilde{I}=(\tilde{G},\tilde{\Sigma})\in\mathcal{I}'$, $|E(\tilde{G})|\leq m/\mu$, and additionally, $\sum_{\tilde{I}=(\tilde{G},\tilde{\Sigma})\in\mathcal{I}'}|E(\tilde{G})|\leq O(|E(G)|)$, and $\mathbf{E}\left[\sum_{\tilde{I}\in\mathcal{I}'}\mathsf{OPT}_{\mathsf{cnwrs}}(\tilde{I})\right]\leq 2^{O((\log m)^{3/4}\log\log m)}\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+|E(G)|)$. We also need to provide an efficient algorithm $\mathcal{A}(I)$, that, given a solution $\varphi(\tilde{I})$ to each instance $\tilde{I}\in\mathcal{I}'$, computes a solution $\varphi$ to instance $I$, with $\mathsf{cr}(\varphi)\leq O\left(\sum_{\tilde{I}\in\mathcal{I}'}\mathsf{cr}(\varphi(\tilde{I}))\right)$.

We denote by $C$ the unique flower cluster of $\mathcal{C}^f$ contained in $G$, by $u^*$ its center vertex, and by $\mathcal{X}=\{X_1,\ldots,X_k\}$ its petals. This phase consists of two steps. In the first step, we compute a layered well-linked decomposition of the graph $G$ with respect to $C$, and perform disengagement of the resulting clusters. In the second step, we modify the flower cluster $C$ and its petals to ensure that every petal is routable in the resulting instance.

### 10.2.1 Step 1: Layered Well-Linked Decomposition and Second Disengagement

In this step, we apply the algorithm from Theorem 4.20 to graph $G$ and cluster $C$, in order to compute a valid layered $\alpha$-well-linked decomposition $(\mathcal{W}, (\mathcal{L}_1, \ldots, \mathcal{L}_r))$ of $G$ with respect to $C$, for $\alpha = \frac{1}{c \log^{2.5} m}$, where $c$ is some large enough constant independent of $m$, and $r \leq \log m$. Note that, since we have assumed that $m$ is sufficiently large, every cluster $W \in \mathcal{W}$ has the $\alpha_0$-bandwidth property, for $\alpha_0 = 1/\log^3 m$. Recall that we are additionally guaranteed that $\bigcup_{W \in \mathcal{W}} V(W) = V(G) \setminus V(C)$, and that, for every cluster $W \in \mathcal{W}$, $|\delta_G(W)| \leq |\delta_G(C)|$. Recall that, for every cluster $W \in \mathcal{W}$ with $W \in \mathcal{L}_i$, we have partitioned the set $\delta_G(W)$ of edges into two subsets: set $\delta^{\text{down}}(W)$ connecting vertices of $W$ to vertices that lie in the clusters of $\{C\} \cup \mathcal{L}_1 \cup \cdots \cup \mathcal{L}_{i-1}$; and set $\delta^{\text{up}}(W)$ containing all remaining edges, and we are guaranteed that $|\delta^{\text{up}}(W)| < |\delta^{\text{down}}(W)|/\log m$.

Lastly, recall that, for every cluster $W \in \mathcal{W}$, there is a collection $\mathcal{P}(W)$ of paths in $G$, routing the edges of $\delta_G(W)$ to edges of $\delta_G(C)$, such that the paths in $\mathcal{P}(W)$ avoid $W$, and cause congestion at most $200/\alpha$. Recall that, from Properties F2 and F6 of the flower cluster, there is a collection $\mathcal{Q} \subseteq \bigcup_{i=1}^{k} \mathcal{Q}_i$ of edge-disjoint paths, routing the edges of $\delta_G(C)$ to the vertex $u^*$, such that all inner vertices on every path lie in $C$. By concatenating the paths in $\mathcal{P}(W)$ and the paths in $\mathcal{Q}$, we obtain a new collection $\mathcal{Q}'(W)$ of paths, that route the edges of $\delta_G(W)$ to vertex $u^*$, so that all inner vertices on every path lie outside $W$, and cause congestion at most $200/\alpha = O(\log^{2.5} m)$.

We partition the set $\mathcal{W}$ of clusters into two subsets $\mathcal{W}^{\text{light}}$, and $\mathcal{W}^{\text{bad}}$, as follows. We apply the algorithm AlgClassifyCluster from Theorem 6.1 to each cluster cluster $W \in \mathcal{W}$ in turn, with parameter $p = 1/(m^*)^4$. If the algorithm returns FAIL, then we add cluster $W$ to $\mathcal{W}^{\text{bad}}$. Recall that the probability that Algorithm AlgClassifyCluster errs, that is, it returns FAIL when $W$ is not $\eta^*$-bad, for $\eta^* = 2^{O((\log m)^{3/4} \log\log m)}$, is at most $1/(m^*)^4$. Otherwise, the algorithm returns a distribution $\mathcal{D}(W)$ over the set $\Lambda_G(W)$ of internal $W$-routers, such that cluster $W$ is $\beta^*$-light with respect to $\mathcal{D}(W)$, where $\beta^* = 2^{O(\sqrt{\log m} \cdot \log\log m)}$. We add $W$ to $\mathcal{W}^{\text{light}}$ in this case. This finishes the algorithm for partitioning the set $\mathcal{W}$ of clusters into $\mathcal{W}^{\text{light}}$ and $\mathcal{W}^{\text{bad}}$. We let $\beta = \max\{\beta^*, \eta^*\}$, so that $\beta \leq 2^{O((\log m)^{3/4} \log\log m)}$.

We say that a bad event $\mathcal{E}_{\text{bad}}$ happens if set $\mathcal{W}^{\text{bad}}$ contains a cluster that is not $\beta$-bad. From the above discussion, $\mathbf{Pr}[\beta_{\text{bad}}] \leq 1/(m^*)^3$, and every cluster $W \in \mathcal{W}^{\text{light}}$ is $\beta$-good with respect to the distribution $\mathcal{D}(W)$ over the set $\Lambda_G(W)$ of internal $W$-routers. Recall that we are also given, for every cluster $W \in \mathcal{W}$, a set $\mathcal{Q}'(W)$ of paths routing the edges of $\delta_G(W)$ to vertex $u^*$ (external $W$-router), with congestion at most $O(\log^{2.5} m) \leq \beta$.

For every cluster $W \in \mathcal{W}$, we define its distribution over the set $\Lambda'_G(W)$ of external $W$-routers to be the distribution that assign probability 1 to the path set $\mathcal{Q}'(W)$. We let $\mathcal{L}$ be a laminar family of clusters of $G$, containing cluster $G$ and every cluster of $\mathcal{W}$. We now apply Algorithm AlgBasicDisengagement to the instance $I = (G, \Sigma)$, using the laminar family $\mathcal{L}$ of clusters, its partition $(\mathcal{W}^{\text{bad}}, \mathcal{W}^{\text{light}} \cup \{G\})$, and distributions $\{\mathcal{D}(W)\}_{W \in \mathcal{W}^{\text{light}}}$ and $\{\mathcal{D}'(W)\}_{W \in \mathcal{L}}$, that is given in Section 5.3.

We denote the resulting family of instances by $\mathcal{I}_2(I)$. Recall that family $\mathcal{I}_2(I)$ of instances contains a single global instance $\hat{I}$, and additionally, for every cluster $W \in \mathcal{W}$, an instance $I_W$.

If we denote the global instance by $\hat{I} = (\hat{G}, \hat{\Sigma})$, then graph $\hat{G}$ is obtained from graph $G$ by contracting every cluster $W \in \mathcal{W}$ into a supernode $v_W$. In particular, the flower cluster $C$ is contained in $\hat{G}$. For every cluster $W \in \mathcal{W}$, if we denote corresponding instance by $I_W = (G_W, \Sigma_W)$, then graph $G_W$ is obtained from graph $G$ by contracting all vertices of $V(G) \setminus V(W)$ into a single vertex $v^*$. In particular, no edge of cluster $C$ may lie in $G_W$, and so every edge of $G_W$ is an edge of set $E^{\mathsf{out}}(\mathcal{C})$, where $\mathcal{C}$ is the set of clusters computed in Phase 1. Therefore, $|E(G_W)| \leq m/(2\mu)$.

Note that the depth of laminar family $\mathcal{L}$ is 1. From Lemma 5.6, if event $\mathcal{E}_{\text{bad}}$ did not happen,

$$\mathbf{E}\left[\sum_{I'\in\mathcal{I}_2(I)}\mathsf{OPT}_{\mathsf{cnwrs}}(I')\right]\leq O\left(\beta^2\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+|E(G)|)\right)$$
$$\leq 2^{O((\log m)^{3/4}\log\log m)}\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+|E(G)|)\,.$$

If bad event $\mathcal{E}_{\text{bad}}$ happened, then $\sum_{I'\in\mathcal{I}_2(I)}\mathsf{OPT}_{\mathsf{cnwrs}}(I')\leq m^3$. Since $\mathbf{Pr}\left[\mathcal{E}_{\text{bad}}\right]\leq 1/(m^*)^3$, we get that overall:

$$\mathbf{E}\left[\sum_{I'\in\mathcal{I}_2(I)}\mathsf{OPT}_{\mathsf{cnwrs}}(I')\right]\leq 2^{O((\log m)^{3/4}\log\log m)}\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+|E(G)|)\,. \tag{16}$$

Additionally, from Lemma 5.2, we get that $\sum_{I'=(G',\Sigma')\in\mathcal{I}_2(I)}|E(G')|\leq O(|E(G)|)$.

Lastly, from Lemma 5.3, there is an efficient algorithm, that, given, for each instance $I'\in\mathcal{I}_2(I)$, a solution $\varphi(I')$, computes a solution for instance $I$ of value at most $\sum_{I'\in\mathcal{I}}\mathsf{cr}(\varphi(I'))$.

Note that the set $\mathcal{I}_2(I)$ of instances has all properties required in Theorem 10.7, with one exception: it is possible that, in the global instance $\hat{I}=(\hat{G},\hat{\Sigma})$, $|E(\hat{G})|>m/(2\mu)$.

In order to overcome this difficulty, we further decompose instance $\hat{I}$ into subinstances, proving the following lemma.

**Lemma 10.8** *There is an efficient randomized algorithm, that either returns FAIL, or computes a $\nu_2$-decomposition $\tilde{\mathcal{I}}$ of instance $\hat{I}$, for $\nu_2=2^{O((\log m)^{3/4}\log\log m)}$, such that, for each instance $\tilde{I}=(\tilde{G},\tilde{\Sigma})\in\tilde{\mathcal{I}}$, $|E(\tilde{G})|\leq m/(2\mu)$. Moreover, if $\mathsf{OPT}_{\mathsf{cnwrs}}(\hat{I})<\frac{m^2}{c''\mu^{13}}$ for some large enough constant $c''$, then the probability that the algorithm returns FAIL is at most $1/(8\mu^4)$.*

We prove the lemma below, after we complete the proof of Theorem 10.7 using it. If the algorithm from Lemma 10.8 returns $\nu_2$-decomposition $\tilde{\mathcal{I}}$ of instance $\hat{I}$, then we return the collection $\tilde{\mathcal{I}}(I)=\tilde{\mathcal{I}}\cup\{I_W\mid W\in\mathcal{W}\}$ of instances, which is now guaranteed to be a $\nu_1$-decomposition of instance $I$, where $\nu_1=2^{O((\log m)^{3/4}\log\log m)}$. We are also guaranteed that, for each instance $\tilde{I}=(\tilde{G},\tilde{\Sigma})\in\tilde{\mathcal{I}}$, $|E(\tilde{G})|\leq m/(2\mu)$.

Assume now that $\mathsf{OPT}_{\mathsf{cnwrs}}(I)<m^2/\left(\mu^{18}\cdot 2^{c'(\log m)^{3/4}\log\log m}\right)$ for some large enough constant $c'$.

Recall that, from Equation 16, $\mathbf{E}\left[\sum_{I'\in\mathcal{I}_2(I)}\mathsf{OPT}_{\mathsf{cnwrs}}(I')\right]\leq 2^{O((\log m)^{3/4}\log\log m)}\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+|E(G)|)$, and in particular $\mathbf{E}\left[\mathsf{OPT}_{\mathsf{cnwrs}}(\hat{I})\right]\leq\nu^*\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+|E(G)|)$, for some $\nu^*=2^{O((\log m)^{3/4}\log\log m)}$.

We say that a bad event $\mathcal{E}'$ happens if $\mathsf{OPT}_{\mathsf{cnwrs}}(\hat{I})>8\mu^4\cdot\nu^*\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+|E(G)|)$. From Markov's inequality, $\mathbf{Pr}\left[\mathcal{E}'\right]\leq 1/(8\mu^4)$.

By letting $c'$ be a large enough constant, we can assume that, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I)<m^2/\left(\mu^{18}\cdot 2^{c'(\log m)^{3/4}\log\log m}\right)$, then $\mathsf{OPT}_{\mathsf{cnwrs}}(I)<\frac{m^2}{16c''\mu^{18}\cdot\nu^*}$, where $c''$ is the constant from Lemma 10.8. Since we have assumed (in the statement of Theorem 3.14) that $m>\mu^{50}$ and since $\mu>\nu^*$, we get that $|E(G)|<\frac{m^2}{16c''\mu^{18}\cdot\nu^*}$. To conclude, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I)<m^2/\left(\mu^{18}\cdot 2^{c'(\log m)^{3/4}\log\log m}\right)$, then $(\mathsf{OPT}_{\mathsf{cnwrs}}(I))+|E(G)|)<\frac{m^2}{8c''\mu^{18}\cdot\nu^*}$.

If, additionally, Event $\mathcal{E}'$ did not happen, then $\mathsf{OPT}_{\mathsf{cnwrs}}(\hat{I})<\frac{m^2}{c''\mu^{11}}$. In this case, the algorithm from Lemma 10.8 may only return FAIL with probability at most $1/(8\mu^4)$. To conclude, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I)<m^2/\left(\mu^{18}\cdot 2^{c'(\log m)^{3/4}\log\log m}\right)$, then our algorithm may return FAIL in only two cases: either (i) event $\mathcal{E}'$ happened (which happens with probability at most $1/(8\mu^4)$); or (ii) $\mathsf{OPT}_{\mathsf{cnwrs}}(\hat{I})<\frac{c''m^2}{\mu^{11}}$, and yet

the algorithm from Lemma 10.8 returns FAIL (which happens with probability at most $1/(8\mu^3)$). Overall, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I) < m^2/\left(\mu^{18}\cdot 2^{c'(\log m)^{3/4}\log\log m}\right)$, then the algorithm only returns FAIL with probability at most $1/(4\mu^4)$.

From now on we focus on the proof of Lemma 10.8. Recall that we have denoted $\hat{I}=(\hat{G},\hat{\Sigma})$, and that graph $\hat{G}$ is obtained from $G$ by contracting every cluster $W\in\mathcal{W}$ into a vertex $v_W$, that we refer to as a *supernode*. We denote the resulting set of supernodes by $U=\{v_W\mid W\in\mathcal{W}\}$. Recall that the flower cluster $C\subseteq\hat{G}$, and the edges of $E(\hat{G})\setminus E(C)$ lie in $E^{\mathsf{out}}(\mathcal{C})$, where $\mathcal{C}$ is the set of clusters computed in Phase 1. Therefore, $|E(\hat{G})\setminus E(C)|\leq m/(160\mu)$. Partition $(\mathcal{L}_1,\ldots,\mathcal{L}_r)$ of the set $\mathcal{W}$ of clusters into $r\leq\log m$ layers naturally defines a partition $L_1,\ldots,L_r$ of the set $U$ of vertices into layers, where vertex $v_W$ lies in layer $L_i$ iff $W\in\mathcal{L}_i$. For convenience, we denote $L_0=V(C)$. For all $1\leq i\leq r$, for every vertex $v\in L_i$, we partition the set $\delta(v)$ of its edges into two subsets: set $\delta^{\mathrm{down}}(v)$ connecting $v$ to vertices of $L_0\cup\cdots\cup L_{i-1}$ and set $\delta^{\mathrm{up}}(v)$ containing all remaining edges, that connect $v$ to vertices of $L_i\cup\cdots\cup L_r$. In the following step, we may move some vertices of $U$ from their current layer to layer $L_0$. The definition of the sets $\delta^{\mathrm{down}}(v'),\delta^{\mathrm{up}}(v')$ of edges is always with respect to the current partition of vertices of $\hat{G}$ into layers. Observe that Property L4 of layered well-linked decomposition ensures the following property:

P1. For every vertex $v\in U$, $|\delta^{\mathrm{up}}(v)|<|\delta^{\mathrm{down}}(v)|/\log m$;

For convenience of notation, in the remainder of this proof we denote instance $\hat{I}=(\hat{G},\hat{\Sigma})$ by $I=(G,\Sigma)$. We use the parameter $m$ from before, so $|E(G)|\leq m$ holds.

### 10.2.2 Step 2: Fixing Petals for Routability

Recall that, as part of the definiton of the flower cluster $C$, we are given a collection $\mathcal{X}=\{X_1,\ldots,X_k\}$ of petals of $C$. Consider now some petal $X_i\in\mathcal{X}$. Let $\hat{E}_i=\delta_G(X_i)\setminus\delta_G(u^*)$, where $u^*$ is the center of the flower cluster $C$. We will use the following definition.

**Definition 10.9** *Let $G$ be a graph, and let $C^f$ be a flower cluster in $G$, with center $u^*$ and a set $\mathcal{X}=\{X_1,\ldots,X_k\}$ of petals. For $1\leq i\leq k$, we say that petal $X_i$ is* routable *in $G$ if there is a collection $\mathcal{Q}'_i=\left\{Q'(e)\mid e\in\hat{E}_i\right\}$ of paths in $G$, where for each edge $e\in\hat{E}_i$, path $Q'(e)$ has $e$ as its first edge, terminates at vertex $u^*$, and its inner vertices are disjoint from $X_i$, such that the paths in $\mathcal{Q}'_i$ cause congestion at most 3000.*

As we show later, if every petal in $\mathcal{X}$ is routable, then we can decompose the current instance $I$ into smaller instances, each of which will correspond to a distinct petal in $\mathcal{X}$ (together with an additional "global" instance). Unfortunately, it is possible that some petals in $\mathcal{X}$ are not routable. We overcome this difficulty by "fixing" the flower cluster $C$. We do so iteratively, while ensuring that Property P1 continues to hold after each iteration. In every iteration, we select some vertex of $U$ to be added to some petal $X_i$ of $\mathcal{X}$. The set $\hat{E}_i=\delta_G(X_i)\setminus\delta_G(u^*)$ of edges is always defined with respect to the current petal $X_i$. In addition to maintaining Property P1, we will maintain the following important property:

P2. For every petal $X_i\in\mathcal{X}$, there is a set $\mathcal{Q}_i=\left\{Q(e)\mid e\in\hat{E}_i\right\}$ of edge-disjoint paths, where for each edge $e\in\hat{E}_i$, path $Q(e)$ has $e$ as its first edge, vertex $u^*$ as its last vertex, and all inner vertices of $Q(e)$ lie in $X_i$.

We now describe the algorithm for fixing the petals of $C$. While there is some petal $X_i\in\mathcal{X}$, and some vertex $v\in U$, such that at least $|\delta_G(v)|/2$ neighbors of $v$ in $G$ lie in $X_i$, we add $v$ to $X_i$, and remove

it from $U$. In other words, we update $X_i$ to be the subgraph of $G$ induced by vertex set $V(X_i) \cup \{v\}$, and we update $C$ to be the subgraph of $G$ induced by vertex set $V(C) \cup \{v\}$. We also remove $v$ from its current layer $L_j$ and add it to $L_0$. It is immediate to verify that Property P1 continues to hold after each iteration. We now show that the same is true for Property P2.

Consider an iteration, when some vertex $v \in U$ was added to some petal $X_i \in \mathcal{X}$. Partition the edges of $\delta_G(v)$ into two subsets: set $\delta'(v)$ connecting vertex $v$ to vertices of $X_i$, and set $\delta''(v)$ containing all remaining edges. From our definitions, at the beginning of the current iteration, $\delta'(v) \subseteq \hat{E}_i$ held. Therefore, set $\mathcal{Q}_i$ contained, for each edge $e \in \delta'(v)$, a path $Q(e)$, connecting $e$ to $u^*$, such that all inner vertices of $Q(e)$ belong to $X_i$. At the end of the current iteration, the edges of $\delta'(v)$ no longer lie in $\hat{E}_i$, and the edges of $\delta''(v)$ are added to $\hat{E}_i$ instead. Since $|\delta''(v)| \leq |\delta'(v)|$, we can define a mapping $M$, that maps every edge of $\delta''(v)$ to a distinct edge of $\delta'(v)$. We update the set $\mathcal{Q}_i$ of paths as follows: first, we remove from it all paths whose first edge lies in $\delta'(v)$. Next, for each edge $e \in \delta''(v)$, we add a new path $Q(e)$ to $\mathcal{Q}_i$, that is obtained by appending $e$ to the original path $Q(e')$, where $e' = M(e)$ is the edge of $\delta'(v)$ to which edge $e$ is mapped. Therefore, Property P2 continues to hold after each iteration. Lastly, we consider the cluster $C$ and its corresponding set $\mathcal{X}$ of petals obtained at the end of the algorithm.

We slightly modify Property F4 of the flower cluster, and replace it with the following property, that we refer to as Modified Property F4:

$$|\delta_G(C)| \leq 96m/\mu^{42} \text{ and } |\bigcup_{i=1}^{k} \delta(X_i)| \leq \frac{192m}{\mu^{42}}.$$

If Properties F1 – F6 hold for a cluster $C'$, with Property F4 replaced with its modified counterpart, then we say that $C'$ is a *modified flower cluster*. We now prove that cluster $C$ is a valid modified cluster.

**Claim 10.10** *Cluster $C$ is a valid modified flower cluster in the current graph $G$.*

**Proof:** It is immediate to verify that throughout the algorithm, Property F1 continues to hold. Recall that, from Property F4 in the definition of a flower cluster, at the beginning of the algorithm, $|\delta_G(C)| \leq 96m/\mu^{42}$ held. We claim that this property continues to hold throughout the algorithm. Indeed, when a vertex $v \in U$ is added to $C$, there is some petal $X_i \in \mathcal{X}$, such that $|\delta'(v)| \geq |\delta''(v)|$, where $\delta'(v)$ contains all edges connecting $v$ to vertices of $X_i$, and $\delta''(v)$ contains all remaining edges of $\delta(v)$. Notice that edges of $\delta'(v)$ are removed from $\delta_G(C)$ at the end of the iteration, while only the edges of $\delta''(v)$ may be added to $\delta_G(C)$ at the end of the current iteration. Therefore, $|\delta_G(C)|$ does not increase, and modified Property F4 continues to hold throughout the algorithm. From the above discussion, whenever a vertex $v$ is added to cluster $C$, $\deg_G(v) \leq 2|\delta_G(C)| \leq 192m/\mu^{42}$ (from Property F4). Therefore, Property F3 holds throughout the algorithm. It is immediate to verify that Properties F2 and F5 continue to hold throughout the algorithm, and we have already established Property F6 for the final cluster $C$. □

Lastly, we show that, once the algorithm terminates, every petal in $\mathcal{X}$ is routable in $G$.

**Claim 10.11** *At the end of the algorithm, every petal of $\mathcal{X}$ is routable in $G$.*

**Proof:** Consider some petal $X_i \in \mathcal{X}$. Recall that we have defined the set $\hat{E}_i = \delta_G(X_i) \setminus \delta_G(u^*)$ of edges, where $u^*$ is the center of the flower cluster $C$. Recall that our goal is to show that there is a collection $\mathcal{Q}'_i = \left\{Q'(e) \mid e \in \hat{E}_i\right\}$ of paths, where for each edge $e \in \hat{E}_i$, path $Q'(e)$ has $e$ as its first edge, terminates at vertex $u^*$, and is internally disjoint from $X_i$, such that the paths in $\mathcal{Q}'_i$ cause

congestion at most 3000. Let $\hat{\mathcal{Q}}_i$ be the set of all paths in graph $G$, where each path $Q \in \hat{\mathcal{Q}}_i$ contains some edge of $\hat{E}_i$ as its first edge, terminates at vertex $u^*$, and is internally disjoint from $X_i$. From the integrality of flow, it is enough to show that there exists a flow $\hat{f}_i$, defined over the set $\hat{\mathcal{Q}}_i$ of paths, in which every edge of $\hat{E}_i$ sends one flow unit, such that flow $\hat{f}_i$ causes congestion at most 3000. From now on we focus on proving that such a flow indeed exists.

For the sake of the proof we will define layer $L_0$ slightly differently than before: we let $L_0 = V(C) \setminus V(X_i \setminus \{u^*\})$. For each index $0 \leq j \leq r$, we let $S_j = L_0 \cup L_1 \cup \cdots \cup L_j$. We then let the set $E^*_j$ of edges contain all edges of $\delta_G(S_j)$, except for those insident to vertex $u^*$. Notice that in particular, since $S_r = V(G) \setminus (V(X_i) \setminus \{u^*\})$, edge set $E^*_r$ is precisely the edge set $\hat{E}_i = E_G(X_i) \setminus \delta_G(u^*)$. For all $0 \leq j \leq r$, we let $\mathcal{P}^*_j$ be the set of all paths $P$, such that the first edge of $P$ lies in $E^*_j$, the last vertex of $P$ is $u^*$, and all inner vertices of $P$ lie in $S_j$. We prove the following claim.

**Claim 10.12** *For all $0 \leq j \leq r$, there is a flow $f^*_j$ defined over the set $\mathcal{P}^*_j$ of paths, in which every edge of $E^*_j$ sends one flow unit, such that the paths in $\mathcal{P}^*_j$ cause congestion at most $\left(1 + \frac{8}{\log m}\right)^j$.*

Note that proof of Claim 10.12 will finish the proof of Claim 10.11. Indeed, as observed already, $E^*_r = \hat{E}_i$, and it is easy to verify that $\mathcal{P}^*_r = \hat{\mathcal{Q}}_i$. In flow $f^*_r$, every edge of $\hat{E}_i$ sends one flow unit, as required, and the congestion of the flow is at most $\left(1 + \frac{8}{\log m}\right)^r \leq 3000$, since $r \leq \log m$. Therefore, in order to complete the proof of Claim 10.11, it is now enough to prove Claim 10.12, which we do next.

**Proof of Claim 10.12.** The proof is by induction on $j$. The base is when $j = 0$. Recall that $S_0 = L_0 = V(C) \setminus (V(X_i) \setminus u^*)$. The set $E^*_0$ of edges is then a subset of $\bigcup_{i' \neq i} \hat{E}_{i'}$. Recall that, from the definition of the flower cluster, for all $1 \leq i' \leq k$, there is a collection $\mathcal{Q}_{i'}$ of edge-disjoint paths routing the edges of $\hat{E}_{i'}$ to vertex $u^*$, with all inner vertices on every path contained in $X_{i'}$. For each index $i \neq i'$, for each edge $e \in \hat{E}_{i'}$, let $Q(e) \in \mathcal{Q}_{i'}$ be the unique path whose first edge is $e$. Observe that $\bigcup_{i' \neq i} \mathcal{Q}_{i'} \subseteq \mathcal{P}^*_0$. By sending one flow unit on each path in $\left\{Q(e) \mid e \in \hat{E}_0\right\}$, we obtain the desired flow $f^*_0$, defined over the set $\mathcal{P}^*_0$ of paths, in which each edge of $E^*_0$ sends one flow unit. The congestion of the flow is 1.

We now prove that the claim holds for an index $1 \leq j \leq r$, provided that it holds for index $j-1$.

Consider some vertex $v \in L_j$. We partition the edges of $\delta^{\text{down}}(v)$ into two subsets: set $\delta^{\text{down}}_1(v)$ containing all edges connecting $v$ to vertices of $X_i$, and set $\delta^{\text{down}}_2(v)$ containing all remaining edges of $\delta^{\text{down}}(v)$. Notice that the edges of $\delta^{\text{down}}_2(v)$ lie in $E^*_{j-1}$ but not in $E^*_j$, while edges of $\delta^{\text{up}}(v)$ lie in $E^*_j$ but not in $E^*_{j-1}$. In fact, since $S_j = S_{j-1} \cup L_j$, $E^*_j \subseteq \left(E^*_{j-1} \setminus \left(\bigcup_{v \in L_j} \delta^{\text{down}}_2(v)\right)\right) \cup \left(\bigcup_{v \in L_j} \delta^{\text{up}}(v)\right)$ (we use inclusion rather than equality since an edge of $\delta^{\text{up}}(v)$ may connect $v$ to a vertex of $L_j$).

Consider again some vertex $v \in L_j$. Recall that $|\delta^{\text{down}}_1(v)| \leq |\deg_G(v)|/2$ (since the algorithm for fixing the flower cluster $C$ has terminated), while $|\delta^{\text{up}}(v)| \leq |\delta^{\text{down}}(v)|/\log m \leq \deg_G(v)/\log m$. Therefore, $|\delta^{\text{down}}_2(v)| \geq \left(\frac{1}{2} - \frac{1}{\log m}\right) \deg_G(v)$, while $|E^*_j \cap \delta_G(v)| \subseteq |\delta^{\text{up}}(v)| + |\delta^{\text{down}}_1(v)| \leq \left(\frac{1}{2} + \frac{1}{\log m}\right) \deg(v)$. Overall, we get that $|E^*_j \cap \delta_G(v)| \leq \left(1 + \frac{8}{\log m}\right) |\delta^{\text{down}}_2(v)|$.

Let $\mathcal{R}_j(v)$ denote the collection of all paths that can be obtained by combining two edges: an edge of $E^*_j \cap \delta_G(v)$ and an edge of $\delta^{\text{down}}_2(v)$; the paths are directed towards edges of $\delta^{\text{down}}_2(v)$. Clearly, there is a flow $f'_v$, defined over the paths in $\mathcal{R}_j(v)$, where every edge of $E^*_j \cap \delta_G(v)$ sends one flow unit, every edge of $\delta^{\text{down}}_2(v)$ receives at most $\left(1 + \frac{8}{\log m}\right)$ flow units, and the flow causes congestion at most $\left(1 + \frac{8}{\log m}\right)$ (for example, we can obtain such a flow by spreading the flow originating at every edge of $E^*_j \cap \delta_G(v)$ evenly among the edges of $\delta^{\text{down}}_2(v)$).

Next, we define a new flow $f_j(v)$, in which every edge of $E^*_j \cap \delta_G(v)$ sends one flow unit via a subset of paths of $\mathcal{P}^*_j$. Consider any flow-path $R \in \mathcal{R}_j(v)$, and assume that $R$ consists of two edges: $e \in$

$E^*_j \cap \delta_G(v)$ and $e' \in \delta^{\text{down}}_2(v)$. Recall that edge $e'$ sends 1 flow unit in flow $f^*_{j-1}$. For every path $P \in \mathcal{P}^*_{j-1}$ whose first edge is $e'$, we consider a path $R^P$ obtained by appending the edge $e$ at the beginning of the path, so that path $R^P$ now starts with edge $e$, and terminates at vertex $u^*$ as before. Observe that path $R^P$ lies in path set $\mathcal{P}^*_j$. Let $x = f^*_{j-1}(P)$ be the amount of flow sent via path $P$ in flow $f_{j-1}$, and let $x' = f'_v(R)$ be the amount of flow sent via path $R$ in flow $f'_v$. We then send $(x \cdot x')$ flow units via path $R^P$ in flow $f_j(v)$. Notice that, since every edge of $E^*_j \cap \delta_G(v)$ sends one flow unit in flow $f'_v$, and every edge of $\delta^{\text{down}}_2(v)$ sends one flow unit in flow $f_{j-1}$, this ensures that every edge of $E^*_j \cap \delta_G(v)$ sends one flow unit in the new flow $f_j(v)$ that we just defined. Moreover, since every edge $e' \in \delta^{\text{down}}_2(v)$ receives at most $\left(1 + \frac{8}{\log m}\right)$ flow units in $f'_v$, for each flow-path $P \in \mathcal{P}^*_{j-1}$ whose first edge is $e'$, the total amount of flow sent along path $P$ in the new flow $f_j(v)$ is at most $\left(1 + \frac{8}{\log m}\right)$ times the amount of flow sent via path $P$ in $f^*_{j-1}$. In other words, we can think of flow $f_j(v)$ as obtained as follows: we start with flow $f^*_{j-1}$, and discard flow on all flow-paths except those whose first edge lies in $\delta^{\text{down}}_2(v)$. Next, we scale the flow on each resulting flow-path by at most factor $\left(1 + \frac{8}{\log m}\right)$. Lastly, we combine the resulting flow with flow $f'_v$.

We are now ready to define the final flow $f^*_j$. Recall that the set $E^*_j$ of edges can be obtained from edge set $E^*_{j-1}$ by first deleting the edges of $\bigcup_{v \in L_j} \delta^{\text{down}}_2(v)$ from it, and then adding a subset of the edges of $\bigcup_{v \in L_j} \delta^{\text{up}}(v)$ to it. For every edge $e \in E^*_j \cap E^*_{j-1}$, for every path $P \in \mathcal{P}^*_{j-1} \cap \mathcal{P}^*_j$, whose first edge is $e$, the flow $f^*_j(P)$ remains the same as the flow $f^*_{j-1}(P)$. This ensures that each edge of $E^*_j \cap E^*_{j-1}$ sends one flow unit in the new flow, as $\mathcal{P}_{j-1} \subseteq \mathcal{P}_j$. For every vertex $v \in L_j$, we use the flow $f_j(v)$ in order to send flow from the edges of $\delta^{\text{up}}(v) \cap E^*_j$. Specifically, for each edge $e \in \delta^{\text{up}}(v) \cap E^*_j$, for every path $P \in \mathcal{P}^*_j$ whose first edge is $e$, we set the flow $f^*_j(P)$ to be equal to the flow sent via this path by $f_j(v)$. This ensures that every edge in $E^*_j \setminus E^*_{j-1}$ sends one flow unit in the new flow $f^*_j$. This finishes the description of the flow $f^*_j$. From the above discussion, every edge of $E^*_j$ sends one flow unit in $f^*_j$. It now remains to analyze the congestion of the flow.

Observe that flow $f^*_j$ can be obtained as follows. We start with the flow $f^*_{j-1}$, and we scale the flow on some of the flow-paths by at most factor $\left(1 + \frac{8}{\log m}\right)$ (this is since for every vertex $v \in L_j$, for each edge $e \in \delta^{\text{down}}_2(v)$, edge $e$ receives at most $\left(1 + \frac{8}{\log m}\right)$ flow units via flow $f'_v$, and this flow utilizes the flow that $e$ sends in $f^*_{j-1}$ in order to reach vertex $u^*$). Lastly, we combine the resulting flow with the flows $f'_v$ for all vertices $v \in L_j$. It is then easy to verify that the congestion caused by flow $f^*_j$ is bounded by the congestion caused by flow $f^*_{j-1}$ times $\left(1 + \frac{8}{\log m}\right)$. Since, from the induction hypothesis, flow $f^*_{j-1}$ causes congestion at most $\left(1 + \frac{8}{\log m}\right)^{j-1}$, flow $f^*_j$ causes congestion at most $\left(1 + \frac{8}{\log m}\right)^{j}$. □

□

### 10.3 Phase 3: Petal-Based Disengagement and the Final Family of Instances

In this subsection we first compute a collection $\mathcal{I}_3$ of subinstances of the current instance $I = (G, \Sigma)$. These subinstances will "almost" have all properties required in Lemma 10.8, except that we will not be able to guarantee that for each resulting subinstance $\tilde{I} = (\tilde{G}, \tilde{E})$, $|E(\tilde{G})| \leq m/(2\mu)$. However, we will guarantee that each such resulting graph $\tilde{G}$ does not have vertices whose degree is at least $m/\mu^4$. This fact will be used in the second part of this subsection in order to further decompose each subinstance of $\mathcal{I}_3$ into smaller subinstances. This will be done using an algorithm similar to that from Phase 1, except that now, since the instances we apply the algorithm to do not have high-degree vertices, we will not obtain any flower clusters, and so each subinstance obtained in this final phase will be sufficiently small.

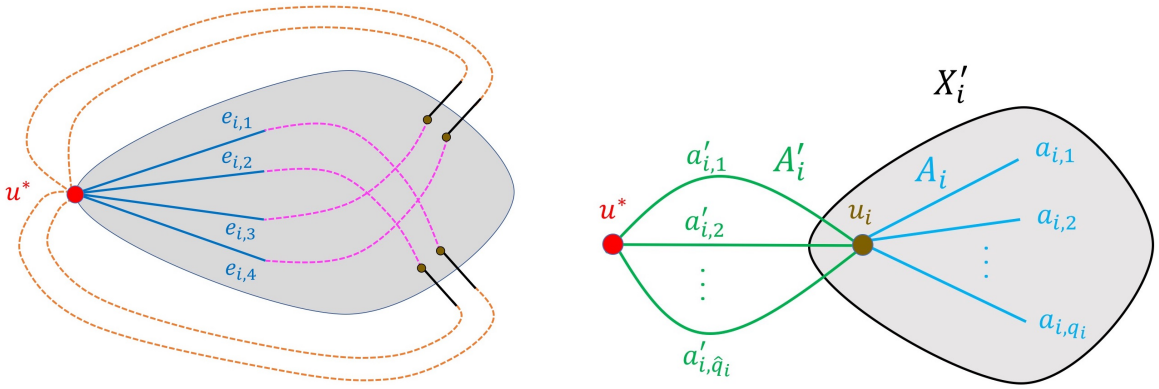
The intuition for the current phase is that we would like to define a set of clusters in the current graph $G$, using the set $\mathcal{X} = \{X_1, \ldots, X_k\}$ of petals of the flower cluster, and then perform basic disengagement, described in Section 5.3 on instance $I$ with the set $\mathcal{X}$ of clusters. Unfortunately, the set $\mathcal{X}$ of clusters is not laminar, as the clusters in $\mathcal{X}$ all share vertex $u^*$. In order to overcome this obstacle, the algorithm in this phase consists of three steps. In the first step we "split" the vertex $u^*$, by creating new vertices $u_1, \ldots, u_k$, each of which is then added to a distinct cluster $X_i$. We show that the optimal solution value to the resulting split instance that we construct is bounded by $\mathsf{OPT}_{\mathsf{cnwrs}}(I)$, and that any solution to this new split instance can be efficiently transformed into a solution to the original instance, by only slightly increasing the solution cost. In the second step, we perform a basic disengagement of the new split instance using the modified set $\mathcal{X}$ of clusters. We show that each of the resulting instances does not contain vertices of degree at least $m/\mu^4$. We also bound the total solution cost and the total number of edges in the new instances. In the third and the final step we further decompose each resulting instance, exploiting the low degrees of its vertices. We now describe each of the steps in turn.

### 10.3.1 Step 1: the Split Instance

Recall that we are given an instance $I = (G, \Sigma)$ of the MCNwRS problem, and a (modified) flower cluster $C \subseteq G$ with center $u^*$, and a set $\mathcal{X} = \{X_1, \ldots, X_k\}$ of petals, such that each petal is routable in $G$.

For all $1 \leq i \leq r$, we let $E_i = E(X_i) \cap \delta_G(u^*)$, and we denote $E_i = (e_{i,1}, \ldots, e_{i,q_i})$, where the edges are indexed according to their order in the ordering $\mathcal{O}_{u^*} \in \Sigma$; in other words, the ordering of the set $\delta_G(u^*)$ of edges in $\Sigma$ is: $\mathcal{O}_{u^*} = (e_{1,1}, \ldots, e_{1,q_1}, e_{2,1}, \ldots, e_{2,q_2}, \ldots, e_{k,1}, \ldots, e_{k,q_k})$. For all $1 \leq i \leq r$, we also let $\hat{E}_i = \delta_G(X_i) \setminus \delta_G(u^*)$, and we denote $|\hat{E}_i| = \hat{q}_i$; see Figure 36(a) for an illustration.

Recall that, from Property F6 of a flower cluster, there is a set $\mathcal{Q}_i$ of edge-disjoint paths routing the edges of $\hat{E}_i$ to the edges of $E_i$, such that every inner vertex on every path lies in $X_i$, and, since petal $X_i$ is routable in $G$, there is a set $\mathcal{Q}'_i$ of paths in graph $G$, routing the edges of $\hat{E}_i$ to vertex $u^*$ such that the paths are internally disjoint from $X_i$ and cause congestion at most 3000.




(a) An original petal $X_i$. Paths of $\mathcal{Q}_i$ are shown in pink and paths of $\mathcal{Q}'_i$ are shown in orange.

(b) New cluster $X'_i$.

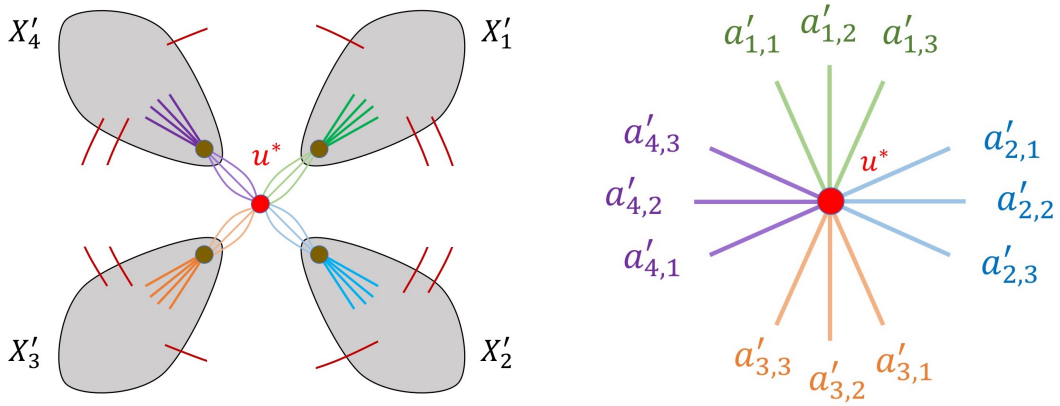
Figure 36: Construction of a split instance $I' = (G', \Sigma')$.

In order to define the new split instance $I' = (G', \Sigma')$, we start with a graph $G' = G \setminus \delta_G(u^*)$. We then add $k$ new vertices $u_1, \ldots, u_k$ to $G'$. Next, we process each index $1 \leq i \leq k$ one by one. When index $i$ is processed, we add a collection $A'_i = \left\{a'_{i,1}, \ldots, a'_{i,\hat{q}_i}\right\}$ of $\hat{q}_i$ parallel edges connecting $u^*$ to $u_i$ (recall that $\hat{q}_i = |\hat{E}_i|$). Additionally, for every edge $e_{i,j} = (u^*, x_{i,j}) \in E_i$, we add a new edge $a_{i,j} = (u_i, x_{i,j})$

to graph $G'$; we view $a_{i,j}$ as a copy of edge $e_{i,j}$, and we will not distinguish between these edges. We denote $A_i = \{a_{i,j} \mid 1 \leq j \leq q_i\}$. In order to complete the construction of graph $G'$, for every edge $e = (u, v) \notin \delta_G(u^*)$ of the graph $G$ whose endpoints lie in different petals, we subdivide the edge $e$ with a new vertex $y_e$.

For all $1 \leq i \leq k$, we let $X'_i$ be the subgraph of $G'$ induced by $(V(X_i) \setminus \{u^*\}) \cup \{u_i\}$. Notice that graph $X'_i$ is completely identical to graph $X_i$, except that vertex $u^*$ is replaced by vertex $u_i$. For all $1 \leq i \leq k$, we denote by $\hat{A}_i = \delta_{G'}(X'_i) \setminus A'_i$, where $A'_i$ is the set of parallel edges connecting $u_i$ to $u^*$ (see Figure 36(b)). It is easy to see that there is a one-to-one correspondence between edges of $\hat{A}_i$ in graph $G'$ and edges of $\hat{E}_i$ in graph $G$.

In order to complete the definition of the split instance $I'$, we need to define its corresponding rotation system $\Sigma'$. It is easy to verify that, every vertex $v \in V(G') \setminus \{u^*, u_1, \ldots, u_k\}$ whose degree in $G'$ is greater than 2, $\delta_{G'}(v) = \delta_G(v)$ holds (we do not distinguish here between edges whose endpoints lie in different petals of $G$ and their subdivided counterparts). For each such vertex, we set the ordering $\mathcal{O}'_v \in \Sigma'$ of the edges of $\delta_{G'}(v)$ to be the same as the ordering $\mathcal{O}_v \in \Sigma$ of the edges of $\delta_G(v)$. Note that $\delta_{G'}(u^*) = A'_1 \cup \cdots \cup A'_k$. We set the ordering $\mathcal{O}'_{u^*} \in \Sigma'$ of the edges of $\delta_{G'}(u^*)$ to be $(a'_{1,1}, \ldots, a'_{1,\hat{q}_1}, a'_{2,1}, \ldots, a'_{2,\hat{q}_2}, \ldots, a'_{k,1}, \ldots, a'_{k,\hat{q}_k})$. In other words, edges in sets $A'_1, \ldots, A'_k$ appear in this order of their sets, and within each set $A'_i$, the edges of $\left\{a'_{i,j}\right\}_{j=1}^{\hat{q}_i}$ are ordered in the increasing order of index $j$. Lastly, for all $1 \leq i \leq k$, we define the ordering $\mathcal{O}'_{u_i} \in \Sigma'$ of the edges of $\delta_{G'}(u_i) = A'_i \cup A_i$ to be: $(a'_{i,1}, a'_{i,2}, \ldots, a'_{i,\hat{q}_i}, a_{i,q_i}, a_{i,q_i-1}, \ldots, a_{i,1})$.




(a) Schematic view of graph $G'$ when $C$ is a 4-petal flower cluster.

(b) The ordering $\mathcal{O}'_{u^*} \in \Sigma'$ of edges of $\delta_{G'}(u^*)$.

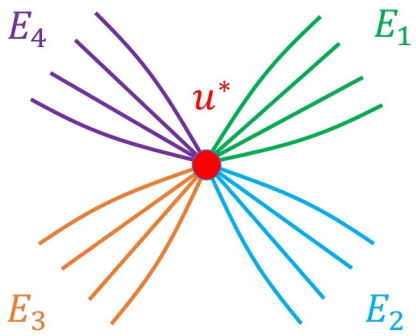
Figure 37: Split instance $I' = (G', \Sigma')$.

This completes the definition of the new split instance $I' = (G', \Sigma')$; see Figure 37 for an illustration. We now establish some of its properties. We start with the following easy observation:

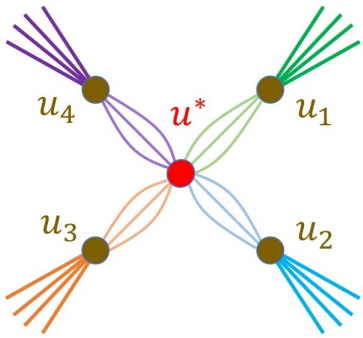
**Observation 10.13** *$|E(G')| \leq 4|E(G)|$, and $\mathsf{OPT}_{\mathsf{cnwrs}}(I') \leq \mathsf{OPT}_{\mathsf{cnwrs}}(I)$.*

The proof of Observation 10.13 is immediate. The first statement is immediate to see. For the second statement, given any solution $\varphi$ to instance $I$, we can obtain a solution $\varphi'$ to instance $I'$ by splitting the vertex $u^*$ to obtain images of vertices $u_1, \ldots, u_k$ and images of the edges in sets $A'_1, \ldots, A'_k$ in a natural way (see Figure 38), and subdividing images of edges whose endpoints lie in different petals of $G$.

The next lemma shows that a solution to instance $I'$ can be transformed into a solution to instance $I$ while only slightly increasing the solution cost. The proof uses arguments similar to those used in




(a) Before: the images of the original vertex $u^*$ and its incident edges in $\varphi$.




(b) After: the images of the new vertices $u, u_1, \ldots, u_k$ and their incident edges in $\varphi'$.

Figure 38: Transforming a solution for instance $I$ into a solution for instance $I'$.

basic and advanced disengagement, but is somewhat tedious, and is deferred to Appendix I.2.

**Lemma 10.14** *There is an efficient algorithm that, given a solution $\varphi'$ to instance $I'$, computes a solution $\varphi$ to instance $I$, with $\mathsf{cr}(\varphi) \leq O(\mathsf{cr}(\varphi'))$.*

### 10.3.2 Step 2: Disengagement of the Petals

In this step, we consider the split instance $I' = (G', \Sigma')$ that was constructed in Step 1 of the current phase, and we will apply Algorithm AlgBasicDisengagement from Section 5.3 to this instance, together with the family $\mathcal{L} = \{X'_1, \ldots, X'_k\}$ of clusters in order to perform a basic disengagement of these clusers, with a parameter $\beta = c(\log m)^{18}$, where $c$ is a large enough constant. Note that the clusters of $\mathcal{L}$ are disjoint, so $\mathcal{L}$ is a laminar family of clusters. In order to be able to use AlgBasicDisengagement, we need to define, for each cluster $X'_i$, a distribution $\mathcal{D}'(X'_i)$ over the set $\Lambda'(X'_i)$ of external routers for $X'_i$.

Consider some cluster $X'_i \in \mathcal{L}$. Recall that petal $X_i$ is routable in $G$, and so there is a set $\mathcal{Q}'_i$ of paths in $G$, routing the edges of $\hat{E}_i = \delta_G(X_i) \setminus \delta_G(u^*)$ to vertex $u^*$, such that the paths in $\mathcal{Q}'_i$ cause congestion at most 3000, and they are internally disjoint from $X_i$. By suitably subdividing the first edge of every path in $\mathcal{Q}'_i$, and by replacing the last edge $e_{i',j}$ on each such path by the corresponding edge $a_{i',j}$, we obtain a collection $\mathcal{Q}''_i$ of paths in graph $G'$, routing the edges of $\hat{A}_i$ to vertices of $\{u_{i'}\}_{i' \neq i}$, such that the paths in $\mathcal{Q}''_i$ are internally disjoint from $X'_i$, and cause congestion at most 3000. Note that, for all $1 \leq i' \leq k$ with $i' \neq i$, the number of paths terminating at vertex $u_{i'}$ is at most $3000|\hat{A}'_{i'}| \leq 3000\hat{q}_i \leq 3000|A'_{i'}|$. Therefore, by appending an edge of $A'_{i'}$ at the end of each such path, for all indices $i' \neq i$, we obtain a set $\mathcal{P}'_i = \left\{P(\hat{a}) \mid \hat{a} \in \hat{A}_i\right\}$ of paths in graph $G'$, that cause congestion at most 3000, such that for each edge $\hat{a} \in \hat{A}_i$, path $P(\hat{a})$ has $\hat{a}$ as its first edge, terminates at vertex $u^*$, and is internally disjoint from $X'_i$. Lastly, for every edge $a'_{i,j} \in A'_i$, we define a path $P(a'_{i,j})$ consisting of only the edge $a'_{i,j}$ itself, and add that path to set $\mathcal{P}'_i$. We have now obtained a set $\mathcal{P}'_i$ of paths in graph $G'$, routing the edges of $\delta_{G'}(X'_i)$ to vertex $u^*$, such that the paths are internally disjoint from $X'_i$. Therefore, $\mathcal{P}'_i \in \Lambda'(X'_i)$. We then let the distribution $\mathcal{D}'(X'_i)$ choose the path set $\mathcal{P}'_i$ with probability 1.

We add each such cluster $X'_i$ to the set $\mathcal{L}^{\text{light}}$ of light clusters, and define, for each such cluster $X'_i$, a distribution $\mathcal{D}(X'_i)$ over the set $\Lambda(X'_i)$ of internal routers for $X'_i$, such that $X'_i$ is $\beta$-light with respect

to $\mathcal{D}(X_i')$. In fact, the disctibution $\mathcal{D}(X_i')$ will select a single set $\mathcal{P}_i \in \Lambda(X_i')$ of paths with probability 1. The set $\mathcal{P}_i$ of paths is constructed as follows. From the properties of the flower cluster, there is a set $\mathcal{Q}_i$ of edge-disjoint paths in graph $G$, routing the edges of $\delta_G(X_i) \setminus \delta_G(u^*)$ to vertex $u^*$, such that every inner vertex on every path lies in $X_i$. Since cluster $X_i'$ can be obtained from $X_i$ by replacing vertex $u_i$ with vertex $u^*$, we obtain a collection $\mathcal{P}_i$ of edge-disjoint paths, routing the edges of $\hat{A}_i$ to vertex $u_i$, such that every inner vertex on every path lies in $X_i'$. For every edge $a'_{i,j} \in A_i'$, we add a path $P(a'_{i,j})$, consisting of the edge $a'_{i,j}$ only, to set $\mathcal{P}_i$. We then obtain a set $\mathcal{P}_i$ of edge-disjoint paths, routing the edges of $\delta_{G'}(X_i')$ to vertex $u_i$ inside $X_i'$, that is, $\mathcal{P}_i \in \Lambda(X_i)$.

Consider the set $\mathcal{I}_3$ of subinstances of $I'$, that is obtained by performing a basic disengagement of instance $I'$ via the tuple $(\mathcal{L}, \mathcal{L}^{\text{bad}}, \mathcal{L}^{\text{light}}, \{\mathcal{D}'(X_i')\}_{i=1}^k, \{\mathcal{D}(X_i')\}_{i=1}^k)$ (here, we set $\mathcal{L}^{\text{bad}} = \emptyset$).

Recall that family $\mathcal{I}_3$ of instances contains a single global instance $\hat{I} = (\hat{G}, \hat{\Sigma})$, where graph $\hat{G}$ is obtained from graph $G'$ by contracting, for all $1 \leq i \leq k$, the vertices of $X_i'$ into a supernode. Additionally, for every cluster $X_i' \in \mathcal{L}$, we obtain an instance $I(X_i') = (G_i, \Sigma_i)$, where graph $G_i$ is obtained from graph $G_i$, by contracting all vertices of $V(G') \setminus V(X_i')$ into a supernode.

We summarize the properties of the resulting family $\mathcal{I}_3$ of instances in the following claim.

**Claim 10.15**
- $\sum_{\tilde{I}=(\tilde{G},\tilde{\Sigma})\in\mathcal{I}_3} |E(\tilde{G})| \leq O(|E(G)|)$;
- $\mathbf{E}\left[\sum_{\tilde{I}\in\mathcal{I}_3} \mathsf{OPT}_{\mathsf{cnwrs}}(\tilde{I})\right] \leq O\left((\mathsf{OPT}_{\mathsf{cnwrs}}(I) + |E(G)|) \cdot \log^{36} m\right)$;
- *There is an efficient algorithm, that, given, for each instance $\tilde{I} \in \mathcal{I}_3$, a solution $\varphi(\tilde{I})$, computes a solution to instance $I$ of cost at most $O\left(\sum_{\tilde{I}\in\mathcal{I}_3} \mathsf{cr}(\varphi(\tilde{I}))\right)$.*

**Proof:** For the first assertion, recall that, from Lemma 5.2 $\sum_{\tilde{I}=(\tilde{G},\tilde{\Sigma})\in\mathcal{I}_3} |E(\tilde{G})| \leq O(|E(G')|)$. Since, from the construction of the split instance, $|E(G')| \leq O(|E(G)|)$, the assertion follows.

In order to prove the second assertion, recall that, from Lemma 5.6, $\mathbf{E}\left[\sum_{\tilde{I}\in\mathcal{I}_3} \mathsf{OPT}_{\mathsf{cnwrs}}(\tilde{I})\right] \leq O\left(\beta^2 \cdot (\mathsf{OPT}_{\mathsf{cnwrs}}(I') + |E(G')|)\right)$. Since, as discussed above, $\mathsf{OPT}_{\mathsf{cnwrs}}(I') \leq \mathsf{OPT}_{\mathsf{cnwrs}}(I)$, $|E(G')| \leq O(|E(G)|)$, and $\beta \leq O(\log^{18} m)$, the assertion follows.

In order to prove the last assertion, we use the algorithm from Lemma 5.3, that, given, for each instance $\tilde{I} \in \mathcal{I}_3$, a solution $\varphi(\tilde{I})$, computes a solution $\varphi'$ for instance $I'$ of cost at most $\sum_{\tilde{I}\in\mathcal{I}_3} \mathsf{cr}(\varphi(\tilde{I}))$. We then use the algorithm from Lemma 10.14 in order to compute a solution $\varphi$ for instance $I$ of cost at most $O(\mathsf{cr}(\varphi')) \leq O\left(\sum_{\tilde{I}\in\mathcal{I}_3} \mathsf{cr}(\varphi(\tilde{I}))\right)$. □

Consider now the global instance $\hat{I} = (\hat{G}, \hat{\Sigma})$. Since graph $\hat{G}$ is obtained from $G'$ by contracting every cluster $X_i'$ into a supernode, for every edge $e \in E(\hat{G})$, either $e$ is incident to $u^*$, or it corresponds to an edge of $E^{\mathsf{out}}(\mathcal{C})$, where $\mathcal{C}$ is the initial collection of clusters that we computed in Phase 1. Recall that, from Equation (15), $|E^{\mathsf{out}}(\mathcal{C})| \leq m/(80\mu)$. Recall that $\deg_{G'}(u^*) = \sum_{i=1}^k \hat{q}_i = \sum_{i=1}^k |\hat{E}_i|$. From Modified Property F4 of the flower cluster, $\sum_{i=1}^k |\hat{E}_i| \leq 200m/\mu^{42}$. Therefore, overall, $|E(\hat{G})| \leq |E^{\mathsf{out}}(\mathcal{C})| + \deg_{G'}(u^*) \leq m/(40\mu)$.

Next, we consider petal-based instances, and we prove that for each such instance, the maximum vertex degree is small.

**Claim 10.16** *For all $1 \leq i \leq k$, if $I(X_i') = (G_i, \Sigma_i)$ is the instance of $\tilde{\mathcal{I}}$ associated with cluster $X_i'$, then every vertex degree in graph $G_i$ is less than $m/\mu^4$.*

**Proof:** Recall that graph $G_i$ is obtained from graph $G'$ by contracting all vertices of $V(G') \setminus V(X_i')$ into a supernode, that we denote by $u'$. Recall that graph $X_i'$ is identical to the petal $X_i$, except that

we replace vertex $u^*$ with vertex $u_i$. From the definition of a flower cluster, every vertex of $X_i$, except for vertex $u^*$, has degree less than $m/\mu^4$ in $G$. The degree of vertex $u_i$ in the new graph is bounded by $q_i + \hat{q}_i$. Here, $q_i = |A_i| = |E_i| \leq m/(2\mu^4)$ from Property F5 of the flower cluster, and $\hat{q}_i \leq 200m/\mu^{42}$ from Modified Property F4 of the flower cluster. Therefore, the degree of $u_i$ in graph $G_i$ is less than $m/\mu^4$. It now remains to bound the degree of the supernode $u'$ in graph $G_i$. The edges incident to $u'$ are the edges of $A'_i \cup \hat{A}_i$, and their number is bounded by $2\hat{q}_i$, which, from the above discussion, is bounded by $400m/\mu^{42}$. □

### 10.3.3 Step 3: Final Decomposition

In this step, we consider each petal-based instance $I(X'_i) = (G_i, \Sigma_i)$ in which $|E(G_i)| > m/(2\mu)$. We further decompose each such instance into subinstances, by exploiting the fact that graph $G_i$ does not have high-degree vertices, using the following lemma.

**Lemma 10.17** *There is an efficient randomized algorithm, that, given an instance $\tilde{I} = (\tilde{G}, \tilde{\Sigma})$ of* MCNwRS *and parameters $m, \mu$, such that $m$ is greater than a large enough constant, $m/(2\mu) < |E(\tilde{G})| \leq 3m$, $\mu \geq 2^{\Omega(\sqrt{\log m})}$, and maximum vertex degree in $\tilde{G}$ is less than $m/\mu^4$, either correctly establishes that $\mathsf{OPT}_{\mathsf{cnwrs}}(I) \geq \Omega\left(\frac{m^2}{\mu^{5.5}}\right)$, or computes a $\nu_3$-decomposition $\tilde{\mathcal{I}}'$ of $\tilde{I}$, for $\nu_3 = 2^{O((\log m)^{3/4} \log\log m)}$, such that, for every instance $\tilde{I}' = (\tilde{G}', \tilde{\Sigma}') \in \tilde{\mathcal{I}}'$, $|E(\tilde{G}')| \leq m/(2\mu)$.*

We prove the lemma below, after we complete the proof of Lemma 10.8 using it. Consider some index $1 \leq i \leq k$. If $|E(G_i)| < m/(2\mu)$, then we let the set $\mathcal{I}(X'_i)$ of subinstances of $I(X'_i)$ consist of a single instance – instance $I(X'_i)$. Otherwise, we apply the algorithm from Lemma 10.17 to instance $I(X'_i) = (G_i, \Sigma_i) \in \mathcal{I}_3$. If the algorithm from Lemma 10.17 computes a $\nu_3$-decomposition $\tilde{\mathcal{I}}$ of $I(X'_i)$, such that, for every instance $\tilde{I}' = (\tilde{G}', \tilde{\Sigma}') \in \tilde{\mathcal{I}}$, $|E(\tilde{G}')| \leq m/(2\mu)$, then we set $\mathcal{I}(X'_i) = \tilde{\mathcal{I}}$. Otherwise, we terminate the algorithm and return FAIL.

If, every time Lemma 10.17 is invoked, it returns a $\nu_3$-decomposition of the corresponding instance $I(X'_i)$, then we output a collection $\left\{\hat{I}\right\} \cup \left(\bigcup_{i=1}^{k} \mathcal{I}(X'_i)\right)$ of instances. From Claim 2.11, it is immediate to verify that this algorithm produces a $\nu_2$-decomposition of instance $I$ for $\nu_2 = O(\nu_3)$ (since the family $\mathcal{I}_3$ of subinstances of $I$ computed in Step 2 of the current phase is an $O(\log^{36} m)$-decomposition of instance $I$, from Claim 10.15), and the graph associated with each instance has at most $m/(2\mu)$ edges.

Assume now that $\mathsf{OPT}_{\mathsf{cnwrs}}(\hat{I}) < \frac{m^2}{c''\mu^{13}}$ for some large enough constant $c''$. Recall that $|E(G)| \leq O(m)$, and, from the statement of Theorem 3.14, $m \geq \mu^{50}$, so $|E(G)| < \frac{m^2}{\mu^{13}}$. Therefore, $(\mathsf{OPT}_{\mathsf{cnwrs}}(\hat{I}) + |E(G)|) < \frac{2m^2}{c''\mu^{13}}$.

Recall that, from Claim 10.15, $\mathbf{E}\left[\sum_{\tilde{I} \in \mathcal{I}_3} \mathsf{OPT}_{\mathsf{cnwrs}}(\tilde{I})\right] \leq O(\mathsf{OPT}_{\mathsf{cnwrs}}(I) + |E(G)|)$. We say that a bad event $\mathcal{E}''$ happens if $\sum_{i=1}^{k} \mathsf{OPT}_{\mathsf{cnwrs}}(I(X'_i)) > c\mu^5(\mathsf{OPT}_{\mathsf{cnwrs}}(I) + |E(G)|)$ for some large enough constant $c$. From Markov's inequality, $\mathbf{Pr}\left[\mathcal{E}''\right] \leq 1/(8\mu^4)$.

If $\mathsf{OPT}_{\mathsf{cnwrs}}(\hat{I}) < \frac{m^2}{c''\mu^{13}}$, and the bad event $\mathcal{E}''$ did not appen, then for all $1 \leq i \leq k$, $\mathsf{OPT}_{\mathsf{cnwrs}}(I(X'_i)) < c\mu^5(\mathsf{OPT}_{\mathsf{cnwrs}}(I) + |E(G)|) \leq \frac{2cm^2}{c''\mu^8}$. Note that our algorithm may only return FAIL if there is some index $1 \leq i \leq k$, such that $|E(G_i)| \geq m/(2\mu)$, and $\mathsf{OPT}_{\mathsf{cnwrs}}(I(X'_i)) \geq \Omega\left(\frac{|E(G_i)|^2}{\mu^{5.5}}\right) \geq \Omega\left(\frac{m^2}{\mu^{7.5}}\right)$. From the above discussion, and since we can choose $c''$ to be a large enough constant compared to $c$, if $\mathsf{OPT}_{\mathsf{cnwrs}}(\hat{I}) < \frac{m^2}{c''\mu^{13}}$, then the algorithm may only return FAIL if $\mathcal{E}''$ happens, which happens with probability at most $1/(8\mu^4)$.

In order to complete Lemma 10.8, and Theorem 3.14, it is now enough to prove Lemma 10.17.

**Proof of Lemma 10.17.** In order to simplify the notation, we denote instance $\tilde{I} = (\tilde{G}, \tilde{\Sigma})$ by $I = (G, \Sigma)$. We will essentially repeat the algorithm from Phase 1, except that, since there are no high-degree vertices in $G$, we do not need to deal with flower cluster, and all instances that we will obtain in the final decomposition will be small.

We start by applying the algorithm from Lemma 10.6 to graph $H = G$, with terminal set $T = \emptyset$, parameter $\tau = 2\mu^{1.1}$, and the parameter $m$ replaced with $3m$. Recall that the maximum vertex degree in $G$ is less than $\frac{m}{\mu^4} < \frac{3m}{\check{c}\tau^3 \log^5(3m)}$, as required.

Assume first that the algorithm from Lemma 10.6 establishes that that $\mathsf{OPT}_{\mathsf{cr}}(G) \geq \Omega\left(\frac{m^2}{\tau^4 \log^5 m}\right) \geq \Omega\left(\frac{m^2}{\mu^{5.5}}\right)$. We then terminate the algorithm and report that $\mathsf{OPT}_{\mathsf{cnwrs}}(I) \geq \Omega\left(\frac{m^2}{\mu^{5.5}}\right)$.

Therefore, we assume from now on that the algorithm from Lemma 10.6 computes a collection $\mathcal{C}'$ of disjoint clusters of $G$, such that every cluster $C \in \mathcal{C}'$ has the $\alpha'$-bandwidth property, where $\alpha' = \Omega\left(\frac{1}{\log^{1.5} m}\right)$. Since $m \geq \mu^4$, we then get that every cluster in $\mathcal{C}'$ has the $\alpha_0 = 1/\log^3 m$-bandwidth property. Additionally, we are guaranteed that, for each such cluster $C \in \mathcal{C}'$, $|E(C)| \leq m/\tau \leq m/(4\mu)$, $\bigcup_{C \in \mathcal{C}'} V(C) = V(G)$, and $|\bigcup_{C \in \mathcal{C}'} \delta_G(C)| \leq m/\tau = m/(2\mu^{1.1})$. Notice that in particular, the number of edges of $G$ with endpoints in different clusters is $|E^{\mathsf{out}}(\mathcal{C}')| \leq m/(2\mu^{1.1})$. Since we have assumed that $|E(G)| \geq m/(2\mu)$, we get that $\sum_{C \in \mathcal{C}'} |\delta_G(C)| \leq |E(G)|/\mu^{0.1}$.

We apply the algorithm from Theorem 7.1 to instance $I = (G, \Sigma)$ of $\mathsf{MCNwRS}$, and the set $\mathcal{C}'$ of clusters. Let $\tilde{\mathcal{I}}'$ be the resulting collection of subinstances of $I$ that the algorithm computes. Recall that the algorithm guarantees that $\tilde{\mathcal{I}}'$ is a $2^{O((\log m)^{3/4} \log\log m)}$-decomposition of $I$, and moreover, for each instance $\tilde{I}' = (\tilde{G}', \tilde{\Sigma}') \in \tilde{\mathcal{I}}'$, there is at most one cluster $C \in \mathcal{C}'$ with $E(C) \subseteq E(\tilde{G}')$, and all other edges of $\tilde{G}'$ lie in set $E^{\mathsf{out}}(\mathcal{C}')$. Since $|E^{\mathsf{out}}(\mathcal{C}')| \leq m/(2\mu^{1.1})$, and, for every cluster $C \in \mathcal{C}'$, $|E(C)| \leq m/(4\mu)$, we are guaranteed that, for every instance $\tilde{I}' = (\tilde{G}', \tilde{\Sigma}') \in \tilde{\mathcal{I}}'$, $|E(\tilde{G}')| \leq m/(2\mu)$. □

# 11 Constructing Internal Routers - Proof of Theorem 6.4

We will repeatedly use the following simple lemma, whose proof is provided in Appendix J.1.

**Lemma 11.1** *Let $G$ be a graph, let $T$ be a set of vertices that are $\alpha$-well-linked in $G$, for some $0 < \alpha < 1$, and let $T'$ be a subset of $T$. Suppose we are given a vertex $x \in V(G)$, and a set $\mathcal{P}$ of paths in $G$, routing the vertices of $T'$ to $x$. Then there is a set $\mathcal{P}'$ of paths routing the vertices of $T$ to $x$, such that, for every edge $e \in E(G)$, $\mathrm{cong}_G(\mathcal{P}', e) \leq \left\lceil \frac{|T|}{|T'|} \right\rceil (\mathrm{cong}_G(\mathcal{P}, e) + \lceil 1/\alpha \rceil)$.*

For conveninence, we denote the contracted graph $H_{|\mathcal{C}}$ by $\hat{H}$, and we denote $|E(\hat{H})| = \hat{m}$. From the statement of Theorem 6.4, $k \geq \hat{m}/\eta$. Observe that, from Claim 4.39, the set $T$ of terminals is $(\alpha\alpha')$-well-linked in $H$. We will assume in the remainder of the proof that $\log m$ is greater than some large enough constant $c_0'$ (whose value we can set later). If this is not the case, then, $n$, and therefore $k$, is bounded by a constant $2^{c_0'}$. We can then use an arbitrary spanning tree $\tau$ of the graph $H$, rooted at an arbitrary vertex $y$, in order to define a set $\mathcal{Q}$ of paths routing all terminals of $T$ to $y$, where for each terminal $t \in T$, the corresponding path $Q_t \in \mathcal{Q}$ is the unique path connecting $t$ to $y$ in the tree $\tau$. Since $|T|$ is bounded by a constant, for every edge $e \in E(H)$, $\mathrm{cong}_H(\mathcal{Q}, e) \leq O(1)$. We then return a distribution $\mathcal{D}$ consisting of a single set $\mathcal{Q}$ that has probability value 1. Therefore, we assume from now on that $\log m > c_0'$ for some large enough constant $c_0'$ whose value we set later.

We start with some intuition. Assume first that graph $H$ contains a grid (or a grid minor) of size $(\Omega(k\alpha\alpha'/\operatorname{poly}\log m) \times \Omega(k\alpha\alpha'/\operatorname{poly}\log m))$, and a collection $\mathcal{P}$ of paths connecting every terminal to a distinct vertex on the first row of the grid, such that the paths in $\mathcal{P}$ cause a low edge-congestion. For this special case, the algorithm of [Sid10] (see also the proof of Lemma D.10 in the full version of

[Chu11]) provides a distribution $\mathcal{D}$ over routers $\mathcal{Q} \in \Lambda(H,T)$ with the required properties. Moreover, if $H$ is a bounded-degree planar graph, with a set $T$ of terminals that is $(\alpha\alpha')$-well-linked, then there is an efficient algorithm to compute such a grid minor, together with the required collection $\mathcal{P}$ of paths. If $H$ is planar but no longer bounded-degree, we can still compute a grid-like structure in it, and apply the same arguments as in [Sid10] in order to compute the desired distribution $\mathcal{D}$. The difficulty in our case is that the input graph $H$ may be far from being planar, and, even though, from the Excluded Grid theorem of Robertson and Seymour [RS86], it must contain a large grid-like structure, without having a drawing of $H$ in the plane with a small number of crossing, we do not know how to compute such a structure[6].

The proof of Theorem 6.4 consists of five steps. In the first step, we will either establish that $\mathsf{OPT}_{\mathsf{cnwrs}}(I)$ is sufficiently large (so the algorithm can return FAIL), or compute a subgraph $\hat{H}' \subseteq \hat{H}$, and a partition $(X,Y)$ of $V(\hat{H}')$, such that each of the clusters $\hat{H}'[X], \hat{H}'[Y]$ has the $\hat{\alpha}$-bandwidth property, for $\hat{\alpha} = \Omega(\alpha/\log^4 m)$, together with a large collection of edge-disjoint paths routing the terminals to the edges of $E_{\hat{H}'}(X,Y)$ in graph $\hat{H}'$. Intuitively, we will view from this point onward the edges of $E_{\hat{H}'}(X,Y)$ as a new set of terminals, that we denote by $\tilde{T}$ (more precisely, we subdivide each edge of $E_{\hat{H}'}(X,Y)$ with a new vertex that becomes a new terminal). We show that it is sufficient to prove an analogue of Theorem 6.4 for this new set $\tilde{T}$ of terminals. The clusters $\hat{H}'[X], \hat{H}'[Y]$ of graph $\hat{H}'$ naturally define a partition $(H_1, H_2)$ of the graph $H$ into two disjoint subgraphs. In the second step, we either establish that $\mathsf{OPT}_{\mathsf{cnwrs}}(I)$ is suffciently large (so the algorithm can return FAIL), or compute some vertex $x$ of $H_1$, and a collection $\mathcal{P}$ of paths in graph $H_1$, routing the terminals of $\tilde{T}$ to $x$, such that the paths in $\mathcal{P}$ cause a relatively low edge-congestion. We exploit this set $\mathcal{P}$ of paths in order to define an ordering of the terminals in $\tilde{T}$, which is in turn exploited in the third step in order to compute a "skeleton" of the grid-like structure. We compute the grid-like structure itself in the fourth step. In the fifth and the final step, we generalize the arguments from [Sid10] and [Chu11] in order to obtain the desired distribution $\mathcal{D}$ over routers $\mathcal{Q} \in \Lambda(H,T)$, by exploiting this grid-like structure.

Before we proceed, we need to consider four simple special cases. In the first case, $\sum_{C\in\mathcal{C}} |\delta_H(C)|^2$ is large. In the second case, we can route a large subset of the terminals to a single vertex of $V(\hat{H}) \cap V(H)$ in the graph $\hat{H}$ via edge-disjont paths. The third case is when $\mathsf{OPT}_{\mathsf{cnwrs}}(H,\Sigma) = 0$, and the fourth special case is when $k < \eta^6$.

**Special Case 1: $\sum_{C\in\mathcal{C}} |\delta_H(C)|^2$ is large.** We consider the case where $\sum_{C\in\mathcal{C}} |\delta_H(C)|^2 \geq \frac{(k\alpha^4\alpha')^2}{c_0 \log^{50} m}$, where $c_0$ is the constant from the statement of Theorem 6.4. For every cluster $C \in \mathcal{C}$, let $\Sigma_C$ be the rotation system for $C$ induced by $\Sigma$. In this case, since we are guaranteed that every cluster $C \in \mathcal{C}$ is $\eta'$-bad, that is, $\mathsf{OPT}_{\mathsf{cnwrs}}(C,\Sigma_C) + |E(C)| \geq |\delta(C)|^2/\eta'$, we get that:

$$\mathsf{OPT}_{\mathsf{cnwrs}}(I) + |E(H \setminus T)| \geq \sum_{C\in\mathcal{C}} \left(\mathsf{OPT}_{\mathsf{cnwrs}}(C,\Sigma_C) + |E(C)|\right) \geq \sum_{C\in\mathcal{C}} \frac{|\delta_H(C)|^2}{\eta'} \geq \frac{(k\alpha^4\alpha')^2}{c_0\eta' \log^{50} m}.$$

Therefore, if $\sum_{C\in\mathcal{C}} |\delta_H(C)|^2 \geq \frac{(k\alpha^4\alpha')^2}{c_0 \log^{50} m}$, the algorithm returns FAIL and terminates. We assume from now on that:

$$\sum_{C\in\mathcal{C}} |\delta_H(C)|^2 < \frac{(k\alpha^4\alpha')^2}{c_0 \log^{50} m}. \tag{17}$$

[6]We note that we need the grid-like structure to have dimensions $(k' \times k')$, where $k'$ is almost linear in $k$. Therefore, we cannot use the known bounds for the Excluded Minor Theorem (e.g. from [CT19]) for general graphs, and instead we need to use an analogue of the stronger version of the theorem for planar graphs.

**Special Case 2: Routing of terminals to a single vertex.** The second special case happens if there exists a collection $\mathcal{P}_0$ of at least $\frac{k\alpha^2}{1024c_{\mathrm{CMG}}^3\log^6 k}$ edge-disjoint paths in graph $\hat{H}$ routing some subset $T_0\subseteq T$ of terminals to some vertex $x$ (here $c_{\mathrm{CMG}}$ is the constant from Claim 4.23).

Note that, if Special Case 1 did not happen, and $c_0$ is a large enough constant, then $x$ may not be a supernode. Indeed, assume that $x=v_C$ for some cluster $C\in\mathcal{C}$. Then:

$$|\delta_H(C)|^2\geq\Omega\left(\frac{(k\alpha^2)^2}{\log^{12}k}\right)\geq\frac{(k\alpha^4\alpha')^2}{c_0\log^{50}m},$$

which is, assuming that $c_0$ is a large enough constant, a contradiction. Therefore, we can assume that $x$ is not a supernode. From Claim 4.41, since the clusters in $\mathcal{C}$ have the $\alpha'$-bandwidth property, there is a collection $\mathcal{P}_0'$ of paths in graph $H$, routing the vertices of $T_0$ to $x$, with edge-congestion at most $\lceil 1/\alpha'\rceil\leq 2/\alpha'$. Since the set $T$ of terminals is $(\alpha\alpha')$-well-linked in graph $H$, from Lemma 11.1, there is a set $\mathcal{Q}$ of paths in graph $H$, routing the vertices of $T$ to $x$ with congestion at most:

$$\left\lceil\frac{|T|}{|T_0|}\right\rceil\left(\frac{2}{\alpha'}+\left\lceil\frac{1}{\alpha\alpha'}\right\rceil\right)\leq O\left(\frac{\log^6 k}{\alpha^3\alpha'}\right).$$

Note that a set $\mathcal{Q}$ of paths with the above properties can be computed efficiently via standard maximum flow algorithm. We return a distribution $\mathcal{D}$ consisting of a single router $\mathcal{Q}$ with probability value 1, and terminate the algorithm. Clearly, for every edge $e\in E(H)$, $\mathbf{E}\left[(\mathrm{cong}(\mathcal{Q},e))^2\right]\leq O\left(\frac{\log^{32}m}{\alpha^{12}(\alpha')^8}\right)$.

**Special Case 3: $\mathsf{OPT}_{\mathsf{cnwrs}}(I)=0$.** Recall that we can efficiently check whether $\mathsf{OPT}_{\mathsf{cnwrs}}(I)=0$, using the algorithm from Theorem 2.7. Assume now that $\mathsf{OPT}_{\mathsf{cnwrs}}(H,\Sigma)=0$. We use the following theorem from [CMT20].

**Lemma 11.2 (Lemma E.2 in [CMT20])** *There is an efficient algorithm, that, given a planar graph $H$ and a subset $T$ of $r$ vertices of $V(H)$ that are $\alpha$-well-linked in $H$ for some $0<\alpha<1$, computes a distribution $\mathcal{D}$ over the routers in $\Lambda(H,T)$, such that the distribution has support size $O(r^2)$, and for each edge $e\in E(H)$,*

$$\mathbf{E}_{(u^*,\mathcal{Q})\sim\mathcal{D}}\left[(\mathrm{cong}_H(\mathcal{Q},e))^2\right]=O\left(\frac{\log r}{\alpha^4}\right).$$

Recall that the set $T$ of terminals is $\alpha$-well-linked in the contracted graph $H_{|\mathcal{C}}$, and every cluster $C\in\mathcal{C}$ has the $\alpha'$-bandwidth property. From Claim 4.39, the set $T$ of terminals is $(\alpha\alpha')$-well-linked in $H$. We then apply the algorithm from Lemma 11.2 to graph $H$, terminal set $T$ and parameter $(\alpha\alpha')$. Let $\mathcal{D}$ be the distribution over the set $\Lambda(H,T)$ of routers that we obtain. Then:

$$\mathbf{E}_{\mathcal{Q}\sim\mathcal{D}}\left[(\mathrm{cong}_H(\mathcal{Q},e))^2\right]=O\left(\frac{\log k}{(\alpha\alpha')^4}\right)\leq O\left(\frac{\log^{32}m}{\alpha^{12}(\alpha')^8}\right).$$

**Special Case 4: $k<\eta^6$, but $\mathsf{OPT}_{\mathsf{cnwrs}}(I)>0$.** Note that $\frac{(k\alpha^4\alpha')^2}{c_0\eta'\log^{50}m}\leq\frac{\eta^{12}}{\eta'}<1$ in this case (as, from the statement of Theorem 6.4, $\eta'>\eta^{13}$). Since we have assumed that $\mathsf{OPT}_{\mathsf{cnwrs}}(I)>0$, we get that $\mathsf{OPT}_{\mathsf{cnwrs}}(I)\geq 1>\frac{(k\alpha^4\alpha')^2}{c_0\eta'\log^{50}m}$. We then simply return FAIL and terminate the algorithm.

In the remainder of the proof, we assume that neither of the four special cases happened. We now describe each step of the algorithm in detail.

### 11.1 Step 1: Splitting the Contracted Graph

In this step, we split the contracted graph $\hat{H}$, using the algorithm summarized in the following theorem.

**Theorem 11.3** *There is an efficient randomized algorithm that returns FAIL with probability at most $1/\operatorname{poly}(k)$, and, if it does not return FAIL, then it computes a subgraph $\hat{H}' \subseteq \hat{H}$ and a partition $(X, Y)$ of $V(\hat{H}')$ such that:*

- *clusters $\hat{H}'[X]$ and $\hat{H}'[Y]$ both have the $\hat{\alpha}'$-bandwidth property in $\hat{H}'$, for $\hat{\alpha}' = \Omega(\alpha/\log^4 m)$; and*
- *there is a set $\mathcal{R}$ of $\Omega(\alpha^3 k/\log^8 m)$ edge-disjoint paths in graph $\hat{H}'$, routing a subset of terminals to edges of $E_{\hat{H}'}(X, Y)$.*

**Proof:** We start by applying the algorithm from Claim 4.23 to graph $\hat{H}$ and the set $T$ of terminals, to obtain a graph $W$ with $V(W) = T$ and maximum vertex degree at most $c_{\mathrm{CMG}} \log^2 k$, and an embedding $\hat{\mathcal{P}}$ of $W$ into $\hat{H}$ with congestion at most $(c_{\mathrm{CMG}} \log^2 k)/\alpha$. Let $\hat{\mathcal{E}}$ be the bad event that $W$ is not a 1/4-expander. Then $\mathbf{Pr}\left[\hat{\mathcal{E}}\right] \leq 1/\operatorname{poly}(k)$. Define graph $\hat{H}'$ as the union of all paths in $\hat{\mathcal{P}}$. We need the following observation.

**Observation 11.4** *If event $\hat{\mathcal{E}}$ did not happen, then the set $T$ of vertices is $\hat{\alpha}$-well-linked in $\hat{H}'$, for $\hat{\alpha} = \frac{\alpha}{4c_{\mathrm{CMG}} \log^2 k}$, and the maximum vertex degree in $\hat{H}'$ is at most $d = \frac{\alpha k}{512 c_{\mathrm{CMG}} \log^2 k}$.*

**Proof:** Assume that Event $\hat{\mathcal{E}}$ did not happen. We first prove that the set $T$ of terminals is $\hat{\alpha}$-well-linked in $\hat{H}$. Consider any paritition $(A, B)$ of vertices of $\hat{H}'$, and denote $T_A = T \cap A$, $T_B = T \cap B$. Assume w.l.o.g. that $|T_A| \leq |T_B|$. Then it is sufficient to show that $|E_{\hat{H}'}(A, B)| \geq \hat{\alpha} \cdot |T_A|$.

Consider the partition $(T_A, T_B)$ of the vertices of $W$, and denote $E' = E_W(T_A, T_B)$. Since $W$ is a 1/4-expander, $|E'| \geq |T_A|/4$ must hold. Consider now the set $\hat{\mathcal{R}} \subseteq \hat{\mathcal{P}}$ of paths containing the embeddings $P(e)$ of every edge $e \in E'$. Each path $R \in \hat{\mathcal{R}}$ connects a vertex of $T_A$ to a vertex of $T_B$, so it must contain an edge of $|E_{\hat{H}}(A, B)|$. Since $|\hat{\mathcal{R}}| \geq |T_A|/4$, and the paths in $\hat{\mathcal{P}}$ cause edge-congestion at most $(c_{\mathrm{CMG}} \log^2 k)/\alpha$, we get that $|E_{\hat{H}}(A, B)| \geq \alpha \cdot |T_A|/(4c_{\mathrm{CMG}} \log^2 k) \geq \hat{\alpha}|T_A|$.

Assume now that maximum vertex degree in $\hat{H}'$ is greater than $d$, and let $x$ be a vertex whose degree is at least $d$. Let $\hat{\mathcal{Q}} \subseteq \hat{\mathcal{P}}$ be the set of all paths containing the vertex $x$. Consider any such path $Q \in \hat{\mathcal{Q}}$. The endpoints of this path are two distinct terminals $t, t' \in T$. We let $Q' \subseteq Q$ be the subpath of $Q$ between the terminal $t$ and the vertex $x$, and we let $\mathcal{Q}' = \left\{Q' \mid Q \in \hat{\mathcal{Q}}\right\}$.

Recall that every vertex in $W$ has degree at most $c_{\mathrm{CMG}} \log^2 k$, and so a terminal in $T$ may be an endpoint of at most $c_{\mathrm{CMG}} \log^2 k$ paths in $\hat{\mathcal{P}}$. Therefore, there is a subset $\mathcal{Q}'' \subseteq \mathcal{Q}'$ of at least $d/(2c_{\mathrm{CMG}} \log^2 k)$ paths in $\hat{H}'$, each of which originates at a distinct terminal. Since paths in $\mathcal{Q}''$ cause congestion at most $(c_{\mathrm{CMG}} \log^2 k)/\alpha$, from Claim 4.2, there is a collection $\mathcal{Q}'''$ of edge-disjoint paths in graph $\hat{H}'$, routing a subset of terminals to $x$ with:

$$|\mathcal{Q}'''| \geq |\mathcal{Q}''| \cdot \frac{\alpha}{c_{\mathrm{CMG}} \log^2 k} \geq \frac{d\alpha}{2c_{\mathrm{CMG}}^2 \log^4 k} \geq \frac{\alpha^2 k}{1024 c_{\mathrm{CMG}}^3 \log^6 k},$$

contradicting the fact that Special Case 2 did not happen. □

Next, we use the following lemma to compute the required sets $X$, $Y$ of vertices. The proof follows immediately from techniques that were introduced in [Chu12] and then refined in [CL12, CC16, Chu16]. Unfortunately, all these proofs assumed that the input graph has a bounded maximum vertex degree, and additionally the proofs are somewhat more involved than the proof that we need here (this is because these proofs could only afford a poly $\log k$ loss in the cardinality of the set $\mathcal{R}$ of paths relatively to $|T|$, while we can afford a poly $\log m$ loss). Therefore, we provide a proof of the lemma in Section J.2 of the Appendix for completeness.

**Lemma 11.5** *There is an efficient algorithm that, given as input an $m$-edge graph $G$, and a subset $T$ of $k$ vertices of $G$ called terminals, together with a parameter $0 < \tilde{\alpha} < 1$, such that the maximum vertex degree in $G$ is at most $\tilde{\alpha}k/64$, and every vertex of $T$ has degree 1 in $G$, either returns FAIL, or computes a partition $(X, Y)$ of $V(G)$, such that:*

- *each of the clusters $G[X]$, $G[Y]$ has the $\tilde{\alpha}'$-bandwidth property, for $\tilde{\alpha}' = \Omega(\tilde{\alpha}/\log^2 m)$; and*
- *there is a set $\mathcal{R}$ of at least $\Omega(\tilde{\alpha}^3 k/\log^2 m)$ edge-disjoint paths in graph $G$, routing a subset of terminals to edges of $E_G(X, Y)$.*

*Moreover, if the set $T$ of vertices is $\tilde{\alpha}$-well-linked in $G$, then the algorithm never returns FAIL.*

We apply the algorithm from Lemma 11.5 to graph $\hat{H}'$, the set $T$ of terminals, and parameter $\tilde{\alpha} = \hat{\alpha} = \frac{\alpha}{4c_{\mathrm{CMG}}\log^2 k}$. Recall that we are guaranteed that the maximum vertex degree in graph $\hat{H}'$ is at most $d = \frac{\alpha k}{512 c_{\mathrm{CMG}}\log^2 k} \leq \frac{\tilde{\alpha}k}{64}$. Note that the algorithm from Lemma 11.5 may only return FAIL if the set $T$ of terminals is not $\tilde{\alpha}$-well-linked in $\hat{H}'$, which, from Observation 11.4, may only happen if event $\hat{\mathcal{E}}$ happened, which in turn may only happen with probability $1/\operatorname{poly}(k)$. If the algorithm from Lemma 11.5 returned FAIL, then we terminate the algorithm and return FAIL as well. Therefore, we assume from now on that the algorithm from Lemma 11.5 did not return FAIL. Let $(X, Y)$ be the partition of $V(\hat{H}')$ that the algorithm returns. We are then guaranteed that each of the clusters $\hat{H}'[X], \hat{H}'[Y]$ has the $\tilde{\alpha}'$-bandwidth property in $\tilde{H}$, where $\tilde{\alpha}' = \Omega(\tilde{\alpha}/\log^2 m) = \Omega(\alpha/\log^4 m)$. The algorithm also ensures that there is a collection $\mathcal{R}$ of edge-disjoint paths in $\hat{H}'$, routing a subset of the terminals to edges of $E_{\hat{H}'}(X, Y)$, with $|\mathcal{R}| \geq \Omega(\tilde{\alpha}^3 k/\log^2 m) \geq \Omega(\alpha^3 k/\log^8 m)$. This completes the proof of Theorem 11.3. □

If the algorithm from Theorem 11.3 returned FAIL (which may only happen with probability at most $1/\operatorname{poly}(k)$), then we terminate the algorithm and return FAIL as well. Therefore, we assume from now on that the algorithm from Theorem 11.3 returned a subgraph $\hat{H}' \subseteq \hat{H}$ and a partition $(X, Y)$ of $V(\hat{H}')$ such that each of the clusters $\hat{H}'[X]$ and $\hat{H}'[Y]$ has the $\hat{\alpha}'$-bandwidth property, for $\hat{\alpha}' = \Theta(\alpha/\log^4 m)$, and there is a set $\mathcal{R}$ of $\Omega(\alpha^3 k/\log^8 m)$ edge-disjoint paths in graph $\hat{H}'$, routing a subset of terminals to edges of $E_{\hat{H}'}(X, Y)$. Notice that we can compute the path set $\mathcal{R}$ with the above properties efficiently, using standard maximum flow algorithms. We assume w.l.o.g. that edges of $E_{\hat{H}}(X, Y)$ do not serve as inner edges on paths in $\mathcal{R}$. Let $E' \subseteq E_{\hat{H}'}(X, Y)$ be the subset of edges containing the last edge on every path in $\mathcal{R}$, so, by reversing the direction of the paths in $\mathcal{R}$, we can view the set $\mathcal{R}$ of paths as routing the edges of $E'$ to the terminals. In the remainder of this step, we will slightly modify the graphs $H$ and $\hat{H}$, and we will continue working with the modified graphs only in the following steps.

Let $\hat{H}'' \subseteq \hat{H}'$ be the graph obtained from $\hat{H}'$ by first deleting all edges of $E_{\hat{H}'}(X, Y) \setminus E'$ from it, and then subdividing every edge $e \in E'$ with a vertex $t_e$. We denote $\tilde{T} = \{t_e \mid e \in E'\}$, and we refer to vertices of $\tilde{T}$ as *pseudo-terminals*. Recall that $|\tilde{T}| = |\mathcal{R}| = \Omega(\alpha^3 k/\log^8 m)$, and there is a set $\mathcal{R}'$ of edge-disjoint paths in the resulting graph $\hat{H}''$, routing the vertices of $\tilde{T}$ to the vertices of $T$. We define $\hat{H}_1 = \hat{H}''[X \cup \tilde{T}]$, the subgraph of $\hat{H}''$ induced by the set $X \cup \tilde{T}$ of vertices, and we define $\hat{H}_2 = \hat{H}''[Y \cup \tilde{T}]$ similarly. From the $\hat{\alpha}'$-bandwidth property of the clusters $\hat{H}'[X]$ and $\hat{H}'[Y]$ in $\hat{H}'$, we are guaranteed that the vertices of $\tilde{T}$ are $\hat{\alpha}'$-well-linked in both $\hat{H}_1$ and in $\hat{H}_2$, where $\hat{\alpha}' = \Theta(\alpha/\log^4 m)$. Let $\mathcal{C}' \subseteq \mathcal{C}$ be the subset of all clusters $C$ whose corresponding supernode $v_C$ lies in graph $\hat{H}''$.

For convenience, we also subdivide, in graph $H$, every edge $e \in E'$, with the vertex $t_e$, so graph $\hat{H}''$ can be now viewed as a subgraph of the contracted graph $H_{|\mathcal{C}}$.

Next, we let $H' \subseteq H$ be the subgraph of $H$ that corresponds to graph $\hat{H}''$. In other words, graph $H'$ is obtained from $\hat{H}''$ by replacing every supernode $v_C$ with the corresponding cluster $C \in \mathcal{C}'$. Equivalently, we can obtain graph $H'$ from $H$, by deleting every edge of $E(\hat{H}) \setminus E(\hat{H}'')$ and every

regular (non-supernode) vertex of $V(\hat{H}) \setminus V(\hat{H}'')$. Additionally, for every cluster $C \in \mathcal{C} \setminus \mathcal{C}'$, we delete all edges and vertices of $C$ from $H'$. We also define a rotation system $\Sigma'$ for graph $H'$, which is naturally induced by $\Sigma$ (vertices $t_e \in \tilde{T}$ all have degree 2, so their corresponding ordering $\mathcal{O}_{t_e}$ of incident edges can be set arbitrarily). Let $I' = (H', \Sigma')$ be the resulting instance of MCNwRS.

We partition the set $\mathcal{C}'$ of clusters into two subsets: set $\mathcal{C}_X$ contains all clusters $C \in \mathcal{C}'$ with $v_C \in X$, and set $\mathcal{C}_Y$ contains all clusters $C \in \mathcal{C}'$ with $v_C \in Y$. We can similarly define the graphs $H_1, H_2 \subseteq H'$, that correspond to the contracted graphs $\hat{H}_1$ and $\hat{H}_2$, respectively: let $X'$ be the set of vertices of $H'$, containing every vertex $x \in V(H')$, such that either $x \in C$ for some cluster $C \in \mathcal{C}_X$, or $x$ is a regular vertex of $\hat{H}''$ lying in $X$. Similarly, we let $Y'$ contain all vertices $y \in V(H)$, such that either $y \in C$ for some cluster $C \in \mathcal{C}_Y$, or $y$ is a regular vertex of $\hat{H}''$ lying in $Y$. We then let $H_1 = H'[X \cup \tilde{T}]$, and $H_2 = H'[Y \cup \tilde{T}]$.

The following observation, summarizing properties of instance $I'$, is immediate.

**Observation 11.6** *Instance $I' = (H', \Sigma')$ of* MCNwRS *satisties the following properties:*

- $\mathsf{OPT}_{\mathsf{cnwrs}}(I') \leq \mathsf{OPT}_{\mathsf{cnwrs}}(I)$*;*
- $\hat{H}'' = H'_{|\mathcal{C}'}$*;*
- $|E(\hat{H}'')| \leq 2|E(\hat{H})| \leq 2\eta k \leq O(|\tilde{T}|\eta \log^8 m/\alpha^3)$*; and*
- *graph $\hat{H}_1$ is a contracted graph of $H_1$ with respect to $\mathcal{C}_X$, and graph $\hat{H}_2$ is a contracted graph of $H_2$ with respect to $\mathcal{C}_Y$. In other words, $\hat{H}_1 = (H_1)_{|\mathcal{C}_X}$, and $\hat{H}_2 = (H_2)_{|\mathcal{C}_Y}$.*

For the third assertion we have used the fact that $k \geq |E(\hat{H})|/\eta$ from the statement of Theorem 6.4, and $|\tilde{T}| \geq \Omega(\alpha^3 k/\log^8 m)$.

Recall that $\Lambda(H', \tilde{T})$ denotes the set of all routers in graph $H'$, with respect to the set $\tilde{T}$ of terminals. Each such router $\mathcal{Q}$ is a set of paths, routing the vertices of $\tilde{T}$ to some vertex of $H'$. Intuitively, from now on we would like to work with instance $I' = (H', \Sigma')$ of MCNwRS, and the new set $\tilde{T}$ of terminals. To this end, we start by showing that, in order to obtain the desired distribution $\mathcal{D}$ over the routers of $\Lambda(H, T)$, it is now sufficient to compute a distribution $\mathcal{D}'$ over the routers of $\Lambda(H', \tilde{T})$, such that for every edge $e \in E(H')$, $\mathbf{E}_{\mathcal{Q}' \sim \mathcal{D}'}\left[(\mathrm{cong}_{H'}(\mathcal{Q}, e))^2\right]$ is low.

**Observation 11.7** *There is an efficient algorithm, that, given an explicit distribution $\mathcal{D}'$ over the routers of $\Lambda(H', \tilde{T})$, such that for every edge $e' \in E(H')$, $\mathbf{E}_{\mathcal{Q}' \sim \mathcal{D}'}\left[(\mathrm{cong}_{H'}(\mathcal{Q}', e'))^2\right] \leq \beta$ holds, computes an explicit distribution $\mathcal{D}$ over the routers of $\Lambda(H, T)$, such that for every edge $e \in E(H)$, $\mathbf{E}_{\mathcal{Q} \sim \mathcal{D}}\left[(\mathrm{cong}_{H}(\mathcal{Q}, e))^2\right] \leq O\left(\frac{\beta \log^{16} m}{\alpha^8 (\alpha')^4}\right)$.*

**Proof:** Recall that $H' \subseteq H$. Consider some router $\mathcal{Q}' \in \Lambda(H', \tilde{T})$, whose probability value in distribution $\mathcal{D}'$ is $p(\mathcal{Q}') > 0$. We compute a router $\mathcal{Q} \in \Lambda(H, T)$ corresponding to $\mathcal{Q}'$, and we assign to $\mathcal{Q}$ the same probability value $p(\mathcal{Q}')$.

We now show an algorithm for computing a router $\mathcal{Q} \in \Lambda(H, T)$ from a router $\mathcal{Q}' \in \Lambda(H', \tilde{T})$. We denote by $x'$ the vertex that serves as the center of the router $\mathcal{Q}'$. Recall that there is a set $\mathcal{R}'$ of edge-disjoint paths in graph $\hat{H}''$, routing the vertices of $\tilde{T}$ to the vertices of $T$, and moreover, a set of paths with these properties can be found efficiently via a standard maximum $s$-$t$ flow computation. Since $\hat{H}'' = H'_{|\mathcal{C}'}$, and every cluster in $\mathcal{C}'$ has the $\alpha'$-bandwidth property in $H'$, from Claim 4.41, we can efficiently compute a set $\mathcal{R}_0$ of edge-disjoint paths in graph $H'$, routing a subset $T_0 \subseteq T$ of terminals to $\tilde{T}$, with $|\mathcal{R}_0| \geq \alpha' \cdot |\mathcal{R}'|/2 = \alpha' \cdot |\tilde{T}|/2 = \Omega(\alpha' \alpha^3 k/\log^8 m)$. By concatenating the paths in $\mathcal{R}_0$ and the paths in $\mathcal{Q}'$, we obtain a collection $\mathcal{R}'_0$ of paths in graph $H'$, routing the terminals of $T_0$ to vertex $x'$, such that for every edge $e \in E(H)$, $\mathrm{cong}_{H'}(\mathcal{R}'_0, e) \leq \mathrm{cong}_{H'}(\mathcal{Q}', e) + 1$. Since the set $T$ of terminals

is $(\alpha\alpha')$-well-linked in graph $H$ (from Claim 4.39), from Lemma 11.1, there exists a collection $\mathcal{Q}$ of paths in graph $H$, routing the terminals in $T$ to vertex $x'$, such that for every edge $e \in E(H)$:

$$\mathrm{cong}_H(\mathcal{Q}, e) \leq \left\lceil \frac{|T|}{|T_0|} \right\rceil \left( \mathrm{cong}_H(\mathcal{R}'_0, e) + \left\lceil \frac{1}{\alpha\alpha'} \right\rceil \right) \leq O\left( \frac{\log^8 m}{\alpha'\alpha^3} \right) \cdot \left( \mathrm{cong}_{H'}(\mathcal{Q}', e) + \frac{2}{\alpha\alpha'} \right).$$

A set $\mathcal{Q}$ of paths with these properties can be computed efficiently via standard maximum $s$-$t$ flow algorithms.

For every router $\mathcal{Q}' \in \Lambda(H', \tilde{T})$, whose probability value in distribution $\mathcal{D}'$ is $p(\mathcal{Q}') > 0$, we have computed a corresponding router $\mathcal{Q} \in \Lambda(H, T)$, and we have assigned to it the same probability value $p(\mathcal{Q}) = p(\mathcal{Q}')$. This completes the definition of the distribution $\mathcal{D}$ over the routers in $\Lambda(H, T)$. From the above discussion, for every edge $e \in E(H)$,

$$\mathbf{E}_{\mathcal{Q}\sim\mathcal{D}}\left[ (\mathrm{cong}_H(\mathcal{Q}, e))^2 \right] \leq O\left( \frac{\log^{16} k}{\alpha^8(\alpha')^4} \right) \cdot \left( \mathbf{E}_{\mathcal{Q}'\sim\mathcal{D}'}\left[ (\mathrm{cong}_{H'}(\mathcal{Q}', e))^2 \right] + 1 \right) \leq O\left( \frac{\beta \log^{16} m}{\alpha^8(\alpha')^4} \right).$$

□

The following immediate corollary is obtained by plugging in the bounds required by Theorem 6.4 into Observation 11.7.

**Corollary 11.8** *There is an efficient algorithm, that, given an explicit distribution $\mathcal{D}'$ over the routers of $\Lambda(H', \tilde{T})$, such that for every edge $e' \in E(H')$, $\mathbf{E}_{\mathcal{Q}'\sim\mathcal{D}'}\left[ (\mathrm{cong}_{H'}(\mathcal{Q}', e'))^2 \right] \leq O\left( \frac{\log^{16} m}{(\alpha\alpha')^4} \right)$ holds, produces an explicit distribution $\mathcal{D}$ over the routers of $\Lambda(H, T)$, such that, for every edge $e \in E(H)$:*

$$\mathbf{E}_{\mathcal{Q}\sim\mathcal{D}}\left[ (\mathrm{cong}_H(\mathcal{Q}, e))^2 \right] \leq O\left( \frac{\log^{32} m}{\alpha^{12}(\alpha')^8} \right).$$

Denote $\tilde{k} = |\tilde{T}|$. Recall that $\tilde{k} \geq \Omega\left( \frac{\alpha^3 k}{\log^8 m} \right)$, so $\frac{(k\alpha^4\alpha')^2}{c_0 \log^{50} m} \leq O\left( \frac{(\tilde{k}\hat{\alpha}'\alpha')^2}{c_0 \log^{20} m} \right)$. We use a large enough constant $c_1$, whose value will be set later, and we set $c_0 = c_1^2$. We can then assume that $\frac{(k\alpha^4\alpha')^2}{c_0 \log^{50} m} \leq \frac{(\tilde{k}\hat{\alpha}'\alpha')^2}{c_1 \log^{20} m}$. In particular, from Equation 17, we get that $\sum_{C\in\mathcal{C}} |\delta_{H'}(C)|^2 < \frac{(k\alpha^4\alpha')^2}{c_0 \log^{50} m} \leq \frac{(\tilde{k}\hat{\alpha}'\alpha')^2}{c_1 \log^{20} m}$. Additionally, if $|E(H' \setminus \tilde{T})| + \mathsf{OPT}_{\mathsf{cnwrs}}(I') > \frac{(\tilde{k}\hat{\alpha}'\alpha')^2}{c_1\eta' \log^{20} m}$, then $|E(H \setminus T)| + \mathsf{OPT}_{\mathsf{cnwrs}}(I) > \frac{(k\alpha^4\alpha')^2}{c_0\eta' \log^{50} m}$.

In order to complete the proof of Theorem 6.4, it is now enough to design a randomized algorithm, that either returns FAIL, or computes a distribution $\mathcal{D}'$ over the routers in $\Lambda(H', \tilde{T})$, such that, for every edge $e \in E(H')$, $\mathbf{E}_{\mathcal{Q}'\sim\mathcal{D}'}\left[ (\mathrm{cong}_{H'}(\mathcal{Q}', e))^2 \right] \leq O\left( \frac{\log^{16} m}{(\alpha\alpha')^4} \right)$. It is enough to ensure that, if $|E(H' \setminus \tilde{T})| + \mathsf{OPT}_{\mathsf{cnwrs}}(H', \Sigma') \leq \frac{(\tilde{k}\hat{\alpha}'\alpha')^2}{c_1\eta' \log^{20} m}$, then the probability that the algorithm returns FAIL is at most $1/2$.

In the remainder of the proof we focus on the above goal. It would be convenient for us to simplify the notation, by denoting $H'$ by $H$, $\Sigma'$ by $\Sigma$, $I'$ by $I$, $\hat{H}''$ by $\hat{H}$, and $\hat{\alpha}'$ by $\tilde{\alpha}$. We also denote $\mathcal{C}'$ by $\mathcal{C}$. We now summarize all properties of the new graphs $H, \hat{H}$ that we have established so far, and in the remainder of the proof of Theorem 6.4 we will only work with these new graphs.

**Summary of the Outcome of Step 1.** We assume from now on that we are given an instance $I = (H, \Sigma)$ of $\mathsf{MCNwRS}$, a set $\tilde{T}$ of terminals in graph $H$, and a collection $\mathcal{C}$ of disjoint subgraphs (clusters) of $H \setminus \tilde{T}$. We denote $|\tilde{T}| = \tilde{k}$. The corresponding contracted graph is denoted by $\hat{H} = H_{|\mathcal{C}}$. We are also given a partition $(X, Y)$ of $V(H) \setminus \tilde{T}$ (note that for convenience of notation, $X$ and $Y$ are now subsets of vertices of $H$, and not of $\hat{H}$), and a parition $\mathcal{C}_X, \mathcal{C}_Y$ of $\mathcal{C}$, such that each cluster $C \in \mathcal{C}_X$ has $V(C) \subseteq X$, and each cluster $C \in \mathcal{C}_Y$ has $V(C) \subseteq Y$. We denote $H_1 = H[X \cup \tilde{T}]$ and

$H_2 = H[Y \cup \tilde{T}]$. We also denote by $\hat{H}_1 = (H_1)_{|\mathcal{C}_X}$ the contracted graph of $H_1$ with respect to $\mathcal{C}_X$, and similarly by $\hat{H}_2 = (H_2)_{|\mathcal{C}_Y}$.

We now summarize the properties of the graphs that we have defined and the relationships between the main parameters.

P1. $\tilde{k} \geq \Omega(\alpha^3 k/\log^8 m)$;

P2. every cluster $C \in \mathcal{C}$ has the $\alpha'$-bandwidth property in $H$;

P3. $|E(\hat{H})| \leq O\left(\frac{\tilde{k} \cdot \eta \log^8 m}{\alpha^3}\right)$ (from Observation 11.6);

P4. every vertex of $\tilde{T}$ has degree 1 in $H_1$, and vertex set $\tilde{T}$ is $\tilde{\alpha}$-well-linked in $\hat{H}_1$, for $\tilde{\alpha} = \Theta(\alpha/\log^4 m)$;

P5. similarly, every vertex of $\tilde{T}$ has degree 1 in $H_2$, and vertex set $\tilde{T}$ is $\tilde{\alpha}$-well-linked in $\hat{H}_2$; and

P6. $\sum_{C \in \mathcal{C}} |\delta_H(C)|^2 < \frac{(\tilde{k}\tilde{\alpha}\alpha')^2}{c_1 \log^{20} m}$, where $c_1$ is some large enough constant, whose value we can set later.

Our goal is to design an efficient randomized algorithm, that either returns FAIL, or computes a distribution $\mathcal{D}$ over the routers in $\Lambda(H, \tilde{T})$, such that, for every edge $e \in E(H)$, $\mathbf{E}_{\mathcal{Q} \sim \mathcal{D}}\left[(\mathrm{cong}_H(\mathcal{Q}, e))^2\right] \leq O\left(\frac{\log^{16} m}{(\alpha\alpha')^4}\right)$. It is enough to ensure that, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I) + |E(H \setminus T)| < \frac{(\tilde{k}\tilde{\alpha}\alpha')^2}{c_1 \eta' \log^{20} m}$, then the probability that the algorithm returns FAIL is at most 1/2.

## 11.2 Step 2: Routing the Terminals to a Single Vertex, and an Expanded Graph

In this step we start by considering the graph $\hat{H}_1$ and the set $\tilde{T}$ of terminals in it. Our goal is to compute a collection $\mathcal{J}$ of paths in graph $\hat{H}_1$, routing all terminals of $\tilde{T}$ to a single regular vertex, such that the paths in $\mathcal{J}$ cause a relatively low congestion in graph $\hat{H}_1$. We show that, if such a collection of paths does not exist, then $\mathsf{OPT}_{\mathsf{cnwrs}}(I)$ is high. Intuitively, we will use the set $\mathcal{J}$ of paths in order to define an ordering of the terminals in $\tilde{T}$, which will in turn be used in order to compute a grid-like structure in graph $H_2$. Once we compute the desired set $\mathcal{J}$ of paths, we will replace the graph $H$ with its low-degree analogue $H^*$, that we refer to as the *expanded graph*. The remaining steps in the proof of Theorem 6.4 will use this expanded graph only.

### 11.2.1 Routing the Terminals to a Single Vertex in $\hat{H}_1$

We process **regular** vertices of $V(\hat{H}_1)$ (that is, vertices of $V(\hat{H}_1) \cap V(H_1)$) one by one. For each such vertex $x$, we compute a set $\mathcal{J}(x)$ of paths in graph $\hat{H}_1$, with the following properties:

- every path in $\mathcal{J}(x)$ originates at a distinct vertex of $\tilde{T}$ and terminates at $x$;
- the paths in $\mathcal{J}(x)$ are edge-disjoint; and
- $\mathcal{J}(x)$ is a maximum-cardinality set of paths in $\hat{H}_1$ with the above two properties.

Note that such a set $\mathcal{J}(x)$ of paths can be computed via a standard maximum $s$-$t$ flow computation. Throughout, we use a parameter $\tilde{k}' = \tilde{k}\alpha^5/(c'\eta \log^{36} m)$, where $c'$ is a large enough constant whose value we set later.

If, for every vertex $x \in V(H_1)$, $|\mathcal{J}(x)| < \tilde{k}'$, then we reurn FAIL and terminate the algorithm. In the following lemma, whose proof is deferred to Section J.3 of Appendix we show that, in this case,

$\mathsf{OPT}_{\mathsf{cnwrs}}(I) \geq \Omega\left(\frac{(\tilde{k}\tilde{\alpha}\alpha')^2}{\eta' \log^{20} m}\right)$ must hold. Note that, since we can set $c_1$ to be a large enough constant, we can ensure that $\mathsf{OPT}_{\mathsf{cnwrs}}(I) > \frac{(\tilde{k}\tilde{\alpha}\alpha')^2}{c_1\eta' \log^{20} m}$ holds in this case. The value of the constant $c'$ that is used in the definition of the parameter $\tilde{k}'$ is set in the proof of the lemma.

**Lemma 11.9** *If, for every vertex $x \in V(\hat{H}_1) \cap V(H_1)$, $|\mathcal{J}(x)| < \tilde{k}'$, then $\mathsf{OPT}_{\mathsf{cnwrs}}(I) \geq \Omega\left(\frac{(\tilde{k}\tilde{\alpha}\alpha')^2}{\eta' \log^{20} m}\right)$.*

From now on we assume that there is some vertex $x \in V(\hat{H}_1) \cap V(H_1)$, for which $|\mathcal{J}(x)| \geq \tilde{k}'$.

### 11.2.2 The Expanded Graph

From now on we fix the vertex $x \in V(\hat{H}_1) \cap V(H_1)$, and we let $\mathcal{J} = \mathcal{J}(x)$ be a set of at least $\tilde{k}'$ edge-disjoint paths in graph $\hat{H}_1$, routing a subset $\tilde{T}_0 \subseteq \tilde{T}$ of terminals to vertex $x$.

We are now ready to define the expanded graph $H^*$. We start with graph $H^*$ being empty, and then process every vertex $u \in V(H_2) \setminus \tilde{T}$ one by one. We now describe an iteration when a vertex $u \in V(H_2) \setminus \tilde{T}$ is processed. We denote by $d(u)$ the degree of the vertex $u$ in graph $H_2$. Let $e_1(u), \ldots, e_{d(u)}(u)$ be the edges that are incident to $u$ in $H_2$, indexed according to their ordering in $\mathcal{O}_u \in \Sigma$. We let $\Pi(u)$ be a $(d(u) \times d(u))$ grid, and we denote the vertices on the first row of this grid by $s_1(u), \ldots, s_{d(u)}(u)$ indexed in their natural left-to-right order. We add the vertices and the edges of the grid $\Pi(u)$ to graph $H^*$. We refer to the edges in the resulting grids $\Pi(u)$ as *inner edges*. Once every vertex $u \in V(H_2) \setminus \tilde{T}$ is processed, we add the vertices of $\tilde{T}$ to the graph $H^*$. Recall that every terminal $t \in \tilde{T}$ has degree 1 in $H_2$. We denote the unique edge $e_t$ incident to $t$ by $e_1(t)$, and we denote $s_1(t) = t$.

Next, we add a collection of *outer edges* to graph $H^*$, as follows. Consider any edge $e = (u, v) \in E(H_2)$. Assume that $e$ is the $i$th edge of $u$ and the $j$th edge of $v$, that is, $e = e_i(u) = e_j(v)$. Then we add an edge $e' = (s_i(u), s_j(v))$ to graph $H^*$, and we view this edge as the *copy of the edge* $e \in E(H_2)$. We will not distinguish between the edge $e$ of $H_2$, and the edge $e'$ of $H^*$.

Our last step is to add vertex $x$ to graph $H^*$, that connects to every terminal $t \in \tilde{T}$ with an edge $(x, t)$, that is also viewed as an outer edge. The following lemma, whose proof is deferred to Section J.4 of Appendix, allows us to compute an ordering $\tilde{\mathcal{O}}$ of the terminals, such that the graph $H^*$ has a drawing $\varphi$ with few crossings, in which the inner edges do not participate in any crossings, and the images of the edges incident to $x$ enter $x$ in order consistent with $\tilde{\mathcal{O}}$.

**Lemma 11.10** *There is an efficient algorithm that computes an ordering $\tilde{\mathcal{O}}$ of the terminals in $\tilde{T}$, such that there is a drawing $\varphi$ of graph $H^*$ with at most $O\left(\mathsf{OPT}_{\mathsf{cnwrs}}(I) \cdot \frac{\eta^2 \log^{74} m}{\alpha^{12}(\alpha')^4}\right) + O\left(\frac{\tilde{k}\eta \log^{37} m}{\alpha^6(\alpha')^2}\right)$ crossings, in which all crossings are between pairs of outer edges. Moreover, if we denote $\tilde{T} = \left\{t_1, \ldots, t_{\tilde{k}}\right\}$, where the terminals are indexed according to the ordering $\tilde{\mathcal{O}}$, and, for each $1 \leq i \leq t_{\tilde{k}}$, denote by $e_i = (t_i, x)$ the edge of $H^*$ connecting $t_i$ to $x$, then the images of the edges $e_1, \ldots, e_{\tilde{k}}$ enter the image of $x$ in this circular order in the drawing $\varphi$.*

From now on we fix the ordering $\tilde{\mathcal{O}}$ of the terminals in $\tilde{T}$ given by Lemma 11.10, and the drawing $\varphi$ of $H^*$ (which is not known to the algorithm).

It will be convenient for us to slightly modify the graph $H^*$ as follows. We denote the terminals by $\tilde{T} = \left\{t_1, \ldots, t_{\tilde{k}}\right\}$, where the terminals are indexed according to the circular ordering $\tilde{\mathcal{O}}$. Let $H'$ be a graph obtained from $H^*$, by first deleting the vertex $x$ from it, and then adding, for all $1 \leq i < \tilde{k}$, an edge $e^*_i = (t_i, t_{i+1})$, and another edge $e^*_{\tilde{k}} = (t_{\tilde{k}}, t_1)$. We denote this set of the newly added edges by $E^*$, and we view them as inner edges. Note that the edges of $E^*$ form a simple cycle, that we denote by $L^*$. We also denote $H'' = H' \setminus E^*$.

We note that the drawing $\varphi$ of $H^*$ can be easily extended to obtain a drawing $\varphi'$ of graph $H'$ in the plane, so that the inner edges of $H'$ do not participate in any crossings, and the image of the cycle $L^*$ (which must be a simple closed curve) is the boundary of the outer face.

In order to do so, we start with the drawing $\varphi$ of $H^*$ on the sphere, and then consider the tiny $x$-disc $D = D_\varphi(x)$, denoting its boundary by $\gamma^*$. For every terminal $t_i \in \tilde{T}$, we denote by $e_i$ the unique edge incident to $t_i$ in $H''$, and by $e'_i = (t_i, x)$. We also denote by $\gamma_i, \gamma'_i$ the images of the edges $e_i, e'_i$ in drawing $\varphi$. Let $p_i$ be the unique point on the intersection of $\gamma'_i$ and $\gamma^*$. We move the image of terminal $t_i$ to point $p_i$. We then modify the image of the edge $e_i$, so that it becomes a concatenation of $\gamma_i$, and the portion of $\gamma'_i$ lying outside the interior of $D$. Lastly, we draw the edges of $E^*$ in a natural way, where edge $e^*_i$ is simply a segment of $\gamma^*$ between the images of $t_i$ and $t_{i+1}$, so that all resulting segments are mutually internally disjoint. Once we delete the vertex $x$ from this drawing, no part of the resulting drawing is contained in the interior of the disc $D$, and the image of the cycle $L^*$ is precisely $\eta$, so we can view the resuting drawing $\varphi'$ of $H'$ as a drawing in the plane, with $D$ being its outer face. Note that this transformation does not increase the number of crossings.

The next observation follows by substituting parameters and bounds that we have already established. The proof is included in Section J.5 of Appendix.

**Observation 11.11** *Let $c_2$ be a large enough constant. We can set the value of constant $c_1$ so that it is large enough, and, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I) < \frac{(\tilde{k}\tilde{\alpha}\alpha')^2}{c_1\eta' \log^{20} m}$, then $\mathsf{cr}(\varphi') \leq \mathsf{cr}(\varphi) \leq \frac{\tilde{k}^2}{c_2\eta^5}$.*

We will se the value of constant $c_2$ later, and the value of constant $c_1$ will then be set using Observation 11.11. It is now enough to ensure that, if $\mathsf{cr}(\varphi') < \frac{\tilde{k}^2}{\eta^5}$, then the probability that the algorithm returns FAIL is at most 1/2.

Let $\Lambda' = \Lambda(H'', \tilde{T})$ be the collection of all routers in graph $H''$ with respect to the set $\tilde{T}$ of terminals. We need the following simple observation, whose proof is deferred to Section J.6 of the Appendix.

**Observation 11.12** *There is an efficient algorithm, that, given an explicit distribution $\mathcal{D}$ over the routers of $\Lambda'$, such that for every* **outer** *edge $e \in E(H'')$, $\mathbf{E}_{\mathcal{Q}\sim\mathcal{D}}\left[(\mathrm{cong}_{H''}(\mathcal{Q}, e))^2\right] \leq \beta$, computes an explicit distribution $\mathcal{D}'$ over the routers in $\Lambda(H, \tilde{T})$, where for every edge $e \in E(H)$, $\mathbf{E}_{\mathcal{Q}'\sim\mathcal{D}'}\left[(\mathrm{cong}_H(\mathcal{Q}', e))^2\right] \leq \beta$.*

### 11.2.3 Summary of Step 2

In the remainder of the proof of Theorem 6.4 we will work with graph $H'$ only. Recall that graph $H'$ contains a set $E^* = \left\{e^*_1, \ldots, e^*_{\tilde{k}}\right\}$ of edges (that are considered to be inner edges), where for all $1 \leq i \leq \tilde{k}$, $e^*_i = (t_i, t_{i+1})$ (we use indexing modulo $\tilde{k}$). The set $E^*$ of edges defines a cycle $L^* = (t_1, \ldots, t_{\tilde{k}})$ in graph $H'$. We also denoted $H'' = H' \setminus E^*$. Recall that graph $H''$ is obtained from a subgraph $H_2 \subseteq H$, by replacing every vertex $v \in V(H_2) \setminus \tilde{T}$ with a grid $\Pi(v)$. All edges lying in the resulting grids $\Pi(v)$, and the edges of $E^*$ are inner edges, while all other edges of $H'$ are outer edges. Each outer edge of $H'$ corresponds to some edge of graph $H_2$, and we do not distinguish between these edges. Note that in graph $H'$, all vertices have degrees at most 4. We will also use the clustering $\mathcal{C}_Y$ of graph $H_2$, and the fact that, from Property P6:

$$\sum_{C\in\mathcal{C}_Y} |\delta_H(C)|^2 < \frac{(\tilde{k}\tilde{\alpha}\alpha')^2}{c_1 \log^{20} m}. \tag{18}$$

We further partition the outer edges of graph $H''$ into two subsets: type-1 outer edges and type-2 outer edges. Consider any outer edge $e$ in graph $H''$, and let $e' = (u, v)$ be the corresponding edge in

graph $H$. If $u$ and $v$ both lie in the same cluster $C \in \mathcal{C}_Y$, then we say that $e$ is a *type-2* outer edge, and otherwise it is a type-1 outer edge. Intuitively, for each type-1 outer edge, there is a corresponding edge in the contracted graph $\hat{H} = H_{|\mathcal{C}}$. From Property P3, we obtain the following observation.

**Observation 11.13** *There is a universal constant $c$ (independent of $c_1$ and $c_2$), such that the total number of type-1 outer edges in $H''$ is bounded by $c\tilde{k} \cdot \eta \log^8 m/\alpha^3$.*

Recall that from Property P4, every vertex of $\tilde{T}$ has 1 in $H_2$, and vertex set $\tilde{T}$ is $\tilde{\alpha}$-well-linked in $\hat{H}_2$. Combining this with the $\alpha'$-bandwidth property of every cluster $C \in \mathcal{C}_Y$ from Property P2, from Claim 4.39, the set $\tilde{T}$ of terminals is $\tilde{\alpha} \cdot \alpha'$-well-linked in $H_2$. Lastly, using the fact that each graph in $\{\Pi(v) \mid v \in V(H_2)\}$ has the 1-bandwidth property, from Claim 4.39, we get the following observation.

**Observation 11.14** *The set $\tilde{T}$ of terminals is $\alpha^*$-well-linked in $H''$, where $\alpha^* = \tilde{\alpha} \cdot \alpha' = \Theta(\alpha\alpha'/\log^4 m)$. Moreover, each terminal in $\tilde{T}$ has degree 1 in $H''$ and degree 3 in $H'$.*

(we have used the fact that $\tilde{\alpha} = \Theta(\alpha/\log^4 m)$ (see Property P4)).

We will restrict our attention to special types of drawings of graph $H'$, called *legal drawings*, that we define next.

**Definition 11.15 (Legal drawing of $H'$)** *We say that a drawing $\varphi^*$ of graph $H'$ in the plane is* legal *if it has the following properties:*

- *no inner edge of $H'$ participates in any crossing of $\varphi^*$, and in particular the image of the cycle $L^*$ is a simple closed curve, denoted by $\gamma^*$; and*
- *$\gamma^*$ is the boundary of the outer face in the drawing.*

We let $\varphi^*$ be a legal drawing of $H'$ with smallest number of crossings, and we denote by $\mathsf{cr}^*$ the number of crossings in $\varphi^*$. From Observation 11.11, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I) < \frac{(\tilde{k}\tilde{\alpha}\alpha')^2}{c_1\eta' \log^{20} m}$, then $\mathsf{cr}^* \leq \frac{\tilde{k}^2}{c_2\eta^5}$.

Denote $\tilde{T} = \left\{t_1, \ldots, t_{\tilde{k}}\right\}$, where the terminals are indexed according their ordering in $\tilde{\mathcal{O}}$. We partition the set $\tilde{T}$ of terminals into four subsets $T_1, \ldots, T_4$, where $T_1, T_2, T_3$ contain $\left\lfloor \tilde{k}/4 \right\rfloor$ consecutive terminals from $\tilde{T}$ each, and $T_4$ contains the remaining terminals, in a natural way using the ordering $\tilde{\mathcal{O}}$, that is, $T_1 = \left\{t_1, \ldots, t_{\lfloor \tilde{k}/4 \rfloor}\right\}$, $T_2 = \left\{t_{\lfloor \tilde{k}/4 \rfloor + 1}, \ldots, t_{2\lfloor \tilde{k}/4 \rfloor}\right\}$, $T_3 = \left\{t_{2\lfloor \tilde{k}/4 \rfloor + 1}, \ldots, t_{3\lfloor \tilde{k}/4 \rfloor}\right\}$, and $T_4 = \left\{t_{3\lfloor \tilde{k}/4 \rfloor + 1}, \ldots, t_{\tilde{k}}\right\}$. Clearly, each of the four sets contains at least $\left\lfloor \tilde{k}/4 \right\rfloor$ terminals.

Recall that in a legal drawing $\varphi$ of $H'$, the image of the cycle $L^*$ is a simple closed curve, that we denoted by $\gamma^*$. It will be convenient for us to view this curve $\gamma^*$ as the boundary of a rectangular area in the plane, that encloses the legal drawing of $H'$. We sometimes refer to this rectangular area as the *bounding box* of the drawing, and denote it by $B^*$. We will think of the terminals in $T_1$ and $T_3$ as appearing on the left and on the right boundaries of $B^*$, respectively, and of the terminals in $T_2$ and $T_4$ as appearing on the top and the bottom boundaries of $B^*$, respectively.

For all $1 \leq i \leq 4$, we let $\tilde{\mathcal{O}}_i$ be the ordering of the terminals in $T_i$ consistent with their ordering on the boundary of $B^*$ (where each ordering $\tilde{\mathcal{O}}_i$ is no longer circular), so that the terminals in sets $T_1$ and in $T_3$ appear in the bottom-to-top order, and the terminals in $T_2$ and $T_4$ appear in their left-to-right order (so $\tilde{\mathcal{O}}$ is obtained by concatenationg $\tilde{\mathcal{O}}_1, \tilde{\mathcal{O}}_2$, the reversed ordering $\tilde{\mathcal{O}}_3$, and the reversed ordering $\tilde{\mathcal{O}}_4$).

Recall that $\Lambda' = \Lambda(H'', \tilde{T})$. Our goal from now on is to design a randomized algorithm, that either computes a distribution $\mathcal{D}$ over the routers of $\Lambda'$, such that for every outer edge $e \in E(H'')$, $\mathbf{E}_{\mathcal{Q} \sim \mathcal{D}}\left[(\mathrm{cong}_{H'}(\mathcal{Q}, e))^2\right] \leq O\left(\frac{\log^{16} m}{(\alpha\alpha')^4}\right)$, or returns FAIL. It is enough to ensure that, if $\mathsf{cr}^* \leq \frac{\tilde{k}^2}{c_2\eta^5}$ for some large enough constant $c_2$, whose value we can set later, then the probability that the algorithm returns FAIL is at most $1/4$.

### 11.3 Step 3: Constructing a Grid Skeleton

In this and the following step we will construct a grid-like structure in graph $H''$. Recall that the set $\tilde{T}$ of terminals is $\alpha^*$-well-linked in graph $H''$. From Theorem 4.17 there is a set $\mathcal{P}'$ of paths in $H''$, routing all terminals of $T_1$ to terminals of $T_3$, with edge-congestion at most $\lceil 1/\alpha^* \rceil$, such that the routing is one-to-one. From Claim 4.2, there is a collection $\mathcal{P}''$ of at least $|T_1|/\lceil 1/\alpha^* \rceil = \left\lfloor \tilde{k}/4 \right\rfloor / \lceil 1/\alpha^* \rceil \geq \alpha^* \tilde{k}/8$ edge-disjoint paths in $H''$, routing some subset of terminals of $T_1$ to a subset of terminal of $T_3$, in graph $H''$. Moreover, since graph $H''$ has maximum vertex degree at most 4, using arguments similar to those in the proof of Claim 4.2, there is a collection $\mathcal{P}$ of $\left\lfloor \alpha^* \tilde{k}/32 \right\rfloor$ **node-disjoint** paths in graph $H''$, routing some subset $A \subseteq T_1$ of terminals, to some subset $A' \subseteq T_3$ of terminals. We can compute such a set $\mathcal{P}$ of paths efficiently using standard maximum $s$-$t$ flow algorithms.

Using similar reasoning, we can compute a collection $\mathcal{R}$ of $\left\lfloor \alpha^* \tilde{k}/32 \right\rfloor$ node-disjoint paths in graph $H''$, routing some subset $B \subseteq T_2$ of terminals, to some subset $B' \subseteq T_4$ of terminals.

Intuitively, after we discard a small subset of paths from each of the sets $\mathcal{P}$ and $\mathcal{R}$, the remaining paths will be used in order to construct a grid-like structure, where paths in $\mathcal{P}$ will serve as horizontal paths of the grid, and paths in $\mathcal{R}$ will serve as vertical paths. If the paths in the resulting sets do not form a grid-like structure, then we will terminate the algorithm with a FAIL. We will prove that, if $\mathsf{cr}^* \leq \frac{\tilde{k}^2}{c_2 \eta^5}$ for a large enough constant $c_2$, then we will construct the grid-like structure successfully with probability at least 3/4.

We denote $\mathcal{P}_0 = \mathcal{P}$ and $\mathcal{R}_0 = \mathcal{R}$. Recall that so far, $|\mathcal{P}_0|, |\mathcal{R}_0| \geq \left\lfloor \alpha^* \tilde{k}/32 \right\rfloor$.

Intuitively, if the dimensions of the grid-like structure that we construct are $(h \times h)$, then we need $h$ to be quite close to $\tilde{k}$, since this grid-like structure will be exploited in order to define the distribution $\mathcal{D}$ over the routers of $\Lambda'$. We will first construct a smaller grid-like structure, that we call a *grid skeleton*. This grid skeleton will be associated with a grid $\Pi^*$ of smaller dimensions, that we sometimes call a *supergrid*. We then extend this grid skeleton to construct a large enough grid-like structure.

We will use two additional parameters. The first parameter is:

$$\lambda = \frac{2^{24} c \cdot \eta \log^8 m}{\alpha^* \alpha^3},$$

where $c$ is the constant from Observation 11.13. Notice that, since $\alpha^* = \Theta(\alpha\alpha'/\log^4 m)$, we get that $\lambda = O\left(\frac{\eta \log^{12} m}{\alpha^4 \alpha'}\right)$. Moreover, since $\eta > \frac{c^* \log^{12} m}{\alpha^4 \alpha'}$ for a large enough constant $c^*$ (from the statement of Theorem 6.4), $\lambda < \eta^2$ holds. The supergrid that we construct will have dimensions $(\Theta(\lambda) \times \Theta(\lambda))$. The second parameter is:

$$\psi = \left\lfloor \frac{\alpha^* \tilde{k}}{64\lambda} \right\rfloor = \left\lfloor \frac{\alpha^3 (\alpha^*)^2 \tilde{k}}{2^{30} c \eta \log^8 m} \right\rfloor.$$

Clearly, $|\mathcal{R}_0|, |\mathcal{P}_0| \geq \lambda\psi$. Note that, since $\alpha^* = \Theta(\alpha\alpha'/\log^4 m)$, $\psi \geq \Omega\left(\frac{\tilde{k}}{\eta} \cdot \frac{\alpha^5 (\alpha')^2}{\log^{16} m}\right)$. Since $\eta > \frac{c^* \log^{16} m}{\alpha^5 (\alpha')^2}$ from the statement of Theorem 6.4, we get that $\psi > \frac{16\tilde{k}}{\eta^2}$. Every cell of the supergrid will be associated with a collection of $\Theta(\psi)$ horizontal paths and $\Theta(\psi)$ vertical paths, that will help us form the grid-like structure.

We discard paths from $\mathcal{P}_0$ and from $\mathcal{R}_0$ arbitrarily, until $|\mathcal{P}_0| = |\mathcal{R}_0| = \lambda\psi$ holds.

We denote by $A_0 \subseteq T_1, A'_0 \subseteq T_3$ the endpoints of the paths in $\mathcal{P}_0$, and we denote by $B_0 \subseteq T_4$, $B'_0 \subseteq T_2$ the endpoints of the paths in $\mathcal{R}_0$.

**Grid Skeleton Construction**

We view the paths in $\mathcal{P}_0$ as directed from vertices of $A_0$ to vertices of $A'_0$. Recall that $A_0 \subseteq T_1$, so the ordering $\tilde{\mathcal{O}}_1$ of the terminals in $T_1$ defines an ordering $\mathcal{O}_{A_0} = \{a_1, \ldots, a_{\lambda\psi}\}$ of the terminals in $A_0$. This ordering in turn defines an ordering $\mathcal{O}_{\mathcal{P}_0}$ of the paths in $\mathcal{P}_0$, as follows: if, for all $1 \leq i \leq \lambda\psi$, $P_i \in \mathcal{P}_0$ is the path originating from $a_i$, then $\mathcal{O}_{\mathcal{P}_0} = \{P_1, \ldots, P_{\lambda\psi}\}$.

Similarly, we view the paths in $\mathcal{R}_0$ as directed from vertices of $B_0$ to vertices of $B'_0$. Ordering $\tilde{\mathcal{O}}_4$ of terminals in $T_4$ defines an ordering $\mathcal{O}_{B_0} = \{b_1, \ldots, b_{\lambda\psi}\}$ of the vertices in $B_0$, which in turn defines an ordering $\mathcal{O}_{\mathcal{R}_0} = \{R_1, \ldots, R_{\lambda\psi}\}$ of paths in $\mathcal{R}_0$, where for all $i$, path $R_i$ originates at vertex $b_i$.

We partition the set $\mathcal{P}_0$ of paths into groups $\mathcal{U}_1, \ldots, \mathcal{U}_\lambda$ of cardinality $\psi$ each, using the ordering $\mathcal{O}_{\mathcal{P}_0}$, so for $1 \leq i < \lambda$, set $\mathcal{U}_i$ is the $i$th set of $\psi$ consecutive paths of $\mathcal{P}_0$. Let $\lambda' = \lfloor (\lambda - 1)/2 \rfloor$. For all $1 \leq i \leq \lambda'$, we let $P^*_i$ be a path that is chosen uniformly at random from set $\mathcal{U}_{2i}$. Let $\mathcal{P}^* = \left\{P^*_1, \ldots, P^*_{\lambda'}\right\}$ be the resulting set of chosen paths. Intuitively, the path in $\mathcal{P}^*$ will serve as the horizontal paths in the grid skeleton that we construct. We then let $\mathcal{P}_1 \subseteq \mathcal{P}_0$ be the set containing all paths in sets $\{\mathcal{U}_{2i-1}\}_{i=1}^{\lambda'+1}$.

We perform similar computation on the set $\mathcal{R}_0$ of paths. First, we partition $\mathcal{R}_0$ into groups $\mathcal{U}'_1, \ldots, \mathcal{U}'_\lambda$ of cardinality $\psi$ each, using the ordering $\mathcal{O}_{\mathcal{R}_0}$, so for $1 \leq i < \lambda$, set $\mathcal{U}'_i$ is the $i$th set of $\psi$ consecutive paths of $\mathcal{R}_0$. For all $1 \leq i \leq \lambda'$, we let $R^*_i$ be a path that is chosen uniformly at random from set $\mathcal{U}'_{2i}$. Let $\mathcal{R}^* = \left\{R^*_1, \ldots, R^*_{\lambda'}\right\}$ be the resulting set of chosen paths. Intuitively, the path in $\mathcal{R}^*$ will serve as the vertical paths in the grid skeleton that we construct. We then let $\mathcal{R}_1 \subseteq \mathcal{R}_0$ be the set containing all paths in sets $\left\{\mathcal{U}'_{2i-1}\right\}_{i=1}^{\lambda'+1}$.

We let $\mathcal{E}_1$ be the bad event that there are two distinct paths $Q, Q' \in \mathcal{R}^* \cup \mathcal{P}^*$, and two distinct edges $e \in E(Q)$, $e' \in E(Q')$, such that the images of $e$ and $e'$ cross in the drawing $\varphi^*$ of $H'$.

**Observation 11.16** *If* $\mathsf{cr}^* < \frac{\tilde{k}^2}{c_2\eta^5}$, *then* $\mathbf{Pr}\left[\mathcal{E}_1\right] \leq 1/64$.

**Proof:** Consider any crossing $(e, e')$ in the drawing $\varphi^*$. We say that crossing $(e, e')$ is *selected* if there are two distinct paths $Q, Q' \in \mathcal{R}^* \cup \mathcal{P}^*$ with $e \in E(Q)$, $e' \in E(Q')$. Notice that $e$ may belong to at most two paths in $\mathcal{R}_0 \cup \mathcal{P}_0$ (one path in each set), and the same is true for $e'$. Each path of $\mathcal{R}_0 \cup \mathcal{P}_0$ is chosen to $\mathcal{R}^* \cup \mathcal{P}^*$ with probability at most $1/\psi$. Therefore, the probability that a path containing $e$, and a path containing $e'$ are chosen to $\mathcal{R}^* \cup \mathcal{P}^*$ is at most $4/\psi^2$. Since $\mathcal{E}_1$ can only happen if at least one crossing is chosen, from the union bound, $\mathbf{Pr}\left[\mathcal{E}_1\right] \leq 4\mathsf{cr}^*/\psi^2$. Since $\psi > \frac{16\tilde{k}}{\eta^2}$, if $\mathsf{cr}^* < \frac{\tilde{k}^2}{c_2\eta^5}$, then:

$$\mathbf{Pr}\left[\mathcal{E}_1\right] \leq \frac{4\mathsf{cr}^*}{\psi^2} \leq \frac{1}{64c_2\eta} \leq \frac{1}{64}.$$

□

We say that a path $Q \in \mathcal{R}_0 \cup \mathcal{P}_0$ is *heavy* iff there are at least $\frac{\psi}{64\lambda}$ crossings $(e, e')$ in $\varphi^*$, such that at least one of the edges $e, e'$ lies on path $Q$. We say that a bad event $\mathcal{E}_2$ happens iff at least one path in $\mathcal{R}^* \cup \mathcal{P}^*$ is heavy.

**Observation 11.17** *If* $\mathsf{cr}^* < \frac{\tilde{k}^2}{c_2\eta^5}$, *then* $\mathbf{Pr}\left[\mathcal{E}_2\right] \leq 1/64$.

**Proof:** Note that every edge of $H''$ may lie on at most two paths of $\mathcal{R}_0 \cup \mathcal{P}_0$, and every crossing $(e, e')$ involves two edges. Therefore, the total number of heavy paths in $\mathcal{R}_0 \cup \mathcal{P}_0$ is bounded by $\frac{4\mathsf{cr}^*}{\psi/(64\lambda)} = \frac{2^8\lambda \cdot \mathsf{cr}^*}{\psi}$. Assuming that $\mathsf{cr}^* < \frac{\tilde{k}^2}{c_2\eta^5}$, and using the fact that $\psi = \Omega\left(\frac{\tilde{k}}{\eta} \cdot \frac{\alpha^5(\alpha')^2}{\log^{16} m}\right)$ and $\lambda = O\left(\frac{\eta \log^{12} m}{\alpha^4\alpha'}\right)$, we get that the total number of heavy paths in $\mathcal{R}_0 \cup \mathcal{P}_0$ is bounded by:

$$\frac{2^8\lambda \cdot \mathsf{cr}^*}{\psi} \leq O\left(\frac{\tilde{k}^2}{c_2\eta^5} \cdot \frac{\eta \log^{12} m}{\alpha^4\alpha'} \cdot \frac{\eta}{\tilde{k}} \cdot \frac{\log^{16} m}{\alpha^5(\alpha')^2}\right) \leq O\left(\frac{\tilde{k} \log^{28} m}{c_2\eta^3\alpha^9(\alpha')^3}\right).$$

Note that each heavy path may be selected to $\mathcal{R}^* \cup \mathcal{P}^*$ with probability at most $1/\psi$. Therfore, using the union bound and the fact that $\psi = \Omega\left(\frac{\tilde{k}}{\eta} \cdot \frac{\alpha^5(\alpha')^2}{\log^{16} m}\right)$, we get that:

$$\mathbf{Pr}\left[\mathcal{E}_2\right] \leq O\left(\frac{\tilde{k}\log^{28} m}{\psi \cdot c_2\eta^3\alpha^9(\alpha')^3}\right) \leq O\left(\frac{\log^{44} m}{c_2\eta^2\alpha^{14}(\alpha')^5}\right).$$

Recall that, from the conditions of Theorem 6.4, $\eta \geq c^* \log^{46} m/(\alpha^{10}(\alpha')^4)$, where $c^*$ is a sufficiently large constant. Therefore, if $\mathsf{cr}^* < \frac{\tilde{k}^2}{c_2\eta^5}$, then $\mathbf{Pr}\left[\mathcal{E}_2\right] \leq 1/64$. □

Let $\mathcal{R}' \subseteq \mathcal{R}_1$, $\mathcal{P}' \subseteq \mathcal{P}_1$ be the sets containing all paths $Q$, such that, in drawing $\varphi^*$, the image of some edge of $Q$ crosses the image of some edge lying on the paths of $\mathcal{R}^* \cup \mathcal{P}^*$. Note that the drawing $\varphi^*$ is not known to us, and so neither are the sets $\mathcal{R}', \mathcal{P}'$ of paths. We will also use the following observation:

**Observation 11.18** *If $\mathcal{E}_2$ did not happen, then $|\mathcal{R}'|, |\mathcal{P}'| \leq \psi/32$.*

**Proof:** Recall that $|\mathcal{R}^*| + |\mathcal{P}^*| \leq \lambda$. If bad event $\mathcal{E}_2$ did not happen, then for each path $Q \in \mathcal{R}^* \cup \mathcal{P}^*$, there are at most $\frac{\psi}{64\lambda}$ crossings in $\varphi^*$, in which edges of $Q$ participate. Therefore, if event $\mathcal{E}_2$ did not happen, there are in total at most $\psi/64$ crossings $(e, e')$ in the drawing $\varphi^*$, where at least one of the edges $e, e'$ lies on a path of $\mathcal{P}^* \cup \mathcal{Q}^*$. Let $E' \subseteq E(H'')$ be the set of all edges $e$, such that there is an edge $e'$ lying on some path of $\mathcal{P}^* \cup \mathcal{Q}^*$, and crossing $(e, e')$ is present in $\varphi^*$. Then $|E'| \leq \psi/32$. Each path in $\mathcal{R}' \cup \mathcal{P}'$ must contain an edge of $E'$. As the paths in each of the sets $\mathcal{R}', \mathcal{P}'$ are disjoint, $|\mathcal{R}'|, |\mathcal{P}'| \leq \psi/32$ must hold. □

**Summary of Step 3.** In this step we have constructed a grid skeleton, that consists of two sets of paths: $\mathcal{P}^* = \left\{P_1^*, \ldots, P_{\lambda'}^*\right\}$, and $\mathcal{R}^* = \left\{R_1^*, \ldots, R_{\lambda'}^*\right\}$, where $\lambda' = \lfloor(\lambda-1)/2\rfloor$. Recall that $\mathcal{P}^* \subseteq \mathcal{P}_0$, and the paths in $\mathcal{P}^*$ are indexed according to their order in $\mathcal{O}_{\mathcal{P}_0}$. Recall that we have also defined the set $\mathcal{P}_1 \subseteq \mathcal{P}_0$ of paths, containing all paths in sets $\{\mathcal{U}_{2i-1}\}_{i=1}^{\lambda'+1}$. It would be convinient for us to re-index the groups $\mathcal{U}_i$ as follows: for $0 \leq i \leq \lambda'$, set $\mathcal{U}_i = \mathcal{U}_{2i+1}$. In other words, the paths of $\mathcal{U}_0$ lie before path $P_1^*$ in the ordering $\mathcal{O}_{\mathcal{P}_0}$, the paths of $\mathcal{U}_{\lambda'}$ lie after $P_{\lambda'}^*$ in this ordering, and, for $1 \leq i < \lambda'$, the paths of $\mathcal{U}_i$ lie between paths $P_i^*$ and $P_{i+1}^*$. Similarly, $\mathcal{R}^* \subseteq \mathcal{R}_0$, and the paths in $\mathcal{R}^*$ are indexed according to their order in $\mathcal{O}_{\mathcal{R}_0}$. We have also defined the set $\mathcal{R}_1 \subseteq \mathcal{R}_0$ of paths, containing all paths in sets $\left\{\mathcal{U}'_{2i-1}\right\}_{i=1}^{\lambda'+1}$. As before, we re-index them as follows: for $0 \leq i \leq \lambda'$, we set $\mathcal{U}'_i = \mathcal{U}'_{2i+1}$. Therefore, the paths of $\mathcal{U}'_0$ lie before path $R_1^*$ in the ordering $\mathcal{O}_{\mathcal{R}_0}$, the paths of $\mathcal{U}'_{\lambda'}$ lie after $R_{\lambda'}^*$ in this ordering, and, for $1 \leq i < \lambda'$, the paths of $\mathcal{U}'_i$ lie between paths $R_i^*$ and $R_{i+1}^*$.

From our definition, if $\mathcal{E}_1$ did not happen, then for every pair $Q, Q' \in \mathcal{P}^* \cup \mathcal{Q}^*$ of distinct paths, their images in $\varphi^*$ do not cross (but note that the image of a single path may cross itself).

We have also defined a set $\mathcal{P}' \subseteq \mathcal{P}_1$ and a set $\mathcal{R}' \subseteq \mathcal{R}_1$ of paths, containing all paths $Q$ whose image crosses the image of some path in $\mathcal{R}^* \cup \mathcal{P}^*$ in drawing $\varphi^*$. From Observation 11.18, if Event $\mathcal{E}_2$ does not happen, then $|\mathcal{P}'|, |\mathcal{R}'| \leq \psi/32$. Note that the sets $\mathcal{P}', \mathcal{R}'$ of paths are not known to the algorithm.

It will be convenient for us to consider the $((\lambda'+1) \times (\lambda'+1))$-grid $\Pi^*$. We view the columns of the grid as corresponding to the left boundary of the bounding box $B^*$, the paths in $\left\{R_1^*, \ldots, R_{\lambda'}^*\right\}$, and the right boundary of the bounding box $B^*$. For convenience, we index the columns of the grid from 0 to $\lambda'+1$, so the left boundary of the bounding box corresponds to column 0, and, for $1 \leq i \leq \lambda'$, path $P_i^*$ represents the $i$th column of the grid, with the right boundary of $B^*$ repersenting the last column. Similarly, we view the bottom boundary of $B^*$, the paths in $\left\{P_1^*, \ldots, P_{\lambda'}^*\right\}$, and the top boundary of $B^*$ as representing the rows of the grid, in the bottom-to-top order. As before, we index the rows of the grid so that the botommost row has index 0 and the topmost row has index $\lambda'+1$. Notice however that the union of the paths in $\mathcal{P}^* \cup \mathcal{R}^*$ does not necessarily form a proper grid graph, as it is possible that, for a pair $P \in \mathcal{P}^*$, $R \in \mathcal{R}^*$ of paths, $P \cap R$ is a collection of several disjoint paths.

We will now consider the drawing $\varphi^*$ of $H''$, and we will use it to define vertical and horizontal strips corresponding to paths in $\mathcal{P}^*$ and $\mathcal{R}^*$, respectively. We will also associate, with each cell of the grid $\Pi^*$, some region of the plane. We assume in the following definitions that Event $\mathcal{E}_1$ did not happen.

Consider first the image $\gamma_i$ of some path $P_i^* \in \mathcal{P}^*$ in the drawing $\varphi^*$. Note that $\gamma_i$ is not necessarily a simple curve. We define two simple curves, $\gamma_i^t$ and $\gamma_i^b$, where $\gamma_i^t$ follows the image of $\gamma_i$ from the top, and $\gamma_i^b$ follows it from the bottom. In other words, we let $\gamma_i^b$ be a simple curve, whose every point lies on $\gamma_i$, that has the same endpoints as $\gamma_i$, such that the following holds: for every point $p \in \gamma_i$, either $p \in \gamma_i^b$, or $p$ lies above $\gamma_i^b$ in the bounding box $B^*$. We define the other curve, $\gamma_i^t$ symmetrically, so curve $\gamma_i$ is contained in the disc whose boundary is $\gamma_i^t \cup \gamma_i^b$ (see Figure 39). For convenience, we let $\gamma_0^t$ be the bottom boundary of the bounding box $B^*$, and $\gamma_{\lambda'+1}^b$ be the top boundary of the bounding box $B^*$. We now define, for all $0 \leq i \leq \lambda'$, a region of the plane that we call the $i$th horizontal strip, and denote by $\mathsf{HStrip}_i$. This strip is simply the closed region of the bounding box between the curves $\gamma_i^t$ and $\gamma_{i+1}^b$.

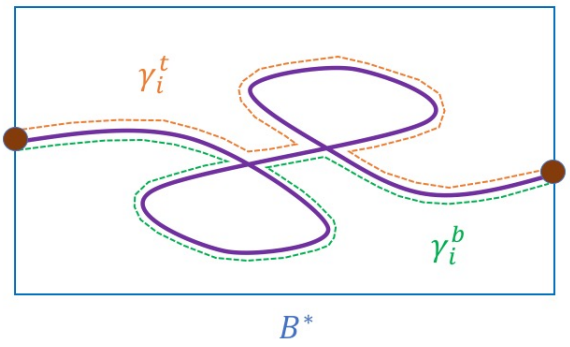



Figure 39: An illustration of curves $\gamma_i^t$ and $\gamma_i^b$. The curve $\gamma_i$ is shown in purple.

For every vertical path $R_i^* \in \mathcal{R}^*$, we also define two curves, $\gamma_i^\ell$ and $\gamma_i^r$, that follow the image $\gamma_i$ of $R_i^*$ in $\varphi^*$ on its left and on its right, respectively. We denote by $\gamma_0^r$ the left boundary of the bounding box $B^*$, and by $\gamma_{\lambda'+1}^\ell$ its right boundary. For all $0 \leq i \leq \lambda'$, we define a vertical strip $\mathsf{VStrip}_i$ to be the closed region of the bounding box $B^*$ betwen $\gamma_i^r$ and $\gamma_{i+1}^\ell$.

The following observation is immediate from the fact that the paths in $\mathcal{P}_0$ are node-disjoint, and so are the paths in $\mathcal{R}_0$.

**Observation 11.19** *If $R \in \mathcal{R}_1$ is a path whose image in $\varphi^*$ intersects the interior of more than one vertical strip in $\{\mathsf{VStrip}_0, \ldots, \mathsf{VStrip}_{\lambda'+1}\}$, then $R \in \mathcal{R}'$. Similarly, if $P \in \mathcal{P}_1$ is a path whose image in $\varphi^*$ intersects the interior of more than one horizontal strip in $\{\mathsf{HStrip}_0, \ldots, \mathsf{HStrip}_{\lambda'+1}\}$, then $P \in \mathcal{P}'$.*

Lastly, for all $0 \leq i, j \leq \lambda'$, we let $\mathsf{CellRegion}_{i,j} = \mathsf{HStrip}_i \cap \mathsf{VStrip}_j$ be a closed region of the plane that we associate with cell $\mathsf{Cell}_{i,j}$ of the grid $\Pi^*$.

## 11.4 Step 4: Constructing a Grid-Like Structure

In this step we further delete some paths from sets $\mathcal{R}_1$ and $\mathcal{P}_1$ to ensure that the resulting paths form a grid-like structure. This is done in three stages. In the first stage, we discard some paths to ensure that every remaining path in $\mathcal{R}_1$ intersects the paths in $\mathcal{P}^*$ "in order" (we formally define this notion later), and we process the paths in $\mathcal{P}_1$ similarly. In the second stage, we associate, with every cell of the grid $\Pi^*$ a collection of horizontal paths and a collection of vertical paths. In the third stage,

we ensure that, for every cell of the grid $\Pi^*$, there are many inersections between its corresponding horizontal and vertical paths.

Before we continue, we discard some paths of $\mathcal{R}_1 \cup \mathcal{P}_1$ that must lie in $\mathcal{R}' \cup \mathcal{P}'$. Specifically, consider some path $P \in \mathcal{P}_1$, and assume that it lies in group $\mathcal{U}_i$, for some $0 \leq i \leq \lambda'$. Let $(a, a')$ be the endpoints of path $P$, with $a \in T_1$ and $a' \in T_3$. Notice that from the definition, if $i > 0$, then $a$ must lie, in the ordering $\tilde{\mathcal{O}}_1$ of the terminals of $T_1$, after the endpoint of the path $P_i^*$ that belongs to $T_1$. Similarly, if $i < \lambda'$, then $a$ must lie before the endpoint of the path $P_{i+1}^*$ that belongs to $T_1$ in the same ordering. In particular, we are guaranteed that, in the drawing $\varphi^*$, the image of $P$ must intersect the interior of the horizontal strip $\mathsf{HStrip}_i$. Consider now the endpoint $a'$ of $P$. If $i > 0$, let $a'_i$ be the endpoint of path $P_i^*$ that lies in $T_3$, and if $i < \lambda'$, let $a'_{i+1}$ be the endpoint of path $P_{i'+1}$ that lies in $T_3$. Note that, if $a'$ lies before $a'_i$ in the ordering $\tilde{\mathcal{O}}_3$ of $T_3$, or if $a'$ lies after $a'_{i+1}$ in the ordering $\tilde{\mathcal{O}}_3$, then the image of $P$ has to intersect the interior of an additional horizontal strip, and, from Observation 11.19, path $P$ must lie in $\mathcal{P}'$. We discard each such path from set $\mathcal{P}_1$ (and from the corresponding set $U_i$). This ensures that, if $P \in U_i$, then its endpoint $a'$ must lie between $a'_i$ and $a'_{i+1}$ in $\tilde{\mathcal{O}}_3$, if $1 \leq i \leq \lambda'$; it must lie before $a'_{i+1}$ if $i = 0$, and it must lie after $a'_i$ if $i = \lambda'$. We process the paths in $\mathcal{R}_1$ similarly, discarding paths as needed. Notice that so far all paths that we have discarded from $\mathcal{P}_1 \cup \mathcal{R}_1$ lie in $\mathcal{P}' \cup \mathcal{R}'$.

### 11.4.1 In-Order Intersection

In this stage we discard some additional paths from $\mathcal{P}_1 \cup \mathcal{R}_1$, to ensure that every remaining path in $\mathcal{P}_1$ interesects the paths in $\mathcal{R}^*$ in-order (notion that we define below); we do the same for paths in $\mathcal{R}_1$. We will ensure that all paths discarded at this stage lie in $\mathcal{P}' \cup \mathcal{R}'$.

Since the definitions and the algorithms for the paths in $\mathcal{P}_1$ and for the paths in $\mathcal{R}_1$ are symmetric, we only describe the algorithm to process the paths in $\mathcal{P}_1$ here.

Let $P \in \mathcal{P}_1$ be any path, that we view as directed from its endpoint that lies in $T_1$ to its endpoint lying in $T_3$. Let $X(P) = \{x_1, \ldots, x_r\}$ denote all vertices of $P$ lying on paths in $R^*$, that is, $X(P) = V(P) \cap \left(\bigcup_{i=1}^{\lambda'} V(R_i^*)\right)$. We assume that the vertices of $X(P)$ are indexed in the order of their appearance on $P$. For each such vertex $x_j$, let $i_j$ be the index of the path $R_{i_j}^* \in \mathcal{R}^*$ containing $x_j$.

**Definition 11.20 (In-order intersection)** *We say that path $P$ intersects the paths of $\mathcal{R}^*$ in-order, if $r \geq \lambda'$, $i_1 = 1$, $i_r = \lambda'$, and, for $1 \leq j < r$, $|i_j - i_{j+1}| \leq 1$.*

Notice that the definition requires that path $P$ intersects every path of $\mathcal{R}^*$ at least once; the first path of $\mathcal{R}^*$ that it intersects must be $R_1^*$, and the last path must be $R_{\lambda'}^*$, and for every consecutive pair $x_j, x_{j+1}$ of vertices in $X(P)$, either both vertices lie on the same path of $\mathcal{R}^*$, or they lie on consecutive paths of $\mathcal{R}^*$. Notice that path $P$ is still allowed to intersect a path of $\mathcal{R}^*$ many times, and may go back and forth across all these paths several times.

**Observation 11.21** *Assume that Event $\mathcal{E}_1$ did not happen. Let $P \in \mathcal{P}_1$ be a path that intersect the paths of $\mathcal{R}^*$ not in-order. Then $P \in \mathcal{P}'$ must hold.*

**Proof:** Assume first that $i_1 \neq 1$, that is, vertex $x_1$ lies on some path $R_i^*$ with $i \neq 1$. Let $p$ be a point on the image of path $P$ in $\varphi^*$ that is very close to its first endpoint, so $p$ lies in the interior of the vertical strip $\mathsf{VStrip}_1$, and let $p'$ be the image of the point $x_1$. Clearly, $p'$ does not lie in the interior or on the boundary of $\mathsf{VStrip}_1$, so the image of path $P$ must cross the right boundary of $\mathsf{VStrip}_1$, which means that the image of some edge of $P$ and the image of some edge of $R_1^*$ cross in $\varphi^*$.

The cases where $i_r \neq \lambda'$, or there is an index $1 \leq j < r$ with $i_j - i_{j+1} > 1$ are treated similarly, as is the case when $r < \lambda'$. □

We discard from $\mathcal{P}_1$ all paths $P$ that intersect the paths of $\mathcal{R}^*$ not in-order. We denote by $\mathcal{P}_2 \subseteq \mathcal{P}_1$ the set of remaining paths. We also update the groups $\mathcal{U}_0, \ldots, \mathcal{U}_{\lambda'}$ accordingly. Observe that so far all paths that we have discarded from $\mathcal{P}_1$ lie in $\mathcal{P}'$. From Observation 11.18, assuming that Events $\mathcal{E}_1$ and $\mathcal{E}_2$ did not happen, the number of paths that we have discarded so far from $\mathcal{P}_1$ is at most $\psi/32$. In particular, for all $0 \leq i \leq \lambda'$, $|\mathcal{U}_i| \geq 31\psi/32$ still holds.

We perform the same transformation on set $\mathcal{R}_1$ of paths, obtaining a new set $\mathcal{R}_2$ of paths, each of which intersects the paths of $\mathcal{P}^*$ in-order. We also update the groups $\mathcal{U}'_0, \ldots, \mathcal{U}'_{\lambda'}$. As before, for all $0 \leq i \leq \lambda'$, $|\mathcal{U}'_i| \geq 31\psi/32$ still holds.

### 11.4.2 Definining Paths Associated with Grid Cells

For every path $P \in \mathcal{P}_2$, for all $1 \leq i \leq \lambda'$, we denote by $v_i(P)$ the first vertex on path $P$ that belongs to the vertical path $R^*_i$; note that, from the definition of in-order intersection, such a vertex must exist. For all $1 \leq i < \lambda'$, we define the $i$th segment of $P$, $\sigma_i(P)$, to be the subpath of $P$ between $v_i(P)$ and $v_{i+1}(P)$. We also let $\sigma_0(P)$ be the subpath of $P$ from its first vertex (which must be a terminal of $T_1$) to $v_1(P)$, and by $\sigma_{\lambda'}(P)$ the subpath of $P$ from $v_{\lambda'}(P)$ to the last vertex of $P$ (which must be a terminal of $T_3$). Note that the sets of edges that lie on paths $\sigma_0(P), \ldots, \sigma_{\lambda'}(P)$ partition $E(P)$.

Similarly, for every path $R \in \mathcal{R}_2$, for all $1 \leq i \leq \lambda'$, we denote by $v_i(R)$ the first vertex on path $R$ that lies on the horizontal path $P^*_i$. For all $1 \leq i < \lambda'$, we define the $i$th segment of $P$, $\sigma_i(R)$, to be the subpath of $R$ between $v_i(R)$ and $v_{i+1}(R)$. We also let $\sigma_0(R)$ be the subpath of $R$ from its first vertex (which must be a terminal of $T_4$) to $v_1(R)$, and by $\sigma_{\lambda'}(R)$ the subpath of $R$ from $v_{\lambda'}(R)$ to the last vertex of $R$ (which must be a terminal of $T_2$).

Consider now some cell $\mathsf{Cell}_{i,j}$ of the grid $\Pi^*$, for some $0 \leq i, j \leq \lambda'$. We define the set $\mathcal{P}^{i,j}$ of horizontal paths, and the set $\mathcal{R}^{i,j}$ of vertical paths associated with cell $\mathsf{Cell}_{i,j}$, as follows. In order to define the set $\mathcal{P}^{i,j}$ of horizontal paths, we consider the group $\mathcal{U}_i \subseteq \mathcal{P}_2$, and, for every path $P \in \mathcal{U}_i$, we include its $j$th segment $\sigma_j(P)$ in $\mathcal{P}^{i,j}$, so $\mathcal{P}^{i,j} = \{\sigma_j(P) \mid P \in \mathcal{U}_i\}$. Similarly, we define $\mathcal{R}^{i,j} = \left\{\sigma_i(R) \mid R \in \mathcal{U}'_j\right\}$. We need the following observation.

**Observation 11.22** *Let $P \in \mathcal{U}_i, R \in \mathcal{U}'_j$ be a pair of paths, for some $1 < i, j < \lambda'$, and assume that their subpaths $\sigma_j(P) \subseteq P, \sigma_i(R) \subseteq R$ do not share any vertices. Then either $P \in \mathcal{P}'$, or $R \in \mathcal{R}'$, or the images of $\sigma_j(P)$ and $\sigma_i(R)$ cross in the drawing $\varphi^*$.*

**Proof:** Assume that $P \notin \mathcal{P}'$ and $R \notin \mathcal{R}'$, that is, the images of the paths $P, R$ do not cross the images of the paths in $\mathcal{P}^* \cup \mathcal{R}^*$ in $\varphi^*$. From the definition of set $\mathcal{U}_i$, the image of $P$ intersects the interior of the horizontal strip $\mathsf{HStrip}_i$, and path $P$ does not share any vertices with the paths of $\mathcal{P}^*$. Therefore, the image of $P$ must be contained in the strip $\mathsf{HStrip}_i$, and it is disjoint from its top and bottom boundaries $\gamma^t_i, \gamma^b_{i+1}$. Using similar reasoning, the image of $R$ is contained in the strip $\mathsf{VStrip}_j$, and it is disjoint from its left and right boundaries, $\gamma^r_j, \gamma^\ell_{j+1}$. Consider now the segment $\sigma_j(P)$ of $P$, whose endpoints lie on $R^*_j$ and $R^*_{j+1}$, respectively. Let $\sigma'_j(P) \subseteq \sigma_j(P)$ be the shortest subpath of $\sigma_j(P)$ whose first endpoint lies on $R^*_j$, and whose last endpoint lies on $R^*_{j+1}$; such a path must exist because we can let $\sigma'_j(P) = \sigma_j(P)$. From the definition of in-order intersection, no inner vertex of $\sigma'_j$ may lie on any path of $\mathcal{R}^*$. It is then easy to verify that the image of $\sigma'_j(P)$ in $\varphi^*$ must be contained in $\mathsf{CellRegion}_{i,j}$, and it must split this region into two subregions: one whose top boundary contains a segment of $\gamma^b_{i+1}$, and one whose bottom boundary contains a segment of $\gamma^t_i$.

Using the same reasoning, we can select a segment $\sigma'_i(R)$, whose first endpoint lies on $P^*_i$, last endpoint lies on $P^*_{i+1}$, and all inner vertices are disjoint from the vertices lying on the paths in $\mathcal{P}^*$. As before, the image of $\sigma'(R)$ must be contained in $\mathsf{CellRegion}_{i,j}$, but it connects a point on its top boundary to a point on its bottom boundary. Therefore, the image of $\sigma'_i(R)$ must cross the image of $\sigma'_j(P)$. □

### 11.4.3 Completing the Construction of the Grid-Like Structure

In order to complete the construction of the grid-like structure, we need to ensure that, for every pair $1 < i, j < \lambda'$ of indices, there are many intersection between the sets $\mathcal{P}^{i,j}$ and $\mathcal{R}^{i,j}$ of paths. More specifically, we need to ensure that every path $\sigma \in \mathcal{P}^{i,j}$ intersects many paths in $\mathcal{R}^{i,j}$, and vice versa. This is needed in order to ensure well-linkedness properties: namely, that the collection of vertices containing the first and the last vertex on every path of $\mathcal{P}^{i,j}$ is sufficiently well-linked in the graph obtained from the union of the paths in $\mathcal{R}^{i,j} \cup \mathcal{P}^{i,j}$. This property, in turn, will be exploited in order to construct the routers of $\Lambda'$ over which the distribution $\mathcal{D}$ will be defined. This motivates the following definition.

**Definition 11.23 (Bad Paths)** *For a pair $0 < i, j < \lambda'$ of indices, we say that a path $P \in \mathcal{U}_i$ is* bad for cell $\mathsf{Cell}_{i,j}$ *if there are at least $\psi/16$ paths in $\mathcal{R}^{i,j}$ that are disjoint from $\sigma_j(P)$. Similarly, we say that a path $R \in \mathcal{U}'_j$ is bad for cell $\mathsf{Cell}_{i,j}$ if there are at least $\psi/16$ paths in $\mathcal{P}^{i,j}$ that are disjoint from $\sigma_i(R)$.*

*Consider now some index $0 < i < \lambda'$. We say that a path $P \in \mathcal{U}_i$ is* bad *if it is bad for at least one cell in $\{\mathsf{Cell}_{i,j} \mid 0 < j < \lambda'\}$. Similarly, for an index $0 < j < \lambda'$, a path $R \in \mathcal{U}'_j$ is bad if it is bad for at least one cell in $\{\mathsf{Cell}_{i,j} \mid 0 < i < \lambda'\}$.*

The following observation bounds the number of bad paths in each group $\mathcal{U}_i$ of horizontal paths, and in each group $\mathcal{U}'_j$ of vertical paths.

**Observation 11.24** *Assume that $\mathsf{cr}^* \leq \frac{\tilde{k}^2}{c_2\eta^5}$, and that neither of the events $\mathcal{E}_1, \mathcal{E}_2$ happenned. Then for all $0 < i < \lambda'$, at most $\psi/16$ paths in $\mathcal{U}_i$ are bad. Similarly, for all $0 < j < \lambda'$, at most $\psi/16$ paths in $\mathcal{U}'_j$ are bad.*

**Proof:** Fix an index $0 < i < \lambda'$, and the corresponding set $\mathcal{U}_i \subseteq \mathcal{P}_2$ of paths. We partition the set of all bad paths in $\mathcal{U}_i$ into two subsets: set $\mathcal{B}_1$ contains all bad paths lying in $\mathcal{P}'$, and set $\mathcal{B}_2$ contains all remaining bad paths. From Observation 11.18, $|\mathcal{B}_1| \leq \psi/32$.

We further partition the set $\mathcal{B}_2$ of bad paths into subsets $\left\{\mathcal{B}_2^j \mid 0 < j < \lambda'\right\}$, where a path $P$ lies in $\mathcal{B}_2^j$ if it is bad for cell $\mathsf{Cell}_{i,j}$ (if path $P$ is bad for several cells, we add it to any of the corresponding sets). Consider now some index $0 < j < \lambda'$, and some path $P \in \mathcal{B}_2^j$. From the definition, there is a set $\Sigma' \subseteq \mathcal{R}^{i,j}$ of at least $\psi/16$ paths that do not share any vertices with $P$. From Observation 11.18, at most $\psi/32$ of these paths may lie in $\mathcal{R}'$. Let $\Sigma'' \subseteq \Sigma'$ be the collection of the remaining paths, whose cardinality is at least $\psi/32$. From Observation 11.22, for every path $\sigma' \in \Sigma'$, the images of $\sigma_j(P)$, and of $\sigma'$ must cross. We let $\chi_j(P)$ denote the set of all crossings $(e, e')$, where $e \in \sigma_j(P)$, and $e'$ is an edge on a path of $\Sigma''$, so $|\chi_j(P)| \geq \psi/32$. We then let $\chi_j = \bigcup_{P \in \mathcal{B}_2^j} \chi_j(P)$, so $|\chi_j| \geq |\mathcal{B}_2^j| \cdot \psi/32$. Lastly, we let $\chi = \bigcup_{j=1}^{\lambda'-1} \chi_j$. Notice that set $\chi$ contains at least $|\mathcal{B}_2| \cdot \psi/32$ distinct crossings in the drawing $\varphi^*$. Assume for contradiction that $|\mathcal{B}_2| > \psi/32$. Then:

$$\mathsf{cr}^* > \frac{\psi^2}{2^{10}} > \frac{\tilde{k}^2}{4\eta^4} > \frac{\tilde{k}^2}{c_2\eta^5},$$

since $\psi > \frac{16\tilde{k}}{\eta^2}$, a contradiction. Therefore, $|\mathcal{B}_2| \leq \psi/32$, and overall there are at most $\psi/16$ bad paths in $\mathcal{U}_i$. The proof for path sets $\mathcal{U}'_j \subseteq \mathcal{R}_2$ is identical. □

For all $0 < i < \lambda'$, we discard every bad path from $\mathcal{U}_i$. If $|\mathcal{U}_i| < \lceil 7\psi/8 \rceil$ for any $i$, then we terminate the algorithm and return FAIL. Notice that in this case, from Observation 11.24, if $\mathsf{cr}^* < \frac{\tilde{k}^2}{c_2\eta^5}$, then at least one of the events $\mathcal{E}_1, \mathcal{E}_2$ must have happened, and the probability for this is at most 1/8.

Therefore, we assume that for all $0 < i < \lambda'$, $|\mathcal{U}_i| \geq \lceil 7\psi/8 \rceil$ holds. We discard additional arbitrary paths from $\mathcal{U}_i$, until $|\mathcal{U}_i| = \lceil 7\psi/8 \rceil$. We then let $\mathcal{P}_3 = \bigcup_{i=1}^{\lambda'} \mathcal{U}_i$ denote the resulting set of paths.

Similarly, for all $0 < j < \lambda'$, we discard every bad path from $\mathcal{U}'_j$. If, as the result, $|\mathcal{U}'_j|$ falls below $\lceil 7\psi/8 \rceil$, we terminate the algorithm and return FAIL. Otherwise, we discard additional arbitrary paths as needed, so that $|\mathcal{U}'_j| = \lceil 7\psi/8 \rceil$ holds. We also let $\mathcal{R}_3 = \bigcup_{j=1}^{\lambda'} \mathcal{U}'_j$.

For all $0 < i, j < \lambda'$, we also update the path sets $\mathcal{P}^{i,j}$ and $\mathcal{R}^{i,j}$ accordingly, discarding the paths that are no longer subpaths of paths in $\mathcal{P}_3 \cup \mathcal{R}_3$. Since we are still guaranteed that $|\mathcal{P}^{i,j}|, |\mathcal{R}^{i,j}| = \lceil 7\psi/8 \rceil$, and since every path that is bad for cell $\mathsf{Cell}_{i,j}$ was discarded, we are guaranteed that every path in $\mathcal{P}^{i,j}$ intersects at least $\frac{7\psi}{8} - \frac{\psi}{16} = \frac{13\psi}{16}$ paths of $\mathcal{R}^{i,j}$ and vice versa. Since we use this fact later, we summarize it in the following observation.

**Observation 11.25** *For all $0 < i, j < \lambda'$, $|\mathcal{P}^{i,j}|, |\mathcal{R}^{i,j}| = \lceil 7\psi/8 \rceil$. Every path in $\mathcal{P}^{i,j}$ intersects at least $\frac{13\psi}{16}$ paths of $\mathcal{R}^{i,j}$ and vice versa.*

This concludes the construction of the grid-like structure.

## 11.5 Step 5: the Routing

Recall that we have denoted by $\Lambda' = \Lambda(H'', \tilde{T})$ the set of all routers in graph $H''$ with respect to the set $\tilde{T}$ of terminals. In this final step we design an efficient algorithm to compute an explicit distribution $\mathcal{D}$ over the routers of $\Lambda'$, such that for every outer edge $e \in E(H'')$, $\mathbf{E}_{\mathcal{Q} \sim \mathcal{D}} \left[ (\mathrm{cong}_{H''}(\mathcal{Q}, e))^2 \right] \leq O\left( \frac{\log^{16} m}{(\alpha\alpha')^4} \right)$.

Our algorithm closely follows the arguments of [Sid10] (see also Lemma D.10 in the full version of [Chu11]), who showed a similar result for a grid graph. In order to provide intuition, we first present their algorithm. Assume that we are given a $(q \times q)$ grid graph $G$ for some integer $q$, and let $T$ be the set of vertices lying on the first row of the grid, that we refer to as terminals. For convenience, assume that $q$ is an integral power of 2. Our goal is to compute a distribution $\mathcal{D}'$ over the routers of in $\Lambda(G, T)$. We need to ensure that, for every edge $e \in E(G)$, the expectation $\mathbf{E}_{\mathcal{Q} \sim \mathcal{D}'} \left[ (\mathrm{cong}_G(\mathcal{Q}, e))^2 \right] \leq O(\log q)$.

For every vertex $v$ in the top right quadrant of the grid, we will define a set $\mathcal{Q}(v)$ of paths in $G$, routing the terminals in $T$ to $v$. Our distribution $\mathcal{D}$ then assigns, to each such router $\mathcal{Q}(v)$, the same probability value $4/q^2$.

We now fix a vertex $v$ in the top right quadrant of the grid, and define the router $\mathcal{Q}(v)$. Let $r = \log(q/4)$. For $0 \leq i \leq r$, let $S_i$ be a square subgrid of $G$, of size $(2^i \times 2^i)$, whose upper right corner has the same column-index as vertex $v$, and the same row-index as the bottom left corner of $S_{i-1}$ (we think of $S_0$ as a $(1 \times 1)$-grid consisting only of vertex $v$). We refer to the subgrids $S_i$ of $G$ as *squares*, and specifically to square $S_i$ as *level-i square*. For all $0 \leq i \leq r$, we denote by $T_i$ the set of vertices lying on the bottom boundary of square $S_i$. Using the well-linkedness of the grids, it is easy to show that for all $1 \leq i \leq r$, there is a collection $\mathcal{P}_i$ of paths in graph $S_i$, routing vertices of $T_i$ to vertices of $T_{i-1}$ with congestion at most 2, such that every vertex of $T_{i-1}$ serves as endpoint of at most two such paths. For $1 \leq i \leq r$, let Let $\mathcal{P}'_i$ be a multipset obtained from set $\mathcal{P}_i$ by creating $2^{r-i+1}$ copies of every path in $\mathcal{P}_i$. Let $T_{r+1} \subseteq T$ be a set of $|T_r|$ vertices lying on the bottom boundary of the grid $G$, that contains, for every vertex $t \in T_r$, vertex $t'$ on the bottom boundary of the grid with the same column index as $t$. Let $P_t$ be the subpath of the corresponding column of $G$ connecting $t$ to $t'$, and denote $\mathcal{P}'_{r+1} = \{P_t \mid t \in T_r\}$.

By concatenating the paths in $\mathcal{P}'_1, \ldots, \mathcal{P}'_{r+1}$, we obtain a collection $\mathcal{Q}'(v)$ of paths in grid $G$, routing the terminals in $T_{r+1}$ to vertex $v$. Notice that for all $0 \leq i \leq r$, for every edge $e$ lying in $S_i$, the congestion on edge $e$ due to paths in $\mathcal{Q}'(v)$ is at most $2^{r-i+2}$. The key in analyzing the expectation $\mathbf{E}_{\mathcal{Q} \sim \mathcal{D}'} \left[ (\mathrm{cong}_G(\mathcal{Q}, e))^2 \right]$ is to notice that, for all $1 \leq i \leq r$, square $S_i$ is a $(2^i \times 2^i)$-subgrid of $G$, whose

upper right corner is chosen uniformly at random from a set of $q^2/4$ possible points. The total number of subgrids of $G$ of size $(2^i \times 2^i)$ that contain $e$ is $2^{2i}$, so the probability that any of them is selected is bounded by $2^{2i+2}/q^2$. Therefore, for all $1 \leq i \leq r$, with probability at most $2^{2i+2-2r}$, edge $e$ belongs to square $S_i$, and in this case, $\mathrm{cong}_G(\mathcal{Q}, e) \leq 2^{r-i+2}$. Therefore, we get that:

$$\mathbf{E}_{\mathcal{Q}\sim\mathcal{D}'}\left[(\mathrm{cong}_G(\mathcal{Q}, e))^2\right] \leq \sum_{i=1}^{r} 2^{2i+2-2r} \cdot 2^{2r-2i+4} \leq O(r) = O(\log q).$$

Using the well-linkedness of the terminals in $T$, it is immediate to extend the set $\mathcal{Q}'(v)$ of paths to a set $\mathcal{Q}^*(v)$ routing all terminals in $T$ to $v$, while increasing the congestion on every edge of $G$ by at most an additive constant and a multiplicative constant factor. This provides the final distribution $\mathcal{D}$ over the routers $\mathcal{Q}(v) \in \Lambda(G, T)$.

We will simulate a similar process on the grid $\Pi^*$, and its corresponding grid-like structure that we have constructed. Notice however that $\Pi^*$ is only a $(\lambda' \times \lambda')$-grid (where $\eta \leq \lambda' \leq \eta^2$), while the number of terminals that we need to route is much larger (comparable to $|\mathcal{R}_3|$). Therefore, we will attempt to route all terminals to a single cell $\mathsf{Cell}_{i,j}$ in the top right quadrant of the grid $\Pi^*$ (in other words, we will route them to vertices lying on paths in $\mathcal{P}^{i,j} \cup \mathcal{R}^{i,j}$). This in itself is not sufficient, since we need to route them to a single vertex of $H''$. This means that we may need to perform some routing within the cell $\mathsf{Cell}_{i,j}$, that is, within the graph obtained from the union of the paths in $\mathcal{P}^{i,j} \cup \mathcal{R}^{i,j}$. While generally such a routing (with low congestion on outer edges) may be difficult to compute, we will select a large collection of cells (called good cells) in the top right quadrant of the grid $\Pi^*$, for which such a routing is easy to obtain. We will then define, for each good cell, the corresponding set of paths routing the terminals to a single vertex $y^* \in V(H'')$. We do so by simulating the process described above: we define square subgrids $\{S_i\}$ of the grid $\Pi^*$, and we associate these subgrids with sets of horizontal and vertical paths (subpaths of some paths in $\mathcal{P}_3 \cup \mathcal{R}_3$), so that the desired well-linkedness properties of graphs corresponding to each subgrid $S_i$ are achieved. Eventually, the distribution $\mathcal{D}$ chooses one of the good cells uniformly at random, and uses the associated router $\mathcal{Q} \in \Lambda'$ in order to route the terminals to a single vertex of $H''$. The analysis of expected congestion squared on every outer edge of $H''$ is very similar to the one outlined above.

We start by defining the notion of good cells of the grid $\Pi^*$, and showing that a large enough number of such cells exist in the upper right quadrant of $\Pi^*$. We will then define square subgrids of $\Pi^*$ and associate sets of paths with each such subgrid to ensure the required well-linkedness properties. Lastly, we show how to construct the desired routing $\mathcal{Q}$ for each good cell.

### 11.5.1 Good Cells

Fix a pair of indices $0 < i, j < \lambda'$, and consider the cell $\mathsf{Cell}_{i,j}$ of the grid $\Pi^*$, and the two corresponding sets $\mathcal{P}^{i,j}$, $\mathcal{R}^{i,j}$ of paths.

**Definition 11.26 (Good cells)** *A path $\sigma \in \mathcal{P}^{i,j}$ is* good *for cell $\mathsf{Cell}_{i,j}$ if $\sigma$ contains no outer edges. We say that cell $\mathsf{Cell}_{i,j}$ is* good *if some path $\sigma \in \mathcal{P}^{i,j}$ is good for $\mathsf{Cell}_{i,j}$; otherwise we say it is* bad.

Assume that cell $\mathsf{Cell}_{i,j}$ is good, and let $\sigma \in \mathcal{P}^{i,j}$ be any horizontal path that is good for this cell. Since $\sigma$ contains no outer edges, there must be a vertex $y \in V(H)$, such that $V(\sigma) \subseteq V(\Pi(y))$. Recall that, from Observation 11.25, $|\mathcal{R}^{i,j}| = \lceil 7\psi/8 \rceil$, and that $\sigma$ intersects at least $13\psi/16$ paths of $\mathcal{R}^{i,j}$. Let $\hat{\mathcal{R}}^{i,j} \subseteq \mathcal{R}^{i,j}$ be a set of $\lceil 13\psi/16 \rceil$ paths, each of which shares at least one vertex with $\sigma$. Note that each such path then must contain a vertex of $\Pi(y)$. We denote by $\mathsf{Portals}^{i,j}$ the set of vertices that contains, for every path $\sigma' \in \hat{\mathcal{R}}^{i,j}$, the first vertex of $\sigma'$ (by definition, each such vertex must lie on path $P_i^*$). For convenience, we denote vertex $y$ of $H$ by $y_{i,j}$.

Let $Z$ be the set of all pairs of indices $\lfloor \lambda'/2 \rfloor \leq i,j < \lambda'$, such that $\mathsf{Cell}_{i,j}$ is good. Next, we show that $|Z|$ is sufficiently large. Our routing algorithm will then choose a pair $(i,j)$ of indices from $Z$ uniformly at random, and route the terminals to the vertices in set $\mathsf{Portals}^{i,j}$, from where they will be routed to vertices of $\Pi(y_{i,j})$, and eventually to some specific vertex of $\Pi(y_{i,j})$.

**Claim 11.27** $|Z| \geq (\lambda')^2/16$.

**Proof:** Let $\mathcal{B}$ be a collection of all bad cells $\mathsf{Cell}_{i,j}$ lying in the top right quadrant, that is, $\lfloor \lambda'/2 \rfloor \leq i,j < \lambda'$. It is enough to show that $|\mathcal{B}| < (\lambda')^2/16$.

Consider now some bad cell $\mathsf{Cell}_{i,j} \in \mathcal{B}$, and any path $Q \in \mathcal{P}^{i,j}$. Since cell $\mathsf{Cell}_{i,j}$ is bad, $Q$ must contain at least one outer edge. We say that $Q$ is a *type-1* bad path for cell $\mathsf{Cell}_{i,j}$ if it contains at least one type-1 outer edge (recall that a type-1 outer edge $e$ corresponds to some edge in graph $H$ that is **not** contained in any cluster of $\mathcal{C}$). Otherwise, every outer edge on path $Q$ is a type-2 outer edge, and in this case we say that $Q$ is a type-2 bad cluster for $\mathsf{Cell}_{i,j}$. We say that cell $\mathsf{Cell}_{i,j}$ is *type-1 bad* if at least $\psi/32$ paths of $\mathcal{P}^{i,j}$ are type-1 bad for this cell, and otherwise it is type-2 bad. We partition the set $\mathcal{B}$ of bad cells into two subsets: set $\mathcal{B}_1$ contains all type-1 bad cells, and set $\mathcal{B}_2$ contains all type-2 bad cells. It is now enough to prove that $|\mathcal{B}_1|, |\mathcal{B}_2| < (\lambda')^2/32$, which we do in the following two observations.

**Observation 11.28** $|\mathcal{B}_1| < (\lambda')^2/32$.

**Proof:** Assume for contradiction that $|\mathcal{B}_1| \geq (\lambda')^2/32$. Consider a type-1 bad cell $\mathsf{Cell}_{i,j} \in \mathcal{B}_1$, and let $\mathcal{Q}^{i,j} \subseteq \mathcal{P}^{i,j}$ be a set of $\lceil \psi/32 \rceil$ paths that are type-1 bad paths for cell $\mathsf{Cell}_{i,j}$. Each path in $\mathcal{Q}^{i,j}$ must contain at least one type-1 bad edge. Since the paths in $\mathcal{Q}^{i,j}$ are edge-disjoint, there is a set $E^{i,j}$ of at least $\psi/32$ type-1 outer edges of $H''$, lying on paths of $\mathcal{Q}^{i,j}$. Since every edge of $H''$ may lie on at most one path in $\mathcal{P}$, the total number of outer edges in $H''$ must be at least:

$$\frac{|\mathcal{B}_1| \cdot \psi}{32} \geq \frac{(\lambda')^2 \cdot \psi}{2^{10}} \geq \frac{\lambda^2 \cdot \psi}{2^{14}},$$

as $\lambda' = \lfloor (\lambda-1)/2 \rfloor \geq \lambda/4$. Recall that $\psi = \left\lfloor \frac{\alpha^* \tilde{k}}{64\lambda} \right\rfloor$ and $\lambda = \frac{2^{24} c \cdot \eta \log^8 m}{\alpha^* \alpha^3}$, where $c$ is the constant from Observation 11.13. Therefore, we get that the total number of outer edges in $H''$ is at least $\frac{2c\tilde{k}\eta \log^8 m}{\alpha^3}$, contradicting Observation 11.13. □

**Observation 11.29** $|\mathcal{B}_2| < (\lambda')^2/32$.

**Proof:** For a cluster $C \in \mathcal{C}$, let $X(C) = \bigcup_{y \in V(C)} V(\Pi(y))$. Note that all terminals of $H$ lie outside of the clusters in $\mathcal{C}$, and so $X(C) \cap \tilde{T} = \emptyset$. If a path $Q \in \mathcal{P}_3 \cup \mathcal{R}_3$ contains a vertex of $X(C)$, then it must contain at least one edge of $\delta_H(C)$. As the paths in $\mathcal{P} \cup \mathcal{R}$ cause edge-congestion at most 2, the total number of paths $Q \in \mathcal{P} \cup \mathcal{R}$ with a non-empty intersection with $X(C)$ is at most $2\delta_H(C)$.

Let $\mathsf{IntPairs} \subseteq \mathcal{P}_3 \times \mathcal{R}_3$ be the collection of all pairs of paths $P \in \mathcal{P}_3$, $R \in \mathcal{R}_3$, such that $P$ and $R$ share at least one vertex. For a cluster $C \in \mathcal{C}$, let $\mathsf{IntPairs}'_C \subseteq \mathsf{IntPairs}$ denote the collection of all pairs $(P,R) \in \mathsf{IntPairs}$ of paths, such that some vertex $v \in X(C)$ lies on both $P$ and $R$. Clearly, if $(P,R) \in \mathsf{IntPairs}'_C$, then each of the paths $P$, $R$ must contain at least one edge of $\delta_H(C)$. Therefore, from the above discussion, $|\mathsf{IntPairs}'_C| \leq 4|\delta_H(C)|^2$. Let $\mathsf{IntPairs}' = \bigcup_{C \in \mathcal{C}_Y} \mathsf{IntPairs}'_C$. Then:

$$|\mathsf{IntPairs}'| \leq \sum_{C \in \mathcal{C}_Y} |\mathsf{IntPairs}'_C| \leq 4 \sum_{C \in \mathcal{C}_Y} |\delta_H(C)|^2.$$

From Equation 18 (see Section 11.2.3), $\sum_{C\in\mathcal{C}_Y}|\delta_H(C)|^2 < \frac{(\tilde{k}\tilde{\alpha}\alpha')^2}{c_1\log^{20}m}$, so we get that:

$$|\mathsf{IntPairs}'| < \frac{4(\tilde{k}\tilde{\alpha}\alpha')^2}{c_1\log^{20}m}, \tag{19}$$

where $c_1$ is an arbitrarily large constant.

In the remainder of the proof, we assume for contradiction that $|\mathcal{B}_2|\geq(\lambda')^2/32$, and we will show that $|\mathsf{IntPairs}'|\geq\frac{4(\tilde{k}\tilde{\alpha}\alpha')^2}{c_1\log^{20}m}$ must hold, contradicting Equation 19.

Consider a type-2 bad cell $\mathsf{Cell}_{i,j}\in\mathcal{B}_2$. Recall that every path in $\mathcal{P}^{i,j}$ contains at least one outer edge, and at most $\psi/32$ such paths contain a type-1 bad edge. Since, from Observation 11.25, $|\mathcal{P}^{i,j}|=\lceil 7\psi/8\rceil$, there is a collection $\Sigma\subseteq\mathcal{P}^{i,j}$ of at least $3\psi/4$ paths $P$, such that all edges on $P$ are either inner edges, or type-2 outer edges. Therefore, if $P\in\Sigma$ is any such path, then there is some cluster $C\in\mathcal{C}$ with $V(P)\subseteq X(C)$. Recall that, from Observation 11.25, each path $P\in\Sigma$ intersects at least $\frac{13\psi}{16}$ paths of $\mathcal{R}^{i,j}$. Clearly, if a path $R\in\mathcal{R}^{i,j}$ intersects a path $P\in\Sigma$, then $(P,R)\in\mathsf{IntPairs}'$. Therefore, intersections between pairs of paths in $\mathcal{P}^{i,j}\times\mathcal{R}^{i,j}$ contribute at least $\frac{13\psi}{16}\cdot\frac{3\psi}{4}\geq\frac{\psi^2}{2}$ pairs to set $\mathsf{IntPairs}'$. Therfore, if we denote by $\mathsf{IntPairs}'_{i,j}$ the collection of all pairs $(P,R)\in\mathsf{IntPairs}'$, where a subpath $\sigma$ of $P$ lies in $\mathcal{P}^{i,j}$, and a subpath $\sigma'$ of $R$ lies in $\mathcal{P}^{i,j}$, and $\sigma,\sigma'$ contain a vertex $v\in X(C)$, for some cluster $C\in\mathcal{C}$, then, from the above discussion, $|\mathsf{IntPairs}'_{i,j}|\geq\frac{\psi^2}{2}$. We claim that for every pair $(P,R)\in\mathsf{IntPairs}'$ of paths, there is at most one pair of indices $0<i,j<\lambda'$, such that $(P,R)\in\mathsf{IntPairs}'_{i,j}$. Indeed, assume that $P\in\mathcal{U}_i$ and $R\in\mathcal{U}'_j$. For a pair $0<i',j'<\lambda'$ of indices, $\mathcal{P}^{i',j'}$ contains a subpath of $P$ iff $i'=i$, and $\mathcal{R}^{i',j'}$ contains a subpath of $R$ iff $j'=j$. So the only pair $(i',j')$ of indices for which $(P,R)\in\mathsf{IntPairs}'_{i',j'}$ may hold is $(i,j)$. Overall, we get that $|\mathsf{IntPairs}'|\geq|\mathcal{B}_2|\cdot\psi^2/2$. Assuming that $|\mathcal{B}_2|\geq(\lambda')^2/32$, since $\lambda'=\lfloor(\lambda-1)/2\rfloor\geq\lambda/4$, we get that $|\mathsf{IntPairs}'|\geq\frac{\lambda^2\psi^2}{1024}$.

Recall that $\psi=\left\lfloor\frac{\alpha^*\tilde{k}}{64\lambda}\right\rfloor$, and, from Observation 11.14, $\alpha^*=\Theta(\tilde{\alpha}\alpha')$. We conclude that:

$$|\mathsf{IntPairs}'|\geq\frac{(\alpha^*)^2\tilde{k}^2}{2^{22}}\geq\Omega\left((\tilde{\alpha}\alpha'\tilde{k})^2\right).$$

Since we can choose $c_1$ to be a sufficiently large constant, this contradicts Equation 19. □ □

### 11.5.2 Square Subgrids and Corresponding Sets of Paths

For integers $1\leq i,j<\lambda'$ and $\ell\leq\min\{i,j\}$, a *square subgrid* $S=S(i,j,\ell)$ *of* $\Pi^*$ (that we also refer to as a *square*) is defined as the collection of cells $\mathsf{CellSet}(S)=\left\{\mathsf{Cell}_{i',j'}\mid i-\ell+1\leq i'\leq i;\quad j-\ell+1\leq j'\leq j\right\}$. Intuitively, $S(i,j,\ell)$ is a subgrid of $\Pi^*$ of size $(\ell\times\ell)$, whose top right corner is the cell $\mathsf{Cell}_{i,j}$.

Given a square $S=S(i,j,\ell)$, we associate with it a collection $\mathcal{P}(S)$ of horizontal paths, and $\mathcal{R}(S)$ of vertical paths, as follows. Intuitively, consider the graph obtained by taking the union of all paths $\mathcal{P}^{i',j'}$, where $\mathsf{Cell}_{i',j'}\in\mathsf{CellSet}(S)$. This graph is a collection of disjoint paths, each of which is a subpath of a distinct path in $\bigcup_{i'=i-\ell+1}^{i}\mathcal{U}_{i'}$; we let $\mathcal{P}(S)$ be this set of paths. Formally, for all $i-\ell+1\leq i'\leq i$, for every path $P\in\mathcal{U}_{i'}$, we include in $\mathcal{P}(S)$ the subpath of $P$ from the first vertex of $\sigma_{j-\ell+1}(P)$ to the last vertex of $\sigma_j(P)$. Similarly, set $\mathcal{R}(S)$ contains, for all $j-\ell+1\leq j'\leq j$, for every path $R\in\mathcal{U}'_{j'}$, the subpath of $R$ from the first vertex of $\sigma'_{i-\ell+1}(R)$ to the last vertex of $\sigma'_i(R)$. Notice that, from Observation 11.25, $|\mathcal{P}(S)|=|\mathcal{R}(S)|=\lceil 7\psi/8\rceil\cdot\ell$.

We denote by $\mathsf{EntryPortals}(S)$ the set of all vertices that serve as the first endpoint of the paths in $\mathcal{P}(S)$, and by $\mathsf{ExitPortals}(S)$ the set of all vertices that serve as the last endpoint of the paths in $\mathcal{P}(S)$. We denote by $G(S)$ the graph obtained by the union of the paths in $\mathcal{P}(S)\cup\mathcal{R}(S)$.

The following claim will be crucial for our algorithm for computing the routing paths for each good cell.

**Claim 11.30** *Let $S = S(i,j,\ell)$ be a square of $\Pi^*$, for some $1 \leq i,j < \lambda'$ and $\ell \leq \min\{i,j\}$, and let $Y \subseteq \mathsf{EntryPortals}(S)$, $Y' \subseteq \mathsf{ExitPortals}(S)$ be two subsets of vertices of cardinality $z$ each, where $z \leq \psi\ell/2$. Then there is a collection $\mathcal{Q}$ of edge-disjoint paths in graph $G(S)$, which is a one-to-one routing from $Y$ to $Y'$.*

**Proof:** Assume for contradiction that the claim is false. Then, from the maximum flow / minimum cut theorem, there is a collection $E'$ of at most $z-1$ edges in graph $G(S)$, such that $G(S) \setminus E'$ contains no path connecting a vertex of $Y$ to a vertex of $Y'$. Recall that each vertex of $Y$ is an endpoint of a distinct path in $\mathcal{P}(S)$, and all paths in $\mathcal{P}(S)$ are edge-disjoint. Since $|Y| = z$, while $|E'| \leq z-1$, there is some path $P \in \mathcal{P}(S)$, whose endpoint $y$ belongs to $Y$, such that $P$ contains no edge of $E'$. Using the same arguments, there is some path $P' \in \mathcal{P}(S)$, whose endpoint $y'$ belongs to $Y'$, that contains no edge of $E'$. Clearly, $P \neq P'$ must hold, as otherwise there is a path in $G(S) \setminus E'$ connecting $y$ to $y'$ – the path $P$. It is now enough to show that there is some path $R \in \mathcal{R}(S)$, that contains no edge of $E'$, but $R \cap P \neq \emptyset$ and $R \cap P' \neq \emptyset$ hold. Indeed, in this case, $P \cup R \cup P' \subseteq G(S) \setminus E'$, and so $y$ remains connected to $y'$ in $G(S) \setminus E'$.

We now show that path $R$ with such properties must exist. Let $\tilde{P} \in \mathcal{P}_3$ be the path with $P \subseteq \tilde{P}$, and assume that $\tilde{P} \in \mathcal{U}_{i'}$ Similarly, let $\tilde{P}' \in \mathcal{P}_3$ be the path with $P' \subseteq \tilde{P}'$, and assume that $\tilde{P}' \in \mathcal{U}_{i''}$ (where possibly $i' = i''$). Consider some index $j - \ell + 1 \leq j' \leq \ell$. Recall $|\mathcal{U}'_{j'}| = \lceil 7\psi/8 \rceil$, and, for every path $R \in \mathcal{U}'_{j'}$, segment $\sigma'_{i'}(R)$, lies in $\mathcal{R}^{i',j'}$, and segment $\sigma'_{i''}(R)$ lies in $\mathcal{R}^{i'',j'}$. Moreover, from Observation 11.25, path $\sigma_{j'}(\tilde{P})$ must intersect at least $\frac{13\psi}{16}$ paths of $\left\{\sigma'_{i'}(R) \mid \mathcal{U}'_{j'}\right\}$, and similarly path $\sigma_{j'}(\tilde{P}')$ must intersect at least $\frac{13\psi}{16}$ paths of $\left\{\sigma'_{i''}(R) \mid \mathcal{U}'_{j'}\right\}$. Therefore, there is a subset $\mathcal{U}''_{j'} \subseteq \mathcal{U}'_{j'}$ of at least $\psi/2$ paths $R$, such that both $P$ and $P'$ intersect the subpath of $R$ that belongs to $\mathcal{R}(S)$. Overall, there are at least $\ell\psi/2$ paths $R \in \mathcal{R}(S)$ that intersect the subpaths of $P$ and of $P'$ that lie in $\mathcal{P}(S)$. Since $z \leq \ell\psi/2$, at least one such path is disjoint from $E'$. □

### 11.5.3 Routing the Terminals to Good Cells

We fix some good cell $\mathsf{Cell}_{i,j}$ in the top right quadrant of the grid, that is, $\lfloor \lambda'/2 \rfloor \leq i,j < \lambda'$. Recall that we have defined a vertex $y_{i,j} \in V(H)$, and a collection $\hat{\mathcal{R}}^{i,j} \subseteq \mathcal{R}^{i,j}$ of $\psi' = \lceil 13\psi/16 \rceil$ paths, each of which contains a vertex of $\Pi(y_{i,j})$. We have also defined a set $\mathsf{Portals}^{i,j}$ of vertices that contains, for every path $\sigma' \in \hat{\mathcal{R}}^{i,j}$, the first vertex on $\sigma'$. Let $y^*_{i,j}$ be an arbitrary vertex of $\Pi(y_{i,j})$.

We define a set $\mathcal{Q}_{i,j}$ of paths in $H''$, routing the terminals of $\tilde{T}$ to vertex $y^*_{i,j}$, so $\mathcal{Q}_{i,j} \in \Lambda'$. In order to do so, we first define a set $\mathcal{Q}'_{i,j}$ of paths, routing a constant fraction of the terminals of $T_4$ to vertices of $\Pi(y_{i,j})$, and then extend this path set in order to obtain routing of all terminals to vertex $y^*_{i,j}$.

**Routing to $\mathsf{Cell}_{i,j}$.** In order to define the routing, we let $z = \lfloor \log(\lambda'/4) \rfloor$, and we define $z+1$ squares $S^{i,j}_0, S^{i,j}_1, \ldots, S^{i,j}_z$. In order to simplify the notation, we will omit the superscript $i,j$ for now.

Square $S_0$ is $S(i,j,1)$, so it consists of a single cell $\mathsf{Cell}_{i,j}$. We denote by $\mathsf{Portals}^{i,j}_0 = \mathsf{Portals}^{i,j}$ the set of $\psi'$ vertices that we have defined. We let $\mathcal{Q}_0$ be the set of $\psi'$ paths, containing, for every path $\sigma' \in \hat{\mathcal{R}}^{i,j}$, a subpath of $\sigma'$ between a vertex of $\mathsf{Portals}^{i,j}_0$ and a vertex of $\Pi(y_{i,j})$. Therefore, $\mathcal{Q}_0$ is a set of $\psi'$ edge-disjoint paths, routing vertices of $\mathsf{Portals}^{i,j}_0$ to vertices of $\Pi(y_{i,j})$, and all paths of $\mathcal{Q}_0$ are contained in $\mathcal{R}(S_0)$. We say that cell $\mathsf{Cell}_{i,j}$ is the bottom right corner of square $S_0$.

Fix some index $1 \leq r \leq z$, and assume that we have defined squares $S_0, \ldots, S_{r-1}$. We now define square $S_r$. We let $S_r = (i_r, j, 2^r)$, so the length of the side of the square is $2^r$, and the coordinates of the top right corner of $S_r$ are $(i_r, j)$; here, $j$ is the column index of the initial cell $\mathsf{Cell}_{i,j}$, and $i_r$ is the cell immediately under the right bottom corner cell of $S_{r-1}$. In other words, if $S_{r-1} =$

$(i_{r-1}, j, 2^{r-1})$, then $i_r = i_{r-1} + 2^{r-1}$. We assume that we have also defined a collection $\mathsf{Portals}_{r-1} \subseteq \mathsf{EntryPortals}(S_{r-1})$, containing $2^{r-1} \cdot \psi'$ vertices. Note that the top boundary of square $S_r$ appears immediately under bottom boundary of square $S_{r-1}$, so $\mathsf{EntryPortals}(S_{r-1}) \subseteq \mathsf{ExitPortals}(S_r)$, and in particular $\mathsf{Portals}_{r-1} \subseteq \mathsf{ExitPortals}(S_r)$. We select an arbitrary subset $\mathsf{Portals}_r \subseteq \mathsf{EntryPortals}(S_r)$ of $2^r \cdot \psi'$ vertices. By partitioning set $\mathsf{Portals}_r$ into two equal-cardinality subsets $Y_1, Y_2$, and applying Claim 11.30 to each of them separately, we obtain two collections $\mathcal{Q}_r^1, \mathcal{Q}_r^2$ of edge-disjoint paths in graph $G(S_r)$, routing vertex sets $Y_1$ and $Y_2$, respectively, to vertex set $\mathsf{Portals}_{r-1}$, in a one-to-one routing. Therefore, there is a set $\mathcal{Q}_r$ of paths in $G(S_r)$, routing vertex set $\mathsf{Portals}_r$ to vertex set $\mathsf{Portals}_{r-1}$ with edge-congestion at most 2, such that every vertex in $\mathsf{Portals}_{r-1}$ is an endpoint of at most two such paths. Moreover, we can compute such set $\mathcal{Q}_r$ of paths efficiently via standard maximum flow.

Lastly, consider the last square $S_z$. We define a subset $T^* \subseteq T_4$ of terminals, as follows. For every vertex $v \in \mathsf{Portals}_z$, let $R_v \in \mathcal{R}$ be the vertical path containing $v$, and let $t_v \in T_4$ be the terminal that serves as an endpoint of path $R_v$. We then let $T^* = \{t_v \mid v \in \mathsf{Portals}_z\}$, and we let $\mathcal{Q}_{z+1}$ be a set of paths containing, for every vertex $v \in \mathsf{Portals}_z$, the subpath of $R_v$ between $t_v$ and $v$. Therefore, set $\mathcal{Q}_{z+1}$ of paths routes terminals of $T^*$ to vertices of $\mathsf{Portals}_z$, and the paths in $\mathcal{Q}_{z+1}$ are edge-disjoint. It is also easy to verify that the paths in $\mathcal{Q}_{z+1}$ do not contain any edges from graphs $G(S_0) \cup \cdots \cup G(S_z)$. Note that, since $\psi = \left\lfloor \frac{\alpha^* \tilde{k}}{64\lambda} \right\rfloor$ and $\alpha^* = \Theta(\alpha\alpha'/\log^4 m)$,

$$|T^*| = 2^z \cdot \psi' = 2^{\lfloor \log(\lambda'/4) \rfloor} \cdot \lceil 13\psi/16 \rceil = \Omega(\lambda' \cdot \psi) = \Omega(\lambda\psi) = \Omega(\alpha^* \tilde{k}) = \Omega(\tilde{k}\alpha\alpha'/\log^4 m).$$

To summarize, we have defined a collection $\{S_0, \ldots, S_z\}$ of squares in the grid $\Pi^*$, where for all $0 \leq r \leq z$, square $S_r$ has dimensions $(2^r \times 2^r)$. The squares are aligned on the right, and are stacked on top of each other, with square $S_0$ containing a single cell, $\mathsf{Cell}_{i,j}$. This guarantees that all corresponding graphs $G(S_r)$ are mutually disjoint, except that, for all $0 \leq r < z$, $V(S_r) \cap V(S_{r+1}) = \mathsf{EntryPortals}(S_r)$. We have defined, for all $0 \leq r \leq z$, a set $\mathsf{Portals}_r \subseteq \mathsf{EntryPortals}(S_r)$ of $2^r \cdot \psi'$ vertices, and a set $\mathcal{Q}_r$ of paths contained in $G(S_r)$, routing vertices of $\mathsf{Portals}_r$ to vertices of $\mathsf{Portals}_{r-1}$ with edge-congestion at most 2, so that every vertex of $\mathsf{Portals}_{r-1}$ serves as an endpoint of at most two such paths. Additionally, in graph $G(S_0)$, we have defined a set $\mathcal{Q}_0$ of $\psi'$ paths routing vertices of $\mathsf{Portals}_0$ to vertices of $\Pi(y_{i,j})$, and an additional set $\mathcal{Q}_{z+1}$ of edge-disjoint paths routing terminals in $T^*$ to vertices of $\mathsf{Portals}_z$ in a one-to-one routing, so that paths in $\mathcal{Q}_{z+1}$ do not contain edges of $G(S_0) \cup \cdots \cup G(S_z)$.

We are now ready to define a set $\mathcal{Q}'_{i,j}$ of paths, routing terminals of $T^*$ to vertices of $\Pi(y_{i,j})$. In order to do so, for all $0 \leq r \leq z$, we let $\mathcal{Q}'_r$ be a multi-set of paths, contianing, for every path $\sigma' \in \mathcal{Q}_r$, $2^{z-r}$ copies of the path $\sigma'$. Therefore, paths in $\mathcal{Q}'_r$ cause edge-congestion $2^{z-r+1}$ in $G(S_r)$. Set $\mathcal{Q}'_{i,j}$ of paths is obtained by concatenating paths in sets $\mathcal{Q}_{z+1}, \mathcal{Q}'_z, \ldots, \mathcal{Q}'_0$. It is easy to verify that paths in $\mathcal{Q}'_{i,j}$ route all terminals in $T^*$ to vertices of $\Pi(y_{i,j})$.

Recall that $|T^*| = \Omega(\tilde{k}\alpha\alpha'/\log^4 m)$, and $|\tilde{T}| = \tilde{k}$. Moreover, from Observation 11.14, The set $\tilde{T}$ of terminals is $\alpha^*$-well-linked in $H''$, where $\alpha^* = \Theta(\alpha\alpha'/\log^4 m)$. From Lemma 11.1, there is a set $\mathcal{Q}_{i,j}$ of paths in graph $H''$, routing all vertices of $\tilde{T}$ to vertices of $\Pi(y_{i,j})$, such that, for every edge $e \in E(H'')$:

$$\mathrm{cong}_{H''}(\mathcal{Q}_{i,j}, e) \leq \left\lceil \frac{\tilde{k}}{|T^*|} \right\rceil \left(\mathrm{cong}_{H''}(\mathcal{Q}'_{i,j}, e) + \lceil 1/\alpha^* \rceil\right) \leq O\left(\frac{\log^4 m}{\alpha\alpha'}\right) \cdot \left(\mathrm{cong}_{H''}(\mathcal{Q}'_{i,j}, e) + \frac{\log^4 m}{\alpha\alpha'}\right).$$

**Distribution $\mathcal{D}$ and Analysis.** The final distribution $\mathcal{D}$ over the routers of $\Lambda'$ is defined as follows. For every pair $(i, j)$ of indices in $Z$, we extend the paths in set $\mathcal{Q}_{i,j}$ via the inner edges of $\Pi(y_{i,j})$ so that each such path terminates at vertex $y^*_{i,j}$, obtaining a router of $\Lambda'$. Each such resulting router $\mathcal{Q}_{i,j}$ is assigned the same distribution $1/|Z|$; recall that, from Claim 11.27, $|Z| \geq (\lambda')^2/16$.

We now fix some outer edge $e \in E(H'')$, and analyze the expectation $\mathbf{E}_{\mathcal{Q}_{i,j} \sim \mathcal{D}}\left[(\mathrm{cong}_{H''}(\mathcal{Q}_{i,j}, e))^2\right]$.

Recall that there is at most one path $P \in \mathcal{P}$ that contains $e$, and at most one path $R \in \mathcal{R}$ containing $e$. Moreover, there is at most one pair $(i_1, j_1)$ of indices with edge $e$ lying on some path of $\mathcal{P}^{i_1,j_1}$, and at most one pair $(i_2, j_2)$ of indices with edge $e$ lying on a path of $\mathcal{R}^{i_2,j_2}$.

We first focus on pair $(i_1, j_1)$ of indices, and the corresponding cell $\mathsf{Cell}_{i_1,j_1}$. Fix some pair $(i, j) \in Z$ of indices, and $0 \leq r \leq z$. If $\mathsf{Cell}_{i_1,j_1} \in S_r^{i,j}$, then segment $\sigma_{j_1}(P)$ of $P$ may lie on at most $2^{z-r+1}$ paths in $\mathcal{Q}'_{i,j}$. Notice that there are at most $2^{2r+2}$ square subgrids $S$ of $\Pi^*$ of dimension $(2^r \times 2^r)$, that contain the cell $\mathsf{Cell}_{i_1,j_1}$. For each such square $S$, there is exactly one pair $(i(S), j(S))$ of indices, for which $S_r^{i(S),j(S)} = S$. Since $|Z| \geq (\lambda')^2/16$, the probability that an index $(i, j) \in Z$ is chosen for which $\mathsf{Cell}_{i_1,j_1}$ lies in the square $S_r^{i,j}$ is at most $O(2^{2r+2}/(\lambda')^2)$. Recall that, if $\mathsf{Cell}_{i_1,j_1} \in S_r^{i,j}$, then $\mathrm{cong}_{H''}(\mathcal{Q}'_{i,j}) \leq 2^{z-r+1} \leq O(\lambda'/2^r)$. Moreover, if $\mathsf{Cell}_{i_1,j_1}$ does not lie in any of the squares $S_0^{i,j}, \ldots, S_z^{i,j}$, then $\mathrm{cong}_{H''}(\mathcal{Q}'_{i,j}) \leq 1$. The analysis for cell $\mathsf{Cell}_{i_2,j_2}$ is symmetric. Therefore, altogether (now taking into account both the cells $\mathsf{Cell}_{i_1,j_1}$ and $\mathsf{Cell}_{i_2,j_2}$), we get that:

$$\mathbf{E}_{(i,j)\in Z}\left[(\mathrm{cong}_{H''}(\mathcal{Q}'_{i,j}, e))^2\right] \leq O(1) + \sum_{r=0}^{z} O\left(\frac{2^{2r+2}}{(\lambda')^2} \cdot \frac{(\lambda')^2}{2^{2r}}\right) = O(z) \leq O(\log m).$$

Lastly, since $\mathrm{cong}_{H''}(\mathcal{Q}_{i,j}, e) \leq O\left(\frac{\log^4 m}{\alpha\alpha'}\right) \cdot \left(\mathrm{cong}_{H''}(\mathcal{Q}'_{i,j}, e) + \frac{\log^4 m}{\alpha\alpha'}\right)$, we get that:

$$\mathbf{E}_{\mathcal{Q}_{i,j}\sim\mathcal{D}}\left[(\mathrm{cong}_{H''}(\mathcal{Q}_{i,j}, e))^2\right] \leq O\left(\frac{\log^{16} m}{(\alpha\alpha')^4}\right).$$

# A Proof of Corollary 1.3

In this section, we provide the proof of Corollary 1.3 from Theorem 1.1 and Theorem 1.2. Suppose we are given a simple $n$-vertex graph $G$ with maximum vertex degree $\Delta$. We use the algorithm from Theorem 1.2 in order to compute an instance $I = (G', \Sigma)$ of MCNwRS, with $m = |E(G')| \leq O\left(\mathsf{OPT}_{\mathsf{cr}}(G) \cdot \mathrm{poly}(\Delta \cdot \log n)\right)$, and $\mathsf{OPT}_{\mathsf{cnwrs}}(I) \leq O\left(\mathsf{OPT}_{\mathsf{cr}}(G) \cdot \mathrm{poly}(\Delta \cdot \log n)\right)$. Notice that, since $G$ is a simple graph, $\mathsf{OPT}_{\mathsf{cr}}(G) \leq |E(G)|^2 \leq n^4$, and $\Delta \leq n$. Therefore, $m = |E(G')| \leq \mathrm{poly}(n)$.

We use the algorithm from Theorem 1.1 to compute a solution to instance $I$ of MCNwRS, such that, w.h.p., the number of crossings in the solution is bounded by $2^{O((\log m)^{7/8} \log\log m)} \cdot (\mathsf{OPT}_{\mathsf{cnwrs}}(I) + m)$. Lastly, using the algorithm from Theorem 1.2, we efficiently compute a drawing of graph $G$, with the number of crossings bounded by:

$$\begin{aligned}&\left(2^{O((\log m)^{7/8} \log\log m)} \cdot (\mathsf{OPT}_{\mathsf{cnwrs}}(I) + m) + \mathsf{OPT}_{\mathsf{cr}}(G)\right) \cdot \mathrm{poly}(\Delta \log n)\\ &\qquad \leq O\left(2^{O((\log n)^{7/8} \log\log n)} \cdot \mathrm{poly}(\Delta)\right) \cdot \mathsf{OPT}_{\mathsf{cr}}(G).\end{aligned}$$

# B Proofs Omitted from Section 2

## B.1 Proof of Theorem 2.7

For every vertex $u \in V(G)$, we denote $d_u = \deg_G(v)$, and we denote $\delta_G(u) = \{e_1(u), \ldots, e_{d_u}(u)\}$, where the edges are indexed according to their order in the rotation $\mathcal{O}_u \in \Sigma$.

In order to prove the theorem, we construct a new graph $G'$, that is obtained from $G$ by replacing every vertex $u \in V(G)$ with the $(d_u \times d_u)$-grid. We show that, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I) = 0$, then graph $G'$ is planar. We then provide an algorithm, that, given a planar drawing of $G'$, computes a solution to instance $I$ of MCNwRS whose cost is 0.

We start by defining the graph $G'$. For every vertex $u \in V(G)$, we let $H_u$ be the $(d_u \times d_u)$ grid. We denote the vertices that appear on the first row of grid $H_u$ by $x_1(u), \ldots, x_{d_u}(u)$, in the natural order of their appearance. In order to construct graph $G'$, we start with the disjoint union of the graphs in $\{H_u\}_{u \in V(G)}$. We then consider every edge $e \in E(G)$ one by one. Let $e = (u, u')$ be any such edge, and assume that $e = e_i(u) = e_j(u')$ (that is, $e$ is the $i$th edge incident to $u$, and the $j$th edge incident to $u'$). We then add edge $e' = (x_i(u), x_j(u'))$ to graph $G'$, and we view edge $e'$ as *representing* the edge $e \in E(G)$. This completes the construction of the graph $G'$. We call the edges of $G'$ that lie in set $\{e' \mid e \in E(G)\}$ *primary edges*, and the remaining edges of $G'$ *secondary edges*. Notice that, from our construction, a vertex of $G'$ may be incident to at most one primary edge. We use the following two observations.

**Observation B.1** *If $\mathsf{OPT}_{\mathsf{cnwrs}}(I) = 0$, then graph $G'$ is planar.*

**Proof:** Assume that $\mathsf{OPT}_{\mathsf{cnwrs}}(I) = 0$, and let $\varphi$ be a solution to instance $I$ of MCNwRS, with $\mathsf{cr}(\varphi) = 0$. We transform drawing $\varphi$ to obtain a planar drawing $\psi$ of the graph $G'$.

In order to do so, for every vertex $u \in V(G)$, we consider the tiny $u$-disc $D(u) = D_\varphi(u)$. For every edge $e_i(u) \in \delta_G(u)$, we denote by $p_i(u)$ the unique point of the image of $e_i(u)$ in $\varphi$ that lies on the boundary of the disc $D(u)$. Note that points $p_1(u), \ldots, p_{d_u}(u)$ must appear on the boundary of disc $D(u)$ in this circular order. If they are encountered in this order as we traverse the boundary of $D(u)$ in counter-clock-wise direction, then we say that vertex $u$ is positive; otherwise we say that it is negative. We let $D'(u)$ be a disc that is contained in $D(u)$, such that the boundaries of $D(u)$ and $D'(u)$ are disjoint.

Consider now a vertex $u \in V(G)$, and let $\psi_u$ be the standard drawing of the grid $H_u$. We let $\tilde{D}(u)$ be a disc in the drawing $\psi_u$, such that the image of the grid $H_u$ is contained in $\tilde{D}(u)$, and the images of vertices $x_1(u), \ldots, x_{d_u}(u)$, that we denote by $p'_1(u), \ldots, p'_{d_u}(u)$ lie on the boundary of $\tilde{D}(u)$, and are encountered in this order as we traverse the boundary of $\tilde{D}(u)$ in the counter-clock-wise direction. We also ensure that the only points of $\psi_u(H_u)$ that lie on the boundary of $\tilde{D}(u)$ are $p'_1(u), \ldots, p'_{d_u}(u)$.

In order to define a planar drawing $\psi$ of graph $G'$, we process every vertex $u \in V(G)$ one by one. Consider any such vertex $u$. If $u$ is a positive vertex, then we plant the drawing $\psi_u$ of $H_u$ inside the disc $D'(u)$ that we defined before, so that the discs $\tilde{D}(u)$ and $D'(u)$ coincide. Observe that, in this case, points $p_1(u), \ldots, p_{d_u}(u)$ are encountered in this order on the boundary of $D(u)$ as we traverse it in counter-clock-wise direction; and similarly, points $p'_1(u), \ldots, p'_{d_u}(u)$ are encountered in this order on the boundary of $D'(u)$ as we traverse it in counter-clock-wise direction. For all $1 \leq i \leq d_u$, we can then define a curve $\gamma_i(u)$ connecting points $p_i(u)$ and $p'_i(u)$, that is contained in $D(u)$, and is internally disjoint from $D'(u)$. Moreover, we can ensure that all curves in $\{\gamma_i(u) \mid 1 \leq i \leq d_u\}$ are disjoint from each other and are internally disjoint from the boundary of $D_u$. If $u$ is a negative vertex, then we repeat the same process, except that we plant a mirror image of the drawing $\psi_u$ of $H_u$ inside the disc $D'(u)$. This allows us to define the set $\{\gamma_i(u) \mid 1 \leq i \leq d_u\}$ of disjoint curves as before, where for $1 \leq i \leq d_u$, curve $\gamma_i(u)$ connects points $p_i(u)$ and $p'_i(u)$, is contained in $D(u)$, and is internally disjoint from $D'(u)$.

So far, for every vertex $u \in V(G)$, we have defined the images of the vertices and the edges of the grid $H_u$ in $\psi$. In order to complete the drawing $\psi$ of graph $G'$, we process the edges $e \in E(G)$ one by one. Consider any such edge $e = (u, u')$, and assume that $e = e_i(u) = e_j(u')$. Note that the image $\varphi(e)$ of edge $e$ contains points $p_i(u)$ and $p_j(u')$. Let $\sigma(e)$ be the segment of $\varphi(e)$ between these two points. Notice that, by construction, $\sigma(e)$ is internally disjoint from all discs in $\{D(u'')\}_{u'' \in V(G)}$. Recall that graph $G'$ contains an edge $e' = (x_i(u), x_j(u'))$ representing edge $e$. We let the image of edge $e'$ in $\psi$ be the concatenation of curves $\gamma_i(u)$ (that connects the image of $x_i(u)$ to point $p_i(u)$; $\sigma(e)$ (connecting $p_i(u)$ to $p_j(u)$); and $\gamma_j(u')$ (connecting $p_j(u')$ to the image of $x_j(u')$). This completes the definition of the drawing $\psi$ of $G'$. It is immediate to verify that it is a planar drawing. □

**Observation B.2** *There is an efficient algorithm, that, given a planar drawing $\varphi'$ of graph $G'$, computes a feasible solution $\varphi$ to instance $I$ of* MCNwRS *with no crossings.*

**Proof:** Consider the planar drawing $\varphi'$ of graph $G'$ on the sphere. Recall that for all $r \geq 1$, the $(r \times r)$-grid graph has a unique planar drawing. Therefore, for every vertex $u \in V(G)$, the drawing of grid $H_u$ that is induced by $\varphi'$ is the standard drawing $\psi_u$ of the grid. Recall that the boundary of the grid $H_u$ is a simple cycle. Let $\gamma_u$ be the closed curve, that is obtained by taking the union of the images of all edges of the boundary of $H_u$. Notice that $\gamma_u$ must be a simple curve, and, moreover, for every pair $u', u''$ of distinct vertices of $G'$, $\gamma_{u'} \cap \gamma_{u''} = \emptyset$. For a vertex $u \in V(G)$, let $D'(u)$ be the disc whose boundary is $\gamma_u$, such that the drawing of $H_u$ in $\varphi'$ is contained in $D'(u)$. We denote by $p^*(u)$ the image of vertex $v_{d_u,d_u}$ of the grid $H_u$, and, for $1 \leq i \leq r$, by $p_i(u)$ the image of vertex $x_i(u)$. Note that points $p_1(u), \ldots, p_{d_u}(u), p^*(u)$ appear in this circular order on the boundary of $D'(u)$. Notice also that it is possible that, for a pair $u' \neq u''$ of vertices of $G$, $D'(u') \subseteq D'(u'')$.

Let $\Gamma$ denote the set of curves that contains, for every primary edge $e'$ of $G'$, its image $\varphi'(e')$. We use the following claim.

**Claim B.3** *There is an efficient algorithm that constructs, for each vertex $u \in V(G)$ and index $1 \leq i \leq d_u$, a curve $\gamma_i(u)$ that is contained in $D'(u)$ and connects $p_i(u)$ to $p^*(u)$. Moreover, for every pair $\gamma, \gamma'$ of distinct curves in set $\Gamma \cup \{\gamma_i(u) \mid u \in V(G); 1 \leq i \leq d_u\}$, every point $p \in \gamma \cap \gamma'$ must be an endpoint of both curves.*

We prove the claim below, after we complete the proof of Observation B.2 using it. We define a

drawing $\varphi$ of graph $G$ as follows. For every vertex $u \in V(G)$, the image $\varphi(u)$ is defined to be $p^*(u)$. Consider now some edge $e = (u, u') \in E(G)$, and assume that $e = e_i(u) = e_j(u')$. We then let the image of $e$ in $\varphi$ be the concatenation of three curves: (i) curve $\gamma_i(u)$, connecting $p^*(u)$ to $p_i(u)$; (ii) the image of edge $e' = (v_i(u), v_j(u')) \in E(G')$ in drawing $\varphi'$, that connects $p_i(u)$ to $p_j(u')$; and (iii) curve $\gamma_j(u')$, connecting $p_j(u')$ to $p^*(u')$. Notice that the resulting curve connects $\varphi(u)$ to $\varphi(u')$, as required. This completes the definition of the drawing $\varphi$ of $G$.

We now show that this is a legal drawing, and that the number of crossings in this drawing is 0. Indeed, assume for contradition that there are two edges $e_1, e_2 \in E(G)$, and that some point $p$ lies in $\varphi(e_1) \cap \varphi(e_2)$. Note that the endpoints of $\varphi(e_2)$ may not be inner points of $\varphi(e_1)$ and vice versa. Therefore, $p$ is an inner point on both $\varphi(e_1)$ and $\varphi(e_2)$. From our construction, there must be two curves $\gamma, \gamma' \in \Gamma \cup \{\gamma_i(u) \mid u \in V(G); 1 \leq i \leq d_u\}$, with $\gamma \subseteq \varphi(e_1)$ and $\gamma' \subseteq \varphi(e_2)$ that contain $p$. From Claim B.3, point $p$ must be an endpoint of both curves. Assume that $e_1 = (u_1, u_1')$, and that $e_1 = e_i(u_1) = e_j(u_1')$. Then, from our construction, $p = p_i(u_1)$ or $p = p_j(u_1')$ must hold. Similarly, assuming that $e_2 = (u_2, u_2')$, and that $e_2 = e_{i'}(u_2) = e_{j'}(u_2')$, we get that $p = p_{i'}(u_2)$ or $p = p_{j'}(u_2')$ must hold. This may only happen if two distinct primary edges of $G'$ are incident to the same vertex of $G'$, which is impossible from our construction. We conclude that $\varphi$ is a valid drawing of $G$ with 0 crossings.

Next, we show that $\varphi$ obeys the rotation system $\Sigma$. Consider some vertex $u \in V(G)$, and a tiny $u$-disc $D_\varphi(u)$. For $1 \leq i \leq d_u$, let $\tilde{p}_i(u)$ be the point on the boundary of $D_\varphi(u)$ that lies on the image of edge $e_i(u)$ in $\varphi$. In particular, point $\tilde{p}_i(u)$ belongs to the curve $\gamma_i(u)$, whose endpoints are $p_i(u), p^*(u)$. Since points $p_1(u), \ldots, p_{d_u}(u)$ appear on the boundary of disc $D'(u)$ in this circular order, and the curves $\gamma_1(u), \ldots, \gamma_{d_u}(u)$ are internally disjoint, points $\tilde{p}_1(u), \ldots, \tilde{p}_{d_u}(u)$ must appear on the boundary of disc $D_\varphi(u)$ in this circular order. We conclude that drawing $\varphi$ of $G$ obeys the rotation system $\Sigma$.

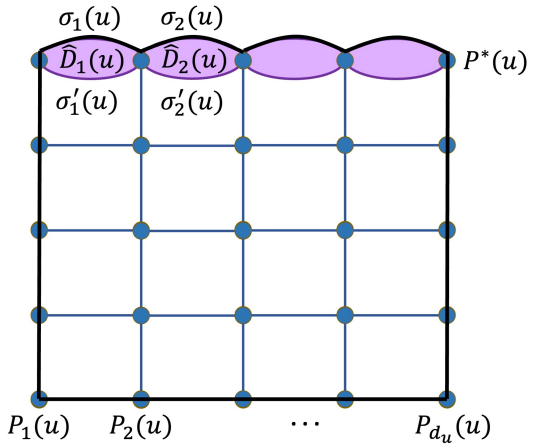
In order to complete the proof of Observation B.2, it is now enough to prove Claim B.3, which we do next.

**Proof of Claim B.3.** Consider a vertex $u \in V(G)$. For convenience, for $1 \leq i, j \leq d_u$, we denote by $v_{i,j}(u)$ the unique vertex of the grid $H_u$ lying in the intersection of its $i$th row and $j$th column.

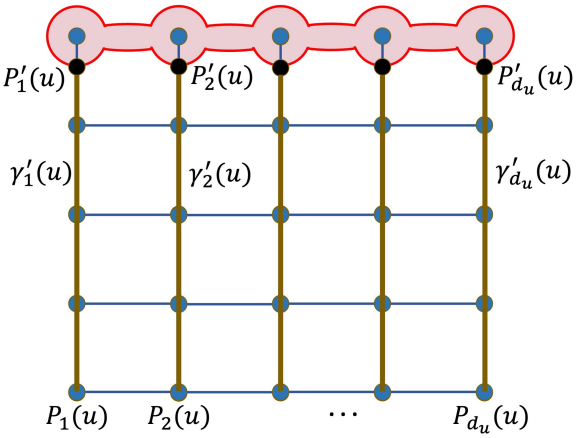
Let $A(u) = \{a_1(u), \ldots, a_{d_u-1}(u)\}$ be the sequence of edges on the last row of the grid $H_u$. Recall that, for $1 \leq i < d_u$, curve $\varphi'(a_i(u))$ is contained in the boundary of the disc $D'(u)$. We denote $\sigma_i(u) = \varphi'(a_i(u))$, and draw another curve $\sigma_i'(u)$, whose endpoints are the same as those of $\sigma_i(u)$, such that $\sigma_i'(u)$ is contained in the interior of $D'(u)$; is internally disjoint from $\sigma_i(u)$ and the images of all edges of $G'$ in $\varphi'$; and it is drawn in parallel to $\sigma_i(u)$ right next to it. Next, we let $\hat{D}_i(u)$ be the disc, whose boundary is the union of the curves $\sigma_i(u)$ and $\sigma_i'(u)$ (see Figure 40(a)). Lastly, we let $\hat{D}(u) \subseteq D'(u)$ be smallest disc, whose interior contains, for all $1 \leq i \leq d_u - 1$, the disc $\hat{D}_i(u)$, and, for all $1 \leq i \leq d_u$, the intersection of the tiny $v_{d_u,i}(u)$-disc $D_{\varphi'}(v_{d_u,i}(u))$ and the disc $D'(u)$ (see Figure 40(b)).

From our construction, the only vertices of $G'$ whose images are contained in disc $\hat{D}(u)$ are the vertices lying in the last row of the grid $H_u$. The only edges of $G'$ that may have a non-empty intersection with disc $\hat{D}(u)$ are the edges of $H_u$ that are incident to the vertices of $H_u$ lying in the last row of the grid.

Consider now some index $1 \leq i \leq d_u$. Let $\gamma_i'(u)$ be the curve obtained by concatenating the images of all edges that lie on the $i$th column of grid $H_u$. We truncate the curve $\gamma_i'(u)$, so that it terminates at a point on the boundary of disc $\hat{D}(u)$, and is internally disjoint from disc $\hat{D}(u)$. We denote by $p_i'(u)$ the point on the boundary of $\hat{D}(u)$ that lies on $\gamma_i'(u)$. Note that curve $\gamma_i'(u)$ connects points $p_i(u)$ and $p_i'(u)$; it is contained in disc $D'(u)$, and it is internally disjoint from disc $\hat{D}(u)$. It is easy to verify that, since drawing $\varphi'$ of $G'$ is planar, curves $\gamma_i'(u), \ldots, \gamma_{d_u}'(u)$ are disjoint from each other. For all $1 \leq i \leq d_u$, we then let $\gamma_i''(u)$ be any simple curve connecting $p_i'(u)$ to $p^*(u)$, that is contained in disc $\hat{D}(u)$; we




(a) Curves in $\{\sigma_i(u), \sigma'_i(u)\}_i$, and the corresponding discs $\left\{\hat{D}_i(u)\right\}_i$. The boundary of disc $D'(u)$ is shown in black.




(b) Disc $\hat{D}(u)$ is shown in red.

Figure 40: Illustration of curves in $\{\sigma_i(u), \sigma'_i(u)\}_i$ and the corresponding discs.

construct the curves $\gamma''_1(u), \ldots, \gamma''_{d_u}(u)$ so that they are internally disjoint from each other. For all $1 \leq i \leq d_u$, we then let $\gamma_i(u)$ be the curve obtained by concatenating $\gamma'_i(u)$ and $\gamma''_i(u)$. It is immediate to verify that curve $\gamma_i(u)$ is contained in $D'(u)$, and it connects $p_i(u)$ to $p^*(u)$. From our construction, it is easy to verify that, for any pair $\gamma, \gamma'$ of distinct curves in set $\Gamma \cup \{\gamma_i(u) \mid u \in V(G); 1 \leq i \leq d_u\}$, every point $p \in \gamma \cap \gamma'$ must be an endpoint of both curves. □ □

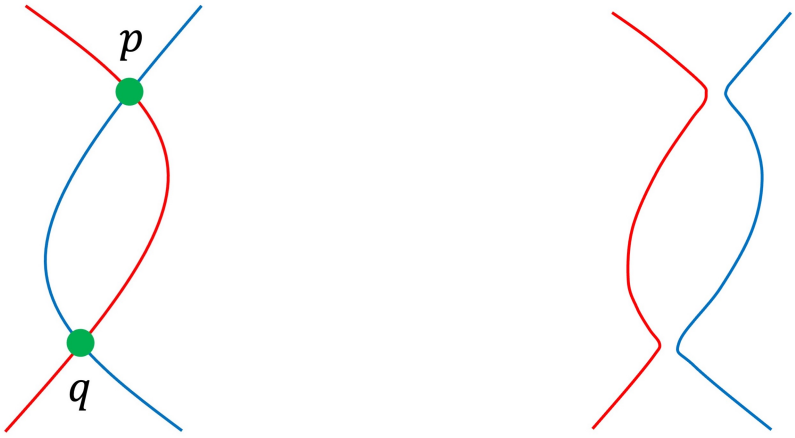
We are now ready to complete the proof of Theorem 2.7. We construct the graph $G'$ as described above, and use the algorithm from Theorem 2.4 to test whether graph $G'$ is planar, and if so, to compute a planar drawing $\varphi'$ of $G'$. If $G'$ is not planar, then we correctly establish that $\mathsf{OPT}_{\mathsf{cnwrs}}(G, \Sigma) \neq 0$, from Observation B.1. If $G'$ is planar, then we apply the algorithm from Observation B.2 to graph $G'$ and its planar drawing $\varphi'$, to compute a valid solution $\varphi$ to instance $I$ of MCNwRS with $\mathsf{cr}(\varphi) = 0$.

## B.2 Proof of Theorem 2.8

We start with any feasible solution $\varphi$ to instance $I$, and then gradually modify it to ensure that every pair of edges in $G$ cross at most once. As long as there is a pair $e, e' \in E(G)$ of distinct edges, whose images cross at least twice in $\varphi$, we perform the following modification step. Let $p, q$ be two crossing points between curves $\varphi(e), \varphi(e')$, that appear consecutively on $\varphi(e)$; in other words, the segment of $\varphi(e)$ between $a$ and $b$ contains no other point that lies on $\varphi(e')$. We "uncross" the images of edges $e$ and $e'$, as shown in Figure 41. (In Sections 4.4.2 and D.17 we provide a more formal description of this uncrossing process, that we refer to as *type-1 uncrossing*). It is easy to see that, after this uncrossing step, the new drawing remains a feasible solution to instance $I$, and the number of crossings in the drawing decreases by at least 2. We continue this process until every pair of edges of $G$ cross at most once in $\varphi$. It is clear that the resulting drawing contains at most $|E(G)|^2$ crossings.

## B.3 Proof of Claim 2.11

Denote $I = (G, \Sigma)$ and $m = |E(G)|$. Recall that from Property D1, $\sum_{I'=(G',\Sigma') \in \mathcal{I}'} |E(G')| \leq m \cdot (\log m)^{O(1)}$, so in particular, for every instance $I' = (G', \Sigma') \in \mathcal{I}'$, $|E(G')| \leq m \cdot (\log m)^{O(1)}$. From the same property, for every instance $I' = (G', \Sigma') \in \mathcal{I}'$, $\sum_{I''=(G'',\Sigma'') \in \mathcal{I}''(I')} |E(G'')| \leq |E(G')| \cdot$




(a) Before: Curves $\varphi(e)$ (red) and $\varphi(e')$ (blue) cross at $p$ and $q$.

(b) After: The modified curves no longer cross at $p$ or at $q$.

Figure 41: Uncrossing two curves.

$(\log(|E(G')|))^{O(1)} \leq |E(G')| \cdot (\log m)^{O(1)}$. Therefore, altogether, we get that:

$$\begin{aligned}\sum_{I''=(G'',\Sigma'')\in\mathcal{I}} |E(G'')| &= \sum_{I'\in\mathcal{I}'} \sum_{I''=(G'',\Sigma'')\in\mathcal{I}''(I')} |E(G'')| \\ &\leq \sum_{I'=(G',\Sigma')\in\mathcal{I}'} |E(G')| \cdot (\log m)^{O(1)} \\ &\leq m \cdot (\log m)^{O(1)},\end{aligned}$$

establishing Property D1.

Next, we establish Property D'2, using the same property of Algorithms $\mathsf{Alg}_1$ and $\mathsf{Alg}_2$:

$$\begin{aligned}\mathbf{E}\left[\sum_{I''\in\mathcal{I}''} \mathsf{OPT}_{\mathsf{cnwrs}}(I'')\right] &= \sum_{I'\in\mathcal{I}'} \mathbf{E}\left[\sum_{I''\in\mathcal{I}''(I')} \mathsf{OPT}_{\mathsf{cnwrs}}(I'')\right] \\ &\leq \sum_{I'=(G',\Sigma')\in\mathcal{I}'} \mathbf{E}\left[\left(\mathsf{OPT}_{\mathsf{cnwrs}}(I') + |E(G')|\right)\cdot\nu''\right] \\ &= \nu''\cdot\left(\sum_{I'=(G',\Sigma')\in\mathcal{I}'} |E(G')|\right) + \nu''\cdot\mathbf{E}\left[\sum_{I'\in\mathcal{I}'} \mathsf{OPT}_{\mathsf{cnwrs}}(I')\right] \\ &\leq O(\nu''\cdot m\cdot(\log m)^{O(1)}) + (\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\cdot(\nu'\nu'') \\ &\leq (\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\cdot\nu''\cdot\max\left\{2\nu', (\log m)^{O(1)}\right\} \\ &\leq (\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\cdot\nu.\end{aligned}$$

Lastly, we establish Property D3 of Algorithm $\mathsf{Alg}$, using the same property of algorithms $\mathsf{Alg}_1$ and $\mathsf{Alg}_2$. Assume that we are given, for every instance $I''\in\mathcal{I}''$, a feasible solution $\varphi(I'')$ to $I''$. We process instances $I'\in\mathcal{I}'$ one by one. For each such instance, we apply Algorithm $\mathsf{Alg}(\mathcal{I}''(I'))$ that is given by Property D3 of the decomposition $\mathcal{I}''(I')$ to solutions $\varphi(I'')$ of instances $I''\in\mathcal{I}''(I')$, to obtain a solution $\varphi(I')$ to instance $I'$ of cost at most $O\left(\sum_{I''\in\mathcal{I}''(I')}\mathsf{cr}(\varphi(I''))\right)$. We then apply the

algorithm $\mathsf{Alg}(\mathcal{I}')$, given by Property D3 of the decomposition $\mathcal{I}'$ of $I$, to the resulting solutions $\varphi(I')$ for instances $I' \in \mathcal{I}'$, to obtain a solution $\varphi(I)$ to instance $I$, whose cost is at most:

$$O\left(\sum_{I' \in \mathcal{I}'} \mathsf{cr}(\varphi(I'))\right) \leq O\left(\sum_{I'' \in \mathcal{I}} \mathsf{cr}(\varphi(I''))\right).$$

# C Proofs Omitted from Section 3

## C.1 Proof of Claim 3.7

The proof is by induction on $h(I)$. The base case is when $h(I) = 0$, so $v(I)$ is a leaf vertex of $T^*$, and hence of $T$. Denote $I = (G, \Sigma)$. From Observation 3.5, either $|E(G)| \leq \mu^{c''}$; or $\mathsf{OPT}_{\mathsf{cnwrs}}(I) = 0$; or $\mathsf{OPT}_{\mathsf{cnwrs}}(I) > |E(G)|^2/\mu^{c''}$. If $\mathsf{OPT}_{\mathsf{cnwrs}}(I) = 0$, then the algorithm returns a solution of cost 0. Otherwise, $\mathsf{OPT}_{\mathsf{cnwrs}}(I) \geq 1$, and, if $|E(G)| \leq \mu^{c''}$, then the algorithm returns the trivial solution, of cost at most $|E(G)|^2 \leq |E(G)| \cdot \mu^{c''}$. Lastly, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I) > |E(G)|^2/\mu^{c''}$, then, since the trivial solution $\varphi'$ is considered by the algorithm, it returns a solution of cost at most $|E(G)|^2 \leq \mathsf{OPT}_{\mathsf{cnwrs}}(I) \cdot \mu^{c''}$.

Assume now that the claim holds for all vertices $v(I)$ of $T^*$ with $h(I) < q$, for some $0 < q \leq \mathsf{dep}(T)$. Consider any vertex $v(I)$ of the tree $T^*$ with $h(I) = q$. Let $v(I_1), \ldots, v(I_k)$ be the child vertices of $v(I)$ in the tree $T^*$. Denote $I = (G, \Sigma)$ and $|E(G)| = m$. Additionally, for all $1 \leq r \leq k$, denote $I_r = (G_r, \Sigma_r)$ and $m_r = |E(G_r)|$.

Since instance $I$ is not a leaf instance of $T$, $|E(G)| \geq \mu^{c''}$ must hold. Since we have assumed that event $\mathcal{E}$ does not happen, either $\mathsf{OPT}_{\mathsf{cnwrs}}(I) > |E(G)|^2/\mu^{c''}$, or $\sum_{r=1}^{k} \mathsf{OPT}_{\mathsf{cnwrs}}(I_r) \leq (\mathsf{OPT}_{\mathsf{cnwrs}}(I) + m) \cdot 2^{c_g(\log m)^{3/4} \log\log m}$ must hold. In the former case, the algorithm is guaranteed to return a solution to $I$ whose cost is at most $|E(G)|^2 \leq \mathsf{OPT}_{\mathsf{cnwrs}}(I) \cdot \mu^{c''}$, since the trivial solution $\varphi'$ is considered as one of the possible solutions. From now on we focus on latter case. For all $1 \leq r \leq k$, let $\varphi_r$ be the solution to instance $I_r$ that the algorithm computes recursively. From the induction hypothesis, for all $1 \leq r \leq k$:

$$\mathsf{cr}(\varphi_r) \leq 2^{\tilde{c} \cdot h(I_r) \cdot (\log m^*)^{3/4} \log\log m^*} \cdot \mu^{c'' \cdot c_g} \cdot \mathsf{OPT}_{\mathsf{cnwrs}}(I_r) + (\log m^*)^{4c_g h(I_r)} \mu^{2c'' \cdot \tilde{c}} \cdot m_r.$$

Notice that, for all $1 \leq r \leq k$, $h(I_r) \leq q - 1$. Moreover, from Theorem 3.1, $\sum_{r=1}^{k} m_r \leq m \cdot (\log m)^{c_g} \leq m \cdot (\log m^*)^{c_g}$. Lastly, as noted already, $\sum_{r=1}^{k} \mathsf{OPT}_{\mathsf{cnwrs}}(I_r) \leq (\mathsf{OPT}_{\mathsf{cnwrs}}(I) + m) \cdot 2^{c_g(\log m)^{3/4} \log\log m}$ must hold. Altogether, we get that:

$$\begin{aligned}
\sum_{r=1}^{k} \mathsf{cr}(\varphi_r) &\leq 2^{\tilde{c} \cdot (q-1) \cdot (\log m^*)^{3/4} \log\log m^*} \cdot \mu^{c'' \cdot c_g} \cdot \sum_{r=1}^{k} \mathsf{OPT}_{\mathsf{cnwrs}}(I_r) + (\log m^*)^{4c_g(q-1)} \mu^{2c'' \cdot \tilde{c}} \cdot \sum_{r=1}^{k} m_r \\
&\leq 2^{\tilde{c} \cdot (q-1) \cdot (\log m^*)^{3/4} \log\log m^*} \cdot \mu^{c'' \cdot c_g} \cdot (\mathsf{OPT}_{\mathsf{cnwrs}}(I) + m) \cdot 2^{c_g(\log m)^{3/4} \log\log m} \\
&\qquad + (\log m^*)^{4c_g(q-1)} \mu^{2c'' \cdot \tilde{c}} \cdot m \cdot (\log m^*)^{c_g} \\
&\leq 2^{\tilde{c} \cdot (q-0.5) \cdot (\log m^*)^{3/4} \log\log m^*} \cdot \mu^{c'' \cdot c_g} \cdot \mathsf{OPT}_{\mathsf{cnwrs}}(I) + (\log m^*)^{4c_g q - 3c_g} \mu^{2c'' \cdot \tilde{c}} \cdot m \\
&\qquad + 2^{\tilde{c} \cdot (q-0.5) \cdot (\log m^*)^{3/4} \log\log m^*} \cdot \mu^{c'' \cdot c_g} \cdot m.
\end{aligned}$$

Since $q \leq \mathsf{dep}(T) \leq \frac{(\log m^*)^{1/8}}{c^* \log\log m^*}$ from Observation 3.2, the last term is bounded by:

$$2^{\tilde{c} \cdot (\log m^*)^{7/8}/c^*} \cdot \mu^{c'' \cdot c_g} \cdot m \leq \mu^{2c'' \cdot c_g} \cdot m$$

(since $\mu = 2^{c^*(\log m^*)^{7/8}\log\log m^*}$). Therefore, we get that:

$$\sum_{r=1}^{k}\mathsf{cr}(\varphi_r) \leq 2^{\tilde{c}\cdot(q-0.5)\cdot(\log m^*)^{3/4}\log\log m^*}\cdot\mu^{c''\cdot c_g}\cdot\mathsf{OPT}_{\mathsf{cnwrs}}(I)+(\log m^*)^{4c_gq-2c_g}\mu^{2c''\cdot\tilde{c}}\cdot m.$$

The solution that our algorithm returns for instance $I$ is obtained by applying Algorithm $\mathsf{AlgCombineDrawings}$ from Theorem 3.1 to solutions $\varphi_1,\ldots,\varphi_k$ to instances $I_1,\ldots,I_k$ (or some other solution the algorithm considers, if its cost is smaller). Since event $\mathcal{E}$ does not happen, the cost of the resulting solution is bounded by:

$$\begin{aligned}
&c_g\cdot\left(\sum_{r=1}^{k}\mathsf{cr}(\varphi_r)\right)+(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\cdot\mu^{c_g}\\
&\quad\leq c_g\cdot 2^{\tilde{c}\cdot(q-0.5)\cdot(\log m^*)^{3/4}\log\log m^*}\cdot\mu^{c''\cdot c_g}\cdot\mathsf{OPT}_{\mathsf{cnwrs}}(I)+\mu^{c_g}\cdot\mathsf{OPT}_{\mathsf{cnwrs}}(I)\\
&\qquad\qquad+c_g\cdot(\log m^*)^{4c_gq-2c_g}\mu^{2c''\cdot\tilde{c}}\cdot m+\mu^{c_g}\cdot m\\
&\quad\leq 2^{\tilde{c}\cdot q\cdot(\log m^*)^{3/4}\log\log m^*}\cdot\mu^{c''\cdot c_g}\cdot\mathsf{OPT}_{\mathsf{cnwrs}}(I)+(\log m^*)^{4c_gq}\mu^{2c''\cdot\tilde{c}}\cdot m,
\end{aligned}$$

as required.

## C.2 Proof of Claim 3.17

We construct the solution $\varphi(I)$ to instance $I$ in three steps. In the first step, we compute a solution $\varphi(I')$ to every instance $I'\in\hat{\mathcal{I}}^{(n)}_{\mathsf{large}}\cup\tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}$, as follows. Consider any instance $I'=(G',\Sigma')\in\hat{\mathcal{I}}^{(n)}_{\mathsf{large}}\cup\tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}$. Recall that we are given a solution $\varphi(I'')$ to every instance $I''\in\overline{\mathcal{I}}(I')$. Recall also that $\overline{\mathcal{I}}(I')$ is a $\nu$-decomposition of instance $I'$. We apply the efficient algorithm $\mathsf{Alg}(\overline{\mathcal{I}}(I'))$ from the definition of $\nu$-decomposition to the drawings in set $\{\varphi(I'')\}_{I''\in\overline{\mathcal{I}}(I')}$, to compute a feasible solution $\varphi(I')$ to instance $I'$, of cost $\mathsf{cr}(\varphi(I'))\leq O\left(\sum_{I''\in\overline{\mathcal{I}}(I')}\mathsf{cr}(\varphi(I''))\right)$. Overall, we get that:

$$\sum_{I'\in\hat{\mathcal{I}}^{(n)}_{\mathsf{large}}\cup\tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}}\mathsf{cr}(\varphi(I'))\leq\sum_{I''\in\mathcal{I}^*}O(\mathsf{cr}(\varphi(I''))). \tag{20}$$

We have now obtained a solution $\varphi(I')$ to every instance $I'\in\left(\hat{\mathcal{I}}_{\mathsf{small}}\cup\hat{\mathcal{I}}^{(n)}_{\mathsf{large}}\cup\tilde{\mathcal{I}}_{\mathsf{small}}\cup\tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}\right)$.

In the second step, we compute a solution $\varphi(I')$ to every instance $I'\in\hat{\mathcal{I}}^{(w)}_{\mathsf{large}}$. Consider any such instance $I'\in\hat{\mathcal{I}}^{(w)}_{\mathsf{large}}$. Recall that we applied the algorithm from Theorem 3.13 to instance $I'$, to obtain a collection $\tilde{\mathcal{I}}(I')$ of instances of $\mathsf{MCNwRS}$. Every instance in the resulting collection belongs to $\tilde{\mathcal{I}}_{\mathsf{small}}$ or to $\tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}$. We use Algorithm $\mathsf{AlgCombineDrawings}'$, that is guaranteed from Theorem 3.13, to compute a solution $\varphi(I')$ to instance $I'$. Since we have assumed that event $\mathcal{E}_2$ did not happen, the cost of the solution is bounded by: $\mathsf{cr}(\varphi(I'))\leq\sum_{\tilde{I}=(\tilde{G},\tilde{\Sigma})\in\tilde{\mathcal{I}}(I')}\mathsf{cr}(\varphi(\tilde{I}))+\mathsf{OPT}_{\mathsf{cnwrs}}(I')\cdot\mu^{c'_g}$. Overall, we get that:

$$\sum_{I'\in\hat{\mathcal{I}}^{(w)}_{\mathsf{large}}}\mathsf{cr}(\varphi(I'))\leq\sum_{I''\in\mathcal{I}^*}O(\mathsf{cr}(\varphi(I''))+\sum_{I'\in\hat{\mathcal{I}}^{(w)}_{\mathsf{large}}}\mathsf{OPT}_{\mathsf{cnwrs}}(I')\cdot\mu^{c'_g}. \tag{21}$$

We now describe the third step. We have so far obtained a solution $\varphi(I')$ to every instance $I'\in\left(\hat{\mathcal{I}}_{\mathsf{small}}\cup\hat{\mathcal{I}}^{(n)}_{\mathsf{large}}\cup\hat{\mathcal{I}}^{(w)}_{\mathsf{large}}\right)$, that is, a solution to every instance in $\hat{\mathcal{I}}$. Recall that $\hat{\mathcal{I}}$ is a $\nu_1$-decomposition of the input instance $I$. Since we have assumed that Event $\mathcal{E}_1$ did not happen, $\sum_{I'\in\hat{\mathcal{I}}}\mathsf{OPT}_{\mathsf{cnwrs}}(I')\leq 100\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\cdot\nu_1$. By combining Inequalities 20 and 21, we get that:

$$\begin{aligned}\sum_{I'\in\hat{\mathcal{I}}}\mathsf{cr}(\varphi(I')) &\leq \sum_{I''\in\mathcal{I}^*} O(\mathsf{cr}(\varphi(I'')) + \sum_{I'\in\hat{\mathcal{I}}^{(w)}_{\mathsf{large}}}\mathsf{OPT}_{\mathsf{cnwrs}}(I')\cdot\mu^{c'_g}\\ &\leq \sum_{I''\in\mathcal{I}^*} O(\mathsf{cr}(\varphi(I'')) + 100\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\cdot\nu_1\cdot\mu^{c'_g} \qquad (22)\\ &\leq \sum_{I''\in\mathcal{I}^*} O(\mathsf{cr}(\varphi(I'')) + (\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\cdot\mu^{O(1)},\end{aligned}$$

since $\nu_1 = 2^{O((\log m)^{3/4}\log\log m)}$ and $\mu \gg \nu_1$.

Lastly, we apply the efficient algorithm $\mathsf{Alg}(\hat{\mathcal{I}})$ that is guaranteed by the definition of $\nu_1$-decomposition to the solutions $\{\varphi(I')\}_{I'\in\hat{\mathcal{I}}}$, to obtain a feasible solution $\varphi(I)$ to instance $I$. The cost of the solution is bounded by $\sum_{I'\in\hat{\mathcal{I}}} O(\mathsf{cr}(\varphi(I'))) \leq \sum_{I''\in\mathcal{I}^*} O(\mathsf{cr}(\varphi(I'')) + (\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\cdot\mu^{O(1)}$, as required.

### C.3 Proof of Observation 3.18

Throughout the proof, we assume that $\mathsf{OPT}_{\mathsf{cnwrs}}(I) \leq |E(G)|^2/\mu^{c'}$ and bad event $\mathcal{E}$ did not happen. Since event $\mathcal{E}_1$ does not happen:

$$\sum_{I'\in\hat{\mathcal{I}}}\mathsf{OPT}_{\mathsf{cnwrs}}(I') \leq 100\nu_1\cdot(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m). \qquad (23)$$

Recall that $\hat{\mathcal{I}} = \hat{\mathcal{I}}_{\mathsf{small}}\cup\hat{\mathcal{I}}^{(n)}_{\mathsf{large}}\cup\hat{\mathcal{I}}^{(w)}_{\mathsf{large}}$. In Step 2 of the algorithm, applied the algorithm from Theorem 3.13 to every instance $I'=(G',\Sigma')\in\hat{\mathcal{I}}^{(w)}_{\mathsf{large}}$, to compute a collection $\tilde{\mathcal{I}}(I')$ of instances of $\mathsf{MCNwRS}$. Consider now any such instance $I'\in\hat{\mathcal{I}}^{(w)}_{\mathsf{large}}$. Since we have assumed that Event $\mathcal{E}_2$ did not happen:

$$\sum_{\tilde{I}\in\tilde{\mathcal{I}}(I')}\mathsf{OPT}_{\mathsf{cnwrs}}(\tilde{I}) \leq \mathsf{OPT}_{\mathsf{cnwrs}}(I')\cdot(\log|E(G')|)^{c'_g} \leq \mathsf{OPT}_{\mathsf{cnwrs}}(I')\cdot(\log m)^{c'_g}.$$

Recall that we have defined $\tilde{\mathcal{I}} = \bigcup_{I'\in\hat{\mathcal{I}}^{(w)}_{\mathsf{large}}}\tilde{\mathcal{I}}(I')$. Combining the above inequality with Equation 23, and recalling that $\nu_1 = 2^{O((\log m)^{3/4}\log\log m)}$, we get that:

$$\sum_{\tilde{I}\in\tilde{\mathcal{I}}}\mathsf{OPT}_{\mathsf{cnwrs}}(\tilde{I}) \leq \sum_{I'\in\hat{\mathcal{I}}^{(w)}_{\mathsf{large}}}\mathsf{OPT}_{\mathsf{cnwrs}}(I')\cdot(\log m)^{c'_g} \leq (\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\cdot 2^{O((\log m)^{3/4}\log\log m)}. \qquad (24)$$

Consider now an instance $I'=(G',\Sigma')\in\hat{\mathcal{I}}^{(n)}_{\mathsf{large}}\cup\tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}$. Since we have assumed that event $\mathcal{E}_3$ does not happen, from the definition of a $\nu$-decomposition:

$$\mathbf{E}\left[\sum_{I''\in\overline{\mathcal{I}}(I')}\mathsf{OPT}_{\mathsf{cnwrs}}(I'')\right] \leq \big(\mathsf{OPT}_{\mathsf{cnwrs}}(I')+|E(G')|\big)\cdot\nu.$$

Recall that $\overline{\mathcal{I}}_{\mathsf{small}} = \bigcup_{I'\in\hat{\mathcal{I}}^{(n)}_{\mathsf{large}}\cup\tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}}\overline{\mathcal{I}}(I')$. Recall also that, from Inequality 1, $\sum_{I'=(G',\Sigma')\in\hat{\mathcal{I}}^{(n)}_{\mathsf{large}}}|E(G')| \leq m\cdot(\log m)^{c'_g}$, and from Inequality 3, $\sum_{I'=(G',\Sigma')\in\tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}}|E(G')| \leq 2m\cdot(\log m)^{c'_g}$. Altogether, we get that:

$$\begin{aligned}\mathbf{E}\left[\sum_{I''\in\overline{\mathcal{I}}_{\mathsf{small}}}\mathsf{OPT}_{\mathsf{cnwrs}}(I'')\right] &\leq \sum_{I'=(G',\Sigma')\in\hat{\mathcal{I}}^{(n)}_{\mathsf{large}}\cup\tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}}\left(\mathsf{OPT}_{\mathsf{cnwrs}}(I')+|E(G')|\right)\cdot\nu\\ &\leq \sum_{I'\in\hat{\mathcal{I}}^{(n)}_{\mathsf{large}}\cup\tilde{\mathcal{I}}^{(n)}_{\mathsf{large}}}\mathsf{OPT}_{\mathsf{cnwrs}}(I')\cdot\nu+4m\cdot(\log m)^{c'_g}\cdot\nu \qquad (25)\\ &\leq (\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\cdot 2^{O((\log m)^{3/4}\log\log m)}\end{aligned}$$

(we have used Equations 23 and 24 in order to bound $\sum_{I'\in\hat{\mathcal{I}}^{(n)}_{\mathsf{large}}}\mathsf{OPT}_{\mathsf{cnwrs}}(I')$ and $\sum_{I'\in\hat{\mathcal{I}}^{(n)}_{\mathsf{large}}}\mathsf{OPT}_{\mathsf{cnwrs}}(I')$, respectively, and the fact that $\nu_1,\nu\leq 2^{O((\log m)^{3/4}\log\log m)}$).
Finally, by combining Equations 23, 24 and 25, we get that:

$$\begin{aligned}\mathbf{E}\left[\sum_{I''\in\mathcal{I}^*}\mathsf{OPT}_{\mathsf{cnwrs}}(I'')\right]&\leq\mathbf{E}\left[\sum_{I''\in\hat{\mathcal{I}}_{\mathsf{small}}}\mathsf{OPT}_{\mathsf{cnwrs}}(I'')+\sum_{I''\in\tilde{\mathcal{I}}_{\mathsf{small}}}\mathsf{OPT}_{\mathsf{cnwrs}}(I'')+\sum_{I''\in\overline{\mathcal{I}}_{\mathsf{small}}}\mathsf{OPT}_{\mathsf{cnwrs}}(I'')\right]\\ &\leq(\mathsf{OPT}_{\mathsf{cnwrs}}(I)+m)\cdot 2^{O((\log m)^{3/4}\log\log m)}.\end{aligned}$$

We denote this expectation by $\eta'$. Let $\hat{\mathcal{E}}$ be the bad event that $\sum_{I''\in\mathcal{I}^*}\mathsf{OPT}_{\mathsf{cnwrs}}(I'')>100\eta'$. From Markov inequality, $\mathbf{Pr}\left[\hat{\mathcal{E}}\mid\neg\mathcal{E}\right]<1/100$.

# D Proofs Omitted from Section 4

## D.1 Proof of Claim 4.2

Denote $k=|\mathcal{P}|$ and $\rho=\mathrm{cong}_G(\mathcal{P})$. We define an undirected $s$-$t$ flow network $H$, as follows. We start with the graph $G$, and set the capacity of every edge in $G$ to be 1. We then add a source vertex $s$, that connects to every vertex $v\in V(G)$ with an edge of capacity $n_S(v)$, and a destination vertex $t$, that connects to every vertex $v\in V(G)$ with an edge of capacity $n_T(v)$. Notice that, by sending $1/\rho$ flow units on every path $P\in\mathcal{P}$, we obtain an $s$-$t$ flow of value $k/\rho$ in this network. From the integrality of flow, since all edge capacities in $H$ are integral, there is an integral $s$-$t$ flow in $H$, of value at least $k/\rho$. This integral flow defines the desired collection $\mathcal{P}'$ of at least $k/\rho$ edge-disjoint paths in graph $G$. We can use standard algorithms for computing maximum $s$-$t$ flow in order to obtain the set $\mathcal{P}'$ of paths with these properties.

## D.2 Proof of Observation 4.6

We first show that $S(\mathcal{P}')=S(\mathcal{P})$ and $T(\mathcal{P}')=T(\mathcal{P})$. We denote by $s$ and $t$ the first and the last endpoints of $P$, respectively, and by $s'$ and $t'$ the first and the last endpoints of $P'$, respectively. From the construction, the first endpoint of $\tilde{P}$ is $s$, the last endpoint of $\tilde{P}$ is $t'$, the first endpoint of $\tilde{P}'$ is $s'$, and the last endpoint of $\tilde{P}'$ is $t$. It is then immediate to verify that $S(\mathcal{P}')=S(\mathcal{P})$ and $T(\mathcal{P}')=T(\mathcal{P})$.
We now prove the second assertion. In order to do so, we assume that both $\tilde{P},\tilde{P}'$ are simple paths, and we will show that $|\Pi^T(\mathcal{P}')|<|\Pi^T(\mathcal{P})|$.
For every vertex $u\in V(G)$, let $N_1(u)$ be the number of triples of $\Pi^T(\mathcal{P})$ in which $u$ participates, and let $N_2(u)$ be the number of triples of $\Pi^T(\mathcal{P}')$ in which $u$ participates. It is enough to show that, for every vertex $u\in V(G)\setminus\{v\}$, $N_2(u)\leq N_1(u)$, and that $N_2(v)<N_1(v)$.

Consider some vertex $u \in V(G) \setminus \{v\}$. We will assign, to every triple $(Q, Q', u) \in \Pi^T(\mathcal{P}')$, a unique triple in $\Pi^T(\mathcal{P})$ that is responsible to it, and we will ensure that every triple in $\Pi^T(\mathcal{P})$ is responsible for at most one such triple.

Consider some triple $(Q, Q', u) \in \Pi^T(\mathcal{P}')$. If neither of the two paths $Q, Q'$ lies in $\left\{\tilde{P}, \tilde{P}'\right\}$, then triple $(Q, Q', u)$ lies in $\Pi^T(\mathcal{P})$ as well, and we make $(Q, Q', u)$ responsible for itself. If $Q = \tilde{P}$ and $Q' = \tilde{P}'$ (or the other way around), then either $Q$ is a subpath of $P$ and $Q'$ is a subpath of $P'$, or the other way around (we use the fact that paths $P, P'$ are simple, so $Q, Q'$ may not be subpaths of the same path). In either case, it is easy to see that triple $(P, P', u)$ lied in $\Pi^T(\mathcal{P})$. We make the triple $(P, P', u)$ responsible for triple $(Q, Q, u)$. The last case is when exactly one of the paths $Q, Q'$ is in $\left\{\tilde{P}, \tilde{P}'\right\}$. We assume w.l.o.g. that $Q = \tilde{P}$, and $Q' \notin \left\{\tilde{P}, \tilde{P}'\right\}$. If $u$ lies on path $P$ between its first vertex and $v$, then triple $(P, Q', u)$ lies in $\Pi^T(\mathcal{P})$, and we make it responsible for $(Q, Q', u)$. Otherwise, triple $(P', Q', u)$ lies in $\Pi^T(\mathcal{P})$, and we make it responsible for $(Q, Q', u)$.

It is easy to see that every triple $(\hat{Q}, \hat{Q}', u) \in \Pi^T(\mathcal{P})$ is responsible for at most one triple in $\Pi^T(\mathcal{P}')$. Indeed, if neither of $\hat{Q}, \hat{Q}'$ lies in $\{P, P'\}$, then triple $(\hat{Q}, \hat{Q}', u)$ may only be responsible for itself. If both $\hat{Q}, \hat{Q}' \in \{P, P'\}$, then triple $(P, P', u)$ may only be responsible for triple $(\tilde{P}, \tilde{P}', u)$. If exactly one of $\hat{Q}, \hat{Q}'$ lies in $\{P, P'\}$, for example, $\hat{Q} = P$, then two cases are possible: if vertex $u$ lies between the first endpoint of $P$ and $v$, then triple $(P, Q', u)$ may only be responsible for triple $(\tilde{P}, Q', u)$, and otherwise it may only be responsible for triple $(\tilde{P}', Q', u)$. We conclude that $N_2(u) \leq N_1(u)$.

Consider now the case where $u = v$, and consider some triple $(Q, Q', v) \in \Pi^T(\mathcal{P}')$. If neither of the two paths $Q, Q'$ lies in $\left\{\tilde{P}, \tilde{P}'\right\}$, then triple $(Q, Q', v)$ lies in $\Pi^T(\mathcal{P})$, and we make $(Q, Q', v)$ responsible for itself. Note that, in case where $u = v$, it is impossible that the triple $(\tilde{P}, \tilde{P}', v)$ lies in $\Pi^T(\mathcal{P}')$. Therefore, it remains to consider the triples $(Q, Q', v)$, where exactly one of the paths $Q, Q'$ lies in $\left\{\tilde{P}, \tilde{P}'\right\}$. We call such triples *problematic triples*, and we assume w.l.o.g. that in each such triple, $Q \notin \left\{\tilde{P}, \tilde{P}'\right\}$. If path $Q$ participates in a problematic triple, then we say that path $Q$ is a *problematic path*.

We denote by $e_a, e'_a$ the two edges on path $P$ that are incident to vertex $v$, and we assume that $e_a$ appears before $e'_a$ on $P$. We denote by $e_b, e'_b$ the two edges on path $P'$ that are incident to $v$, and we assume that $e_b$ appears before $e'_b$ on $P'$. Recall that path $\tilde{P}$ contains edges $e_a$ and $e'_b$, while path $\tilde{P}'$ contains edges $e_b$ and $e'_a$. Recall that edges $e_a, e_b, e'_a, e'_b$ must appear in this circular order in $\mathcal{O}_v \in \Sigma$, since paths $P$ and $P'$ are transversal (recall that the ordering is unoriented). We use the edges of $\{e_a, e_b, e'_a, e'_b\}$ to partition the edge set $\delta_G(v) \setminus \{e_a, e_b, e'_a, e'_b\}$ into four subsets: set $E_1$ of edges appearing between $e_a$ and $e_b$ in $\mathcal{O}_v$; set $E_2$ of edges appearing between $e_b$ and $e'_a$; set $E_3$ of edges appearing between $e'_a$ and $e'_b$, and set $E_4$ of all remaining edges, that must appear between $e'_b$ and $e_a$ (see Figure 42).

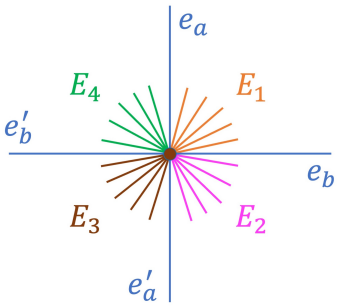



Figure 42: A schematic view of edges $e_a$, $e_b$, $e'_a$, $e'_b$ and edge sets $\{E_i\}_{1 \leq i \leq 4}$.

Consider now some problematic path $Q$, and denote by $e(Q), e'(Q)$ the two edges that lie on $Q$ and are incident to $v$. Note that $e(Q), e'(Q)$ must lie in different sets of $\{E_1, \ldots, E_4\}$. Since path $\tilde{P}$ contains edges $e_a, e'_b$, while path $\tilde{P}'$ contains edges $e'_a, e_b$, in order for path $Q$ to be problematic, at least one of the two edges $e(Q), e'(Q)$ must lie in one of the sets $E_2, E_4$. We assume w.l.o.g. that $e(Q) \in E_2$. If $e'(Q) \in E_4$, then both $(Q, \tilde{P}, v)$ and $(Q, \tilde{P}', v)$ are problematic pairs. But in this case, both $(Q, P, v)$ and $(Q, P', v)$ lied in $\Pi^T(\mathcal{P})$. We make triple $(Q, P, v)$ responsible for $(Q, \tilde{P}, v)$, and we make triple $(Q, P', v)$ responsible for $(Q, \tilde{P}', v)$. Otherwise, $e'_Q \in E_1$ or $e'_Q \in E_3$ must hold. In the either case, the only problematic triple involving path $Q$ is $(Q, \tilde{P}', v)$. In the former case, $(Q, P', v) \in \Pi^T(\mathcal{P})$, and we make this triple responsible for $(Q, \tilde{P}', v)$, while in the latter case, $(Q, P, v) \in \Pi^T(\mathcal{P})$, and we make this triple responsible for $(Q, \tilde{P}', v)$. So far, we have assigned, to every triple $(Q, Q', v) \in \Pi^T(\mathcal{P}')$, a distinct triple $(\hat{Q}, \hat{Q}', v) \in \Pi^T(\mathcal{P})$ that is responsible for it. Note that triple $(P, P', v)$ is not responsible for any triple $(Q, Q', v) \in \Pi^T(\mathcal{P}')$, so $N_2(v) < N_1(v)$. We conclude that $|\Pi^T(\mathcal{P}')| < |\Pi^T(\mathcal{P})|$.

### D.3 Proof of Lemma 4.7

We first preprocess the set $\mathcal{R}$ of paths by removing cycles from the paths, to obtain a collection $\mathcal{R}$ of simple paths. The algorithm is iterative. Throughout the algorithm, we maintain a set $\hat{\mathcal{R}}$ of paths in $G$, that is initialized to be $\mathcal{R}$. The algorithm proceeds in iterations, as long as $\Pi^T(\hat{\mathcal{R}}) \neq \emptyset$. An iteration is executed as follows. Let $(P, P', v)$ be any triple in $\Pi^T(\hat{\mathcal{R}})$. We perform path splicing of $P$ and $P'$ at vertex $v$, obtaining two new paths $\tilde{P}$ and $\tilde{P}'$. We then remove cycles from $\tilde{P}$ and $\tilde{P}'$, to obtain two simple paths, which are then added to $\hat{\mathcal{R}}$, replacing the paths $P$ and $P'$. Note that, from Observation 4.6, multisets $S(\hat{\mathcal{R}}), T(\hat{\mathcal{R}})$ remain unchanged after the execution of the iteration. It is also easy to verify, from the definition of the splicing procedure, that, for every edge $e \in E(G)$, $\mathrm{cong}_G(\hat{\mathcal{R}}, e)$ may not increase after the iteration execution. Moreover, if the paths $\tilde{P}, \tilde{P}'$ obtained after the splicing procedure are simple, then $|\Pi^T(\mathcal{R})|$ is guaranteed to decrease after the current iteration, while otherwise, $\sum_{R \in \hat{\mathcal{R}}} |E(R)|$ must decrease. We conclude that, after every iteration of the algorithm, either $\sum_{R \in \hat{\mathcal{R}}} |E(R)|$ decreases, or $\sum_{R \in \hat{\mathcal{R}}} |E(R)|$ remains unchanged and $|\Pi^T(\hat{\mathcal{R}})|$ decreases. Since $|\Pi^T(\hat{\mathcal{R}})| \leq |\hat{\mathcal{R}}|^2 \cdot |V(G)|$, the number of iterations in the algorithm is bounded by $|\hat{\mathcal{R}}|^2 \cdot |V(G)| \cdot |E(G)|$, and so the algorithm is efficient. The output $\mathcal{R}'$ of the algorithm is the set $\hat{\mathcal{R}}$ of paths that is obtained when the algorithm terminates. From the above discussion, we get that $S(\mathcal{R}') = S(\mathcal{R})$ and $T(\mathcal{R}') = T(\mathcal{R})$. Once the algorithm terminates, the paths in set $\mathcal{R}' = \hat{\mathcal{R}}$ are non-transversal with respect to $\Sigma$. Lastly, from the above discussion, for every edge $e \in E(G)$, $\mathrm{cong}_G(\hat{\mathcal{R}}, e)$ may not increase over the course of the algorithm, and so $\mathrm{cong}_G(\mathcal{R}', e) \leq \mathrm{cong}_G(\mathcal{R}, e)$ must hold.

### D.4 Proof of Lemma 4.8

We use the following claim.

**Claim D.1** *Let $G$ be a graph, $S$ a subset of vertices in $G$, and $x, y \in S$ two distinct vertices. Assume that $(A, B)$ is a minimum cut separating $x$ from $S \setminus \{x\}$ with $x \in A$, and $(A', B')$ is a minimum cut separating $y$ from $S \setminus \{y\}$ with $y \in A'$. Consider another cut $(\hat{A}, \hat{B})$, where $\hat{A} = A \setminus A'$, and $\hat{B} = V(G) \setminus \hat{A}$. Then $(\hat{A}, \hat{B})$ is a minimum cut separating $x$ from $S \setminus \{x\}$ in $G$.*

**Proof:** Since $(A', B')$ is a cut separating $y$ from $S \setminus \{y\}$ with $y \in A'$, we get that $A' \cap S = \{y\}$. Similarly, $A \cap S = \{x\}$. Therefore, $\hat{A} \cap S = \{x\}$, and so $(\hat{A}, \hat{B})$ is indeed a cut separating $x$ from $S \setminus \{x\}$. It now remains to show that $|E(\hat{A}, \hat{B})| \leq |E(A, B)|$. Denote $\hat{A}' = A' \setminus A$, and $\hat{B}' = V(G) \setminus \hat{A}'$. Using the same argument as above, $(\hat{A}', \hat{B}')$ is a cut separating $y$ from $S \setminus \{y\}$.

From submodularity of cuts, for any pair $X, Y$ of vertex subsets in a graph $G$, $|\delta_G(X)| + |\delta_G(Y)| \geq$

$|\delta_G(X \setminus Y)| + |\delta_G(Y \setminus X)|$. Therefore:

$$|\delta_G(A)| + |\delta_G(A')| \geq |\delta_G(A \setminus A')| + |\delta_G(A' \setminus A)| = |\delta_G(\hat{A})| + |\delta_G(\hat{A}')|.$$

Notice however that $(A, B)$ is a minimum cut separating $x$ from $S \setminus \{x\}$, so $|\delta_G(A)| = |E(A, B)| \leq |E(\hat{A}, \hat{B})| = |\delta_G(\hat{A})|$. Similarly, since $(A', B')$ is a minimum cut separating $y$ from $S \setminus \{y\}$, we get that $|\delta_G(A')| = |E(A', B')| \leq |E(\hat{A}', \hat{B}')| = |\delta_G(\hat{A}')|$. We conclude that $|\delta_G(A)| + |\delta_G(A')| = |\delta_G(\hat{A})| + |\delta_G(\hat{A}')|$ must hold. If we assume for contradiction that $(\hat{A}, \hat{B})$ is not a minimum cut separating $x$ from $S \setminus \{x\}$, then $|\delta_G(A)| < |\delta_G(\hat{A})|$ must hold, and so $|\delta_G(A')| > |\delta_G(\hat{A}')|$, a contradiction to the minimality of the cut $(A', B')$. We conclude that $(\hat{A}, \hat{B})$ is a minimum cut separating $x$ from $S \setminus \{x\}$. □

We now complete the proof of Lemma 4.8 using Claim D.1. Recall that we are given a set $S = \{s_1, \ldots, s_k\}$ of vertices of graph $G$. We first compute, for all $1 \leq i \leq k$, a minimum cut separating $\{s_i\}$ from $S \setminus \{s_i\}$ in $G$, that we denote by $(U_i, \overline{U_i})$, with $s_i \in U_i$. For each $1 \leq i \leq k$, we then let $A_i = U_i \setminus (\bigcup_{1 \leq j \leq k, j \neq i} U_j)$. Clearly, for all $1 \leq i < j \leq k$, $A_i \cap A_j = \emptyset$.

Consider now some index $1 \leq i \leq k$. We claim that $(A_i, V(G) \setminus A_i)$ is a minimum cut separating $s_i$ from $S \setminus \{s_i\}$ in graph $G$. For convenience, assume that $i = k$ (the other cases are symmetric). For all $1 \leq j < k$, let $Z_j = U_k \setminus (U_1 \cup U_2 \cup \cdots \cup U_j)$, so that $Z_{k-1} = A_k$. Set $Z_0 = U_k$. By applying Claim D.1 to each of the sets $Z_0, \ldots, Z_{k-1}$ in turn, and using the fact that, for all $1 \leq j \leq k-1$, $Z_j = Z_{j-1} \setminus U_j$, we get that, for all $0 \leq j \leq k-1$, $(Z_j, V(G) \setminus Z_j)$ is a minimum cut separating $s_k$ from $S \setminus \{s_k\}$ in $G$.

It remains to compute, for each $1 \leq i \leq k$, a set $\mathcal{Q}_i$ of paths routing the edges of $\delta(A_i)$ to $s_i$. Fix an index $1 \leq i \leq k$. We construct a flow network as follows. Let $H_i$ be the graph obtained from $G[A_i] \cup \delta_G(A_i)$, by contracting all vertices that do not belong to $A_i$ into a single vertex, that we denote by $t_i$. We set the capacity of every edge in $H_i$ to be 1, and compute a maximum $s_i$-$t_i$ flow in the resulting network. From the max-flow / min-cut theorem, the value of the resulting flow must be $|\delta_G(A_j)|$, and from the integrality of flow we can ensure that the resulting flow is integral. We can then use this flow to obtain a set $\mathcal{Q}_i = \{Q_i(e) \mid e \in \delta_G(e)\}$ of edge-disjoint paths, where, for all $e \in \delta_G(e)$, path $Q_i(e)$ has $e$ as its first edge, $s_i$ as its last vertex, and all inner vertices of $Q_i(e)$ are contained in $A_i$.

## D.5 Proof of Theorem 4.11

Let $0 < \eta < 1$ be some parameter. In order to avoid confusion, throughout this proof, we will refer to $\eta$-balanced cuts as $\eta$-edge-balanced cuts. We now define the notion of $\eta$-vertex-balanced cuts, that will be used in this proof. We say that a cut $(A, B)$ in a graph $G$ is *$\eta$-vertex-balanced* if $|A|, |B| \leq \eta \cdot |V(G)|$. We say that a cut $(A, B)$ is a *minimum $\eta$-vertex-balanced cut in $G$* if $(A, B)$ is an $\eta$-vertex-balanced cut of minimum value $|E(A, B)|$. We need the following theorem.

**Theorem D.2 (Corollary 2 in [ARV09])** *For every constant $1/2 < \eta < 1$, there is another constant $\eta < \eta' < 1$, and an efficient algorithm, that, given any **simple** connected graph $G$ with $n$ vertices, computes an $\eta'$-vertex-balanced cut $(A, B)$ in $G$, whose value $|E(A, B)|$ is at most $\beta_{\text{ARV}}(n)$ times the value of a minimum $\eta$-vertex-balanced cut of $G$.*

We now turn to prove Theorem 4.11. Let $G$ be the input graph, with $|E(G)| = m$. For every vertex $v \in V(G)$, we denote by $d_v = \deg_G(v)$ the degree of $v$ in $G$. For each such vertex $v \in V(G)$, we denote $\delta_G(v) = \{e_1(v), \ldots, e_{d_v}(v)\}$, where the edges are indexed arbitrarily, and we let $K_v$ be a complete graph on $d_v$ vertices. We denote $V(K_v) = \{x_1(v), \ldots, x_{d_v}(v)\}$.

We construct a new graph $H$ as follows. First, we let $H$ be a disjoint union of graphs $K_v$, for all $v \in V(G)$. We call all edges in $\bigcup_{v \in V(G)} E(K_v)$ *internal edges*. Next, we consider the edges of the

graph $G$ one by one. Consider any such edge $e = (v, v')$, and assume that $e = e_i(v) = e_j(v')$. In other words, $e$ is the $i$th edge incident to $v$ and the $j$th edge incident to $v'$. We add the edge $e' = (x_i(v), x_j(v'))$ to graph $H$, and we view this edge as the *copy of the edge* $e$. We call the resulting set $\{e' \mid e \in E(H)\}$ of edges *external edges of* $H$. This completes the definition of the graph $H$. Note that $|V(H)| = \sum_{v \in V(G)} d_v = 2m$, and every vertex of $H$ is incident to exactly one external edge.

Consider now any cut $(A', B')$ in graph $H$. We say that cut $(A', B')$ is *canonical* if, for every vertex $v \in V(G)$, either $V(K_v) \subseteq A'$, or $V(K_v) \subseteq B'$.

Let $(X, Y)$ be a minimum $\hat{\eta}$-edge-balanced cut in graph $G$, and let $\rho = |E_G(X, Y)|$ denote its value. We start with the following observation.

**Observation D.3** *There is an $\eta_1$-vertex-balanced cut in graph $H$ of value at most $\rho$, for $\eta_1 = \frac{1+\hat{\eta}}{2}$.*

**Proof:** We construct a cut $(X', Y')$ in graph $H$ using the cut $(X, Y)$ in $G$, as follows. We start with $X', Y' = \emptyset$. For every vertex $v \in V(H)$, if $v \in X$, then we add all vertices of $K_v$ to $X'$, and otherwise we add them to $Y'$. It is immediate to verify that the value of the resulting cut $(X', Y')$ is $|E_H(X', Y')| = |E_G(X, Y)| = \rho$.

We now show that cut $(X', Y')$ is $\eta_1$-vertex-balanced. In order to do so, it is enough to show that $|X'|, |Y'| \leq \eta_1 \cdot |V(H)|$. We show that $|X'| \leq \eta_1 \cdot |V(H)|$. The proof for $Y'$ is symmetric. Indeed:

$$|X'| = \sum_{v \in X} |V(K_v)| = \sum_{v \in X} d_v = 2|E_G(X)| + |E_G(X, Y)| \leq |E_G(X)| + m \leq \hat{\eta} m + m \leq \frac{1+\hat{\eta}}{2} \cdot |V(H)|,$$

which is bounded by $\eta_1 |V(H)|$ (we have used the fact that $|V(H)| = 2m$). □

We can now use the algorithm from Theorem D.2 to compute an $\eta_2$-vertex-balanced cut $(X', Y')$ in graph $H$, whose value is at most $\rho' = \beta_{\text{ARV}}(2m) \cdot \rho$. Here, $\eta_1 < \eta_2 < 1$ is some constant. Note that, if cut $(X', Y')$ were canonical, we could immediately obtain a corresponding cut $(A, B)$ in graph $G$, whose value is at most $\rho'$, with the guarantee that $(A, B)$ is a $\hat{\eta}'$-edge-balanced cut, for some constant $\hat{\eta}'$. We use the following observation in order to convert the cut into a canonical one.

**Observation D.4** *There is an efficient algorithm, that, given an $\eta'$-vertex-balanced cut $(X', Y')$ in graph $H$ of value $\rho'$, for some $0 < \eta' < 1$, computes a canonical $\eta^*$-vertex-balanced cut $(X^*, Y^*)$ in graph $H$ of value $\rho^* \leq O(\rho')$, for $\eta^* = \max\left\{\frac{1+\eta'}{2}, 0.95\right\}$.*

**Proof:** For every vertex $v \in V(G)$, we denote $X_v = X' \cap V(K_v)$ and $Y_v = Y' \cap V(K_v)$. Notice that graph $K_v$ contributes $|X_v| \cdot |Y_v|$ edges to the cut $(X', Y')$. We say that vertex $v$ is *indecisive* if $|X_v|, |Y_v| \geq \frac{1-\eta'}{2} \cdot d_v$, and we say that it is *decisive* otherwise.

We modify the cut $(X', Y')$ in two steps. In the first step, we construct a new cut $(X'', Y'')$ in graph $H$ as follows. We start from $(X'', Y'') = (X', Y')$. We then consider every decisive vertex $v \in V(G)$ one by one. Consider any such vertex $v$, and recall that either $|X_v| < \frac{1-\eta'}{2} \cdot d_v$ holds, or $|Y_v| < \frac{1-\eta'}{2} \cdot d_v$. In the former case, we move the vertices of $X_v$ to $Y''$, while in the latter case we move the vertices of $Y_v$ to $X''$. Notice that $|X_v| \cdot |Y_v|$ edges of $K_v$ lie in the cut $(X', Y')$. At the end of the current iteration, no internal edges of $K_v$ contribute to the cut $(X'', Y'')$, but we may have added new external edges to the cut: if vertices of $X_v$ were moved to $Y''$, then we may have added up to $|X_v|$ such new edges (edges incident to vertices of $X_v$), and otherwise we may have added up to $|Y_v|$ such new edges. In either case, it is easy to see that $|E(X'', Y'')|$ may not grow as the result of the current iteration. The first step terminates once every decisive vertex of $G$ is processed. Notice that the total number of new vertices that we may have added to set $X''$ over the course of this step is at most:

$$\frac{1-\eta'}{2} \cdot \sum_{v \in V(G)} d_v \leq (1-\eta')m.$$

Since we are guaranteed that $|X'| \leq \eta' \cdot (2m)$, we get that, at the end of the current step, $|X''| \leq \eta' \cdot (2m) + (1-\eta')m \leq \frac{1+\eta'}{2} \cdot (2m)$ holds. Similarly, $|Y''| \leq \frac{1+\eta'}{2} \cdot (2m)$. We conclude that $(X'', Y'')$ is an $\eta''$-vertex-balanced cut in $H$, of value at most $\rho'$, where $\eta'' = \frac{1+\eta'}{2}$.

In the second step, we construct the final cut $(X^*, Y^*)$ in $H$ by taking care of indecisive vertices. Assume first that there is some indecisive vertex $v \in V(H)$ with $d_v \geq m/10$. Notice that, in this case, the number of edges that graph $K_v$ contributes to cut $(X', Y')$ is at least $|X_v| \cdot |Y_v| \geq \frac{1-\eta'}{4} \cdot (d_v)^2 \geq \frac{1-\eta'}{400} \cdot m^2$. Therefore, $\rho' > \frac{1-\eta'}{400} \cdot m^2$ must hold. Consider now a new cut $(X^*, Y^*)$ in graph $H$, where $X^* = V(K_v)$ and $Y^* = V(H) \setminus X^*$. Notice that $|X^*| = d_v \leq m$ and $|Y^*| \leq 2m - |X^*| \leq 2m \cdot 0.95$ holds. Therefore, cut $(X^*, Y^*)$ is 0.95-vertex-balanced. Additionally, $|E_H(X^*, Y^*)| \leq d_v \leq m \leq O(\rho')$. We then return the cut $(X^*, Y^*)$ as the outcome of the algorithm. We assume from now on that for every indecisive vertex $v \in V(H)$, $d_v < m/10$.

We start with $(X^*, Y^*) = (X'', Y'')$, and then process every indecisive vertex $v \in V(G)$ one by one. Consider an iteration when vertex $v$ is processed. Recall that graph $K_v$ contributes at least $|X_v| \cdot |Y_v| \geq \max\{|X_v|, |Y_v|\}$ edges to the cut $(X^*, Y^*)$. If $|X^*| < |Y^*|$, then we move the vertices of $Y_v$ from $Y^*$ to $X^*$. Notice that, after this transformation, the inner edges of $K_v$ no longer contribute to the cut, and at most $|Y_v|$ new outer edges are added to the cut. Therefore, the value of the cut does not increase. Otherwise, $|X^*| \geq |Y^*|$, and we move the vertices of $X_v$ from $X^*$ to $Y^*$. Using the same argument as before, the value of the cut does not increase. This completes the description of an iteration. Consider the cut $(X^*, Y^*)$ that is obtained at the end of the algorithm, after all indecisive vertices are processed.

We now show that $|X^*|, |Y^*| \leq \eta^* \cdot (2m)$. We prove this for $X^*$, and the proof for $Y^*$ is symmetric. We consider two cases. First, if no new vertices were added to $X^*$ over the course of the second step, then $|X^*| \leq |X''| \leq \frac{1+\eta'}{2} \cdot (2m)$. Assume now that some vertices were added to $X^*$, and let $v$ be the last indecisive vertex of $G$, for which the vertices of $K_v$ were added to $X^*$. Then before vertex $v$ was processed, $|X^*| \leq |Y^*|$ held. Since $d_v \leq m/10$ from our assumption, at the end of the iteration when $v$ was processed, $|X^*| \leq 1.1m$ held. Since no new vertices were added to $X^*$ in subsequent iterations, we get that $|X^*| \leq 1.1m \leq \eta^* \cdot (2m)$ holds at the end of the algorithm. We conclude that $(X^*, Y^*)$ is an $\eta^*$-balanced cut, of value at most $O(\rho')$. □

By applying the algorithm from Observation D.4 to the $\eta'$-vertex-balanced cut $(X', Y')$ in graph $H$, we obtain a canonical $\eta^*$-vertex-balanced cut $(X^*, Y^*)$ in graph $H$, with $\eta^* = \max\left\{\frac{1+\eta'}{2}, 0.95\right\}$, whose value is $\rho^* \leq O(\rho') = O(\beta_{\text{ARV}}(m)) \cdot \rho$. We use this cut in order to construct a cut $(A, B)$ in $G$ as follows: for every vertex $v \in V(G)$, if $V(K_v) \subseteq X^*$, then vertex $v$ is added to $A$, and otherwise it is added to $B$. Notice that $|E_G(A)| \leq \sum_{v \in A} d_v/2 \leq |X^*|/2 \leq \eta^* \cdot m$. Similarly, $|E_G(B)| \leq \eta^* \cdot m$. Therefore, cut $(A, B)$ is $\eta^*$-edge-balanced. Additionally, $|E_G(A, B)| \leq |E_H(A^*, B^*)| \leq O(\beta_{\text{ARV}}(m)) \cdot \rho$.

## D.6 Proof of Theorem 4.12

We use the following theorem from [LT79].

**Theorem D.5 (Theorem 4 from [LT79])** *Let $G$ be any **simple** $n$-vertex planar graph with weights $w_v \geq 0$ on its vertices $v \in V(G)$, such that $\sum_{v \in V(G)} w_v \leq 1$. Then there is a partition $(A, B, C)$ of $V(G)$, such that no edge connects a vertex of $A$ to a vertex in $B$; $\sum_{v \in A} w_v, \sum_{v \in B} w_v \leq 2/3$; and $|C| \leq \sqrt{8n}$.*

In order to prove Lemma 4.12, we define a new simple planar graph $G'$ that is obtained by modifying graph $G$, using its optimal drawing. We then apply Theorem D.5 to graph $G'$, and transform the resulting partition $(A, B, C)$ of $V(G')$ into a (3/4)-edge-balanced cut of graph $G$, whose value is at most $O(\sqrt{\mathsf{OPT}_{\mathsf{cr}}(G) + \Delta \cdot m})$.

In order to define graph $G'$, we first define an intermediate graph $G_1$. Consider the input graph $G$ and its optimal drawing $\varphi$ in the plane. For every vertex $v \in V(G)$, we denote $d_v = \deg_G(v)$, and we denote $\delta_G(v) = \{e_1(v), \ldots, e_{d_v}(v)\}$, where the edges are indexed according to the order in which their images enter the image of $v$ in $\varphi$, in the counter-clock-wise direction. We let $H_v$ be the $(d_v \times d_v)$-grid, and we denote the set of the vertices on the first row of the grid by $X(v) = \{x_1(v), \ldots, x_{d_v}(v)\}$, where the vertices are indexed in their natural order. In order to define graph $G_1$, we start with the disjoint union of all grids in $\{H_v\}_{v \in V(G)}$. We refer to the edges that lie in these grids as *internal edges*. Next, we process every edge $e \in E(G)$ one by one. Consider any such edge $e = (v, v')$, and assume that $e = e_i(v) = e_j(v')$, that is, $e$ is the $i$th edge incident to $v$ and the $j$th edge incident to $v'$. We then add an edge $e' = (x_i(v), x_j(v'))$ to graph $G_1$. We think of edge $e'$ as the *copy* of the edge $e$ in $G_1$. The edges in set $\{e' \mid e \in E(G)\}$ are called *external edges* of graph $G_1$. We say that a cut $(A, B)$ in graph $G_1$ is *canonical* if, for every vertex $v \in V(G)$, either $V(H_v) \subseteq A$, or $V(H_v) \subseteq B$ holds. Note that a canonical cut $(A, B)$ in graph $G_1$ naturally defines a cut $(A', B')$ of the same value on graph $G$, where a vertex $v \in V(G)$ is added to $A'$ if $V(H_v) \subseteq A$, and it is added to $B'$ otherwise. Lastly, note that the optimal drawing $\varphi$ of $G$ defines a drawing $\varphi_1$ of $G_1$ with the same number of crossings. In order to obtain drawing $\varphi_1$ of $G_1$, we start with the drawing $\varphi$ of $G$, and then inflate the image of every vertex $v \in V(G)$, so that it becomes a disc $D(v)$. We place another smaller disc $D'(v)$ inside $D(v)$, so that the boundaries of both discs are disjoint. We then place the standard drawing of the grid $H_v$ inside disc $D'(v)$, so that vertices $x_1(v), \ldots, x_{d_v}(v)$ appear on the boundary of the disc $D'(v)$ in this counter-clock-wise order. By slightly extending the images of the edges $e_1(v), \ldots, e_{d_v}(v)$ inside $D(v) \setminus D'(v)$, we can ensure that the image of each such edge $e_i(v)$ terminates at the image of the vertex $x_i(v)$. Once all vertices of $V(G)$ are processed in this manner, we obtain a drawing $\varphi_1$ of graph $G_1$, in which the number of crossings is bounded by $\mathsf{cr}(\varphi) = \mathsf{OPT}_{\mathsf{cr}}(G)$.

In order to obtain the final graph $G'$, we start with $G' = G_1$, and we denote $V(G_1) = X$. Next, for every **external** edge $e' \in E(G_1)$, we subdivide the edge with a new vertex $u_{e'}$. In other words, if $e' = (x_i(v), x_j(v'))$, then we replace the edge with a path consisting of two edges: $(x_i(v), u_{e'})$, and $(u_{e'}, x_j(v'))$. We denote this new set of vertices representing the external edges of $G_1$ by $U = \{u_{e'} \mid e \in E(G)\}$. Note that drawing $\varphi_1$ of graph $G_1$ can be easily transformed into a drawing of this new graph, without increasing the number of crossings. Denote the resulting drawing by $\varphi_2$. In our last step, for every crossing point $p$ between a pair $a, a'$ of edges in drawing $\varphi_2$, we replace point $p$ with a new vertex $y_p$. In other words, if $a = (s, t)$ and $a' = (s', t')$, then we add a new vertex $y_p$ to the graph. We then replace edge $a = (s, t)$ with two new edges, $(s, y_p)$ and $(y_p, t)$, and we similarly replace edge $a'$ with two new edges, $(s', y_p)$ and $(y_p, t')$. We continue processing every crossing point in drawing $\varphi_2$ one by one in this manner, until no more crossings remain. We denote this new set of vertices, that represent all crossing points in the original drawing $\varphi_2$, by $Y$. Note that $|Y| = \mathsf{OPT}_{\mathsf{cr}}(G)$. This completes the definition of the graph $G'$. Observe that $V(G') = X \cup Y \cup U$, and so:

$$|V(G')| = \sum_{v \in V(G)} (d_v)^2 + m + \mathsf{OPT}_{\mathsf{cr}}(G) \leq \Delta \cdot \sum_{v \in V(G)} d_v + m + \mathsf{OPT}_{\mathsf{cr}}(G) \leq 3\Delta m + \mathsf{OPT}_{\mathsf{cr}}(G).$$

It is also immediate to verify that $G'$ is a simple planar graph, and that the maximum vertex degree in $G'$ is at most 4. We now assign weights $w_v$ to vertices $v \in V(G')$, as follows: every vertex $u_{e'} \in U$ is assigned weight $1/m$, and all other vertices are assigned weight 0. It is immediate to verify that $\sum_{v \in V(G')} w_v = 1$.

From Theorem D.5, there is a partition $(A, B, C)$ of $V(G')$, such that no edge connects a vertex of $A$ to a vertex in $B$; $\sum_{v \in A} w_v, \sum_{v \in B} w_v \leq 2/3$; and $|C| \leq \sqrt{8|V(G')|} \leq \sqrt{24\Delta m + 8\mathsf{OPT}_{\mathsf{cr}}(G)}$. We convert this partition of vertices of $G'$ into a $(3/4)$-balanced cut in graph $G$ in three steps. In the first step, we use the partition $(A, B, C)$ of $V(G')$ in order to construct a cut $(A_1, B_1)$ in graph $G_1$. In the second step, we transform this cut into a canonical cut $(A_2, B_2)$ in graph $G_1$. Lastly, in the third step,

we use this canonical cut in order to define the final cut $(A^*, B^*)$ in graph $G$. We now describe each of the steps in turn.

**Step 1: Cut in Graph $G_1$.** We define a cut $(A_1, B_1)$ in graph $G_1$ as follows. We start with $A_1 = B_1 = \emptyset$, and then process every vertex $v \in V(G)$ one by one. When vertex $v \in V(G)$ is processed, we consider every vertex $x \in V(H_v)$. If $x \in A \cup C$, then we add $x$ to $A_1$, and otherwise we add $x$ to $B_1$.

Consider now the resulting cut $(A_1, B_1)$ in graph $G_1$. We first claim that $|E_{G_1}(A_1, B_1)| \leq 4|C|$. In order to prove this, we assign, to every edge $e \in E_{G_1}(A_1, B_1)$, some vertex $x \in C$ that is *responsible* for $e$, and we will ensure that every vertex of $C$ is responsible for at most 4 edges of $E_{G_1}(A_1, B_1)$. Consider some edge $e \in E_{G_1}(A_1, B_1)$. If either of the endpoints of $e$ lies in set $C$, then we assign $e$ to that endpoint. Otherwise, there must be some vertex $x$ of graph $G'$ that subdivided the edge $e$ (so either $x \in U$ or $x \in Y$ holds), and $x \in C$. In this case, we assign $e$ to this vertex $x$. Since the degree of every vertex in $G'$ is at most 4, every vertex of $C$ may be assigned to at most 4 edges of $E_{G_1}(A_1, B_1)$, and so we conclude that $|E_{G_1}(A_1, B_1)| \leq 4|C| \leq 4 \cdot \sqrt{24\Delta m + 8\mathsf{OPT}_{\mathsf{cr}}(G)}$.

Next, we bound the total number of external edges in $E_{G_1}(A_1)$ and in $E_{G_1}(B_1)$. For convenience, denote by $E'$ the set of all external edges in graph $G_1$. Consider some external edge $e' = (x_i(v), x_j(v')) \in E'$. Let $P(e')$ be the path that replaced the edge $e'$ in graph $G'$. Recall that path $P(e')$ is a path connecting $x_i(v)$ to $x_j(v')$, it contains the vertex $u_{e'}$ representing the edge $e'$, and possibly additional vertices representing the crossing points of edge $e'$ with other edges. We claim that either vertex $y_{e'}$ lies in $A$, or some vertex of $P(e')$ (including possibly $x_i(v)$ or $x_j(v')$) must lie in $C$. Indeed, assume that $y_{e'} \notin A$. If none of the vertices of $P(e')$ lie in $C$, then $y_{e'} \in B$, while $x_i(v), x_j(v') \in A$ must hold. This is impossible since there are no edges connecting vertices of $A$ to vertices of $B$. Therefore, either $y_{e'} \in A$, or at least one vertex on $P(e')$ lies in $C$. Since every vertex of $G'$ has degree at most 4, every vertex of $C$ may lie on at most 4 paths in $\{P(e') \mid e \in E(G)\}$. Since the weight of every vertex in $U$ is $1/m$, we get that:

$$|E' \cap E_{G_1}(A_1)| \leq |U \cap A| + 4|C| \leq m \cdot \sum_{v \in A} w_v + 4|C| \leq 2m/3 + 4 \cdot \sqrt{24\Delta m + 8\mathsf{OPT}_{\mathsf{cr}}(G)}.$$

Using similar reasoning, $|E' \cap E_{G_1}(B_1)| \leq 2m/3 + 4 \cdot \sqrt{24\Delta m + 8\mathsf{OPT}_{\mathsf{cr}}(G)}$.

**Step 2: Canonical Cut in Graph $G_1$.** In this step we construct a cut $(A_2, B_2)$ in graph $G_1$ that is canonical, by gradually modifying the cut $(A_1, B_1)$. For every vertex $v \in V(G)$, we denote $n_A(v) = |X(v) \cap A_1|$, and $n_B(v) = |X(v) \cap B_1|$. We use the following simple observation.

**Observation D.6** *For every vertex $v \in V(G)$, $|E(H_v) \cap E_{G_1}(A_1, B_1)| \geq \min\{n_A(v), n_B(v)\}$.*

**Proof:** We partition the columns of the grid $H_v$ into two subsets, $\mathcal{W}', \mathcal{W}''$, as follows. For $1 \leq i \leq d_v$, the $i$th column of the grid is added to set $\mathcal{W}'$ if vertex $x_i(v)$ (the vertex of the $i$th column that lies on the first row of the grid) lies in $A_1$. Otherwise, the $i$th column is added to $\mathcal{W}''$.

We now consider three cases. The first case happens if, for every row $R$ of the grid $H_v$, at least one edge of $R$ lies in $E_{G_1}(A_1, B_1)$. Clearly, in this case, $|E(H_v) \cap E_{G_1}(A_1, B_1)| \geq d_v \geq \min\{n_A(v), n_B(v)\}$. The second case happens if, for every column $W \in \mathcal{W}'$, at least one edge of $W$ lies in $E_{G_1}(A_1, B_1)$. In this case, $|E(H_v) \cap E_{G_1}(A_1, B_1)| \geq n_A(v) \geq \min\{n_A(v), n_B(v)\}$. Lastly, the third case happens if, for every column $W \in \mathcal{W}''$, at least one edge of $W$ lies in $E_{G_1}(A_1, B_1)$. In this case, $|E(H_v) \cap E_{G_1}(A_1, B_1)| \geq n_B(v) \geq \min\{n_A(v), n_B(v)\}$.

We now claim that at least one of the above three cases has to happen. Indeed, assume otherwise. Then there is some row $R$ of the grid, and two columns $W \in \mathcal{W}'$, $W' \in \mathcal{W}''$, such that no edge of

$E(R) \cup E(W') \cup E(W'')$ lies in $E_{G_1}(A_1, B_1)$. Assume that $W'$ is the $i$th column and $W''$ is the $j$th column of the grid $H_v$. Since $x_i(v) \in A_1$, $x_j(v) \in B_1$, and $R \cup W' \cup W''$ is a connected graph, this is impossible. □

We say that a vertex $v \in V(G)$ is *indecisive* iff $n_A, n_B \geq d_v/32$; otherwise we say that vertex $v$ is *decisive*. We start with $(A_2, B_2) = (A_1, B_1)$, and then gradually modify this cut, by processing the decisive vertices one by one. When such a vertex $v$ is processed, if $n_A < d_v/32$, then we move the vertices of $V(H_v) \cap A_2$ from $A_2$ to $B_2$. Notice that this transformation adds up to $n_A$ new external edges to cut $E_{G_1}(A_2, B_2)$ – the edges incident to the vertices of $X(v) \cap A_2$. However, from Observation D.6, at least $n_A = |X(v) \cap A_2|$ edges of $H_v$ contributed to the cut $E_{G_1}(A_2, B_2)$ before this transformation, and they no longer contribute to the cut after the transformation. Therefore, $|E_{G_1}(A_2, B_2)|$ does not increase as the result of this transformation. If $n_A \geq d_v/32$, then $n_B < d_v/32$ must hold, and we move the vertices of $V(H_v) \cap B_2$ from $B_2$ to $A_2$. Using the same arguments as before, $|E_{G_1}(A_2, B_2)|$ does not increase.

Consider the cut $(A_2, B_2)$ of $G_1$ that is obtained after all decisive vertices are processed. From the above discussion, $|E_{G_1}(A_2, B_2)| \leq |E_{G_1}(A_1, B_1)| \leq 4 \cdot \sqrt{24\Delta m + 8\mathsf{OPT}_{\mathsf{cr}}(G)}$. Moreover, the total number of new edges of $E'$ that were added to $E_{G_1}(A_2)$ is bounded by $\sum_{v \in V(G)} d_v/32 \leq m/16$. Since $|E' \cap E_{G_1}(A_1)| \leq 2m/3 + 4 \cdot \sqrt{24\Delta m + 8\mathsf{OPT}_{\mathsf{cr}}(G)}$, if $\mathsf{OPT}_{\mathsf{cr}}(G) \leq m^2/2^{40}$ and $\Delta \leq m/2^{40}$, then

$$|E' \cap E_{G_1}(A_2)| \leq m/16 + 2m/3 + 4 \cdot \sqrt{24\Delta m + 8\mathsf{OPT}_{\mathsf{cr}}(G)} \leq 3m/4.$$

Using the same reasoning, if $\mathsf{OPT}_{\mathsf{cr}}(G) \leq m^2/2^{40}$, then $|E' \cap E_{G_1}(B_2)| \leq 3m/4$.

Next, we construct a canonical cut $(A_3, B_3)$ in graph $G_1$, by starting with $(A_3, B_3) = (A_2, B_2)$, and processing every indecisive vertex $v \in V(G)$ one by one. When vertex $v$ is processed, if $|E' \cap E_{G_1}(A_3)| \leq |E' \cap E_{G_1}(B_3)|$, then we move all vertices of $H_v \cap B_3$ from $B_3$ to $A_3$, and otherwise we move all vertices of $H_v \cap A_3$ from $A_3$ to $B_3$. Note that in either case, the total number of external edges that are added to cut $(A_3, B_3)$ is bounded by $d_v$. From Observation D.6, the number of edges of $H_v$ that contributed to the cut $(A_3, B_3)$ before this transformation is at least $\min\{n_A, n_B\}$. Since vertex $v$ is indecisive, $n_A, n_B \geq d_v/32$. Once every indecisive vertex of $G$ is processed, we obtain the final cut $(A_3, B_3)$ in graph $G_1$, that must be canonical. From the above discussion:

$$|E_{G_1}(A_3, B_3)| \leq 32 \cdot |E_{G_1}(A_2, B_2)| \leq 32|E_{G_1}(A_1, B_1)| \leq O\left(\sqrt{\Delta m + \mathsf{OPT}_{\mathsf{cr}}(G)}\right).$$

We claim that $|E' \cap E_{G_1}(A_3)|, |E' \cap E_{G_1}(B_3)| \leq 3m/4$. We prove this for $A_3$, as the proof for $B_3$ is symmetric. If no new vertices were added to set $A_3$, then $|E' \cap E_{G_1}(A_3)| \leq |E' \cap E_{G_1}(A_2)| \leq 3m/4$ holds. Assume now that new vertices were added to $A_3$, and let $v$ be the last indecisive vertex that was processed by the algorithm, for which vertices of $H_v$ were added to $A_3$. Then, before vertex $v$ was processed, $|E' \cap E_{G_1}(A_3)| \leq |E' \cap E_{G_1}(B_3)|$ held; therefore, $|E' \cap E_{G_1}(A_3)| \leq m/2$. Note that moving the vertices of $V(H_v) \cap B_3$ from $B_3$ to $A_3$ could have added at most $d_v \leq \Delta \leq m/2^{40}$ new edges to $E' \cap E_{G_1}(A_3)$, and so $|E' \cap E_{G_1}(A_3)| \leq 3m/4$ must hold at the end of this iteration. Since the iteration when $v$ was processed is the last iteration when vertices were added to $A_3$, we conclude that $|E' \cap E_{G_1}(A_3)| \leq 3m/4$ holds at the end of the algorithm. Using the same reasoning, $|E' \cap E_{G_1}(B_3)| \leq 3m/4$ holds as well.

**Step 3: Balanced Cut in $G$.** We are now ready to define the final cut $(A^*, B^*)$ in graph $G$. We add to $A^*$ every vertex $v \in V(G)$ with $V(H_v) \subseteq A_3$, and we add all remaining vertices of $V(G)$ to $B^*$. It is easy to verify that $|E_G(A^*, B^*)| = |E_{G_1}(A_3, B_3)| \leq O\left(\sqrt{\Delta m + \mathsf{OPT}_{\mathsf{cr}}(G)}\right)$. Additionally, $|E_G(A^*)| \leq |E' \cap E_{G_1}(A_3)| \leq 3m/4$, and similarly $|E_G(B^*)| \leq |E' \cap E_{G_1}(B_3)| \leq 3m/4$. We conclude that $(A^*, B^*)$ is a $(3/4)$-edge-balanced cut in graph $G$, whose value is at most $O(\sqrt{\mathsf{OPT}_{\mathsf{cr}}(G) + \Delta \cdot m})$.

## D.7 Proof of Observation 4.14

The proof is practically identical to the proof of Observation D.6. Consider a cut $(A, B)$ in graph $H$, and denote $n_A = |S \cap A|$ and $n_B = |S \cap B|$. It is enough to show that $|E(A, B)| \geq \min\{n_A, n_B\}$.

We partition the columns of the grid graph $H$ into two subsets, $\mathcal{W}', \mathcal{W}''$, as follows. For $1 \leq i \leq r$, the $i$th column of the grid is added to set $\mathcal{W}'$ if the unique vertex of $S$ lying in the $i$th column belongs to $A$. Otherwise, the $i$th column is added to $\mathcal{W}''$.

We now consider three cases. The first case happens if, for every row $R$ of the grid $H$, at least one edge of $R$ lies in $E(A, B)$. Clearly, in this case, $|E(A, B)| \geq r \geq \min\{n_A, n_B\}$. The second case happens if, for every column $W \in \mathcal{W}'$, at least one edge of $W$ lies in $E(A, B)$. In this case, $|E(A, B)| \geq n_A \geq \min\{n_A, n_B\}$. Lastly, the third case happens if, for every column $W \in \mathcal{W}''$, at least one edge of $W$ lies in $E(A, B)$. In this case, $|E(A, B)| \geq n_B \geq \min\{n_A, n_B\}$.

We now claim that at least one of the above three cases has to happen. Indeed, assume otherwise. Then there is some row $R$ of the grid, and two columns $W \in \mathcal{W}'$, $W' \in \mathcal{W}''$, such that no edge of $E(R) \cup E(W') \cup E(W'')$ lies in $E(A, B)$. But the unique vertex of $S \cap V(W)$ lies in $A$, the unique vertex of $S \cap V(W')$ lies in $B$, and $R \cup W' \cup W''$ is a connected graph, a contradiction.

## D.8 Proof of Theorem 4.17

We construct an $s$-$t$ flow network, as follows. We start with the graph $G$, and then add a new source vertex $s$, that connects to every vertex in $T_1$ with an edge. We also add a new destination vertex $t$, and connect every vertex of $T_2$ to $t$ with an edge. Denote the resulting graph by $H$. For every edge $e \in E(H)$, if $e$ is incident to $s$ or to $t$, then we set its capacity $c(e) = 1$, and otherwise we set $c(e) = \lceil 1/\alpha \rceil$. Note that the capacity of every edge in the resulting flow network is integral.

We show below that the value of the maximum $s$-$t$ flow in the resulting flow network is $k = |T_1|$. From the integrality of flow, we can then compute an integral $s$-$t$ flow $f$ of value $k$ in $H$. Let $\mathcal{P}$ be the set of all $s$-$t$ paths in graph $H$. Since flow $f$ is integral, and since the capacity of every edge incident to $s$ and to $t$ is 1, for every path $P \in \mathcal{P}$, $f(P) = 0$ or $f(P) = 1$ holds. Moreover, if $\mathcal{P}' \subseteq \mathcal{P}$ is the set of all paths $P$ with $f(P) = 1$, then $|\mathcal{P}'| = k$. Since the capacity of every edge in $\{(s, x) \mid x \in T_1\}$, and the capacity of every edge in $\{(y, x) \mid y \in T_1\}$ is 1, each such edge belongs to exactly one path in $\mathcal{P}'$. Therefore, set $\mathcal{P}'$ of paths naturally defines a one-to-one routing $\mathcal{Q}$ of vertices of $T_1$ to vertices of $T_2$ in graph $G$, with $\mathrm{cong}_G(\mathcal{Q}) \leq \lceil 1/\alpha \rceil$. In order to complete the proof of the theorem, it is now enough to show that the value of the maximum $s$-$t$ flow in graph $H$ is at least $k$.

Assume for contradiction that this is not the case. Consider a minimum $s$-$t$ cut $(A, B)$ in graph $H$. From our assumption, the value of the cut is less than $k$. We partition the set $T_1$ of vertices into two subsets: set $T_1' = T_1 \cap A$ and set $T_1'' = T_1 \cap B$. Note that, for every vertex $x \in T_1''$, its corresponding edge $(s, x)$ belongs to the cut $E(A, B)$. We denote by $E_1 = \{(s, x) \mid x \in T_1''\}$ the corresponding set of edges. We also partition the set $T_2$ of vertices into two subsets: set $T_2' = T_2 \cap B$ and set $T_2'' = T_2 \cap A$. Note that, for every vertex $y \in T_2''$, its corresponding edge $(y, t)$ belongs to the cut $E(A, B)$. We denote by $E_2 = \{(y, t) \mid y \in T_2''\}$ the corresponding set of edges.

Lastly, we denote by $E' = E(A, B) \setminus (E_1 \cup E_2)$ the set of the remaining edges in the cut $(A, B)$. Note that each edge in $E'$ has capacity $\lceil 1/\alpha \rceil$, while each edge in $E_1 \cup E_2$ has capacity 1. Since we have assumed that the value of the cut $(A, B)$ is less than $k$, we get that:

$$|T_1''| + |T_2''| + |E'| \cdot \lceil 1/\alpha \rceil = |E_1| + |E_2| + |E'| \cdot \lceil 1/\alpha \rceil = \sum_{e \in E_H(A,B)} c(e) < k \tag{26}$$

We define a cut $(A', B')$ in graph $G$ using cut $(A, B)$ as follows: $A' = A \setminus \{s\}$ and $B' = B \setminus \{t\}$. Notice

that $|E_G(A',B')| = |E'|$. From Equation (26), we then get that:

$$|E_G(A',B')| = |E'| < \alpha \cdot (k - |T_1''| - |T_2''|)$$

Since $|T_1'| = k - |T_1''|$ and $|T_2'| = k - |T_2''|$, we get that:

$$|E_G(A',B')| < \alpha \cdot \min\left\{|T_1'|, |T_2'|\right\}.$$

Lastly, since $T_1' \subseteq A'$ and $T_2' \subseteq B'$, we get that $|E_G(A',B')| < \alpha \cdot \min\{|T \cap A'|, |T \cap B'|\}$, contradicting the fact that the set $T$ of vertices is $\alpha$-well-linked in $G$.

### D.9 Proof of Theorem 4.19

Assume first that $0 < \alpha \leq 1/m$. Then we simply let $\mathcal{R} = \{S\}$. Since $\alpha \leq 1/m$, and $S$ is a connected graph, it is easy to verify that it has the $\alpha$-bandwidth property in graph $G$. For each edge $e \in \delta_G(S)$, we simply let $P(e)$ be the path that contains the single edge $e$. It is easy to verify that cluster set $\mathcal{R} = \{S\}$, and the set $\mathcal{P}(S) = \{P(e) \mid e \in \delta_G(S)\}$ of paths have all required properties.

We assume from now on that $\frac{1}{m} < \alpha < \min\left\{\frac{1}{64\beta_{\text{ARV}}(m)\cdot\log m}, \frac{1}{48\log^2 m}\right\}$ holds.

Our algorithm maintains a collection $\mathcal{R}$ of clusters of $S$, that is initialized to $\mathcal{R} = \{S\}$. Throughout the algorithm, we ensure that the following invariants hold:

I1. all clusters in $\mathcal{R}$ are mutually disjoint;

I2. $\bigcup_{R\in\mathcal{R}} V(R) = V(S)$; and

I3. for every cluster $R \in \mathcal{R}$, $|\delta_G(R)| \leq |\delta_G(S)|$.

For a given collection $\mathcal{R}$ of clusters with the above properties, we define a *budget* $b(e)$ for every edge $e \in E(G)$, as follows. If $e \in \delta_G(S)$, and the endpoint of $e$ that lies in $S$ belongs to a cluster $R \in \mathcal{R}$, then we set the budget $b(e) = 1 + 8\alpha \cdot \beta_{\text{ARV}}(m) \cdot \log_{3/2}(|\delta_G(R)|)$. If edge $e$ has its endpoints in two distinct clusters $R, R' \in \mathcal{R}$, then we set $b(e) = 2 + 8\alpha \cdot \beta_{\text{ARV}}(m) \cdot \log_{3/2}(|\delta_G(R)|) + 8\alpha \cdot \beta_{\text{ARV}}(m) \cdot \log_{3/2}(|\delta_G(R')|)$. Otherwise, we set $b(e) = 0$. Notice that, for every edge $e \in E(G)$, $b(e) \leq 3$ always holds. Additionally, for every edge $e \in \bigcup_{R\in\mathcal{R}} \delta_G(R)$, $b(e) \geq 2$ if the endpoints of $e$ lie in two different clusters of $\mathcal{R}$, and $b(e) \geq 1$ if $e \in \delta_G(S)$. Therefore, if we denote by $B = \sum_{e\in E(G)} b(e)$ the total budget in the system, then, throughout the algorithm, $B \geq \sum_{R\in\mathcal{R}} |\delta_G(R)|$ holds. Lastly, observe that, at the beginning of the algorithm, $B \leq |\delta_G(S)| \cdot (1 + O(\alpha \cdot \beta_{\text{ARV}}(m) \cdot \log m))$. Throughout the algorithm, we will modify the clusters in set $\mathcal{R}$, leading to changes in the budgets of the edges of $G$. We will ensure however that the total budget $B$ never increases, and so, if $\mathcal{R}$ is the final set of clusters that we obtain, then $\sum_{R\in\mathcal{R}} |\delta_G(R)| \leq B \leq (1 + O(\alpha \cdot \beta_{\text{ARV}}(m) \cdot \log m)) = (1 + O(\alpha \cdot \log^{1.5} m))$ holds.

Throughout the algorithm, we maintain a partition of the set $\mathcal{R}$ of clusters into two subsets: set $\mathcal{R}^A$ of *active* clusters, and set $\mathcal{R}^I$ of *inactive* clusters. We will ensure that the following additional invariant holds:

I4. every cluster $R \in \mathcal{R}^I$ has the $\alpha$-bandwidth property.

Additionally, we will store, with every inactive cluster $R \in \mathcal{R}^I$, a set $\mathcal{P}(R) = \{P(e) \mid e \in \delta_G(R)\}$ of paths in graph $G$, (that we refer to as *witness set of paths for* $R$), such that $\text{cong}_G(\mathcal{P}(R)) \leq 100$, and, for every edge $e \in \delta_G(R)$, path $P(e)$ has $e$ as its first edge and some edge of $\delta_G(S)$ as its last edge, and all inner vertices of $P(e)$ lie in $V(S) \setminus V(R)$. At the beginning of the algorithm, we set

$\mathcal{R}^A = \mathcal{R} = \{S\}$ and $\mathcal{R}^I = \emptyset$. Clearly, all invariants hold at the beginning of the algorithm. We then proceed in iterations, as long as $\mathcal{R}^A \neq \emptyset$.

In order to execute an iteration, we select an arbitrary cluster $R \in \mathcal{R}^A$ to process. We will either establish that $R$ has the $\alpha$-bandwidth property in graph $G$ and compute a witness set $\mathcal{P}(R)$ of paths for $R$ (in which case $R$ is moved from $\mathcal{R}^A$ to $\mathcal{R}^I$); or we will modify the set $\mathcal{R}$ of clusters so that the total budget of all edges decreases by at least $1/m$. An iteration that processes a cluster $R \in \mathcal{R}^A$ consists of two steps. The purpose of the first step is to either establish the $\alpha$-bandwidth property of cluster $R$, or to replace it with a collection of smaller clusters in $\mathcal{R}$. The purpose of the second step is to either compute the witness set $\mathcal{P}(R)$ of paths for cluster $R$, or to modify the set $\mathcal{R}$ of clusters so that the total budget of all edges decreases. We now describe each of the two steps in turn.

**Step 1: Bandwidth Property.** Let $R \in \mathcal{R}^A$ be any active cluster, and let $R^+$ be the augmentation of $R$ in graph $G$. Recall that $R^+$ is a graph that is obtained from $G$ through the following process. First, we subdivide every edge $e \in \delta_G(R)$ with a vertex $t_e$, and we let $T = \{t_e \mid e \in \delta_G(R)\}$ be the resulting set of vertices. We then let $R^+$ be the subgraph of the resulting graph induced by vertex set $V(R) \cup T$. We apply Algorithm $\mathcal{A}_{\text{ARV}}$ for computing approximate sparsest cut to graph $R^+$, with the set $T$ of vertices, to obtain a $\beta_{\text{ARV}}(m)$-approximate sparsest cut $(X, Y)$ in graph $R^+$ with respect to vertex set $T$. We now consider two cases. The first case happens if $|E(X,Y)| \geq \alpha \cdot \beta_{\text{ARV}}(m) \cdot \min\{|X \cap T|, |Y \cap T|\}$. In this case, we are guaranteed that the minimum sparsity of any $T$-cut in graph $R^+$ is at least $\alpha$, or equivalently, set $T$ of vertices is $\alpha$-well-linked in $R^+$. From Observation 4.16, cluster $R$ has the $\alpha$-bandwidth property in graph $G$. In this case, we proceed to the second step of the algorithm.

Assume now that $|E(X,Y)| < \alpha \cdot \beta_{\text{ARV}}(m) \cdot \min\{|X \cap T|, |Y \cap T|\}$. Since $\alpha \leq \min\left\{\frac{1}{64\beta_{\text{ARV}}(m)\cdot \log m}, \frac{1}{48\log^2 m}\right\}$, we get that the sparsity of the cut $(X, Y)$ is less than 1. Consider now any vertex $t \in T$, and let $v$ be the unique neighbor of $t$ in $R^+$. We can assume w.l.o.g. that either $t, v$ both lie in $X$, or they both lie in $Y$. Indeed, if $t \in X$ and $v \in Y$, then moving vertex $t$ from $X$ to $Y$ does not increase the sparsity of the cut $(X, Y)$. This is because, for any two real numbers $1 \leq a < b$, $\frac{a-1}{b-1} \leq \frac{a}{b}$. Similarly, if $t \in Y$ and $v \in X$, then moving $t$ from $Y$ to $X$ does not increase the sparsity of the cut $(X, Y)$. Therefore, we assume from now on, that for every vertex $t \in T$, if $v$ is the unique neighbor of $t$ in $R^+$, then either both $v, t \in X$, or both $v, t \in Y$.

Consider now the partition $(X', Y')$ of $V(R)$, where $X' = X \setminus T$ and $Y' = Y \setminus T$. It is easy to verify that $|\delta_G(R) \cap \delta_G(X')| = |X \cap T|$, and $|\delta_G(R) \cap \delta_G(Y')| = |Y \cap T|$. Let $E' = E_G(X', Y')$, and assume w.l.o.g. that $|\delta_G(R) \cap \delta_G(X')| \leq |\delta_G(R) \cap \delta_G(Y')|$. Then $|E'| < \alpha \cdot \beta_{\text{ARV}}(m) \cdot |\delta_G(R) \cap \delta_G(X')|$ must hold. We remove cluster $R$ from sets $\mathcal{R}$ and $\mathcal{R}^A$, and we add instead every connected component of graphs $G[X']$ and $G[Y']$ to both sets. It is immediate to verify that $\mathcal{R}$ remains a collection of disjoint clusters of $G$, and that $\bigcup_{R' \in \mathcal{R}} V(R') = V(G)$. Since $|E'| < \min\{|\delta_G(R) \cap \delta_G(X')|, |\delta_G(R) \cap \delta_G(Y')|\}$, we get that for every cluster $C$ that we just added to $\mathcal{R}$, $|\delta_G(C)| \leq |\delta_G(R)| \leq |\delta_G(S)|$ (from Invariant I3). Therefore, all invariants continue to hold. We now show that the total budget $B$ decreases by at least $1/m$ as the result of this operation.

Note that the only edges whose budgets may change as the result of this operation are edges of $\delta_G(R) \cup E'$. Observe that, for each edge $e \in \delta_G(R) \cap \delta_G(Y')$, its budget $b(e)$ may not increase. Since we have assumed that $|\delta_G(R) \cap \delta_G(X')| \leq |\delta_G(R) \cap \delta_G(Y')|$, and since $|E'| < |\delta_G(R)|/8$, we get that $|\delta_G(X')| \leq 2|\delta_G(R)|/3$. Therefore, for every edge $e \in \delta_G(X') \cap \delta_G(R)$, its budget $b(e)$ decreases by at least $8\alpha \cdot \beta_{\text{ARV}}(m) \cdot \log_{3/2}(|\delta_G(R)|) - 8\alpha \cdot \beta_{\text{ARV}}(m) \cdot \log_{3/2}(|\delta_G(X')|)$. Since $|\delta_G(X')| \leq 2|\delta_G(R)|/3$, we get that $\log_{3/2}(|\delta_G(R)|) \leq \log_{3/2}(3|\delta_G(X')|/2) \leq 1 + \log_{3/2}(|\delta_G(X')|$. We conclude that the budget $b(e)$ of each edge $e \in \delta_G(X') \cap \delta_G(R)$ decreases by at least $8\alpha \cdot \beta_{\text{ARV}}(m)$. On the other hand, the budget of every edge $e \in E'$ increases by at most 3. Since $|E'| \leq \alpha \cdot \beta_{\text{ARV}}(m) \cdot |\delta_G(R) \cap \delta_G(X')|$, we get that the

decrease in the budget $B$ is at least:

$$\begin{aligned}
&8\alpha\cdot\beta_{\text{ARV}}(m)\cdot|\delta_G(X')\cap\delta_G(R)|-3|E'|\\
&\qquad\geq 8\alpha\cdot\beta_{\text{ARV}}(m)\cdot|\delta_G(X')\cap\delta_G(R)|-3\alpha\cdot\beta_{\text{ARV}}(m)\cdot|\delta_G(R)\cap\delta_G(X')|\\
&\qquad\geq 5\alpha\cdot\beta_{\text{ARV}}(m)\cdot|\delta_G(R)\cap\delta_G(X')|\\
&\qquad> 1/m,
\end{aligned}$$

since $\alpha\geq 1/m$. To conclude, if $|E(X,Y)|<\alpha\cdot\beta_{\text{ARV}}(m)\cdot\min\{|X\cap T|,|Y\cap T|\}$, then we have modified the set $\mathcal{R}$ of clusters, so that all invariants continue to hold, and the total budget $B$ decreases by at least $1/m$. In this case, we terminate the current iteration.

From now on we assume that $|E(X,Y)|>\alpha\cdot\beta_{\text{ARV}}(m)\cdot\min\{|X\cap T|,|Y\cap T|\}$, which, as observed already, implies that cluster $R$ has the $\alpha$-bandwidth property. We now proceed to describe the second step of the algorithm.

**Step 2: Witness Set of Paths.** In the second step, we attempt to compute a witness set $\mathcal{P}(R)$ of paths for cluster $R$. If we succeed in doing so, we will move cluster $R$ from $\mathcal{R}^A$ to $\mathcal{R}^I$. Otherwise, we will further modify the set $\mathcal{R}$ of clusters, so that all invariants continue to hold, and the total budget decreases by at least $1/m$.

We construct the following flow network. Starting from graph $G$, we contract all vertices of $R$ into a source vertex $s$, and we contract all vertices of $V(G)\setminus V(S)$ into a destination vertex $t$. Denote the resulting graph by $H$, and observe that $\delta_H(s)=\delta_G(R)$, and $\delta_H(t)=\delta_G(S)$. We set the capacity $c(e)$ of every edge incident to $s$ to 1, and the capacity of every other edge in graph $H$ to 100. We then compute the maximum $s$-$t$ flow $f$ in the resulting flow network.

We consider two cases. The first case is when the value of the flow $f$ is $|\delta_H(s)|$. Since all edge capacities are integral, we can assume that flow $f$ is integral as well. Note that in this case, for every path $P$ connecting $s$ to $t$, either $f(P)=0$ or $f(P)=1$ must hold, as the capacities of all edges incident to $s$ are 1. Therefore, flow $f$ naturally defines a collection $\mathcal{P}'(R)$ of $s$-$t$ paths, with $\text{cong}_H(\mathcal{P}'(R))\leq 100$, where each edge $e\in\delta_G(s)$ serves as the first edge of exactly one such path. Set $\mathcal{P}'(R)$ of paths then naturally defines a witness set $\mathcal{P}(R)=\{P(e)\mid e\in\delta_G(R)\}$ of paths for cluster $R$ in graph $G$, with $\text{cong}_G(\mathcal{P}(R))\leq 100$, where, for every edge $e\in\delta_G(R)$, path $P(e)$ has $e$ as its first edge and some edge of $\delta_G(S)$ as its last edge, with all inner vertices of $P(e)$ lying in $V(S)\setminus V(R)$. We then move cluster $R$ from $\mathcal{R}^A$ to $\mathcal{R}^I$ and terminate the current iteration. It is easy to verify that all invariants continue to hold, and the total budget $B$ does not change.

It remains to consider the second case, where the value of the flow $f$ in $H$ is less than $|\delta_H(s)|$. We compute a minimum $s$-$t$ cut $(A',B')$ in graph $H$, whose value is less than $|\delta_H(s)|$. We partition the set $E(A',B')$ of edges into two subsets: set $E'=E(A',B')\cap\delta_H(s)$, and set $E''=E(A',B')\setminus E'$. Recall that the capacity of every edge in $E'$ is 1, while the capacity of every edge in $E''$ is 100. Therefore, $|E'|+100|E''|<|\delta_G(R)|$.

Observe that cut $(A',B')$ in $H$ naturally defines cut $(A,B)$ of graph $S$: we let $A=(A'\setminus\{s\})\cup V(R)$, and $B=V(S)\setminus A$. Notice that $\delta_G(A)=E_H(A',B')$.

Let $\mathcal{A}$ denote the set of all connected components of graph $S[A]=G[A]$. Let $\mathcal{X}$ denote the set of all clusers $R'\in\mathcal{R}$ with $R'\cap A\neq\emptyset$. For each such cluster $R'\in\mathcal{X}$, let $\mathcal{Y}(R')$ be the set of all connected components of $R'\setminus A$. We need the following observation.

**Observation D.7** *For every cluster $C\in\mathcal{A}$, $|\delta_G(C)|\leq|\delta_G(R)|$. Additionally, for every cluster $R'\in\mathcal{X}$, and every cluster $R''\in\mathcal{Y}(R')$, $|\delta_G(R'')|\leq|\delta_G(R')|$.*

**Proof:** Consider first some cluster $C\in\mathcal{A}$. Clearly, $\delta_G(C)\subseteq\delta_G(A)\subseteq E_H(A',B')$. Since $|E_H(A',B')|<|\delta_H(s)|=|\delta_G(R)|$, we get that $|\delta_G(C)|\leq|\delta_G(R)|$.

Consider now some cluster $R' \in \mathcal{X}$. Denote by $R'_A$ the subgraph of $R'$ induced by $V(R') \cap A$, and denote by $R'_B = R' \setminus A$. Let $E_1 = \delta_G(R') \cap \delta_G(R'_A)$, $E_2 = \delta_G(R') \cap \delta_G(R'_B)$, and $\hat{E} = E_G(R'_A, R'_B)$. We show below that $|\hat{E}| \leq |E_1|$ must hold. Assume for now that this is true, and consider any cluster $R'' \in \mathcal{Y}(R')$. Since $R''$ is a connected component of $R'_B$, we get that $\delta_G(R'') \subseteq E_2 \cup \hat{E}$. Therefore, if $|\hat{E}| \leq |E_1|$, then $|\delta_G(R'')| \leq |E_2| + |\hat{E}| \leq |E_1| + |E_2| = |\delta_G(R')|$ holds.

It now remains to prove that $|\hat{E}| \leq |E_1|$. Consider the cut $(A', B')$ in graph $H$, and recall that it is a minimum $s$-$t$ cut. From the definition of the cut $(A, B)$, the edges of $\hat{E}$ belong to the edge set $E_H(A', B')$. Since none of these edges is incident to $s$, the capacity of every edge in $\hat{E}$ is 100. Consider now a new $s$-$t$ cut $(A'', B'')$ in graph $H$, where $A'' = A' \setminus V(R'_A)$ and $B'' = B' \cup V(R'_B)$. Note that the edges of $\hat{E}$ no longer contribute to this cut, and the only new edges that were added to this cut are the edges of $E_1$, each of which has a capacity that is either 1 or 100. Therefore, $\sum_{e \in E_H(A'', B'')} c(e) \leq \sum_{e \in E_H(A', B')} c(e) - 100|\hat{E}| + 100|E_1|$. Since $(A', B')$ is a minimum $s$-$t$ cut in graph $H$, $|\hat{E}| \leq |E_1|$ must hold. □

We perform the following modifications to the sets $\mathcal{R}, \mathcal{R}^I$ and $\mathcal{R}^A$ of clusters. First, we remove cluster $R$ from $\mathcal{R}$ and from $\mathcal{R}^A$, and we add every cluster of $\mathcal{A}$ to both sets instead. Next, we consider every cluster $R' \in \mathcal{X}$ one by one. We remove each such cluster $R'$ from $\mathcal{R}$, and we add instead every cluster in $\mathcal{Y}(R')$ to $\mathcal{R}$. We also remove cluster $R'$ from the cluster set in $\left\{\mathcal{R}^I, \mathcal{R}^A\right\}$ to which it belongs, and we add every cluster of $\mathcal{Y}(R')$ to set $\mathcal{R}^I$. This completes the description of the modification of the sets $\mathcal{R}, \mathcal{R}^I, \mathcal{R}^A$ of clusters. Note that all clusters in $\mathcal{R}$ remain disjoint, and $\bigcup_{R'' \in \mathcal{R}'} V(R'') = V(S)$ continues to hold. Moreover, from Observation D.7, combined with Invariant I3, for every cluster $R' \in \mathcal{R}$, $|\delta_G(R')| \leq |\delta_G(S)|$ continues to hold. It remains to show that the total budget $B$ decreases by at least $1/m$.

Consider any edge $e \in E(G) \setminus E''$, whose budget, at the end of the current step, is non-zero. Assume first that the budget of $e$ was non-zero at the beginning of the current step. Then, from Observation D.7, the budget of $e$ could not have increased as the result of the current step. If the budget of an edge $e$ was 0 at the beginning of the current step and is non-zero at the end of the current step, then $e \in E''$ must hold. Therefore, the only edges whose budget may have increased as the result of the current step are edges of $E''$.

For each edge $e \in E''$, its budget may have grown from 0 to at most 3, while the number of all such edges is $|E''| < (|\delta_G(R)| - |E'|)/100$. Therefore, the total increase in the budget $B$ due to the edges of $E''$ is at most $(|\delta_G(R)| - |E'|)/30$. We show that this increase is compensated by the decrease in the budgets of the edges of $\delta_G(R) \setminus E'$. Consider any edge $e \in \delta_G(R) \setminus E'$. Edge $e$ had budget at least 1 originally, but after the current iteration, since the endpoints of $e$ both lie in $A$, its budget becomes 0. Therefore, the decrease in the budget $B$ due to the edges of $\delta_G(R) \setminus E'$ is at least $|\delta_G(R) \setminus E'|$. Overall, we get that the decrease in the budget $B$ is at least:

$$|\delta_G(R) \setminus E'| - (|\delta_G(R)| - |E'|)/30 \geq 1/2.$$

This concludes the description of an iteration. The algorithm terminates when $\mathcal{R}^I = \emptyset$ holds, at which point we obtain the final set $\mathcal{R}$ of clusters, together with the witness sets $\{\mathcal{P}(R)\}_{R \in \mathcal{R}}$ of paths, that, from the invariants, have all required properties. In particular, as observed above, since $B \geq \sum_{R \in \mathcal{R}} |\delta_G(R)|$, and $B$ never increases over the course of the algorithm, $\sum_{R \in \mathcal{R}} |\delta_G(R)| \leq B \leq (1 + O(\alpha \cdot \beta_{\text{ARV}}(m) \cdot \log m)) = (1 + O(\alpha \cdot \log^{1.5} m))$ holds. It remains to prove that the algorithm is efficient. Clearly, the algorithm for executing every iteration is efficient. We now show that the number of iterations is bounded by $O(m^3)$. Consider any iteration $i$ of the algorithm. Recall that, as the result of iteration $i$, either the budget $B$ decreased by at least $1/m$ (in which case we say that $i$ is a type-1 iteration); or budget $B$ did not change, but the number of clusters in set $\mathcal{R}^I$ decreases by 1 (in which case we say that $i$ is a type-2 iteration). It is then immediate to see that the number of type-1 iterations, over the course of the algorithm, is bounded by $O(m^2)$. Since every cluster of $\mathcal{R}$ must

contain at least one vertex, and $|V(S)| \leq m$ (because $S$ is a connected graph), the number of type-2 iterations executed between every consecutive pair of type-1 iterations is bounded by $O(m)$. Therefore, the total number of iterations of the algorithm is $O(m^3)$, and so the algorithm is efficient.

### D.10 Proof of Theorem 4.20

Note that by letting $c$ be a large enough constant, we can ensure that $\alpha < \min\left\{\frac{1}{64\beta_{\mathrm{ARV}}(m)\cdot\log m}, \frac{1}{48\log^2 m}\right\}$ holds.

The algorithm starts with layer $\mathcal{L}_0$ containing a single subgraph of $H$ – the subgraph $C$, and then performs iterations. The input to iteration $i$ is layers $\mathcal{L}_0, \mathcal{L}_1, \ldots, \mathcal{L}_{i-1}$, each of which is a collection of disjoint clusters of $H$. We ensure that all clusters in $\bigcup_{i'=0}^{i-1}\mathcal{L}_{i'}$ are mutually disjoint, and each cluster $W \in \bigcup_{i'=1}^{i-1}\mathcal{L}_{i'}$ has $\alpha$-bandwidth property. We let $S_i$ be the subgraph of $H$ induced by vertex set $\bigcup_{i'=0}^{i-1}\bigcup_{W\in\mathcal{L}_{i'}}V(W)$, and we let $E_i = \delta_H(S_i)$. In subsequent iterations, we will create layers $\mathcal{L}_i, \mathcal{L}_{i+1}, \ldots$, each of which will contain clusters that are disjoint from $S_i$. Notice that, for all $1 \leq i' < i$, for every cluster $W \in \mathcal{L}_{i'}$, the partition of the edges of $\delta_H(W)$ into $\delta^{\mathrm{down}}(W)$ and $\delta^{\mathrm{up}}(W)$ is now settled, since the layers $\mathcal{L}_0, \ldots, \mathcal{L}_{i'-1}$ will not undergo any changes in subsequent iterations, and the edges of $E_i \cap \delta_H(W)$ are guaranteed to lie in $\delta^{\mathrm{up}}(W)$. We will ensure that, for all $1 \leq i' < i$, every cluster in $\mathcal{L}_{i'}$ has properties L2, L3 and L4. We will also ensure that $|E_i| \leq |\delta_H(C)|/2^{i-1}$. The algorithm terminates once $S_i = H$. Since we ensure that for all $i$, $|E_i| \leq |\delta_H(C)|/2^{i-1}$, the number of iterations is bounded by $\log m$.

We now describe the execution of the $i$th iteration. We start by considering the subgraph $S_i' = H\backslash V(S_i)$ of $H$. We apply the algorithm from Theorem 4.19 to graph $H$, its subgraph $S = S_i'$, and the parameter $\alpha = \frac{1}{c\log^{2.5} m}$. As observed already, $\alpha < \min\left\{\frac{1}{64\beta_{\mathrm{ARV}}(m)\cdot\log m}, \frac{1}{48\log^2 m}\right\}$ holds. (If graph $S_i'$ is not connected, then we apply the algorithm to every connected component of $S_i'$ separately). Let $\mathcal{W}_i$ be the collection of clusters that the algorithm returns. Recall that we are guaranteed that the sets $\{V(W)\}_{W\in\mathcal{W}_i}$ of vertices partition $V(S_i')$, and for each such cluster $W \in \mathcal{W}_i$, $|\delta_H(W)| \leq |\delta_H(S_i')| = |\delta_H(S_i)| \leq |\delta_H(C)|$. We are also guaranteed that each cluster $W \in \mathcal{W}_i$ has the $\alpha$-bandwidth property, and that $\sum_{W\in\mathcal{W}_i}|\delta_H(W)| \leq |\delta_H(S_i')|\cdot\left(1+O(\alpha\cdot\log^{3/2} m)\right) = |E_i|\cdot\left(1+O(\alpha\cdot\log^{3/2} m)\right)$. Since $\alpha = \frac{1}{c\log^{2.5} m}$ for a large enough constant $c$, we can ensure that $\sum_{W\in\mathcal{W}_i}|\delta_H(W)| \leq |E_i|\cdot\left(1+\frac{1}{1000\log m}\right)$. If there is a cluster $W \in \mathcal{W}_i$ with $|E_H(W)| < |\delta_H(W)|/(64\log m)$, then remove $W$ from $\mathcal{W}_i$ and add each of its vertices as a separate cluster to $\mathcal{W}_i$. Clearly, we have increased the sum $\sum_{W\in\mathcal{W}_i}|\delta_H(W)|$ by a factor of at most $(1+1/(32\log m))$ (since each edge appears in at most two sets of $\{\delta_H(W)\}_{W\in\mathcal{W}_i}$). Therefore, for the resulting set $\mathcal{W}_i$, we get that:

$$\sum_{W\in\mathcal{W}_i}|\delta_H(W)| \leq |E_i|\cdot\left(1+\frac{1}{1000\log m}\right)\cdot\left(1+\frac{1}{32\log m}\right) \leq |E_i|\cdot\left(1+\frac{1}{16\log m}\right).$$

Recall that the algorithm from Theorem 4.19 also computes, for each cluster $W \in \mathcal{W}$, a set $\mathcal{P}'(W)$ of paths in graph $H$ routing the edges of $\delta_H(W)$ to edges of $\delta_H(S_i') = E_i$, such that the paths of $\mathcal{P}'(W)$ avoid $W$ and cause congestion at most 100 in $H$.

We partition the set $\mathcal{W}_i$ of clusters into two subsets: set $\mathcal{W}_i'$ contains all clusters $W \in \mathcal{W}_i$, such that $|\delta_H(W)\setminus E_i| < |\delta_H(W)\cap E_i|/\log m$, and set $\mathcal{W}_i''$ contains all remaining clusters. We then set $\mathcal{L}_i = \mathcal{W}_i'$. This finishes the description of the iteration. Recall that we define $S_{i+1}$ to be the subgraph of $H$ induced by the set $V(S_i)\cup\left(\bigcup_{W\in\mathcal{L}_i}V(W)\right)$ of vertices, and $E_{i+1} = \delta_H(S_{i+1})$. We now analyze the iteration. First, from the algorithm, every cluster $W \in \mathcal{L}_i$ satisfies Property L3, since the first inequality is guaranteed by Theorem 4.19, and we have replaced every cluster that does not satisfies the second inequality of Property L3 by single-vertex clusters. We now prove the following claim.

**Claim D.8** $|E_{i+1}| \leq |E_i|/2$.

**Proof:** We partition the set $E_{i+1}$ of edges into two subsets. The first set, $E'_{i+1}$, contains all edges of $E_{i+1}$ that lie in the sets $\{\delta_H(W) \setminus E_i\}_{W \in \mathcal{W}'_i}$. Since, for each cluster $W \in \mathcal{W}'_i$, $|\delta_H(W) \setminus E_i| \leq |\delta_H(W) \cap E_i|/\log m$, we get that $|E'_{i+1}| \leq |E_i|/\log m$. The second set, $E''_{i+1}$, contains all remaining edges of $E_{i+1}$. It is easy to verify that every edge of $E''_{i+1}$ belongs to set $\delta_H(W) \cap E_i$ of edges for some cluster $W \in \mathcal{W}''_i$.

Recall that $\sum_{W \in \mathcal{W}_i} |\delta_H(W)| \leq |E_i| \cdot \left(1 + \frac{1}{16 \log m}\right)$. Therefore, since $E_i \subseteq \bigcup_{W \in \mathcal{W}_i} \delta_H(W)$, we get that $\sum_{W \in \mathcal{W}''_i} |\delta_H(W) \setminus E_i| \leq \frac{|E_i|}{16 \log m}$. For every cluster $W \in \mathcal{W}''_i$, $|\delta_H(W) \setminus E_i| \geq \frac{|\delta_H(W) \cap E_i|}{\log m}$ from the definition of set $\mathcal{W}''_i$. Therefore:

$$|E''_{i+1}| = \sum_{W \in \mathcal{W}''_i} |\delta_H(W) \cap E_i| \leq (\log m) \cdot \sum_{W \in \mathcal{W}''_i} |\delta_H(W) \setminus E_i| \leq \frac{|E_i|}{16}.$$

Altogether, $|E_{i+1}| = |E'_{i+1}| + |E''_{i+1}| \leq |E_i|/2$. □

Recall that we have already established that, for every cluster $W \in \mathcal{L}_i$, $|\delta_H(W)| \leq |\delta_H(C)|$. Consider any such cluster $W \in \mathcal{L}_i$. Since layers $\mathcal{L}_0, \ldots, \mathcal{L}_i$ will remain unchanged in the remainder of the algorithm, the partition of the edge set $\delta_H(W)$ into $\delta^{\text{up}}(W)$ and $\delta^{\text{down}}(W)$ is now settled, and moreover $\delta^{\text{down}}(W) = \delta_H(W) \cap E_i$, while $\delta^{\text{up}}(W) = \delta_H(W) \setminus E_i$. From the definition of cluster set $\mathcal{W}'_i = \mathcal{L}_i$, we get that, for every cluster $W \in \mathcal{L}_i$, $|\delta^{\text{up}}(W)| \leq |\delta^{\text{down}}(W)|/\log m$ holds. Therefore, property L4 holds for every cluster $W \in \mathcal{L}_i$. Recall that we have already established property L2 for each such cluster as well.

The algorithm terminates once $S_i = H$. Let $r$ denote the index of the last iteration. Since, for all $i$, $|E_{i+1}| \leq |E_i|/2$, $r \leq \log m$ holds. We let $\mathcal{W} = \bigcup_{i=1}^{r} \mathcal{L}_i$ be the final collection of clusters. We now claim that $(\mathcal{W}, (\mathcal{L}_1, \ldots, \mathcal{L}_r))$ is a valid layered $\alpha$-well-linked decomposition of $H$ with respect to $C$. Note that our algorithm immediately guarantees Property L1, and we have already established Properties L2 – L4 of the decomposition. In order to establish property L5, observe that for all $1 \leq i \leq r$:

$$\sum_{W \in \mathcal{L}_i} |\delta_H(W)| = \sum_{W \in \mathcal{L}_i} (|\delta^{\text{up}}(W)| + |\delta^{\text{down}}(W)|) \leq \sum_{W \in \mathcal{L}_i} 2|\delta^{\text{down}}(W)| = \sum_{W \in \mathcal{L}_i} 2|\delta_H(W) \cap E_i| = 2|E_i|.$$

Therefore,

$$\sum_{W \in \mathcal{W}} |\delta_H(W)| \leq 2 \sum_{1 \leq i \leq r} |E_i| \leq 4|E_1| = 4|\delta_H(C)|.$$

We conclude that property L5 holds for the decomposition. Lastly, it remains to establish property L6. We do so using the following claim.

**Claim D.9** *For all $0 \leq i < r$, there is a collection $\mathcal{R}_{i+1} = \{R(e) \mid e \in E_{i+1}\}$ of paths such that, for every edge $e \in E_{i+1}$, path $R(e)$ has $e$ as its first edge, and some edge $e' \in E_i$ as its last edge. Moreover, every edge $e' \in E_i$ participates in at most one path of $\mathcal{R}_{i+1}$, the paths in $\mathcal{R}_{i+1}$ cause congestion at most $\lceil 1/\alpha \rceil$, and for each path $R(e) \in \mathcal{R}_{i+1}$, all inner vertices on $R(e)$ lie in $S_{i+1} \setminus S_i$.*

**Proof:** We partition the set $E_{i+1}$ of edges into two subsets: $E'_{i+1} = E_i \cap E_{i+1}$, and $E''_{i+1} = E_{i+1} \setminus E_i$. For each edge $e \in E'_{i+1}$, the path $R(e)$ consists of a single edge – the edge $e$.

Observe that $E''_{i+1} \subseteq \left(\bigcup_{W \in \mathcal{L}_i} \delta_H(W)\right) \setminus E_i$. For every cluster $W \in \mathcal{L}_i$, we let $\hat{E}(W) = \delta_H(W) \cap E''_{i+1}$. Clearly, $\hat{E}(W) \subseteq \delta^{\text{up}}(W)$. Recall that $\delta^{\text{down}}(W) = \delta_H(W) \cap E_i$, and $|\delta^{\text{down}}(W)| > |\delta^{\text{up}}(W)|$. From Corollary 4.18, there is a set $\mathcal{R}(W)$ of paths, that is a one-to-one routing of edges in $\delta^{\text{up}}(W)$ to a

subset of edges in $\delta^{\text{down}}(W)$, such that, for each path $Q \in \mathcal{R}(W)$, all its edges, except for the first and the last, belong to $E(W)$, and the paths in $\mathcal{R}(W)$ cause congestion at most $\lceil 1/\alpha \rceil$. For each edge $e \in \hat{E}(W) \subseteq \delta^{\text{up}}(W)$, we let $R(e) \in \mathcal{R}(W)$ be the unique path whose first edge is $e$. We have now defined, for each edge $e \in E_{i+1}$, a path $R(e)$, whose first edge is $e$, last edge lies in $E_i$, and all inner vertices lie in $S_{i+1} \setminus S_i = \bigcup_{W \in \mathcal{L}_i} V(W)$. It is immediate to verify that the resulting set $\mathcal{R}_{i+1}$ of paths causes congestion at most $\lceil 1/\alpha \rceil$, and that each edge of $E_i$ participates in at most one such path. □

We obtain the following immediate corollary of Claim D.9.

**Corollary D.10** *For all $0 \leq i < r$, there is a collection $\mathcal{R}'_{i+1} = \{R'(e) \mid e \in E_{i+1}\}$ of paths such that, for every edge $e \in E_{i+1}$, path $R'(e)$ has $e$ as its first edge, some edge of $\delta_H(C)$ as its last edge, and all its inner vertices are contained in $S_{i+1} \setminus C$. Moreover, every edge in $\delta_H(C)$ may participate in at most one path in $\mathcal{R}'_{i+1}$, and the paths in $\mathcal{R}'_{i+1}$ cause congestion at most $\lceil 1/\alpha \rceil$.*

**Proof:** The proof is by induction on $i$. For $i = 0$, we let the set $\mathcal{R}'_1$ of paths contain, for each edge $e \in \delta_H(C) = \delta_H(S_1)$, a path $R'(e)$ that consists of a single edge - the edge $e$.

Assume now that we have defined the sets $\mathcal{R}'_1, \ldots, \mathcal{R}'_i$ of paths. In order to define the set $\mathcal{R}'_{i+1} = \{R'(e) \mid e \in E_{i+1}\}$ of paths, consider any edge $e \in E_{i+1}$, and let $R(e) \in \mathcal{R}_{i+1}$ be the unique path that has $e$ as its first edge. Denote by $e' \in E_i$ the last edge on path $R(e)$, and consider the path $R'(e') \in \mathcal{R}_i$, connecting $e'$ to an edge of $\delta_H(C)$. We obtain the path $R'(e)$ by concatenating $R(e)$ and $R'(e')$. It is easy to verify that the resulting set $\mathcal{R}'_{i+1} = \{R'(e) \mid e \in E_{i+1}\}$ of paths has all required properties. □

We are now ready to establish Property L6 of the decomposition. Consider some layer $\mathcal{L}_i$, for $1 \leq i \leq r$, and some cluster $W \in \mathcal{L}_i$. Recall that, when the algorithm from Theorem 4.19 was applied to cluster $S'_i$ in iteration $i$, it returned a set $\mathcal{P}'(W)$ of paths in graph $H$, routing the edges of $\delta_H(W)$ to edges of $\delta_H(S'_i) = E_i$, such that the paths of $\mathcal{P}'(W)$ avoid $W$ and cause congestion at most 100 in $H$. We can assume without loss of generality that, if an edge of $E_i$ lies on a path of $\mathcal{P}'(W)$, then it is the last edge on that path. Equivalently, no path of $\mathcal{P}'(W)$ may contain a vertex of $S_i$ as its inner vertex. Consider now some edge $e \in \delta_H(W)$, and let $P'(e) \in \mathcal{P}'(W)$ be the path whose first edge is $e$. Let $e' \in E_i$ be the last edge on path $P'(e)$, and let $R'(e')$ be the unique path in $\mathcal{R}'_i$ that has $e'$ as its first edge. Recall that the last edge of $\mathcal{R}'_i$ lies in $\delta_H(C)$. Moreover, since all inner vertices on path $R'(e')$ lie in $S_i \setminus C$, no inner vertex of path $R'(e')$ may lie in $W$. By concatenating the paths $P'(e)$ and $R'(e')$, we obtain a path $P(e)$, whose first edge is $e$, and last edge lies in $\delta_H(C)$. From the above discussion, no inner vertex of $P(e)$ lies in $W$. We then let $\mathcal{P}(W) = \{P(e) \mid e \in \delta_H(W)\}$. We have now obtained a set of paths routing the edges of $\delta_H(W)$ to edges of $\delta_H(C)$, such that the paths in $\mathcal{P}(W)$ avoid $W$. It now remains to analyze the congestion that this set of paths causes in graph $H$. As the set $\mathcal{P}'(W)$ of paths causes congestion at most 100, every edge $e' \in E_i$ may participate in at most 100 such paths. Since the congestion caused by the set $\mathcal{R}'_i$ of paths is at most $\lceil 1/\alpha \rceil$, and since no vertex of $S_i$ may serve as an inner vertex on a path of $\mathcal{P}'(W)$, the total congestion caused by paths in $\mathcal{P}(W)$ is at most $100 \cdot \lceil \frac{1}{\alpha} \rceil \leq \frac{200}{\alpha}$.

## D.11 Proof of Claim 4.23

For convenience, we sometimes refer to vertices of $T$ as *terminals*. We use the cut-matching game of Khandekar, Rao and Vazirani [KRV09], defined as follows. The game is played between two players, called the cut player and the matching player. The input to the game is an even integer $N$. The game is played in iterations. We start with a graph $W$, whose vertex set $V$ has cardinality $N$, and the edge set is empty. In every iteration, some edges are added to $W$. The game ends when $W$ becomes a $\frac{1}{2}$-expander. The goal of the cut player is to construct a $\frac{1}{2}$-expander in as few iterations as possible, whereas the goal of the matching player is to prevent the construction of the expander for as long as possible. The iterations proceed as follows. In every iteration $j$, the cut player chooses a partition

$(Z_j, Z'_j)$ of $V$ with $|Z_j| = |Z'_j|$, and the matching player chooses a perfect matching $M_j$ that matches the nodes of $Z_j$ to the nodes of $Z'_j$. The edges of $M_j$ are then added to $W$. Khandekar, Rao, and Vazirani [KRV09] showed that there is an efficient randomized algorithm for the cut player (that is, an algorithm that, in every iteration $j$, given the current graph $W$, computes a partition $(Z_j, Z'_j)$ of $V$ with $|Z_j| = |Z'_j|$), that guarantees that after $O(\log^2 N)$ iterations, with high probability, graph $W$ is a $(1/2)$-expander, regardless of the specific matchings chosen by the matching player.

We use the above cut-matching game in order to compute an expander $W$ with vertex set $T$, and to embed it into $G$, using standard techniques. If $|T|$ is an even integer, then we start with the graph $W$ containing the vertices of $T$; otherwise, we let $t \in T$ be an arbitrary vertex, and we start with $V(W) = T \setminus \{t\}$. Initially, $E(W) = \emptyset$. We then perform iterations. In the $i$th iteration, we apply the algorithm of the cut player to the current graph $W$, and obtain a partition $(Z_i, Z'_i)$ of its vertices with $|Z_j| = |Z'_j|$. Using the algorithm from Theorem 4.17, we compute a collection $\mathcal{P}_i$ of paths in graph $G$, routing vertices of $Z_i$ to vertices of $Z'_i$, so that every vertex of $Z_i \cup Z'_i$ is an endpoint of exactly one path in $\mathcal{P}_i$, and the paths in $\mathcal{P}_i$ cause congestion at most $\lceil 1/\alpha \rceil \leq 2/\alpha$ in $G$. Let $M_i$ be the perfect matching between vertices of $Z_i$ and vertices of $Z'_i$ defined by the set $\mathcal{P}_i$ of paths: that is, we add to $M_i$ a pair $(t, t')$ of vertices iff some path in $\mathcal{P}_i$ has endpoints $t$ and $t'$. We then treat $M_i$ as the response of the matching player, and add the edges of $M_i$ to $W$, completing the current iteration of the game.

Let $W$ be the graph obtained after $i^* = O(\log^2 k)$ iterations, that is guaranteed to be a $1/2$-expander with high probability. We then set $\mathcal{P} = \bigcup_{i=1}^{i^*} \mathcal{P}_i$. It is immediate to verify that $\mathcal{P}$ is an embedding of $W$ into $G$. Since each set $\mathcal{P}_i$ of paths causes congestion $O(1/\alpha)$, and $i^* \leq O(\log^2 k)$, the paths in $\mathcal{P}$ cause congestion $O((\log^2 k)/\alpha)$. If $|T|$ is even, then we have constructed the desired expander and its embedding into $G$ as required. If $|T|$ is odd, then we add the terminal $t$ to the graph $W$. Let $P$ be any path in graph $G$, connecting $t$ to any terminal $t' \in T \setminus \{t\}$; such a path must exist since the set of terminals is $\alpha$-well-linked in $G$. We then add edge $(t, t')$ to graph $W$, and we let its embedding be $P(e) = P$; we add path $P$ to $\mathcal{P}$. It is easy to verify that this final graph $W$ is $1/4$ expander, provided that the original graph $W$ obtained at the end of the cut-matching game was a $1/2$-expander. We have also obtained an embedding of $W$ into $G$ with congestion $O((\log^2 k)/\alpha)$. Lastly, since the number of iterations in the cut-matching game is $O(\log^2 k)$, and the set of edges that is added to $W$ in every iteration is a maching, we get that the maximum vertex degree in $W$ is $O(\log^2 k)$.

### D.12 Proof of Observation 4.24

We assume that $c > 2^{120}$ is a large enough constant. Let $\Delta$ denote the maximum vertex degree in $W$. Since $c$ is a large enough constant, we can assume that $\Delta \leq c^{1/8} \log^2 k < k/2^{40}$. Assume for contradiction that $\mathsf{OPT}_{\mathsf{cr}}(W) < k^2/(c \log^8 k)$. From Lemma 4.12, there is a $(3/4)$-edge-balanced cut $(A, B)$ in $W$, with:

$$|E_W(A,B)| \leq O(\sqrt{\mathsf{OPT}_{\mathsf{cr}}(W) + \Delta \cdot |E(W)|}) \leq O(\sqrt{\mathsf{OPT}_{\mathsf{cr}}(W) + c^{1/4} \cdot k \cdot \log^4 k}) \leq O\left(\frac{k}{c^{1/4} \log^4 k}\right).$$

(We have used the fact that, since $k > c$, and $c$ is a large enough constant, $\log^4 k < \frac{k}{c^{3/4} \log^8 k}$.)

Since cut $(A, B)$ is a $(3/4)$-edge-balanced cut, $|E_W(A)| \leq 3|E(W)|/4$. Therefore, $|E_W(B)| \geq |E(W)|/4 - |E_W(A,B)| \geq |E(W)|/8$. Since the degree of every vertex in $W$ is at most $\Delta \leq c^{1/8} \log^2 k$, we get that $|B| \geq \frac{|E(W)|}{8\Delta} \geq \frac{|E(W)|}{8c^{1/8} \log^2 k}$. Using the same reasoning, $|A| \geq \frac{|E(W)|}{8c^{1/8} \log^2 k}$. Since graph $W$ is a $\frac{1}{4}$-expander, $|E_W(A,B)| \geq \frac{1}{4} \cdot \min\{|A|, |B|\} \geq \frac{1}{4} \cdot \frac{|E(W)|}{8c^{1/8} \log^2 k} > \frac{k}{32c^{1/8} \log^2 k}$ must hold, a contradiction.

## D.13 Proof of Corollary 4.25

The proof relies on known results for routing on expanders, that are summarized in the next claim, that is well-known, and follows immediately from the results of [LR99]. A proof can be found, e.g. in [Chu12].

**Claim D.11 (Corollary C.2 in [Chu12])** *There is an efficient randomized algorithm that, given as input an $n$-vertex $\alpha$-expander $H$, and any partial matching $M$ over the vertices of $H$, computes, for every pair $(u,v) \in M$, a path $P(u,v)$ connecting $u$ to $v$ in $H$, such that with high probability, the set $\{P(u,v) \mid (u,v) \in M\}$ of paths causes congestion $O(\log^2 n/\alpha)$ in $H$.*

We start by computing a graph $W$ with $V(W) = T$, and its embedding $\mathcal{P} = \{P(e) \mid e \in E(W)\}$ into $G$ with congestion $O((\log^2 k)/\alpha)$ using the algorithm from Claim 4.23 (recall that, with high probability, $W$ is an $(1/4)$-expander). Next, we use the algorithm from Claim D.11 to compute a collection $\mathcal{R}' = \{R'(u,v) \mid (u,v) \in M\}$ of paths in graph $W$, where for all $(u,v) \in M$, path $R(u,v)$ connects $u$ to $v$ in $W$, such that the congestion of the set $\mathcal{R}'$ of paths in $W$ is $O(\log^2 k)$ with high probability.

Lastly, we consider the paths $R'(u,v) \in \mathcal{R}'$ one by one. We transform each such path $R'(u,v)$ into a path $R(u,v)$ connecting $u$ to $v$ in graph $G$ by replacing, for every edge $e \in R'(u,v)$, the edge $e$ with the path $P(e) \in \mathcal{P}$ embedding the edge $e$ into $G$. Since the paths in $\mathcal{R}'$ with high probability cause congestion at most $O(\log^2 k)$ in $W$, while the paths in $\mathcal{P}$ cause congestion $O(\log^2 k/\alpha)$ in $G$, we get that with high probability, the paths in the resulting set $\mathcal{R} = \{R(u,v) \mid (u,v) \in M\}$ cause congestion $O(\log^4 k/\alpha)$ in $G$.

## D.14 Proof of Corollary 4.26

We partition the set $E(K)$ of edges into $3z$ matchings $M_1, \ldots, M_{3z}$, and then use Corollary 4.25 to compute, for each $1 \leq i \leq 3z$, a set $\tilde{\mathcal{R}}_i = \left\{\tilde{P}(e) \mid e \in M_i\right\}$ of paths in graph $G$, where for all $e = (t,t') \in M_i$, path $\tilde{P}(e)$ connects $t$ to $t'$, and with high probability, the paths in $\tilde{\mathcal{P}}_i$ cause edge-congestion $O((\log^4 z)/\alpha)$ in graph $G$. Let $\tilde{\mathcal{P}} = \bigcup_{i=1}^{3z} \tilde{\mathcal{P}}_i$. Then $\tilde{\mathcal{P}}$ is an embedding of $K_z$ into $G$, and with high probability, the congestion of this embedding is $O((z \log^4 z)/\alpha)$.

## D.15 Proof of Lemma 4.31

Suppose we are given a graph $G$, and, for every vertex $v \in V(G)$, an oriented circular ordering $(\mathcal{O}_v, b_v)$ of edges in $\delta_G(v)$. We say that a drawing $\varphi$ of $G$ *obeys the oriented orderings* $\{(\mathcal{O}_v, b_v)\}_{v \in V}$ *at the vertices of* $G$ if, for every vertex $v \in V(G)$, the oriented circular order in which the images of the edges of $\delta_G(v)$ enter $v$ in $\varphi$ is $(\mathcal{O}_v, b_v)$. We use the following theorem from [PSŠ11].

**Theorem D.12 (Corollary 5.6 of [PSŠ11])** *There is an efficient algorithm, that, given a two-vertex loopless multigraph $G$ (so $V(G) = \{v, v'\}$ and $E(G)$ only contains parallel edges connecting $v$ to $v'$), and, for each vertex $v \in V(G)$, an oriented ordering $(\mathcal{O}_v, b_v)$ of its incident edges, computes a drawing $\varphi$ of $G$ that obeys the given oriented orderings, such that $\mathsf{cr}(\varphi)$ is at most twice the minimum number of crossings of any drawing of $G$ that obeys the given oriented orderings.*

The proof of Lemma 4.31 easily follows Theorem D.12. Recall that we are given a pair $(\mathcal{O}, b)$, $(\mathcal{O}', b')$ of oriented orderings of a collection $U$ of elements. We construct a two-vertex loopless graph $G$ with oriented ordering on its vertices, as follows. Denote $U = \{u_1, \ldots, u_r\}$. The vertex set of $G$ is $\{v, v'\}$. The edge set of $G$ consists of $r$ parallel edges connecting $v$ to $v'$, that we denote by $e_{u_1}, e_{u_2}, \ldots e_{u_r}$, respectively. The oriented ordering $(\mathcal{O}, b)$ of the elements of $U$ naturally defines an oriented ordering

$(\hat{\mathcal{O}}, b)$ of the edges of $G$, and similarly, the oriented ordering $(\mathcal{O}, b')$ of the elements of $U$ defines an oriented ordering $(\hat{\mathcal{O}}, b')$ of the edges of $G$. We define the oriented orderings for the vertices of $G$ as follows: $(\mathcal{O}_v, b_v) = (\hat{\mathcal{O}}, -b)$ and $(\mathcal{O}_{v'}, b_{v'}) = (\hat{\mathcal{O}}', b')$.

Consider any drawing $\varphi$ of $G$ on the sphere that obeys the oriented orderings for $v, v'$ defined above. Let $D' = D_\varphi(v')$ be a tiny $v'$-disc. For all $1 \leq i \leq r$, we denote the unique point on the image of edge $e_{u_i}$ that lies on the boundary of $D'$ by by $p'_i$. Similarly, we let $\hat{D} = D_\varphi(v)$ be a tiny $v$-disc, and, for all $1 \leq i \leq r$, we denote the unique point on the image of edge $e_{u_i}$ that lies on the boundary of $\hat{D}$ by $p_i$. Let $D$ be the disc whose boundary is the same as the boundary of $\hat{D}$, but whose interior is disjoint from that of $\hat{D}$. Then $D' \subseteq D$, and the boundaries of the two discs are disjoint. Furthermore, points $p_1, \ldots, p_r$ appear on the boundary of $D$ according to the oriented ordering $(\mathcal{O}, b)$, and points $p'_1, \ldots, p'_r$ appear on the boundary of $D'$ according to the oriented ordering $(\mathcal{O}', b')$. For all $1 \leq i \leq r$, let $\gamma_i$ be the segment of the image of the edge $e_{u_i}$ between points $p_i$ and $p'_i$. Then $\{\gamma_i \mid 1 \leq i \leq r\}$ is a set of reordering curves for the orderings $(\mathcal{O}, b)$ and $(\mathcal{O}', b')$, and moreover, the cost of this curve set is exactly the number of crossings in $\varphi$.

Using a similar reasoning, any set $\Gamma$ of reordering curves for the orderings $(\mathcal{O}, b)$ and $(\mathcal{O}', b')$ can be converted into a drawing of graph $G$ that obeys the oriented orderings at vertices $v$ and $v'$, in which the number of crossings is exactly the cost of $\Gamma$. Therefore, there is a drawing of $G$ that obeys the oriented orderings at $v$ and $v'$, whose number of crossings is bounded by $\mathsf{dist}((\mathcal{O}, b), (\mathcal{O}', b'))$. We apply the algorithm from Theorem D.12 to graph $G$, and then compute a set of reordering curves from the resulting drawing of $G$ as described above. From the above discussion, the cost of the resulting set of curves is at most $2 \cdot \mathsf{dist}((\mathcal{O}, b), (\mathcal{O}', b'))$.

## D.16 Proof of Corollary 4.32

The proof easily follows from Lemma 4.31. We denote $\delta_G(v) = \{e_1, \ldots, e_r\}$, and, for all $1 \leq i \leq r$, we let $p_i$ be the unique point of $\varphi(e_i)$ lying on the boundary of the disc $D$. Let $\sigma_i$ be the segment of $\varphi(e_i)$ that is disjoint from the interior of the disc $D$. In other words, if $e_i = (v, u_i)$, then $\sigma_i$ is a curve connecting $\varphi(u_i)$ to $p_i$. Note that the points $p_1, \ldots, p_r$ appear on the boundary of $D$ according to the circular ordering $\mathcal{O}'_v$ of their corresponding edges. We assume w.l.o.g. that the orientation of the ordering is $b'_v = -1$.

Let $D'$ be another disc, that is contained in $D$, with $\varphi(v)$ lying in the interior of $D'$, such that the boundaries of $D$ and $D'$ are disjoint. We place points $p'_1, \ldots, p'_r$ on the boundary of the disc $D'$, so that all resulting points are distinct, and they appear on the boundary of $D'$ in the order $\mathcal{O}_v$ of their corresponding edges, using a positive orientation of the ordering. For all $1 \leq i \leq r$, we can compute a simple curve $\gamma_i$, connecting $\varphi(v)$ to $p'_i$, such that $\gamma_i$ is contained in $D'$ and only intersects the boundary of $D'$ at its endpoint $p'_i$. We also ensure that all resulting curves $\gamma_1, \ldots, \gamma_r$ are mutually internally disjoint. Using the algorithm from Lemma 4.31, we compute a collection $\Gamma = \{\gamma'_1, \ldots, \gamma'_r\}$ of reordering curves, where for $1 \leq i \leq r$, curve $\gamma'_i$ connects $p_i$ to $p'_i$, is contained in $D$, and is disjoint from the interior of $D'$. Note that the total number of crossings between the curves in $\Gamma$ is at most $2 \cdot \mathsf{dist}((\mathcal{O}_v, 1), (\mathcal{O}'_v, -1))$. For all $1 \leq i \leq r$, we define a new image of the edge $e_i$ to be the concatenation of the curves $\sigma_i, \gamma'_i$, and $\gamma_i$. The images of all remaining edges and vertices of $G$ remain unchanged. Denote the resulting drawing of the graph $G$ by $\varphi'$. It is immediate to verify that the edges of $\delta_G(v)$ enter the image of $v$ in the order $\mathcal{O}_v$ in $\varphi'$, and that the drawings of $\varphi$ and $\varphi'$ are identical except for the segments of the images of the edges in $\delta_G(v)$ that lie inside the disc $D$. It is also immediate to verify that $\mathsf{cr}(\varphi') \leq \mathsf{cr}(\varphi) + 2 \cdot \mathsf{dist}((\mathcal{O}_v, 1), (\mathcal{O}'_v, -1))$.

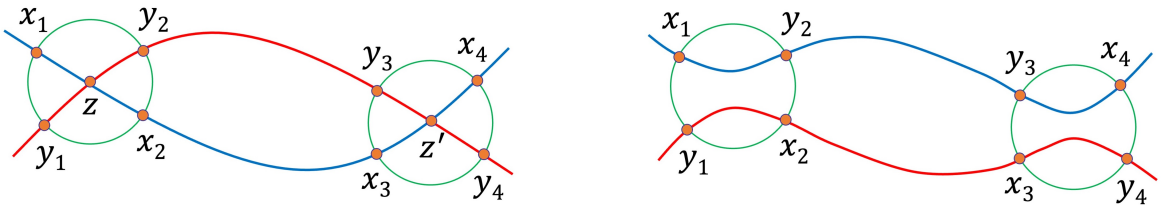
We repeat the same algorithm again, only this time the points $p'_1, \ldots, p'_r$ are placed on the boundary of disc $D'$ in the order $\mathcal{O}_v$ of their corresponding edges, using a negative orientation of the ordering. The remainder of the algorithm remains unchanged, and produces a drawing $\varphi''$ of $G$. As before, the

edges of $\delta_G(v)$ enter the image of $v$ in the order $\mathcal{O}_v$ in $\varphi''$, and the drawings of $\varphi$ and $\varphi''$ are identical except for the segments of the images of the edges in $\delta_G(v)$ that lie inside the disc $D$. Moreover, $\mathsf{cr}(\varphi'') \leq \mathsf{cr}(\varphi) + 2 \cdot \mathsf{dist}((\mathcal{O}_v, -1), (\mathcal{O}'_v, -1))$. Let $\varphi^*$ be the drawing with smaller number of crossings, among $\varphi'$ and $\varphi''$. Our algorithm returns the drawing $\varphi^*$ as its final outcome. From the above discussion, $\mathsf{cr}(\varphi^*) \leq \mathsf{cr}(\varphi) + 2 \cdot \mathsf{dist}(\mathcal{O}_v, \mathcal{O}'_v)$.

## D.17 Proof of Theorem 4.33

Let $Z$ be the set of crossings points between the curves of $\Gamma$. For each such crossing point $z \in Z$, we consider a tiny disc $D_z$, that contains the point $z$ in its interior. We select the discs $D_z$ to ensure that all such discs are disjoint, and, moreover, if $z$ is a crossing point between curves $\gamma_1, \gamma_2$, then for every curve $\gamma \in \Gamma \setminus \{\gamma_1, \gamma_2\}$, $\gamma \cap D_z = \emptyset$, while for every curve $\gamma \in \{\gamma_1 \cup \gamma_2\}$, $\gamma \cap D_z$ is a simple open curve whose endpoints lie on the boundary of $D_z$.

We start with $\Gamma'_1 = \Gamma_1$, and then iteratively modify the curves in $\Gamma'_1$, as long as there is a pair of distinct curves $\gamma_1, \gamma_2 \in \Gamma'_1$ that cross more than once. Each iteration is executed as follows. Let $\gamma_1, \gamma_2 \in \Gamma'_1$ be a pair of curves that cross more than once, and let $z, z'$ be two crossing points between $\gamma_1, \gamma_2$, that appear consecutively on $\gamma_1$. In other words, no other point that appears between $z$ and $z'$ on $\gamma_1$ may belong to $\gamma_2$. We denote by $s_1, t_1$ the endpoints of $\gamma_1$, such that $z$ appears closer to $s_1$ than $z'$ on $\gamma_1$. Similarly, we denote by $s_2, t_2$ the endpoints of $\gamma_2$, such that $z$ appears closer to $s_2$ than $z'$ on $\gamma_2$. We denote by $x_1, x_2$ the two points of $\gamma_1$ that lie on the boundary of disc $D_z$, and denote by $x_3, x_4$ the two points of $\gamma_1$ lying on the boundary of disc $D_{z'}$, such that the points $x_1, z, x_2, x_3, z', x_4$ appear on $\gamma_1$ in this order. We define points $y_1, y_2, y_3, y_4$ on $\gamma_2$ similarly (see Figure 43(a)).




(a) Before: Curve $\gamma_1$ is shown in blue and curve $\gamma_2$ is shown in red. The disc on the left is $D_z$, and the disc on the right is $D_{z'}$.

(b) After: Curve $\gamma'_1$ is shown in blue and curve $\gamma'_2$ is shown in red.

Figure 43: An iteration of the algorithm for performing a type-1 uncrossing.

In order to execute the iteration, we slightly modify the curves $\gamma_1, \gamma_2$, by "swapping" their segments between points $x_2, x_3$ and $y_2, y_3$, respectively, and slightly nudging them inside the discs $D_z, D_{z'}$, as show in Figure 43. We now describe the construction of the new curves $\gamma'_1, \gamma'_2$ more formally. Note that the points $x_1, x_2, y_1, y_2$ appear on the boundary of $D_z$ clockwise in either the order $(x_1, y_1, x_2, y_2)$ or the order $(x_1, y_2, x_2, y_1)$. Therefore, we can find two disjoint simple curves $\eta_1$ and $\eta_2$ that are contained in disc $D_z$, with $\eta_1$ connecting $x_1$ to $y_2$, and $\eta_2$ connecting $y_1$ to $x_2$. Similarly, we compute two disjoint simple curves curves $\eta'_1, \eta'_2$, that are contained in disc $D_{z'}$, with $\eta'_1$ connecting $y_3$ to $x_4$, and $\eta'_2$ connecting $x_3$ to $y_4$ (see Figure 43(b)).

We let $\gamma'_1$ be a curve, that is constructed by concatenating the following five curves: (1) the segment of $\gamma_1$ from $s_1$ to $x_1$; (2) curve $\eta_1$; (3) the segment of $\gamma_2$ from $y_2$ to $y_3$; (4) curve $\eta'_1$; and (5) the segment of $\gamma_1$ from $x_4$ to $t_1$. Similarly, let $\gamma'_2$ be a curve, that is constructed by concatenating the following

five curves: (1) the segment of $\gamma_2$ from $s_2$ to $y_1$; (2) curve $\eta_2$; (3) the segment of $\gamma_1$ from $x_2$ to $x_3$; (4) curve $\eta_2'$; and (5) the segment of $\gamma_2$ from $y_4$ to $t_2$. We then remove any self loops from the two curves, to obtain the final curves $\gamma_1', \gamma_2'$, that replace the curves $\gamma_1, \gamma_2$ in $\Gamma_1'$. Note that $\gamma_1'$ has the same endpoints as $\gamma_1$, and the same is true for $\gamma_2'$ and $\gamma_2$. It is also easy to verify that, at the end of the iteration, the number of crossings between the curves of $\Gamma_1' \cup \Gamma_2$ strictly decreases, and the number of crossings between the curves of $\Gamma_1'$ and the curves of $\Gamma_2$, that we denoted by $\chi(\Gamma_1', \Gamma_2)$, does not grow. Moreover, for every curve $\gamma \in \Gamma_2$, the number of crossings of $\gamma$ with the curves in $\Gamma_1'$ may not grow either. Once the algorithm terminates, we obtain the desired set $\Gamma_1'$ of curves, in which every pair of distinct curves crosses at most once. From the above discussion, it is immediate to verify that the curves in $\Gamma_1'$ have all required properties. Since the curves in $\Gamma$ are in general position, the number of iterations is bounded by the number of crossing points between the curves.

## D.18 Proof of Claim 4.34

We start by constructing a collection $\Gamma' = \{\gamma_i' \mid 1 \leq i \leq k\}$ of curves that are in general position, such that for all $1 \leq i \leq k$, curve $\gamma_i'$ has $s_i$ and $t_i$ as its endpoints, and is contained in disc $D$. In order to construct the set $\Gamma'$ of curves, we let $p$ be any point in the interior of the disc $D$, and $r > 0$ be some real number, such that a radius-$r$ circle centered at point $p$ is contained in the disc $D$. For all $1 \leq i \leq k$, let $\ell_i$ be a straight line, connecting point $s_i$ to $p$, and $\ell_i'$ a straight line, connecting point $t_i$ to $p$. We can assume that both lines are contained in the disc $D$, by stretching the disc as needed. For all $1 \leq i \leq k$, we choose a radius $0 < r_i < r$, so that $0 < r_1 < \cdots < r_k < r$ holds. For an index $1 \leq i \leq k$, we let $C_i$ be the boundary of a radius-$r_i$ circle centered at point $p$, and we let $q_i, q_i'$ be the points on lines $\ell_i$ and $\ell_i'$, respectively, that lie on $C_i$. We let curve $\gamma_i'$ be a concatenation of three curves: the segment of $\ell_i$ from $s_i$ to $q_i$; a segment of $C_i$ between $q_i$ and $q_i'$; and the segment of $\ell_i'$ from $q_i'$ to $t_i$. Consider the resulting set $\Gamma' = \{\gamma_i' \mid 1 \leq i \leq k\}$ of curves. Clearly, for all $1 \leq i \leq k$, curve $\gamma_i'$ has $s_i$ and $t_i$ as its endpoints, and is contained in disc $D$. It is also easy to verify that curves of $\Gamma'$ are in general position.

Next, we use the algorithm from Theorem 4.33 to perform a type-1 uncrossing of the curves in $\Gamma'$. Specifically, we set $\Gamma_1 = \Gamma'$ and $\Gamma_2 = \emptyset$. We denote by $\Gamma = \Gamma_1' = \{\gamma_i \mid 1 \leq i \leq k\}$ the set of curves that the algorithm outputs. Recall that, for all $1 \leq i \leq k$, curve $\gamma_i$ has $s_i$ and $t_i$ as its endpoints; the curves in $\Gamma$ are in general position; and every pair of curves in $\Gamma$ cross at most once. From the description of the type-1 uncrossing operation, it is easy to verify that all curves in $\Gamma$ are contained in the disc $D$.

Consider now two pairs $(s_i, t_i), (s_j, t_j)$ of points, with $i \neq j$. Note that curve $\gamma_i$ partitions the disc $D$ into two regions, that we denote by $F$ and $F'$. If the two pairs $(s_i, t_i), (s_j, t_j)$ cross, then $s_j, t_j$ may not lie on the boundary of the same region, and so curve $\gamma_j$ must cross curve $\gamma_i$ exactly once. If the two pairs do not cross, then $s_j, t_j$ either both lie on the boundary of $F$, or they both lie on the boundary of $F'$. It is then impossible that curves $\gamma_i, \gamma_j$ cross exactly once, and, since every pair of curves cross at most once, they cannot cross.

## D.19 Proof of Theorem 4.37

We start with an initial set $\Gamma' = \{\gamma'(Q) \mid Q \in \mathcal{Q}\}$ of curves, where, for each path $Q \in \mathcal{Q}$, $\gamma'(Q)$ is the image of the path $Q$ in $\varphi$. In other words, $\gamma'(Q)$ is the concatenation of the images of the edges of $Q$ in $\varphi$. Note however that the resulting set $\Gamma'$ of curves may not be in general position. This is since a vertex $v \in V(G)$ may serve as an inner vertex on more than two paths of $\mathcal{Q}$, and in such a case its image $\varphi(v)$ serves as an inner point of more than two curves in $\Gamma'$. Let $V' \subseteq V(G)$ be the set of all vertices $v \in V(G)$, such that more than two paths in $\mathcal{Q}$ contain $v$.

In our first step, we transform the set $\Gamma'$ of curves so that the resulting curves are in general position,

while ensuring that the endpoints of each curve $\gamma'(Q)$ remain unchanged, and each such curve $\gamma'(Q)$ remains aligned with the graph $\bigcup_{Q'\in\mathcal{Q}} Q'$. We do so by performing a *nudging operation* around every vertex $v \in V'$, as follows.

Consider any vertex $v \in V'$, and let $\mathcal{Q}(v) \subseteq \mathcal{Q}$ be the set of all paths containing vertex $v$. Note that $v$ must be an inner vertex on each such path. For convenience, we denote $\mathcal{Q}(v) = \{Q_1, \ldots, Q_z\}$. Consider the tiny $v$-disc $D = D_\varphi(v)$. For all $1 \leq i \leq z$, denote by $a_i$ and $b_i$ the two points on curve $\gamma(Q_i)$ that lie on the boundary of disc $D$. We use the algorithm from Claim 4.34 to compute a collection $\{\sigma_1, \ldots, \sigma_z\}$ of curves, such that, for all $1 \leq i \leq z$, curve $\sigma_i$ connects $a_i$ to $b_i$, and the interior of the curve is contained in the interior of $D$. Recall that every pair of resulting curves crosses at most once, and every point in the interior of $D$ may be contained in at most two curves. For all $1 \leq i \leq z$, we modify the curve $\gamma(Q_i)$, by replacing the segment of the curve that is contained in disc $D$ with $\sigma_i$. Once every vertex $v \in V'$ is processed, we obtain a collection $\Gamma'' = \{\gamma''(Q) \mid Q \in \mathcal{Q}\}$ of curves, where for every path $Q \in \mathcal{Q}$, curve $\gamma''(Q)$ connects $\varphi(s(Q))$ to $\varphi(t(Q))$. Moreover, it is easy to verify that each resulting curve $\gamma''(Q) \in \Gamma''$ is aligned with the drawing of the graph $\bigcup_{Q'\in\mathcal{Q}} Q'$ induced by $\varphi$, and that the curves in $\Gamma''$ are in general position.

We let $S$ be the multiset of points that contains, for every curve $\gamma''(Q) \in \Gamma''$ its first endpoint $\varphi(s(Q))$, and we let $T$ be the multiset of points contianing the last endpoint of each such curve.

We initially let, for each path $Q \in \mathcal{Q}$, $\gamma(Q)$ be the curve obtained by deleting all loops from $\gamma''(Q)$, and we denote by $\Gamma = \{\gamma(Q) \mid Q \in \mathcal{Q}\}$ the resulting set of curves. We gradually modify the curves in $\Gamma$ in order to eliminate all crossings between them. Throughout the algorithm, we ensure that for each path $Q \in \mathcal{Q}$, curve $\gamma(Q)$ originates at point $\varphi(s(Q))$, and moreover, if $e_1(Q)$ is the first edge on path $Q$, then segment $\varphi(e_1(Q)) \cap D_\varphi(s(Q))$ is contained in $\gamma(Q)$. We also ensure that the multiset containing the last point on every curve of $\Gamma$ remains unchanged throughout the algorithm. Let $P$ be the collection of all points $p$, such that at least two curves of $\Gamma$ contain $p$ as an inner point. We perform iterations, as long as $P \neq \emptyset$. Each iteration is executed as follows. Let $p \in P$ be any point, and let $Q, Q' \in \mathcal{Q}$ be two paths whose corresponding curves $\gamma(Q), \gamma(Q')$ contain the point $p$. Let $x, x'$ be the two points of $\gamma(Q)$ that lie on the boundary of the tiny $p$-disc $D(p)$, with $x$ appearing before $x'$ on $\gamma(Q)$. Let $y, y'$ be the two points of $\gamma(Q')$, that lie on the boundary of $D(p)$, with $y$ appearing before $y'$ on $\gamma(Q')$. We now consider two cases. In the first case, the circular clock-wise ordering of points $x, x', y, y'$ on the boundary of $D(p)$ is either $(x, y, x', y')$, or $(x, y', x', y)$. In this case, the two pairs $(x, y')$ and $(y, x')$ of points on the boundary of $D(p)$ do not cross. Therefore, from Claim 4.34, we can construct two disjoint curves $\sigma, \sigma'$ that are contained in $D(p)$, with $\sigma$ connecting $x$ to $y'$ and $\sigma'$ connecting $y$ to $x'$. We let $\gamma'(Q)$ be a curve that is obtained by concatenating the segment of $\gamma(Q)$ from its first endpoint to $x$; the curve $\sigma$; and the segment of $\gamma(Q')$ from $y'$ to its last endpoint. Similarly, we let $\gamma'(Q')$ be a curve that is obtained by concatenating the segment of $\gamma(Q')$ from its first endpoint to $y$; the curve $\sigma'$; and the segment of $\gamma(Q)$ from $x'$ to its last endpoint. We then replace $\gamma(Q)$ with $\gamma'(Q)$ and $\gamma(Q')$ with $\gamma'(Q')$ in $\Gamma$. In the second case, the circular clock-wise ordering of points $x, x', y, y'$ on the boundary of $D(p)$ must be either $(x, x', y, y')$, or $(x', x, y, y')$, or $(x, x', y', y)$, or $(x', x, y', y)$. In either case, the two pairs $(x, x')$ and $(y, y')$ of points on the boundary of $D(p)$ do not cross. Therefore, from Claim 4.34, we can construct two disjoint curves $\sigma, \sigma'$ that are contained in $D(p)$, with $\sigma$ connecting $x$ to $x'$ and $\sigma'$ connecting $y$ to $y'$. We modify curve $\gamma(Q)$ by replacing its segment that is contained in $D(p)$ with $\sigma$, and we similarly replace the segment of $\gamma(Q')$ that is contained in $D(p)$ with $\sigma'$. If either of the new curves $\gamma(Q), \gamma(Q')$ has loops, we turn the corresponding curve into a simple one by removing all loops from it. We also update the set $P$ of points, by removing from it points that no longer belong to two curves in $\Gamma$. This finishes the description of an iteration. It is easy to verify that no new crossing points between curves in $\Gamma$ are created, the curves in $\Gamma$ remain in general position, and each such curve is aligned with the drawing of the graph $\bigcup_{Q''\in\mathcal{Q}} Q''$ induced by $\varphi$. It is also easy to verify that the invariants continue to hold.

Consider the set $\Gamma$ of curves that we obtain at the end of the algorithm. Clearly, the curves in $\Gamma$ do not cross with each other, and they are aligned with the drawing of the graph $\bigcup_{Q\in\mathcal{Q}} Q$ induced by $\varphi$. It is also immediate to verify that they have all remaining required properties. Since $|P|$ strictly decreases from iteration to iteration, the number of iterations is bounded by the number of crossings of the drawing $\varphi$ of $G$, which is in turn bounded by the input size. Each iteration can be executed in time polynomial in the input size, so the algorithm is efficient.

## D.20 Proof of Corollary 4.38

For each edge $e \in E(G) \setminus E(C)$, we let $n_e = \mathrm{cong}_G(\mathcal{Q}, e)$. Let $H$ be a new graph, with $V(H) = V(G)$, whose edge set consists of the set $E(C)$ of edges, and, for each edge $e \in E(G) \setminus E(C)$, a set $J(e)$ of $n_e$ parallel copies of the edge $e$. Note that the drawing $\varphi$ of graph $G$ naturally defines a drawing $\varphi'$ of graph $H$. In order to obtain the drawing $\varphi'$ of $H$, we start with the drawing $\varphi$ of $G$, and then, for every edge $e \in E(G) \setminus E(C)$ with $n_e > 0$, we draw the edges of $J(e)$ in parallel to $\varphi(e)$, very close to it. We also delete the images of all edges $e \in E(G) \setminus E(C)$ with $n_e = 0$. Note that, for every edge $e \in E(C)$, the number of crossings between $\varphi'(e)$ and the images of the edges of $E(H) \setminus E(C)$ in the drawing $\varphi'$ is at most $\sum_{e'\in E(G)\setminus E(C)} \chi(e, e') \cdot \mathrm{cong}_G(\mathcal{Q}, e')$, where $\chi(e, e')$ is the number of crossings between $\varphi(e)$ and $\varphi(e')$.

The set $\mathcal{Q}$ of paths in graph $G$ naturally defines a set $\mathcal{Q}'$ of edge-disjoint paths in graph $H$, where, for each edge $e \in E(G) \setminus E(C)$, for every path $Q \in \mathcal{Q}$ containing the edge $e$, we replace $e$ with a distinct edge of $J(e)$ on path $Q$. In particular, the multisets $S(\mathcal{Q})$, $S(\mathcal{Q}')$ of vertices containing the first vertex of every path in set $\mathcal{Q}$ and $\mathcal{Q}'$, respectively, remain unchanged, and the same is true regarding the multisets $T(\mathcal{Q}), T(\mathcal{Q}')$ of paths, containing the last vertex of every path in set $\mathcal{Q}$ and $\mathcal{Q}'$, respectively.

We apply the algorithm from Theorem 4.37 to graph $H$, the drawing $\varphi'$ of $H$, and the set $\mathcal{Q}'$ of edge-disjoint paths in $H$. Let $\Gamma' = \{\gamma'(Q') \mid Q' \in \mathcal{Q}'\}$ be the resulting set of curves. Recall that, for every path $Q \in \mathcal{Q}$, there is a distinct path $Q' \in \mathcal{Q}'$, that is obtained from $Q$ by replacing each edge $e \in E(Q)$ with one of its copies. For each path $Q \in \mathcal{Q}$, we then let $\gamma(Q) = \gamma'(Q')$, and we consider the resulting set $\Gamma = \{\gamma(Q) \mid Q \in \mathcal{Q}\}$ of curves. The algorithm from Theorem 4.37 ensures that the curves in $\Gamma$ do not cross each other, and that, for every path $Q \in \mathcal{Q}$, $s(\gamma(Q)) = \varphi(s(Q))$. It also guarantees that the multiset $T(\Gamma)$ is precisely the multiset $\{\varphi(t(Q)) \mid Q \in \mathcal{Q}\}$. Lastly, consider any edge $e \in E(C)$. Since the curves in set $\Gamma'$ are aligned with the drawing of the graph $\bigcup_{Q'\in\mathcal{Q}'} Q'$ induced by $\varphi'$, the number of crossings between $\varphi'(e) = \varphi(e)$ and the curves in set $\Gamma' = \Gamma$ is bounded by the number of crossings between $\varphi'(e)$ and the images of the edges of $E(H) \setminus E(C)$ in drawing $\varphi'$ of $H$, which is, in turn, bounded by $\sum_{e'\in E(G)\setminus E(C)} \chi(e, e') \cdot \mathrm{cong}_G(\mathcal{Q}, e')$.

## D.21 Proof of Claim 4.39

Consider any $T$-cut $(A, B)$ in graph $G$, and denote $T_A = T \cap A$ and $T_B = T \cap B$. Assume without loss of generality that $|T_A| \le |T_B|$. It is enough to show that $|E_G(A, B)| \ge (\alpha_1\alpha_2) \cdot |T_A|$. Assume for contradiction that this is not the case.

Denote $H = G_{|\mathcal{C}}$. We partition the set $V(H)$ of vertices into two subsets: set $V' = V(H) \cap V(G)$ of regular vertices, and set $V'' = \{v_C \mid C \in \mathcal{C}\}$ of supernodes. Note that $T \subseteq V'$ must hold. We use the cut $(A, B)$ in $G$, in order to construct a cut $(A', B')$ in graph $H$, with $A' \cap T = T_A$ and $B' \cap T = T_B$, such that $|E_H(A', B')| < \alpha_2 \cdot |T_A|$, contradicting the fact that vertex set $T$ is $\alpha_2$-well-linked in graph $G_C$.

In order to construct the cut $(A', B')$ in $H$, we first process every vertex $v \in V'$ one by one. For each such vertex $v$, if $v \in A$, then we add $v$ to $A'$, and otherwise we add it to $B'$. Notice that this process guarantees that $T_A \subseteq A'$ and $T_B \subseteq B'$.

Next, we process every cluster $C \in \mathcal{C}$ one by one. Notice that partition $(A, B)$ of $V(G)$ naturally defines a partition $(A_C, B_C)$ of $V(C)$, where $A_C = A \cap V(C)$ and $B_C = B \cap V(C)$. We denote $E'_C = E_G(A_C, B_C)$, $E_1(C) = \delta_G(A_C) \setminus E'_C$, and $E_2(C) = \delta_G(B_C) \setminus E'_C$. If $|E_1(C)| \leq |E_2(C)|$, then we add supernode $v_C$ to $B'$, and otherwise we add it to $A'$. Assume w.l.o.g. that $v_C$ was added to $B'$, so $|E_1(C)| \leq |E_2(C)|$ holds. From the $\alpha_1$-bandwidth property of $C$, we get that $|E'_C| \geq \alpha_1 \cdot |E_1(C)|$. Notice that the edges of $E'_C$ lie in the cut $(A, B)$ in graph $G$, but they do not contribute to the cut $(A', B')$ in graph $H$. On the other hand, edges of $E_1(C)$ may lie in $E_H(A', B') \setminus E_G(A, B)$. We *charge* the edges of $E_1(C)$ to the edges of $E'$. Since $|E'| \geq \alpha_1 \cdot |E_1(C)|$, every edge of $E'$ pays at most $1/\alpha_1$ units for the edges of $E_1(C)$, so the total charge to the edges of $E'$ is $|E_1(C)|$.

Once every cluster $C \in \mathcal{C}$ is processed, we obtain the final cut $(A', B')$ in graph $H$. For every edge $e \in E_H(A', B')$, either $e \in E_G(A, B)$, or $e$ is charged to some edges of $E_G(A, B) \setminus E_H(A', B')$. Since the charge to every edge of $E_G(A, B) \setminus E_H(A', B')$ is at most $1/\alpha_1$, we get that $|E_H(A', B')| \leq |E_G(A, B)|/\alpha_1$. Since we have assumed that $|E_G(A, B)| < (\alpha_1\alpha_2) \cdot |T_A|$, we get that $|E_H(A', B')| < \alpha_2 \cdot |T_A| = \alpha_2 \cdot |T \cap A'|$, contradicting the fact that vertex set $T$ is $\alpha_2$-well-linked in $H$.

## D.22 Proof of Corollary 4.40

Let $G^+$ be the graph obtained from $G$ by subividing each edge $e \in \delta_G(R)$ with a new vertex $t_e$. Denote $T = \{t_e \mid e \in \delta_G(R)\}$. Recall that the augmentation $R^+$ of cluster $R$ in $G$ is defined to be the subgraph of $G^+$ induced by vertex set $V(R) \cup T$. It is immediate to verify that every cluster $C \in \mathcal{C}$ has the $\alpha_1$-bandwidth property in graph $R^+$. Furthermore, from Observation 4.16, the set $T$ of vertices is $\alpha_2$-well-linked in graph $R^+_{|\mathcal{C}}$. By applying Claim 4.39 to graph $R^+$, vertex set $T$ and collection $\mathcal{C}$ of clusters, we get that $T$ is $(\alpha_1 \cdot \alpha_2)$-well-linked in graph $R^+$. From Observation 4.16, cluster $R$ has the $(\alpha_1 \cdot \alpha_2)$-bandwidth property in $G$.

## D.23 Proof of Claim 4.41

For convenience, we denote $|T| = k$. We assume w.l.o.g. that the paths in $\mathcal{P}$ are simple, and we direct each such path towards $x$. We then graduately modify the paths in $\mathcal{P}$, by processing the clusters of $\mathcal{C}$ one by one.

Consider any cluster $C \in \mathcal{C}$, and let $\mathcal{P}(C) \subseteq \mathcal{P}$ be the subset of paths that contain the supernode $v_C$. For each path $P \in \mathcal{P}(C)$, let $e_P(C)$ and $e'_P(C)$ denote the edges appearing immediately before and immediately after $v_C$ on $P$. We denote $E_1(C) = \{e_P(C) \mid P \in \mathcal{P}(C)\}$ and $E_2(C) = \{e'_P(C) \mid P \in \mathcal{P}(C)\}$. We use the algorithm from Corollary 4.18, to compute a collection $\mathcal{R}(C)$ of paths that is a one-to-one routing of the edges of $E_1(C)$ to the edges of $E_2(C)$, such that all inner vertices on the paths of $\mathcal{R}(C)$ lie in $C$, and every edge in $E(C)$ participates in at most $\lceil 1/\alpha \rceil$ such paths. We modify the paths in set $\mathcal{P}(C)$ as follows. First, for each path $P \in \mathcal{P}(C)$, we delete the vertex $v_C$ from $P$, together with its two incident edges. Let $P_1, P_2$ be the two resulting subpaths of $P$. We then let $\mathcal{P}_1(C) = \{P_1 \mid P \in \mathcal{P}(C)\}$, and $\mathcal{P}_2(C) = \{P_2 \mid P \in \mathcal{P}(C)\}$. Lastly, let $\mathcal{P}'(C)$ be the set of paths obtained by concatenating the paths in $\mathcal{P}_1(C), \mathcal{R}(C)$ and $\mathcal{P}_2(C)$. We delete from $\mathcal{P}$ the paths that belong to $\mathcal{P}(C)$, and add the paths of $\mathcal{P}'(C)$ instead. It is easy to verify that the resulting set $\mathcal{P}$ of paths still routes the vertices of $T$ to $x$.

Once we process every cluster $C \in \mathcal{C}$, we obtain a collection $\mathcal{P}'$ of $k$ paths, routing the vertices of $T$ to vertex $x$ in graph $G$. Since the paths of $\mathcal{P}$ at the beginning of the algorithms are edge-disjoint, for each edge $e \in E(G) \setminus \left(\bigcup_{C \in \mathcal{C}} E(C)\right)$, $\mathrm{cong}_G(\mathcal{P}', e) \leq 1$. From our construction, for each edge $e \in \bigcup_{C \in \mathcal{C}} E(C)$, $\mathrm{cong}_G(\mathcal{P}', e) \leq \lceil 1/\alpha \rceil \leq 2/\alpha$. Lastly, we apply the algorithm from Claim 4.2, to graph $G$ and the set $\mathcal{P}'$ of paths, to obtain a collection $\mathcal{P}''$ of at least $\alpha k/2$ edge-disjoint paths in graph $G$, where each path in $\mathcal{P}''$ connects a distinct vertex of $T$ to $x$.

## D.24 Proof of Claim 4.42

Let $\varphi^*$ be an optimal solution to instance $I$ of MCNwRS. Let $G'$ be the graph that is obtained from $G$ by subdividing every edge $e \in \bigcup_{C\in\mathcal{C}} \delta_G(C)$ with a vertex $t_e$, and let $T = \left\{t_e \mid e \in \bigcup_{C\in\mathcal{C}} \delta_G(C)\right\}$ be the resulting set of new vertices. For every cluster $C \in \mathcal{C}$, we denote by $T_C = \{t_e \mid e \in \delta_G(C)\}$, and we let $C^+$ be the subgraph of $G'$ induced by vertex set $V(C) \cup T_C$. From Observation 4.16, vertex set $T_C$ is $\alpha$-well-linked in $C^+$. Observe that drawing $\varphi^*$ of $G$ naturally defines a drawing $\varphi'$ of graph $G'$, with $\mathsf{cr}(\varphi') = \mathsf{cr}(\varphi^*)$. We denote $\mathcal{C} = \{C_1, \ldots, C_r\}$, where the clusters are indexed arbitrarily. For $1 \le i \le r$, we let $\mathcal{C}_i = \{C_1, \ldots, C_i\}$, and we let $G'_i = G'_{|\mathcal{C}_i}$. We also denote $G'_0 = G'$. We perform $r$ iterations. The input to the $i$th iteration is a drawing $\varphi'_{i-1}$ of the graph $G'_{i-1}$, and the output is a drawing $\varphi'_i$ of the graph $G'_i$. We set $\varphi'_0 = \varphi'$.

We now describe the $i$th iteration, for $1 \le i \le r$. For convenience, we denote $C_i = C$. Corollary 4.28 guarantees that there is a distribution $\mathcal{D}(C)$ over the set $\Lambda(C)$ of internal $C$-routers, such that, for every edge $e \in E(C)$, $\mathbf{E}_{\mathcal{Q}(C)\sim\mathcal{D}(C)}\left[\mathrm{cong}(\mathcal{Q}(C), e)\right] \le O((\log|\delta_G(C)|)^4/\alpha) \le O((\log^4 m)/\alpha)$. We let $\mathcal{Q}(C) = \{Q(e) \mid e \in \delta_G(C)\}$ be an internal $C$-router sampled from the distribution $\mathcal{D}(C)$, and we denote by $u(C)$ the center of the router $\mathcal{Q}(C)$. Note that the set $\mathcal{Q}(C)$ of paths in graph $G$ naturally defines a set of paths in graph $C^+$, routing the vertices of $T_C$ to vertex $u(C)$. Abusing the notation, we denote this set of paths by $\mathcal{Q}(C)$ as well.

Applying the algorithm from Corollary 4.38 to graph $G'_{i-1}$, its drawing $\varphi'_{i-1}$, subgraph $G'_{i-1} \setminus C^+$, and the set $\mathcal{Q}(C)$ of paths, we obtain a collection $\Gamma(C) = \left\{\gamma(e) \mid e \in \delta_{G'_{i-1}}(C)\right\}$ of curves, such that, for every edge $e \in \delta_{G'_{i-1}}(C)$, curve $\gamma(e)$ originates at the image of the endpoint of $e$ that lies in $T_C$, and terminates at the image of $u(C)$. Furthermore, the curves in $\Gamma$ do not cross each other, and, for every edge $e \in E(G'_{i-1}) \setminus E(C^+)$, the number of crossings between $\varphi'_{i-1}(e)$ and the curves in $\Gamma(C)$ is bounded by $\sum_{e'\in E(C^+)} \chi(e, e') \cdot \mathrm{cong}_{G'}(\mathcal{Q}(C), e')$, where $\chi(e, e')$ is the number of crossings between $\varphi'_{i-1}(e)$ and $\varphi'_{i-1}(e')$. For every edge $e' \in E(C^+)$, and every crossing $(e, e')_p$ between $e$ and $e'$ in $\varphi'_{i-1}$, we *charge* this crossing $\mathrm{cong}_G(\mathcal{Q}(C), e')$ units, and we say that crossing $(e, e')_p$ is *responsible* for $\mathrm{cong}_{G'}(\mathcal{Q}(C), e')$ new crossings between the edge $e$ and the curves in $\Gamma(C)$. Therefore, the total charge to all crossings between $e$ and the edges of $E(C^+)$ is at least the total number of crossings between $\varphi'_{i-1}(e)$ and the curves in $\Gamma(C)$. We obtain a drawing $\varphi'_i$ of the graph $G_i$ as follows. We start from the drawing $\varphi'_{i-1}$ of graph $G'_{i-1}$, and delete all edges and vertices of $C_i^+ \setminus T_C$ from it. We place the image of the supernode $v_{C_i}$ at the image of the vertex $u(C_i)$ in $\varphi'_{i-1}$. For every edge $e \in \delta_{G'_i}(v_{C_i})$, we let $\gamma(e) \in \Gamma$ be the new image of the edge $e$. This concludes the definition of the drawing $\varphi'_i$ of graph $G'_i$. Note that:

$$\mathsf{cr}(\varphi'_i) - \mathsf{cr}(\varphi'_{i-1}) \le \sum_{e\in E(G'_{i-1})\setminus E(C_i^+)} \sum_{e'\in E(C_i^+)} \chi(e, e') \cdot \mathrm{cong}_{G'}(\mathcal{Q}(C), e').$$

Once every cluster $C \in \mathcal{C}$ is processed in this manner, we obtain the final drawing $\varphi$ of $G_{|\mathcal{C}}$, by suppressing the images of the vertices of $T$ in the drawing $\varphi'_r$ of the graph $G'_r$. Note that for every vertex $x \in V(G_{|\mathcal{C}}) \cap V(G)$, we did not modify the images of the edges of $\delta_G(x)$ inside the tiny $\varphi^*$-disc $D_{\varphi^*}(x)$, so the order in which these edges enter the image of $x$ continues to be $\mathcal{O}_x$. It now remains to bound the number of crossings in the drawing $\varphi$. We only bound the number of new crossings that were added due to the transformations that we perform.

Consider some crossing $(e, e')_p$ in the drawing $\varphi'$ of graph $G'$. If neither of the edges $e, e'$ lie in $\bigcup_{C\in\mathcal{C}} E(C^+)$, then no new crossings between these edges were introduced, and this crossing was not charged for any new crossings. Assume next that $e \in E(C_i^+)$, for some cluster $C_i \in \mathcal{C}$, and $e' \notin \bigcup_{C\in\mathcal{C}} E(C^+)$. Then crossing $(e, e')_p$ may be responsible for up to $\mathrm{cong}_{G'}(\mathcal{Q}(C_i), e)$ new crossings. Each of these new crossings is between the image of $e'$ and the images of the edges of $\delta_{G'_i}(v_{C_i})$, and so they cannot be responsible for any additional new crossings. Since $\mathbf{E}\left[\mathrm{cong}_{G'}(\mathcal{Q}(C_i), e)\right] \le$

$O((\log^4 m)/\alpha)$, the total expected number of crossings for which crossing $(e,e')_p$ is responsible is at most $O((\log^4 m)/\alpha)$.

Lastly, assume that $e \in E(C_i^+)$ and $e' \in E(C_j^+)$, for $C_i, C_j \in \mathcal{C}$. If $i = j$, then crossing $(e,e')_p$ is not responsible for any new crossings. Assume now without loss of generality that $i < j$. After cluster $C_i$ is processed, crossing $(e,e')_p$ may be responsible for at most $\mathrm{cong}_{G'}(\mathcal{Q}(C_i), e)$ new crossings. All these new crossings are between the images of the edges of $\delta_{G'_i}(v_{C_i})$ and the image of edge $e'$. Once cluster $C_j$ is processed, each of the resulting crossings may in turn be responsible for at most $\mathrm{cong}_{G'}(\mathcal{Q}(C_j), e')$ new crossings. Each of these new crossings is between images of edges in $\delta_{G'_j}(v_{C_i})$ and images of edges in $\delta_{G'_j}(v_{C_j})$, so they in turn will not be responsible for any new crossing. Therefore, overall, crossing $(e,e')_p$ may be responsible for up to $\mathrm{cong}_{G'}(\mathcal{Q}(C_i), e) \cdot \mathrm{cong}_{G'}(\mathcal{Q}(C_j), e')$ new crossings. Since $\mathrm{cong}_{G'}(\mathcal{Q}(C_i), e)$ and $\mathrm{cong}_{G'}(\mathcal{Q}(C_j), e')$ are independent random variables, and the expected value of each of these variables is at most $O((\log^4 m)/\alpha)$, the expected number of crossings for which crossing $(e,e')_p$ is responsible is at most $O((\log^8 m)/\alpha^2)$. We conclude that $\mathbf{E}\left[\mathsf{cr}(\varphi)\right] \leq O((\log^8 m)/\alpha^2) \cdot \mathsf{cr}(\varphi') \leq O((\log^8 m)/\alpha^2) \cdot \mathsf{OPT}_{\mathsf{cnwrs}}(I)$. Therefore, there exists a drawing $\varphi$ of the contracted graph $G_{|\mathcal{C}}$, with $\mathsf{cr}(\varphi) \leq O((\mathsf{OPT}_{\mathsf{cnwrs}}(I) \cdot \log^8 m)/\alpha^2)$, in which, for every regular vertex $x \in V(G_{|\mathcal{C}}) \cap V(G)$, the ordering of the edges of $\delta_G(x)$ as they enter $x$ in $\varphi$ is consistent with the rotation $\mathcal{O}_x \in \Sigma$.

# E Proofs Omitted from Section 5

## E.1 Proof of Lemma 5.3

In this proof, we assume that all drawings are on the sphere. For every cluster $C \in \mathcal{L}$, denote by $\mathcal{W}(C) \subseteq \mathcal{L}$ the set of all child clusters of $C$, and by $\mathcal{W}^*(C) \subseteq \mathcal{L}$ the set of all descendant clusters of $C$. We define a new instance $I'_C = (G'_C, \Sigma'_C)$ of $\mathsf{MCNwRS}$ associated with cluster $C$, as follows. If $C = G$, then $G'_C = G$ and $\Sigma'_C = \Sigma$. Otherwise, graph $G'_C$ is obtained from graph $G$, by contracting all vertices of $V(G) \setminus V(C)$ into a supernode $v^*$. Rotation system $\Sigma'_C$ is defined as follows. Note that $\delta_{G'_C}(v^*) = \delta_G(C)$. We define the rotation $\mathcal{O}_{v^*} \in \Sigma'_C$ to be $\mathcal{O}(C)$. For every other vertex $v \in V(G'_C)$, $\delta_{G'_C}(v) = \delta_G(v)$ holds, and its rotation $\mathcal{O}_v \in \Sigma'_C$ remains the same as in $\Sigma$. This completes the definition of instance $I'_C$. Notice that instance $I_C$ can be obtained from instance $I'_C$ by contracting, for each cluster $C' \in \mathcal{W}(C)$, the vertices of $C'$ into a supernode $v_{C'}$, and then setting the rotation of the edges incident to this supernode to $\mathcal{O}(C')$.

We prove by induction that there is an efficient algorithm, that, given a cluster $C \in \mathcal{L}$, and solutions $\{\varphi(I_{C'})\}_{C' \in \mathcal{W}^*(C)}$ to instances associated with the descendant clusters of $C$, computes a solution $\varphi'(I'_C)$ to instance $I'_C$, of cost at most $\sum_{C' \in \mathcal{W}^*(C)} \varphi(I_{C'})$. Since $I'_G = I$, this will complete the proof of the lemma.

The proof is by induction of the length of the longest path in the partitioning tree $\tau(\mathcal{L})$ between $v(C)$ and its descendant. The base of the induction is when cluster $C$ is the leaf of the tree $\tau(\mathcal{L})$. In this case, $I'_C = I_C$ holds, and we let $\varphi'(I'_C) = \varphi(I_C)$.

For the induction step, we consider some cluster $C \in \mathcal{L}$, whose corresponding vertex $v(C)$ is not a leaf vertex of the tree $\tau(\mathcal{L})$. Assume that $\mathcal{W}(C) = \{C_1, \ldots, C_r\}$. For convenience, for each $1 \leq i \leq r$, we denote the supernode $v_{C_i}$ representing cluster $C_i$ in graph $G_C$ by $v_i$. By applying the induction hypothesis to every cluster $C_i \in \mathcal{W}(C)$, we obtain a solution $\varphi'_i = \varphi'(I'_{C_i})$ to instance $I'_{C_i}$ of $\mathsf{MCNwRS}$, whose cost is $\mathsf{cr}(\varphi'_i) \leq \sum_{C' \in \mathcal{W}^*(C_i)} \mathsf{cr}(\varphi(I_{C'}))$. It is now enough to show an efficient algorithm that constructs a solution $\varphi'(I'_C)$ to instance $I'_C$, whose cost is at most $\mathsf{cr}(\varphi(I_C)) + \sum_{i=1}^{r} \mathsf{cr}(\varphi'_i) \leq \sum_{C' \in \mathcal{W}^*(C)} \mathsf{cr}(\varphi(I_{C'}))$.

We start with the solution $\tilde{\varphi} = \varphi(I_C)$ to instance $I_C$, and we process the clusters $C_1, \ldots, C_r$ one by one, gradually modifying the drawing $\tilde{\varphi}$. We now describe the iteration when cluster $C_i$ is processed. We denote $\delta_{G'_C}(v_i) = \delta_G(C_i) = \left\{e^i_1, \ldots, e^i_{q_i}\right\}$, where the edges are indexed according to their ordering

in $\mathcal{O}(C_i)$. For all $1 \leq j \leq q_i$, we denote $e^i_j = (x_j, y_j)$, where $x_j \in C_i$. Let $D_i = D_{\tilde{\varphi}}(v_i)$ be a tiny $v_i$-disc in the drawing $\tilde{\varphi}$. For all $1 \leq j \leq q_i$, we denote by $p^i_j$ the unique point on the image of edge $e^i_j$ that lies on the boundary of the disc $D_i$, and we let $\gamma(e^i_j)$ denote the segment of the image of $e^i_j$ that is disjoint from the interior of $D_i$. Therefore, $\gamma(e^i_j)$ connects the image of vertex $y_j$ to point $p^i_j$. Notice that points $p^i_1, \ldots, p^i_{q_i}$ appear on the boundary of $D_i$ in this circular order. If the orientation of this ordering is positive, then we say that vertex $v_i$ is positive, and otherwise we say that it is negative. We erase the parts of the images of all edges in the interior of disc $D_i$, and we erase the image of the vertex $v_i$ from the current drawing. We place another disc $D'_i$ inside $D_i$, so that $D'_i \subseteq D_i$, and the boundaries of both discs are disjoint.

Next, we consider the drawing $\varphi'_i$ of the graph $G'_{C_i}$. We let $\hat{D}_i = D_{\varphi'_i}(v^*)$ be a tiny $v^*$-disc in this drawing. Recall that $\delta_{G'_{C_i}}(v^*) = \delta_G(C_i)$. For all $1 \leq j \leq q_i$, we denote by $\hat{p}^i_j$ the unique point on the image of the edge $e^i_j$ in $\varphi'_i$ that lies on the boundary of the disc $\hat{D}_i$. Note that points $\hat{p}^i_1, \ldots, \hat{p}^i_{q_i}$ must appear on the boundary of the disc $\hat{D}_i$ in this circular order, from the definition of the rotation $\mathcal{O}_{v^*} \in \Sigma'_{C_i}$. We assume w.l.o.g. that, if vertex $v_i$ is positive, then the orientation of this ordering is negative, and otherwise it is positive (if this is not the case then we simply flip the drawing $\varphi'_i$). Let $\hat{D}'_i$ be the disc that has the same boundary as $\hat{D}_i$ but whose interior is disjoint from that of $\hat{D}_i$ (so $\hat{D}'_i$ is the complement of disc $\hat{D}_i$; recall that the drawing $\varphi'_i$ is on the sphere). For all $1 \leq j \leq q_i$, we denote by $\gamma'(e^i_j)$ the segment of the image of edge $e^i_j$ that lies inside $\hat{D}'_i$. Therefore, $\gamma'(e^i_j)$ connects the image of vertex $x_j$ to point $\hat{p}^i_j$.

We copy the disc $\hat{D}'_i$, together with its contents (in $\varphi'_i$), to the current drawing $\tilde{\varphi}$, so that the boundaries and the interiors of the discs $\hat{D}'_i$ and $D'_i$ coincide. Assume w.l.o.g. that vertex $v_i$ is positive. Then points $p^i_1, \ldots, p^i_{q_i}$ appear on the boundary of disc $D'_i$ in this counter-clock-wise order, while points $\hat{p}^i_1, \ldots, \hat{p}^i_{q_i}$ appear on the boundary of disc $\hat{D}'_i$ in this counter-clock-wise order. Therefore, we can compute a collection $\{\sigma_1, \ldots, \sigma_{q_i}\}$ of mutually disjoint curves, where for all $1 \leq j \leq q_i$, curve $\sigma_j$ has endpoints $p^i_j$ and $\hat{p}^i_j$, and all inner points of $\sigma_j$ lie in $D_i \setminus D'_i$, and are disjoint from the boundary of $D_i$. For all $1 \leq j \leq q_i$, we now define the image of the edge $e_j = (x_j, y_j)$ to be the concatenation of the curves $\gamma^i_j, \sigma_j$, and $\hat{\gamma}^i_j$.

Once every cluster $C_i \in \mathcal{W}(C)$ is processed in this manner, we obtain a solution $\varphi'(I'_C)$ to instance $I'_C$ of MCNwRS. It is immediate to verify that the total number of crossings in this solution is at most $\mathsf{cr}(\varphi(I_C)) + \sum_{i=1}^{r} \mathsf{cr}(\varphi'_i) \leq \sum_{C' \in \mathcal{W}^*(C)} \mathsf{cr}(\varphi(I_{C'}))$. The lemma follows by letting $\varphi$ be the solution $\varphi'(I_{C'})$ that we construct for instance $I_{C'}$, where $C' = G$.

## E.2 Proof of Lemma 5.6

Throughout the proof, we denote $|E(G)| = m$.

Consider first a cluster $C \in \mathcal{L}^{\text{light}}$. Recall that, in order to define instance $I_C$, we used the distribution $\mathcal{D}(C)$ over the internal $C$-routers, where $C$ is $\beta$-light with respect to $\mathcal{D}(C)$. We selected a router $\mathcal{Q}(C)$ from the distribution $\mathcal{D}(C)$ at random, whose center vertex is denoted by $u(C)$. We then used the algorithm from Lemma 4.7 to compute a non-transversal set $\tilde{\mathcal{Q}}(C)$ of paths, routing all edges of $\delta_G(C)$ to vertex $u(C)$, so $\tilde{\mathcal{Q}}(C)$ is also an internal $C$-router. The set $\tilde{\mathcal{Q}}(C)$ of paths was used in order to define the ordering $\mathcal{O}(C)$ of the edges of $\delta_G(C)$, which was in turned used in order to define instance $I_C$.

Consider now a cluster $C \in \mathcal{L}^{\text{bad}}$. We apply the algorithm from Corollary 4.28 to $C$, obtaining a distribution $\mathcal{D}(C)$ over the set $\Lambda(C)$ of internal $C$-routers, such that, for every edge $e \in E(C)$, $\mathbf{E}_{\mathcal{Q} \sim \mathcal{D}}[\text{cong}(\mathcal{Q}(C), e)] \leq O(\log^4 m/\alpha_0) \leq O(\log^{16} m)$. We then select a router $\mathcal{Q}(C)$ from the distribution $\mathcal{D}(C)$ at random, and denote by $u(C)$ its center vertex. We view the paths of $\mathcal{Q}(C)$ as being

directed towards $u(C)$. Next, we use the algoritm from Lemma 4.7 to compute a non-transversal set $\tilde{\mathcal{Q}}(C)$ of paths, routing all edges of $\delta_G(C)$ to vertex $u(C)$, so $\tilde{\mathcal{Q}}(C)$ is also an internal $C$-router. The algorithm ensures that, for every edge $e \in E(G)$, $\mathrm{cong}_G(\tilde{\mathcal{Q}}(C), e) \leq \mathrm{cong}_G(\mathcal{Q}(C), e)$.

Consider an optimal solution $\varphi^*$ to instance $I$ of $\mathsf{MCNwRS}$. For every cluster $C \in \mathcal{L}$, denote by $\chi(C)$ the set of all crossings $(e, e')_p$ in the drawing $\varphi^*$, where at least one of the edges $e, e'$ lies in $E(C) \cup \delta_G(C)$. Recall that $\mathcal{I} = \{I_C \mid C \in \mathcal{L}\}$. The proof of the lemma follows from the following claim.

**Claim E.1** *For every cluster $C \in \mathcal{L}$, $\mathbf{E}\left[\mathsf{OPT}_{\mathsf{cnwrs}}(I_C)\right] \leq O(\beta^2 \cdot (|\chi(C)| + |E(C)|))$.*

Indeed, for all $1 \leq i \leq \mathsf{dep}(\mathcal{L})$, let $\mathcal{L}_i \subseteq \mathcal{L}$ be the set of all clusters that lie at level $i$ of the laminar family. Note that all clusters in $\mathcal{L}_i$ are mutually disjoint. Therefore, every crossing $(e, e')_p$ of the drawing $\varphi^*$ may contribute to the sets $\chi(C)$ of at most four clusters of $\mathcal{L}_i$: at most two clusters $C$ with $e \in E(C) \cup \delta_G(C)$, and at most two clusters $C'$ with $e \in E(C') \cup \delta_G(C')$. Therefore:

$$\sum_{C \in \mathcal{L}_i} \mathbf{E}\left[\mathsf{OPT}_{\mathsf{cnwrs}}(I_C)\right] \leq \sum_{C \in \mathcal{L}_i} O(\beta^2 \cdot (|\chi(C)| + |E(C)|)) \leq O(\beta^2 \cdot (\mathsf{OPT}_{\mathsf{cnwrs}}(I) + |E(G)|)).$$

Summing this up over all $1 \leq i \leq \mathsf{dep}(\mathcal{L})$, we get that $\mathbf{E}\left[\sum_{I' \in \mathcal{I}} \mathsf{OPT}_{\mathsf{cnwrs}}(I')\right] \leq O(\mathsf{dep}(\mathcal{L}) \cdot \beta^2 \cdot (\mathsf{OPT}_{\mathsf{cnwrs}}(I) + |E(G)|))$. In order to complete the proof of Lemma 5.6, it is now enough to prove Claim E.1, which we do next.

In the remainder of this proof, we fix a cluster $C \in \mathcal{L}$. Recall that there is a distribution $\mathcal{D}'(C)$ over the set $\Lambda'(C)$ of external $C$-routers, such that for every edge $e$ of $E(G \setminus C)$,

$$\mathbf{E}_{\mathcal{Q}'(C) \sim \mathcal{D}'(C)}\left[\mathrm{cong}_G(\mathcal{Q}'(C), e)\right] \leq \beta.$$

We sample an external $C$-router $\mathcal{Q}'(C)$ from the distribution $\mathcal{D}'(C)$. We view the paths of $\mathcal{Q}'(C)$ as being directed towards vertex $u'(C)$, that is the center of the router. We apply the algorithm from Lemma 4.7 to obtain a collection $\tilde{\mathcal{Q}}'(C)$ of non-transversal paths, routing the edges of $\delta_G(C)$ to $u'(C)$, such that, for every edge $e \in E(G)$, $\mathrm{cong}_G(\tilde{\mathcal{Q}}'(C), e) \leq \mathrm{cong}_G(\mathcal{Q}'(C), e)$. In particular, $\tilde{\mathcal{Q}}'(C)$ is also an external $C$-router with center vertex $u'(C)$.

In order to simplify the notation, we denote $\tilde{\mathcal{Q}}'(C)$ by $\mathcal{Q}'$, and $\tilde{\mathcal{Q}}(C)$ by $\mathcal{Q}$. To summarize, $\mathcal{Q}'$ is an external $C$-router, and we are guaranteed that, for every edge $e \in E(G \setminus C)$, $\mathbf{E}\left[\mathrm{cong}_G(\mathcal{Q}', e)\right] \leq \beta$. Set $\mathcal{Q}$ of paths is an internal $C$-router. If $C \in \mathcal{L}^{\mathrm{bad}}$, then for every edge $e \in E(C)$, $\mathbf{E}\left[\mathrm{cong}(\mathcal{Q}, e)\right] \leq \beta$, while, if $C \in \mathcal{L}^{\mathrm{light}}$ then, for every edge $e \in E(C)$, $\mathbf{E}\left[(\mathrm{cong}_G(\mathcal{Q}, e))^2\right] \leq \beta$. We denote the center vertex of router $\mathcal{Q}$ by $u$, and the center vertex of router $\mathcal{Q}'$ by $u'$.

We denote by $\mathcal{W} = \{C_1, \ldots, C_r\}$ the set of child-clusters of $C$. We partition $\mathcal{W}$ into two subsets: $\mathcal{W}^{\mathrm{bad}} = \mathcal{W} \cap \mathcal{L}^{\mathrm{bad}}$, and $\mathcal{W}^{\mathrm{light}} = \mathcal{W} \cap \mathcal{L}^{\mathrm{light}}$. For convenience, for all $1 \leq i \leq r$, we denote the internal $C_i$-router $\tilde{\mathcal{Q}}(C_i)$ that we have constructed by $\mathcal{Q}_i$, and its center vertex by $u_i$. Recall that, if $C_i \in \mathcal{L}^{\mathrm{bad}}$, then for every edge $e \in E(C_i)$, $\mathbf{E}\left[\mathrm{cong}(\mathcal{Q}_i, e)\right] \leq \beta$, while, if $C_i \in \mathcal{L}^{\mathrm{light}}$ then, for every edge $e \in E(C_i)$, $\mathbf{E}\left[(\mathrm{cong}_G(\mathcal{Q}_i, e))^2\right] \leq \beta$.

It will be convenient for us to also define another cluster $C_0$ to be the connected component of $G \setminus C$ containing the vertex $u'$ – the center vertex of the external $C$-router $\mathcal{Q}'$. It is immediate to verify that the set $\mathcal{Q}'$ of paths is an internal $C_0$-router, and for consistency we denote it by $\mathcal{Q}_0$. We also denote the center vertex of this router by $u_0 = u'$. Recall that each one of the sets $\mathcal{Q}_0, \mathcal{Q}_1, \ldots, \mathcal{Q}_r$ of paths is non-transversal with respect to $\Sigma$. For convenience, we denote $R = C \setminus (\bigcup_{i=1}^{r} C_i)$, and $\mathcal{Q}^* = \bigcup_{i=0}^{r} \mathcal{Q}_i$.

Lastly, it will be convenient for us to assume that no edge of $G$ has its endpoints in two distinct clusters of $C_0, C_1, \ldots, C_r$. For each such edge $e$, we subdivide the edge with a new vertex $v_e$, that is added to graph $R$ as an isolated vertex. Note that, for all $0 \leq i < j \leq r$, the only vertices that may be shared by paths in $\mathcal{Q}_i$ and paths in $\mathcal{Q}_j$ are vertices of $V(R)$, which must serve as endpoints of those paths.

The remainder of the proof of Claim E.1 consists of three steps. In the first step, we will define a new graph $H$ by slightly modifying graph $G$, and compute its drawing $\psi$. In the second step, we use graph $H$ and its drawing $\psi$ in order to construct an initial drawing $\varphi'$ of graph $G_C$ (associated with instance $I_C = (G_C, \Sigma_C) \in \mathcal{I}$ of MCNwRS). This drawing, however, may not obey all rotations in $\Sigma_C$. In the third and the last step, we modify drawing $\varphi'$ to obtain the final drawing $\varphi''$ of $G_C$, that is a valid solution to instance $I_C$ of MCNwRS. We now describe each of the three steps in turn.

### E.2.1 Step 1: Graph $H$

We assign, to every edge $e \in E(G)$, an integer $n_e \geq 0$, as follows. For every edge $e \in E(R) \cup \delta_G(R)$, we let $n_e = 1$. For every other edge $e \in E(G) \setminus E(R)$, we let $n_e = \mathrm{cong}_G(\mathcal{Q}^*, e)$.

In order to construct graph $H$, we start with $V(H) = V(G)$. For every edge $e = (u, v) \in E(G)$ with $n_e > 0$, we add a set $J(e)$ of $n_e$ parallel edges $(u, v)$ to graph $H$, and we call the edges of $J(e)$ *copies of edge* $e$. This completes the definition of the graph $H$. Note that graph $H$ is a random graph, as values $\{n_e\}_{e \in E(G)}$ are random variables.

Consider the optimal solution $\varphi^*$ to instance $I$ of MCNwRS. We use $\varphi^*$ in order to define a drawing $\psi$ of the graph $H$, in a natural way: for every vertex $v \in V(H)$, its image in $\psi$ remains the same as in $\varphi^*$. For every edge $e \in E(G)$ with $n(e) \geq 1$, we draw the edges of $J(e)$ in parallel to the image of $e$ in $\varphi^*$, immediately next to it, so that their images do not cross. Consider the resulting drawing $\psi$ of graph $H$, and let $(e'_1, e'_2)_p$ be a crossing of $\psi$. Assume that $e'_1 \in J(e_1)$ and $e'_2 \in J(e_2)$. Then the images of the edges $e_1, e_2$ cross in $\varphi^*$, at some point $p'$ that is very close to point $p$. We say that crossing $(e_1, e_2)_{p'}$ of $\varphi^*$ is *responsible* for the crossing $(e'_1, e'_2)_p$ of $\psi$.

Next, we classify the crossings of the drawing $\psi$ into three types, and we bound the expected number of crossings of some of the types. Consider some crossing $(e'_1, e'_2)_p$ of $\psi$, and let $(e_1, e_2)_{p'}$ be the crossing of $\varphi^*$ that is responsible for $(e'_1, e'_2)_p$. We say that crossing $(e'_1, e'_2)_p$ is a *type-1 crossing* if there is some cluster $C_i \in \mathcal{W}^{\mathrm{light}}$, with $e_1, e_2 \in E(C_i)$. We say that it is a *type-2 crossing* if there is an index $0 \leq i \leq r$ with $e_1, e_2 \in E(C_i)$, and either $i = 0$ or $C_i \in \mathcal{W}^{\mathrm{bad}}$ holds. We say that $(e'_1, e'_2)_p$ is a *type-3 crossing* otherwise.

We now bound the expected number of type-1 and type-3 crossings. We do not bound the number of type-2 crossings, as all such crossings will eventually be eliminated.

**Type-1 crossings.** Consider some cluster $C_i \in \mathcal{W}^{\mathrm{light}}$, and some crossing $(e_1, e_2)_{p'}$ of $\varphi^*$, such that $e_1, e_2 \in E(C_i)$. Notice that crossing $(e_1, e_2)_{p'}$ lies in $\chi(C_i)$. The number of type-1 crossings in $\varphi$ that this crossing is responsible for is $n_{e_1} \cdot n_{e_2} \leq n_{e_1}^2 + n_{e_2}^2$. Observe that, for an edge $e \in E(C_i)$, $n_e = \mathrm{cong}_G(\mathcal{Q}_i, e)$, and so $\mathbf{E}\left[n_e^2\right] = \mathbf{E}\left[(\mathrm{cong}_G(\mathcal{Q}_i, e))^2\right] \leq \beta$. We conclude that the total expected number of type-1 crossings in $\psi$ is bounded by:

$$\sum_{C_i \in \mathcal{W}^{\mathrm{light}}} \sum_{(e_1,e_2)_{p'} \in \chi(C_i)} \mathbf{E}\left[n_{e_1}^2 + n_{e_2}^2\right] \leq \sum_{C_i \in \mathcal{W}^{\mathrm{light}}} O(\beta \cdot |\chi(C_i)|) \leq O(\beta \cdot |\chi(C)|).$$

**Type-3 crossings.** Consider some crossing $(e_1, e_2)_{p'}$ of $\varphi^*$. If edges $e_1, e_2$ lie in the same cluster $C_i$, for $0 \leq i \leq r$, then this crossing may not be responsible for any type-3 crossings in $\psi$. Assume now that this is not the case. Then the total number of type-3 crossings that $(e_1, e_2)_{p'}$ is responsible for is at most $n_{e_1} \cdot n_{e_2}$. Furthermore $n_{e_1}, n_{e_2}$ are independent random variables, each of which has expectation at most $\beta$. Therefore, the expected number of type-3 crossings for which crossing $(e_1, e_2)_{p'}$ is responsible is at most $\beta^2$. Note that, at least one of the edges $e_1, e_2$ must lie in $E(C) \cup \delta_G(C)$, so crossing $(e_1, e_2)_{p'}$ must lie in $\chi(C)$. We conclude that the total expected number of type-3 crossings

in $\varphi$ is at most $|\chi(C)| \cdot \beta^2$.

Consider now an index $0 \leq i \leq r$, and let $E_i = \delta_G(C_i)$. From our definition, for every edge $e \in E_i$, $n_e = 1$. Recall that we have defined a set $\mathcal{Q}_i = \{Q(e) \mid e \in E_i\}$ of paths in graph $G$, routing all edges of $E_i$ to the vertex $u_i \in V(C_i)$. The paths in $\mathcal{Q}_i$ are non-transversal with respect to $\Sigma$, and all their inner vertices are contained in $C_i$. We will now define a corresponding set $\hat{\mathcal{Q}}_i = \left\{\hat{Q}(e) \mid e \in E_i\right\}$ of paths in graph $H$, routing the edges of $E_i$ to the same vertex $u_i$, such that the paths in $\hat{\mathcal{Q}}_i$ are edge-disjoint. In order to do so, we assign, to every path $Q(e) \in \mathcal{Q}_i$, for every edge $e' \in E(Q(e)) \setminus \{e\}$, a copy of the edge $e'$ from $J(e')$, such that every copy of edge $e'$ is assigned to a distinct path. We will then obtain path $\hat{Q}(e)$ from path $Q(e)$ by replacing every edge $e' \in E(Q(e)) \setminus \{e\}$ with its copy that was assigned to $Q(e)$.

Consider any edge $e' \in E(C_i)$. If $e'$ is not incident to the vertex $u_i$, then we assign each copy of $e'$ to a distinct path in $\mathcal{Q}_i$ that contains $e'$ arbitrarily. Assume now that edge $e'$ is incident to vertex $u_i$, and that $C_i \notin \mathcal{W}^{\text{light}}$. In this case, as before, we assign each copy of $e'$ to a distinct path in $\mathcal{Q}_i$ that contains $e'$ arbitrarily. It now remains to consider the case where $C_i \in \mathcal{W}^{\text{light}}$, and edges that are incident to vertex $u_i$. We need to assign copies of such edges to paths in $\mathcal{Q}_i$ more carefully. The goal of this more careful assignment is to achieve the following property: if we denote $E_i = \{e_1, \ldots, e_{h_i}\}$, where the edges are indexed according to the ordering $\mathcal{O}(C_i)$, and, for each such edge $e_j$, we denote by $e'_j$ be the last edge on path $\hat{Q}(e_j)$ (that we are trying to construct), then the images of the edges $\left\{e'_1, \ldots, e'_{h_i}\right\}$ enter the image of $u_i$ in the drawing $\psi$ of graph $H$ in this circular order. We now describe the procedure for assigning copies of edges of $\delta_G(u_i)$ to paths in $\mathcal{Q}_i$.

We start by revisiting the definition of the ordering $\mathcal{O}(C_i)$ of the edges of $E_i = \delta_G(C_i)$, which is an ordering that is guided by the set $\mathcal{Q}_i$ of paths and the rotation system $\Sigma$. Denote $\delta_G(u_i) = \left\{a^i_1, \ldots, a^i_{z_i}\right\}$, where the edges are indexed according to their circular ordering $\mathcal{O}_{u_i} \in \Sigma$. We assume w.l.o.g. that the orientation of this ordering in the drawing $\varphi^*$ of $G$ is negative (or clock-wise). For all $1 \leq j \leq z_i$, let $\mathcal{Q}^j_i \subseteq \mathcal{Q}_i$ the set of paths in $\mathcal{Q}_i$ whose last edge is $a^i_j$. We defined an ordering $\hat{\mathcal{O}}_i$ of the paths in $\mathcal{Q}_i$, where the paths in sets $\mathcal{Q}^1_i, \ldots, \mathcal{Q}^{z_i}_i$ appear in the natural order of their indices, and for all $1 \leq j \leq z_i$, the ordering of the paths in set $\mathcal{Q}^j_i$ is arbitrary. We denote $\mathcal{Q}^j_i = \left\{Q(e^{i,j}_1), Q(e^{i,j}_2), \ldots, Q(e^{i,j}_{m_{i,j}})\right\}$, and assume that these paths are indexed according to the ordering that we have chosen when defining $\hat{\mathcal{O}}_i$.

Ordering $\hat{\mathcal{O}}_i$ of the paths in $\mathcal{Q}_i$ was then used to define the ordering $\mathcal{O}(C_i)$ of the edges in $E_i$: we obtain the ordering $\mathcal{O}(C_i)$ from $\hat{\mathcal{O}}_i$ by replacing, for every path $Q(e^{i,j}_\ell) \in \mathcal{Q}_i$, the path $Q(e^{i,j}_\ell)$ in $\hat{\mathcal{O}}$ with the edge $e^{i,j}_\ell$ (the first edge of $Q(e^{i,j}_\ell)$).

Consider now some edge $a^i_j \in \delta_G(u_i)$. Recall that we have defined a set $J(a^i_j)$ of $n_{a^i_j} = m_{i,j}$ copies of the edge $a^i_j$. We denote these copies by $\hat{a}^{i,j}_1, \ldots, \hat{a}^{i,j}_{m_{i,j}}$, where the copies are indexed according to the order in which their images enter the image of vertex $u_i$ in the drawing $\psi$ of $H$, in the clock-wise direction. For all $1 \leq \ell \leq m_{i,j}$, we assign the copy $\hat{a}^{i,j}_\ell$ of edge $a^i_j$ to the path $Q(e^{i,j}_\ell)$. This completes the assignment of the copies of the edges incident to vertex $u_i$ to the paths of $\mathcal{Q}_i$.

We now define a set $\hat{\mathcal{Q}}_i = \left\{\hat{Q}(e) \mid e \in E_i\right\}$ of paths in graph $H$, routing the edges of $E_i$ to vertex $u_i$, as follows. For every edge $e \in E_i$, path $\hat{Q}(e)$ is obtained from the path $Q(e) \in \mathcal{Q}_i$ by replacing every edge $e' \in E(Q(e)) \setminus \{e\}$ with the copy of $e'$ that was assigned to path $Q(e)$. The following observation summarizes the properties of the path set $\hat{\mathcal{Q}}_i$, that follow immediately from our construction.

**Observation E.2** *Paths in set $\hat{\mathcal{Q}}_i = \left\{\hat{Q}(e) \mid e \in E_i\right\}$ route the edges of $E_i$ to vertex $u_i$ in graph $H$, and all inner vertices on all paths in $\hat{\mathcal{Q}}_i$ lie in $V(C_i)$. Moreover, the paths of $\hat{\mathcal{Q}}_i$ are edge-disjoint. Additionally, if $C_i \in \mathcal{W}^{\text{light}}$, then the following holds. Denote $E_i = \{e_1, \ldots, e_{h_i}\}$, where the edges are*

*indexed according to the ordering $\mathcal{O}(C_i)$. For each such edge $e_j$, let $e'_j$ be the last edge on path $\hat{Q}(e_j)$. Then the images of the edges $e'_1, \ldots, e'_{h_i}$ enter the image of $u_i$ in the drawing $\psi$ of graph $H$ in the circular order of their indices.*

### E.2.2 Step 2: Initial Drawing of Graph $G_C$

In this step we exploit the drawing $\psi$ of graph $H$ that we have constructed in the first step, in order to construct an initial drawing $\varphi'$ of graph $G_C$.

In order to construct the drawing $\varphi'$ of graph $G_C$, we start with the drawing $\psi$ of graph $H$, and then gradually modify it. We place the image of the vertex $v^*$ in $\varphi'$ at point $\psi(u_0)$, and, for all $1 \leq i \leq r$, we place the image of the vertex $v_{C_i}$ at point $\psi(u_i)$. Intuitively, the images of the vertices and the edges of $R$ will remain unchanged. For all $0 \leq i \leq r$, we will utilize the images of the paths of $\hat{\mathcal{Q}}_i$ in $\psi$ in order to draw the edges of $E_i$. There are two issues with this approach. First, we did not bound the expected number of type-2 crossings in $\psi$, so there may be many crossings between pairs of edges lying on paths of $\mathcal{Q}_i$, where $i = 0$, or $C_i \in \mathcal{W}^{\mathrm{bad}}$. We take care of this issue by performing a type-2 uncrossing for each such path set $\hat{\mathcal{Q}}_i$, to obtain the drawings of the edges in $E_i$. The second problem then remains for indices $i$ with $C_i \in \mathcal{W}^{\mathrm{light}}$. Since several paths from $\hat{\mathcal{Q}}_i$ may share the same vertex, there could be points that lie on images of multiple paths of $\hat{\mathcal{Q}}_i$. We take care of this latter issue by employing a nudging procedure. We now describe each of these two operations in turn.

**Uncrossing.** We consider indices $i$ for which either $i = 0$ or $C_i \in \mathcal{W}^{\mathrm{bad}}$ holds one by one. Consider any such index $i$. We view every path of $\hat{\mathcal{Q}}_i$ as being directed towards the vertex $u_i$. We use the algorithm from Theorem 4.37 in order to compute a type-2 uncrossing, that produces, for every edge $e \in E_i$, a directed curve $\gamma(e)$, that connects the image of the endpoint of $e$ that lies in $R$ to the image of $u_i$ in $\psi$. Recall that we are guaranteed that the curves in the resulting set $\Gamma_i = \{\gamma(e) \mid e \in E_i\}$ do not cross each other, and each such curve is aligned with the drawing of graph $\bigcup_{\hat{Q}(e) \in \hat{\mathcal{Q}}_i} \hat{Q}(e)$ induced by $\psi$.

Let $\psi'$ be a drawing obtained from $\psi$ as follows. For every index $i$ with $i = 0$ or $C_i \in \mathcal{W}^{\mathrm{bad}}$, we delete the images of all vertices of $V(C_i)$ and all edges with at least one endpoint in $V(C_i)$ from the drawing. If $i = 0$, then we place the image of vertex $v^*$ at point $\psi(u_0)$, and otherwise we place the image of vertex $v_{C_i}$ at point $\psi(u_i)$. For every edge $e \in E_i$, we then let $\gamma(e) \in \Gamma_i$ be the image of the edge $e$.

Note that this uncrossing step has eliminated all type-2 crossings, and every crossing in the resulting drawing $\psi'$ corresponds to a distinct type-1 or type-3 crossing of $\psi$. Therefore, the expected number of crossings of $\psi'$ is bounded by $O(\beta^2 \cdot |\chi(C)|)$. We call all crossings that are currently present in drawing $\psi'$ *primary crossings*.

**Nudging.** We now consider the indices $i$ with $C_i \in \mathcal{W}^{\mathrm{light}}$ one by one. When such an index $i$ is considered, we delete the images of all vertices of $V(C_i)$ and all edges with at least one endpoint in $V(C_i)$ from the current drawing $\psi'$. We then place the image of vertex $v_{C_i}$ at point $\psi(u_i)$. For every edge $e \in E_i$, we initially let $\gamma(e)$ be the image of the path $\hat{Q}(e) \in \hat{\mathcal{Q}}_i$ in $\psi$, and we add $\gamma(e)$ to the current drawing as the image of the edge $e$. Note that the curves in $\{\gamma(e) \mid e \in E_i\}$ enter the image of $v_{C_i}$ in the order $\mathcal{O}(C_i)$ of their corresponding edges in $E_i$, from Observation E.2. However, it is possible that, for some vertex $x \in V(C_i)$, point $\psi(x)$ lies on more than two curves from $\{\gamma(e) \mid e \in E_i\}$.

We process each vertex $x \in V(C_i) \setminus \{u_i\}$ one by one. Consider any such vertex $x$, and let $\mathcal{Q}^x \subseteq \hat{\mathcal{Q}}_i$ be the set of all paths containing vertex $x$. Note that $x$ must be an inner vertex on each such path. For convenience, we denote $\mathcal{Q}^x = \{Q(e_1), \ldots, Q(e_z)\}$. Consider the tiny $x$-disc $D = D_\psi(x)$. For all $1 \leq j \leq z$, denote by $s_j$ and $t_j$ the two points on the curve $\gamma(e_j)$ that lie on the boundary of disc $D$. We use the algorithm from Claim 4.34 to compute a collection $\{\sigma_1, \ldots, \sigma_z\}$ of curves, such that, for

all $1 \leq j \leq z$, curve $\sigma_j$ connects $s_j$ to $t_j$, and the interior of the curve is contained in the interior of $D$. Recall that every pair of resulting curves crosses at most once, and every point in the interior of $D$ may be contained in at most two curves. Consider now a pair $\sigma_\ell, \sigma_{\ell'}$ of curves, and assume that these two curves cross. Recall that, from Claim 4.34, this may only happen if the two pairs $(s_\ell, t_\ell)$, $(s_{\ell'}, t_{\ell'})$ of points cross. Denote by $e_1, e_2$ the two edges that lie on path $\hat{Q}(e_\ell)$ immediately before and immediately after vertex $x$, and denote by $e'_1, e'_2$ the two edges that lie on path $\hat{Q}(e_{\ell'})$ immediately before and immediately after vertex $x$. We assume that edges $e_1, e_2$ are copies of edges $\hat{e}_1, \hat{e}_2$ of $G$, and similarly, $e'_1, e'_2$ are copies of edges $\hat{e}'_1, \hat{e}'_2$ of $G$, respectively. Assume first that there are four distinct edges in set $\{\hat{e}_1, \hat{e}'_1, \hat{e}_2, \hat{e}'_2\}$. From the fact that the two pairs $(s_\ell, t_\ell)$, $(s_{\ell'}, t_{\ell'})$ of points cross, we get that these four edges must appear in the rotation $\mathcal{O}_x \in \Sigma$ in the order $(\hat{e}_1, \hat{e}'_1, \hat{e}_2, \hat{e}'_2)$. Since the paths of $\mathcal{Q}_i$ are non-transversal with respect to $\Sigma$, this is impossible. Therefore, we conclude that paths $Q(e_\ell), Q(e_{\ell'})$ must share an edge that is incident to $x$. If $e^*$ is an edge incident to $x$ that the two paths share, then we say that $e^*$ is *responsible for the crossing between $\sigma_\ell$ and $\sigma_{\ell'}$*.

For all $1 \leq j \leq z$, we modify the curve $\gamma(e_j)$, by replacing the segment of the curve that is contained in disc $D$ with $\sigma_j$. Once every vertex $x \in V(C_i) \setminus \{u_i\}$ is processed, we obtain the final set $\Gamma'_i = \{\gamma'(e) \mid e \in E_i\}$ of curves, which are now guaranteed to be in general position. For every edge $e \in E_i$, we modify the image of edge $e$ in the current drawing, by replacing it with the new curve $\gamma'(e)$. As before, curves of $\Gamma'_i$ enter the image of $v_{C_i}$ according to the ordering $\mathcal{O}(C_i)$.

Once every index $i$ with $C_i \in \mathcal{W}^{\text{light}}$ is processed, we obtain a valid drawing $\varphi'$ of the graph $G_C$. In this drawing, for every index $i$ with $C_i \in \mathcal{W}^{\text{light}}$, the images of the edges in set $E_i = \delta_{G_C}(v_{C_i})$ enter the image of vertex $v_{C_i}$ according to the ordering $\mathcal{O}_{v_{C_i}} \in \Sigma_C$, which is precisely $\mathcal{O}(C_i)$. However, for indices $i$ with $C_i \in \mathcal{W}^{\text{bad}}$, this property may not hold, and the edges incident to $v^*$ may enter the image of $v^*$ in an arbitrary order. For every other vertex $v$ of $G_C$, the rotation $\mathcal{O}_v \in \Sigma_C$ is identical to the rotation $\mathcal{O}_v \in \Sigma$, and is obeyed by the current drawing $\varphi'$. We modify the drawing $\varphi'$ to obtain a drawing that is consistent with the rotation system $\Sigma_C$ in the third step. Notice however that the nudging operation may have introduced some new crossings. Each such new crossing must be contained in a disc $D_\psi(x)$, for some vertex $x$ that must lie in some cluster $C_i \in \mathcal{W}^{\text{light}}$. We call all such new crossings *secondary crossings*. We now bound the total number of secondary crossings.

Fix an index $i$ with $C_i \in \mathcal{W}^{\text{light}}$, and consider some vertex $x \in V(C_i)$. Every secondary crossing that is contained in $\mathcal{D}_\psi(x)$ is a crossing between a pair $\sigma_\ell, \sigma_{\ell'}$ of curves that we have defined when processing vertex $x$, and each such crossing was charged to an edge of $G$ that is incident to $x$, whose copies lie on the corresponding two paths $\hat{Q}(e_\ell), \hat{Q}(e_{\ell'}) \in \hat{\mathcal{Q}}_i$. If $e$ is an edge that is incident to $x$ in $G$, then there are at most $(\text{cong}_G(\mathcal{Q}_i, e))^2$ pairs of paths in $\mathcal{Q}_i$ that contain $e$, and each such pair of paths may give rise to a single secondary crossing in $D_\psi(x)$ that is charged to edge $e$. Therefore, the total expected number of secondary crossings that are contained in discs $D_\psi(x)$ for vertices $x \in V(C_i)$ is bounded by:

$$\sum_{e \in E(C_i)} O(\mathbf{E}\left[(\text{cong}_G(\mathcal{Q}_i, e))^2\right]) \leq O(\beta \cdot |E(C_i)|),$$

since $C_i \in \mathcal{W}^{\text{light}}$.

We conclude that the total expected number of secondary crossings in $\varphi'$ is at most $\sum_{C_i \in \mathcal{W}^{\text{light}}} O(\beta \cdot |E(C_i)|) \leq O(\beta \cdot |E(C)|)$, and the total number of all crossings in $\varphi'$ is at most $O(\beta^2 \cdot (|\chi(C)| + |E(C)|))$.

### E.2.3 Step 3: the Final Drawing

So far we have obtained a drawing $\varphi'$ of graph $G_C$, that obeys the rotations $\mathcal{O}_v \in \Sigma_C$ for all vertices $v \in V(G_C)$, except possibly for vertex $v^*$, and vertices $v_{C_i}$, for $C_i \in \mathcal{W}^{\text{bad}}$. We now fix this drawing

to obtain a final drawing $\varphi''$ of $G_C$ that obeys the rotation system $\Sigma_C$.

Let $U = \{v^*\} \cup \left\{v_{C_i} \mid C_i \in \mathcal{W}^{\mathrm{bad}}\right\}$. For each vertex $x \in U$, we denote by $\hat{\mathcal{O}}(x) = \mathcal{O}_x \in \Sigma_C$ the rotation associated with vertex $x$ in the rotation system $\Sigma_C$, and by $\hat{\mathcal{O}}'(x)$ the circular order in which the edges of $\delta_{G_C}(x)$ enter the image of $x$ in the current drawing $\varphi'$. Note that, for a vertex $x = v_{C_i}$, where $C_i \in \mathcal{W}^{\mathrm{bad}}$, if we denote by $\Sigma(C_i)$ the rotation system induced by $\Sigma$ for cluster $C$, then the following must hold:

$$\mathsf{dist}(\hat{\mathcal{O}}(x), \hat{\mathcal{O}}'(x)) \leq |\delta_{G_C}(x)|^2 = |\delta_G(C_i)|^2 \leq \beta \cdot (\mathsf{OPT}_{\mathsf{cnwrs}}(C_i, \Sigma(C_i)) + |E(C_i)|) \leq \beta(|\chi(C_i)| + |E(C_i)|).$$

(we have used the fact that cluster $C_i$ is a $\beta$-bad cluster). We use the following claim, whose proof appears in Section E.3, in order to bound $\mathsf{dist}(\hat{\mathcal{O}}(v^*), \hat{\mathcal{O}}'(v^*))$.

**Claim E.3** $\mathbf{E}\left[\mathsf{dist}(\hat{\mathcal{O}}(v^*), \hat{\mathcal{O}}'(v^*))\right] \leq \beta^2 \cdot (|\chi(C)| + |E(C)|)$.

In order to compute the final drawing $\varphi''$ of graph $G_C$, we process the vertices $x \in U$ one by one. When vertex $x$ is processed, we apply the algorithm from Corollary 4.32 to it. The algorithm modifies the current drawing of graph $G_C$ within the tiny $x$-disc $D(x)$ to ensure that the images of the edges of $\delta_{G_C}(x)$ enter the image of $x$ in the circular order $\hat{\mathcal{O}}(x)$. This modification increases the number of crossings in the current drawing by at most $2 \cdot \mathsf{dist}(\hat{\mathcal{O}}(x), \hat{\mathcal{O}}'(x))$. Once every vertex of $U$ is processed, we obtain the final drawing $\varphi''$ of graph $G_C$, which obeys the rotation system $\Sigma_C$. Moreover, $\mathsf{cr}(\varphi'') \leq \mathsf{cr}(\varphi') + \sum_{x \in U} 2 \cdot \mathsf{dist}(\hat{\mathcal{O}}(x), \hat{\mathcal{O}}'(x))$. Recall that $\mathbf{E}\left[\mathsf{cr}(\varphi')\right] \leq O(\beta^2) \cdot (|\chi(C)| + |E(C)|)$, and that, for every vertex $x = v_{C_i}$ with $C_i \in \mathcal{W}^{\mathrm{bad}}$, $\mathsf{dist}(\hat{\mathcal{O}}(x), \hat{\mathcal{O}}'(x)) \leq \beta(|\chi(C_i)| + |E(C_i)|)$. Combining this with Claim E.3, we get that:

$$\mathbf{E}\left[\mathsf{cr}(\varphi'')\right] \leq O(\beta^2) \cdot (|\chi(C)| + |E(C)|) + \sum_{C_i \in \mathcal{W}^{\mathrm{bad}}} O(\beta) \cdot (|\chi(C_i)| + |E(C_i)|) \leq O(\beta^2 \cdot (|\chi(C)| + |E(C)|)).$$

This completes the proof of Claim E.1.

## E.3 Proof of Claim E.3

Clearly, $\mathsf{dist}(\hat{\mathcal{O}}(v^*), \hat{\mathcal{O}}'(v^*)) \leq |\delta_{G_C}(v^*)|^2 = |\delta_G(C)|^2$. If $C \in \mathcal{L}^{\mathrm{bad}}$, then cluster $C$ is $\beta$-bad, and so $\mathsf{dist}(\hat{\mathcal{O}}(v^*), \hat{\mathcal{O}}'(v^*)) \leq |\delta_G(C)|^2 \leq \beta \cdot (|\chi(C)| + |E(C)|)$ from the definition of $\beta$-bad clusters. Therefore, we assume from now on that $C \in \mathcal{L}^{\mathrm{light}}$.

For convenience of notation, we denote $v^*$ by $u'$, and we denote $\hat{\mathcal{O}}(v^*)$ and $\hat{\mathcal{O}}'(v^*)$ by $\mathcal{O}$ and $\mathcal{O}'$, respectively. We also denote $E' = \delta_G(C) = \delta_{G_C}(u')$. In order to prove the claim, we will construct a collection $\Gamma = \{\gamma(e) \mid e \in E'\}$ of curves in the plane, all of which connect two points $p$ and $q$. We will ensure that the order in which the curves enter the point $p$ is precisely $\mathcal{O}'$, and the order in which they enter the point $q$ is $\mathcal{O}$. We will also ensure that the curves of $\Gamma$ are in general position. By showing that the expected number of crossings between the curves in $\Gamma$ is relatively small, we will obtain the desired bound on $\mathbf{E}\left[\mathsf{dist}(\mathcal{O}, \mathcal{O}')\right]$.

In order to construct the curves in $\Gamma$, we consider again the intance $I$ of $\mathsf{MCNwRS}$ and its optimal solution $\varphi^*$. Recall that we have computed an internal $C$-router $\mathcal{Q} = \{Q(e) \mid e \in E'\}$, where for each edge $e \in E'$, path $Q(e)$ originates with edge $e$, terminates at vertex $u$, and all its inner vertices lie in $C$. The paths in $\mathcal{Q}$ are non-transversal with respect to $\Sigma$, and, for every edge $e' \in E(C)$, $\mathbf{E}\left[(\mathrm{cong}_G(\mathcal{Q}, e'))^2\right] \leq \beta$. We have also constructed an external $C$-router $\mathcal{Q}' = \{Q'(e) \mid e \in E'\}$, where for each edge $e \in E'$, path $Q(e)$ originates with edge $e$, terminates at vertex $u'$, and all its inner vertices are disjoint from $C$. The paths in $\mathcal{Q}'$ are non-transversal with respect to $\Sigma$, and, for every edge $e' \in E(G \setminus C)$, $\mathbf{E}\left[\mathrm{cong}_G(\mathcal{Q}', e')\right] \leq \beta$. We note that path set $\mathcal{Q}'$ is exactly the same as path set

$\mathcal{Q}_0$ – the internal router for $C_0$, that we used in the first step of the algorithm. Intuitively, we would like to let $p$ be the image of vertex $u'$ and $q$ the image of vertex $u$ in $\varphi^*$. For every edge $e \in E'$, we would like to use the concatenation of the images of paths $Q(e)$ and $Q'(e)$ in $\varphi^*$ in order to construct the curve $\gamma(e)$. This approach has several problems. First, the paths in sets $\mathcal{Q}$ and $\mathcal{Q}'$ may share edges and vertices, and so the resulting curves may not be in a general position. Second, there could be many crossings between edges lying on the paths of $\mathcal{Q}'$, which may lead to many crossings between curves of $\Gamma$. We take care of all these issues in the following three steps. In the first step, we take care of the congestion issue by constructing a graph $H'$ and its drawing $\psi'$. The construction is somewhat similar to the construction of graph $H$, in that we make several copies of some of the edges of $G$, in a way that allows us to define edge-disjoint paths in graph $H'$ replacing the path sets $\mathcal{Q}$ and $\mathcal{Q}'$. In the second step, we perform uncrossing of curves corresponding to the paths in $\mathcal{Q}'$, in order to eliminate some of the crossings. In the third step we perform nudging of curves corresponding to the paths in $\mathcal{Q}$. We now describe each of these steps in turn.

**Graph $H'$.** For each edge $e \in E'$, we set $n'_e = 1$. For an edge $e \in E(C)$, we set $n'_e = \mathrm{cong}_G(\mathcal{Q}, e)$, and for an edge $e \in E(G \setminus C)$, we set $n'_e = \mathrm{cong}_G(\mathcal{Q}', e)$. For every other edge $e$, we set $n'_e = 0$. Note that for each edge $e \in E(G \setminus C)$, $n'_e = n_e$ holds, where $n_e$ is the parameter that we have used in the construction of graph $H$ in Step 1 of the algorithm.

In order to construct graph $H'$, we start with $V(H') = V(G)$. For every edge $e = (x, y) \in E(G)$ with $n'_e \neq 0$, we add a new set $J'(e)$ of $n'_e$ parallel edges connecting $x$ to $y$ to graph $H'$, that we view as *copies of edge $e$*. As before, we use the drawing $\varphi^*$ of $G$ in order to compute a drawing $\psi'$ of graph $H'$. For every vertex $x \in V(H')$, we let its image in $\psi'$ be $\varphi^*(x)$. For every edge $e \in E(G)$ with $n'_e \neq 0$, we draw the edges of $J'(e)$ in parallel to the image of $e$ in $\varphi^*$. Recall that for each edge $e \in E(G \setminus C)$, $n'_e = n_e$ holds, and so set $J'(e)$ of copies of $e$ can be thought of as being identical to the set $J(e)$ of copies of $e$ that we have constructed for graph $H$. We ensure that the specific drawing of the edges of $J'(e)$ in $\psi'$ is identical to the drawing of these edges in $\psi$.

We now bound the expected number of crossings in the resulting drawing $\psi'$ of graph $H'$. Consider any such crossing $(e'_1, e'_2)_{p'}$, and assume that $e'_1 \in J'(e_1)$, $e'_2 \in J'(e_2)$ holds for some edges $e_1, e_2 \in E(G)$. Then there must be some crossing $(e_1, e_2)_p$ in the drawing $\varphi^*$ of $G$, with point $p$ lying very close to point $p'$. We say that crossing $(e_1, e_2)_p$ of $\varphi^*$ is *responsible* for the crossing $(e'_1, e'_2)_{p'}$ of $\psi'$. It is immediate to verify that every crossing $(e_1, e_2)_p$ of $\varphi^*$ may be responsible for at most $n'_{e_1} \cdot n'_{e_2}$ crossings of $\psi'$.

We classify the crossings of $\psi'$ into three types. Let $(e'_1, e'_2)_{p'}$ be a crossing of $\psi'$, and let $(e_1, e_2)_p$ be the crossing of $\varphi^*$ responsible for it. We say that $(e'_1, e'_2)_{p'}$ is a *type-1 crossing* if $e_1, e_2 \in E(C)$. We say that it is a *type-2 crossing* if $e_1, e_2 \in E(G) \setminus (E(C) \cup \delta_G(C))$. Otherwise, we say that it is a type-3 crossing. We now bound the expected number of type-1 and type-3 crossings; type-2 crossings will eventually be eliminated.

In order to bound the expected number of type-1 crossings, consider any crossing $(e_1, e_2)_p$ of $\varphi^*$ with $e_1, e_2 \in E(C)$. Recall that this crossing may be responsible for at most $n'_{e_1} \cdot n'_{e_2} \leq (n'_{e_1})^2 + (n'_{e_2})^2$ type-1 crossings of $\psi'$, and moreover, $(e_1, e_2)_p \in \chi(C)$ must hold. Since we have assumed that $C \in \mathcal{L}^{\mathrm{light}}$, for every edge $e \in E(C)$, $\mathbf{E}\left[(n'_e)^2\right] = \mathbf{E}\left[(\mathrm{cong}_G(\mathcal{Q}, e))^2\right] \leq \beta$. Therefore, the expected number of type-1 crossings of $\psi'$ for which $(e_1, e_2)_p$ is responsible for is at most $\mathbf{E}\left[(n'_{e_1})^2 + (n'_{e_1})^2\right] \leq 2\beta$. Overall, the total expected number of type-1 crossings in $\psi'$ is then bounded by $O(\beta \cdot |\chi(C)|)$.

In order to bound the expected number of type-3 crossings, consider any crossing $(e_1, e_2)_p$ of $\varphi^*$, and assume that neither $e_1, e_2 \in E(C)$ nor $e_1, e_2 \in E(G) \setminus (E(C) \cup \delta_G(C))$ holds. Recall that this crossing may be repsonsible for at most $n'_{e_1} \cdot n'_{e_2}$ type-3 crossings of $\psi'$. Moreover, $n'_{e_1}, n'_{e_2}$ are independent random variables, and the expected value of each such variable is at most $\beta$. Therefore, the expected number of type-3 crossings of $\psi'$ for which crossing $(e_1, e_2)_p$ of $\varphi^*$ is responsible for is at most $\beta^2$.

Notice that one of the edges $e_1, e_2$ lies in $E(C) \cup \delta_G(C)$, and so $(e_1, e_2)_p \in \chi(C)$ must hold. Overall, the total expected number of type-3 crossings in $\psi'$ is then bounded by $O(\beta^2 \cdot |\chi(C)|)$.

We conclude that the total expected number of type-1 and type-3 crossings in $\psi'$ is at most $O(\beta^2 \cdot |\chi(C)|)$.

Next, we construct a set $\hat{\mathcal{Q}}' = \left\{\hat{Q}'(e) \mid e \in E'\right\}$ of edge-disjoint paths in graph $H'$, routing the edges of $E'$ to vertex $u'$. In order to do so, we assign, for every path $Q'(e) \in \mathcal{Q}'$, for every edge $e' \in E(Q'(e)) \setminus \{e\}$, a copy of edge $e'$ to path $Q'(e)$. Observe that path set $\mathcal{Q}'$ in graph $G$ was denoted by $\mathcal{Q}_0$ in Step 1 of the algorithm, and, as observed before, for every edge $e' \in E(G) \setminus (E(C) \cup \delta_G(C))$, $n_{e'} = n'_{e'}$ and so $J(e') = J'(e')$. For each such edge $e' \in E(G) \setminus (E(C) \cup \delta_G(C))$, we assign a distinct copy of $e' \in J'(e')$ to every path in $\mathcal{Q}'$ that contains $e'$. We ensure that this assignment is exactly the same as the assignment done in Step 1 of the algorithm. For each edge $e \in E'$, we then obtain a path $\hat{Q}'(e)$ in graph $H'$ from path $Q'(e)$ by replacing each edge $e' \in E(Q'(e)) \setminus \{e\}$ with the copy of $e'$ that was assigned to $e'$. We then denote $\hat{\mathcal{Q}}' = \left\{\hat{Q}'(e) \mid e \in E'\right\}$. From our construction, $\hat{\mathcal{Q}}'$ is a set of edge-disjoint paths in graph $H'$, routing the edges of $E'$ to vertex $u'$, and all inner vertices on the paths in $\hat{\mathcal{Q}}'$ are disjoint from $C$. Moreover, from our construction, $\hat{\mathcal{Q}}' = \hat{\mathcal{Q}}_0$ – the set of edge-disjoint paths in $H$ that we have constructed in Step 1 of the algorithm.

We also construct a set $\hat{\mathcal{Q}} = \left\{\hat{Q}(e) \mid e \in E'\right\}$ of edge-disjoint paths in graph $H'$, routing the edges of $E'$ to vertex $u$. In order to do so, we assign, for every path $Q(e) \in \mathcal{Q}$, for every edge $e' \in E(Q(e)) \setminus \{e\}$, a copy of edge $e'$ to path $Q(e)$. Consider now any edge $e' \in E(C)$. If edge $e'$ is not incident to vertex $u$, then we assign every copy of $e$ in $J'(e')$ to a distinct path of $\mathcal{Q}$ containing $e'$ arbitrarily. If edge $e'$ is incident to vertex $u$, then we perform the assignment more carefully, using the same procedure that we used for every cluster $C_i \in \mathcal{W}^{\text{light}}$ in order to assign, for each edge $e'$ incident to $u_i$, copies of $e'$ to paths in $\mathcal{Q}_i$; we do not repeat the description of the procedure there. For each edge $e \in E'$, we obtain a path $\hat{Q}(e)$ in graph $H'$ from path $Q(e)$ by replacing each edge $e' \in E(Q(e)) \setminus \{e\}$ with the copy of $e'$ that was assigned to $e'$. We then denote $\hat{\mathcal{Q}} = \left\{\hat{Q}(e) \mid e \in E'\right\}$. Recall that $\mathcal{O} = \mathcal{O}(C)$ is a circular ordering of the edges of $E' = \delta_G(C)$ that is guided by the set $\mathcal{Q}$ of paths and the rotation system $\Sigma$. As in Step 1 of the algorithm, our assignment of edges incident to vertex $u$ ensures the following crucial property. Denote $E' = \{e_1, \ldots, e_k\}$, where the edges are indexed according to the ordering $\mathcal{O}$. For each such edge $e_j$, let $e'_j$ be the last edge on path $\hat{Q}(e_j)$. Then the images of edges $e'_1, \ldots, e'_k$ enter the image of the vertex $u$ in the drawing $\psi'$ of $H'$ in the order of their indices.

**Uncrossing.** We view every path of $\hat{\mathcal{Q}}'$ as being directed towards the vertex $u'$. We use the algorithm from Theorem 4.37 in order to compute a type-2 uncrossing, that produces, for every edge $e_j \in E'$, a directed curve $\hat{\gamma}'(e_j)$, that connects the image of the endpoint of $e_j$ lying in $C$ to $u'$. Recall that we are guaranteed that the curves in the resulting set $\hat{\Gamma}' = \{\hat{\gamma}'(e_j) \mid e_j \in E'\}$ do not cross each other, and each such curve is aligned with the drawing of graph $\bigcup_{\hat{Q}'(e_\ell) \in \hat{\mathcal{Q}}'} \hat{Q}'(e_\ell)$ induced by $\psi'$ (which is identical to the drawing induced by $\psi$). Notice that the steps that we have followed in constructing the set $\hat{\Gamma}'$ of curves are identical to those we followed in order to construct the set $\Gamma_0$ of curves, and so the resulting two sets of curves are identical. In particular, the order in which the curves of $\hat{\Gamma}$ enter the image of vertex $u'$ in $\psi'$ is exactly $\mathcal{O}'$.

We now construct another set of curves, $\hat{\Gamma} = \{\hat{\gamma}(e_j) \mid e_j \in E'\}$, by letting, for each edge $e_j \in E'$, $\hat{\gamma}(e_j)$ be the image of the path $\hat{Q}(e_j) \in \hat{\mathcal{Q}}$ in the drawing $\psi'$ of $H'$. From our construction of the set $\hat{\mathcal{Q}}$ of paths, the order in which the curves of $\hat{\Gamma}$ enter the image of vertex $u$ in $\psi$ is exactly $\mathcal{O}$. For every edge $e_j \in E'$, we then let $\gamma(e_j)$ be a curve, connecting the images of $u$ and $u'$ in $\psi'$, obtained by combining the curves $\hat{\gamma}(e_j)$ and $\hat{\gamma}'(e_j)$. In order to combine the two curves, let $y_j$ be the endpoint of $e_j$ lying in $C$, and let $p_j$ be the unique point on $\psi'(e_j)$ that lies on the boundary of the tiny $y_j$-disc $D_{\psi'}(y_j)$.

We truncate curve $\hat{\gamma}'(e_j)$ so it connects point $p_j$ to the image of vertex $u'$, and we truncate the curve $\hat{\gamma}(e_j)$, so it connects point $p_j$ to the image of vertex $u$. We then concatenate the resulting two curves to obtain the curve $\gamma(e_j)$.

Consider the resulting set $\Gamma = \{\gamma(e_j) \mid e_j \in E'\}$ of curves. From the above discussion, the curves enter the image of $u'$ in $\psi'$ according to the ordering $\mathcal{O}'$, and they enter the image of $u$ according to the ordering $\mathcal{O}$. The total number of crossings between the curves in $\Gamma$ is bounded by $\mathsf{cr}(\psi')$. We call all such crossings *primary crossings*. Recall that the expected number of primary crossings is at most $O(\beta^2 \cdot |\chi(C)|)$. However, the curves in $\Gamma$ may not be in general position. This is since some vertices $x \in V(C)$ may lie on a number of paths in $\hat{\mathcal{Q}}$. In the next step we perform "nudging" around such vertices, to ensure that the resulting curves are in general position.

**Nudging.** The nudging procedure and its analysis are identical to those from Step 2 of the algorithm. We only need to perform nudging of the curves in $\Gamma$ around vertices $x \in V(C) \setminus \{u\}$.

We process each vertex $x \in V(C) \setminus \{u\}$ one by one. Consider any such vertex $x$, and let $\mathcal{Q}^x \subseteq \hat{\mathcal{Q}}$ be the set of all paths containing vertex $x$. Note that $x$ must be an inner vertex on each such path. For convenience, we denote $\mathcal{Q}^x = \left\{\hat{Q}(e_1), \ldots, \hat{Q}(e_z)\right\}$. Consider the tiny $x$-disc $D = D_{\psi'}(x)$. For all $1 \leq j \leq z$, denote by $s_j$ and $t_j$ the two points on the curve $\gamma(e_j)$ that lie on the boundary of disc $D$. We use the algorithm from Claim 4.34 to compute a collection $\{\sigma_1, \ldots, \sigma_z\}$ of curves, such that, for all $1 \leq j \leq z$, curve $\sigma_j$ connects $s_j$ to $t_j$, and the interior of the curve is contained in the interior of $D$. Recall that every pair of resulting curves crosses at most once, and every point in the interior of $D$ may be contained in at most two curves. Consider now a pair $\sigma_\ell, \sigma_{\ell'}$ of curves, and assume that these two curves cross. Recall that, from Claim 4.34, this may only happen if the two pairs $(s_\ell, t_\ell)$, $(s_{\ell'}, t_{\ell'})$ of points cross. Denote by $e_1, e_2$ the two edges that lie on path $\hat{Q}(e_\ell)$ immediately before and immediately after vertex $x$, and denote by $e_1', e_2'$ the two edges that lie on path $\hat{Q}(e_{\ell'})$ immediately before and immediately after vertex $x$. We assume that edges $e_1, e_2$ are copies of edges $\hat{e}_1, \hat{e}_2$ of $G$, and similarly, $e_1', e_2'$ are copies of edges $\hat{e}_1', \hat{e}_2'$ of $G$, respectively. Assume first that there are four distinct edges in set $\{\hat{e}_1, \hat{e}_1', \hat{e}_2, \hat{e}_2'\}$. From the fact that the two pairs $(s_\ell, t_\ell)$, $(s_{\ell'}, t_{\ell'})$ of points cross, we get that these four edges must appear in $\mathcal{O}_x \in \Sigma$ in the order $(\hat{e}_1, \hat{e}_1', \hat{e}_2, \hat{e}_2')$. Since the paths of $\mathcal{Q}$ are non-transversal with respect to $\Sigma$, this is impossible. Therefore, we conclude that paths $Q(e_\ell), Q(e_{\ell'})$ must share an edge that is incident to $x$. If $e^*$ is an edge incident to $x$ that the two paths share, then we say that $e^*$ is *responsible for the crossing between $\sigma_\ell$ and $\sigma_{\ell'}$*.

For all $1 \leq j \leq z$, we modify the curve $\gamma(e_j)$, by replacing the segment of the curve that is contained in disc $D$ with $\sigma_j$. Once every vertex $x \in V(C) \setminus \{u\}$ is processed, we obtain the final set $\Gamma^* = \{\gamma^*(e) \mid e \in E'\}$ of curves, which are now guaranteed to be in general position. Notice that as before, the curves in $\Gamma^*$ enter the image of $u'$ according to the ordering $\mathcal{O}'$, and they enter the image of $u$ according to the ordering $\mathcal{O}$. It now only remains to bound the expected number of crossings between the curves of $\Gamma^*$.

Notice that the nudging operation may have introduced some new crossings. Each such new crossing must be contained in a disc $D_{\psi'}(x)$, for some vertex $x \in V(C) \setminus \{u\}$. We call all such new crossings *secondary crossings*. We now bound the expected number of secondary crossings.

Consider some vertex $x \in V(C) \setminus \{u\}$. Every secondary crossing that is contained in $\mathcal{D}_{\psi'}(x)$ is a crossing between a pair $\sigma_\ell, \sigma_{\ell'}$ of curves that we have defined when processing vertex $x$, and each such crossing was charged to an edge of $G$ that is incident to $x$. If $e$ is an edge of $G$ that is incident to $x$, then there are at most $(\mathrm{cong}_G(\mathcal{Q}, e))^2$ pairs of paths in $\mathcal{Q}^x$ that contain copies of $e$, and each such pair of paths may give rise to a single secondary crossing in $D_{\psi'}(x)$ that is charged to edge $e$. Therefore,

the total expected number of secondary crossings is bounded by:

$$\sum_{e\in E(C)} O(\mathbf{E}\left[(\mathrm{cong}_G(\mathcal{Q},e))^2\right]) \leq O(\beta\cdot|E(C)|),$$

since we have assumed that $C\in\mathcal{W}^{\mathrm{light}}$.

Overall, the expected number of crossings between the curves in $\Gamma^*$ is at most $O(\beta^2\cdot(|\chi(C)|+|E(C)|))$, proving that $\mathbf{E}\left[\mathsf{dist}(\mathcal{O},\mathcal{O}')\right]\leq O(\beta^2\cdot(|\chi(C)|+|E(C)|))$.

# F Proofs Omitted from Section 6

## F.1 Proof of Theorem 6.3

Note that, since graph $G$ is connected, $|V(G)|\leq m+1\leq 2m$ must hold. Throughout, we use a parameter $\eta'=c\eta\log_{3/2}m\log_2 m$, were $c$ is a large enough constant, whose value we set later.

The algorithm maintains a collection $\mathcal{C}$ of disjoint clusters of $G\setminus T$, such that $\bigcup_{C\in\mathcal{C}}V(C)=V(G)\setminus T$. Set $\mathcal{C}$ of clusters is partitioned into two subsets: set $\mathcal{C}^A$ of active clusters and set $\mathcal{C}^I$ of inactive clusters. We will ensure that every cluster $C\in\mathcal{C}^I$ has the $\alpha'$-bandwidth property. Set $\mathcal{C}^I$ of inactive clusters is, in turn, partitioned into three subsets, $\mathcal{C}^I_1,\mathcal{C}^I_2$, and $\mathcal{C}^I_3$. For every cluster $C\in\mathcal{C}^I_3$, we will define a vertex $u(C)\in V(C)$, and an internal $C$-router $\mathcal{Q}(C)$, whose center vertex is $u(C)$, such that the paths in $\mathcal{Q}(C)$ are edge-disjoint. For every cluster $C\in\mathcal{C}^I_1$, we will ensure that $|E(C)|\leq O(\eta^4\log^8 m)\cdot|\delta_G(C)|$ holds. Lastly, for every cluster $C\in\mathcal{C}^I_2$, we will ensure that $\mathsf{OPT}_{\mathsf{cr}}(C)\geq\Omega(|E(C)|^2/(\eta^2\operatorname{poly}\log m))$, and $|E(C)|>\Omega(\eta^4|\delta_G(C)|\log^8 m)$. We start with $\mathcal{C}^I=\emptyset$, and $\mathcal{C}^A$ containing a single cluster $G\setminus T$ (note that graph $G\setminus T$ is connected since $G$ is connected and every terminal has degree 1). The algorithm terminates once $\mathcal{C}^A=\emptyset$, and once this happens, we return $\mathcal{C}^I$ as the algorithm's outcome.

In order to bound the number of edges in the contracted graph $E(G_{|\mathcal{C}})$, we will use edge budgets and vertex budgets, that are defined as follows.

**Edge Budgets.** If an edge $e$ belongs to the boundary $\delta_G(C)$ of a cluster $C\in\mathcal{C}$, then, if $C\in\mathcal{C}^I$, we set the budget $B_C(e)=1$, and otherwise we set it to be $B_C(e)=\log_{3/2}(|\delta_G(C)|)$. If cluster $C$ is the unique cluster with $e\in\delta_G(C)$, then we set $B(e)=B_C(e)$. If there are two clusters $C\neq C'\in\mathcal{C}$ with $e\in\delta_G(C)$ and $e\in\delta_G(C')$, then we set $B(e)=B_C(e)+B_{C'}(e)$. Lastly, if no cluster $C\in\mathcal{C}$ with $e\in\delta_G(C)$ exists, then we set $B(e)=0$.

**Vertex Budgets.** Vertex budgets are defined as follows. For every cluster $C\in\mathcal{C}^A$, for every vertex $v\in V(C)$, we set the budget $B(v)=\frac{c\deg_C(v)\log_{3/2}m\cdot\log_2(|E(C)|)}{8\eta'}$, where $c$ is the constant used in the definition of $\eta'$. The budgets of all other vertices are set to 0.

**Cluster Budgets and Total Budget.** For a cluster $C\in\mathcal{C}$, we define its edge-budget $B^E(C)=\sum_{e\in\delta_G(C)}B_C(e)$, and its vertex-budget $B^V(C)=\sum_{v\in V(C)}B(v)$. The total budget of a cluster $C\in\mathcal{C}$ is $B(C)=B^E(C)+B^V(C)$, and the total budget in the system is $B^*=\sum_{C\in\mathcal{C}}B(C)=\sum_{e\in E(G)}B(e)+\sum_{v\in V(G\setminus T)}B(v)$.

Notice that at the beginning of the algorithm, the budget of every vertex $v\in V(G)\setminus T$ is bounded by:

$$\frac{c\cdot\deg_{G\setminus T}(v)\cdot\log_{3/2}m\cdot\log_2|E(G)|}{8\eta'}\leq\frac{\deg_G(v)}{8\eta},$$

the budget of every edge incident to a vertex in $T$ is at most $\log_{3/2}(|T|) \leq 16 \log m$, while the budget of every other edge is 0. Therefore, the total budget $B^*$ in the system at the beginning of the algorithm is:

$$\frac{m}{4\eta} + 16k \log m \leq \frac{m}{\eta},$$

since $k \leq \frac{m}{16\eta \log m}$ from the statement of Theorem 6.3.

We will ensure that, throughout the algorithm, the total budget $B^*$ never increases. Since, from the definition, $B^* \geq \sum_{C \in \mathcal{C}} |\delta_G(C)|$, this ensures that, when the algorithm terminates, $|E(G_{|\mathcal{C}})| \leq m/\eta$, so the set $\mathcal{C}^I$ of clusters and its partition $(\mathcal{C}_1^I, \mathcal{C}_2^I, \mathcal{C}_3^I)$ is a valid output of the algorithm.

As mentioned above, the algorithm starts with $\mathcal{C}^I = \emptyset$, and $\mathcal{C}^A$ contains a single cluster – cluster $G \setminus T$. As long as $\mathcal{C}^A \neq \emptyset$, we perform iterations, where in each iteration we select an arbitrary cluster $C \in \mathcal{C}^A$ to process. We now describe the execution of an iteration in which cluster $C \in \mathcal{C}^A$ is processed. The algorithm for processing cluster $C$ consists of three steps, that we describe next.

**Step 1: Bandwidth Property.** In this step we will either establish that $C$ has the $\alpha'$-bandwidth property, or we will partition it into smaller clusters that will replace $C$ in set $\mathcal{C}^A$. Let $C^+$ be the augmentation of cluster $C$. Recall that $C^+$ is a graph that is obtained as follows. We start with the graph $G$, and we subdivide every edge $e \in \delta_G(C)$ with a vertex $t_e$, letting $T(C) = \{t_e \mid e \in \delta_G(C)\}$ be this new set of vertices. We then let $C^+$ be the subgraph of the resulting graph induced by $V(C) \cup T(C)$. From Observation 4.16, cluster $C$ has the $\alpha'$-bandwidth property iff the set $T(C)$ of vertices is $\alpha'$-well-linked in $C^+$. We apply the algorithm $\mathcal{A}_{\text{ARV}}$ to graph $C^+$ and terminal set $T(C)$, to obtain an $\beta_{\text{ARV}}(m) = O(\sqrt{\log m})$-approximate sparsest cut $(X, Y)$ in graph $C^+$ with respect to the set $T(C)$ of terminals.

We can assume without loss of generality that, for every vertex $t_e \in T(C)$, if $t_e \in X$, and $e' = (t_e, v)$ is the unique edge that is incident to $t_e$ in $C^+$, then $v \in X$ as well (as otherwise we can move $t_e$ to $Y$, making the cut only sparser). Similarly, if $t_e \in Y$, then $v \in Y$ as well. We assume w.l.o.g. that $|X \cap T(C)| \leq |Y \cap T(C)|$. We then consider two cases. First, if $|E(X, Y)| \geq \alpha' \cdot \beta_{\text{ARV}}(m) \cdot |X \cap T(C)|$, then we are guaranteed that vertex set $T(C)$ is $\alpha'$-well-linked in $C^+$, and therefore cluster $C$ has the $\alpha'$-bandwidth property. Assume now that $|E(X, Y)| < \alpha' \cdot \beta_{\text{ARV}}(m) \cdot |X \cap T(C)|$.

Let $X' = X \setminus T(C)$ and $Y' = Y \setminus T(C)$, so $(X', Y')$ is a partition of $C$. Note that $|T(C) \cap X| = |\delta_G(C) \cap \delta_G(X')|$ and similarly $|T(C) \cap Y| = |\delta_G(C) \cap \delta_G(Y')|$.

We remove cluster $C$ from $\mathcal{C}^A$, and we add all connected components of $C[X']$ and $C[Y']$ to $\mathcal{C}^A$ instead. Observe that we are still guaranteed that $\bigcup_{C' \in \mathcal{C}} V(C') = V(G) \setminus T$. We now show that the total budget in the system does not increase as the result of this step.

Since $|X'|, |Y'| < |V(C)|$, it is immediate to verify that, for every vertex $v$ of $C$, its budget may only decrease. The only edges whose budget may increase are the edges of $E_C(X', Y')$. The number of such edges is bounded by $\alpha' \cdot \beta_{\text{ARV}}(m) \cdot |X \cap T(C)| = \alpha' \cdot \beta_{\text{ARV}}(m) \cdot |\delta_G(C) \cap \delta_G(X')|$, and the budget of each of them increases by at most $2 \log_{3/2} m \leq 8 \log m$, so the total increase in the budget of all edges due to this step is bounded by:

$$8\alpha' \cdot \beta_{\text{ARV}}(m) \cdot |\delta_G(C) \cap \delta_G(X')| \cdot \log m \leq |\delta_G(C) \cap \delta_G(X')|,$$

since $\alpha' = \frac{1}{16 \beta_{\text{ARV}}(m) \cdot \log m}$.

Consider now some edge $e \in \delta_G(C) \cap \delta_G(X')$. Since we have assumed that $|X \cap T(C)| \leq |Y \cap T(C)|$, it is easy to verify that $|\delta_G(X')| \leq 2|\delta_G(C)|/3$. Therefore, if an endpoint of an edge $e \in \delta_G(C)$ belongs to a new cluster $C' \subseteq G[X']$, then the new budget $B_{C'}(e)$ becomes at most:

$$\log_{3/2}(|\delta_G(X')|) \leq \log_{3/2}(|\delta_G(C)|) - 1.$$

The total decrease in the global budget due to the edges of $\delta_G(X') \cap \delta_G(C)$ is then at least $|\delta_G(C) \cap \delta_G(X')|$. We conclude that overall the global budget does not increase.

We assume from now on that algorithm $\mathcal{A}_{\text{ARV}}$ returned a cut $(X, Y)$ of $C^+$ with $|E(X, Y)| \geq \alpha' \cdot \beta_{\text{ARV}}(m) \cdot |X \cap T(C)|$, and so cluster $C$ has the $\alpha'$-bandwidth property.

Assume now that $|E(C)| \leq (\eta')^4 |\delta_G(C)|$. From the definition of $\eta'$, we are then guaranteed that $|E(C)| \leq O(\eta^4 \log^8 m) \cdot |\delta_G(C)|$. We then remove cluster $C$ from $\mathcal{C}^A$ and add it to the set $\mathcal{C}^I$ of inactive clusters, and to the set $\mathcal{C}^I_1$ of clusters. Therefore, we assume from now on that $|E(C)| > (\eta')^4 |\delta_G(C)|$.

**Step 2: Sparse Balanced Cut.** In this step, we apply the algorithm from Theorem 4.11 to graph $C$ with parameter $\hat{c} = 3/4$, to obtain a $\hat{c}'$-edge-balanced cut $(Z, Z')$ of $C$ (where $1/2 < \hat{c}' < 1$), whose value is at most $O(\beta_{\text{ARV}}(m))$ times the value of a minimum 3/4-edge-balanced cut of $C$. We say that this step is *successful* if $|E_G(Z, Z')| < |E(C)|/\eta'$. Assume first that the step was successful. Then we remove cluster $C$ from set $\mathcal{C}^A$, and add all connected components of graphs $C[Z], C[Z']$ to set $\mathcal{C}^A$ insead. We now show that the total budget in the system does not increase as the result of this step. Observe that the budget of every vertex may only decrease, and the same is true for the budget of every edge, except for the edges in set $\delta_G(C) \cup E_G(Z, Z')$. The budget of each such edge may increase by at most $2 \log_{3/2} m$, so the total increase in the budgets of all edges is bounded by $(|\delta_G(C)| + |E_G(Z, Z')|) \cdot 2 \log_{3/2} m \leq \frac{4|E(C)| \cdot \log_{3/2} m}{\eta'}$ (we have used the fact that $|E(C)| > (\eta')^4 |\delta_G(C)|$). We now show that this increase in total budget is compensated by the decrease in the budgets of the vertices of $Z$.

Assume without loss of generality that $|E(Z)| \leq |E(Z')|$. Recall that for every vertex $v \in Z$, its original budget is: $B(v) = \frac{c \deg_C(v) \log_{3/2} m \cdot \log_2(|E(C)|)}{8\eta'}$. From our assumption that $|E(Z)| \leq |E(Z')|$, $\log_2(|E(Z)|) \leq \log_2(|E(C)|) - 1$. The new budget of vertex $v$ is:

$$B'(v) = \frac{c \deg_Z(v) \log_{3/2} m \cdot \log_2(|E(Z)|)}{8\eta'} \leq \frac{c \deg_Z(v) \log_{3/2} m \cdot (\log_2(|E(C)|) - 1)}{8\eta'}.$$

Therefore, for every vertex $v \in Z$, its budget decreases by at least $\frac{c \deg_Z(v) \log_{3/2} m}{8\eta'}$.

From the definition of a (3/4)-edge-balanced cut, $|E(Z')| \leq \hat{c}' |E(C)|$, for some universal constant $\hat{c}'$. In particular:

$$\sum_{v \in Z} \deg_Z(v) \geq |E(C)| - |E(Z')| \geq (1 - \hat{c}')|E(C)|.$$

Overall, the budget of the vertices in $Z$ decreases by at least:

$$\frac{c \log_{3/2} m}{8\eta'} \cdot \sum_{v \in Z} \deg_Z(v) \geq \frac{c \log_{3/2} m}{8\eta'} \cdot (1 - \hat{c}') \cdot |E(C)|.$$

Since $\hat{c}' < 1$, and we can set $c$ to be a large enough constant, we can ensure that this is at least $\frac{4|E(C)| \cdot \log_{3/2} m}{\eta'}$, so the overall budget in the system does not increase.

If the current step is successful, then we replace cluster $C$ with the connected components of $C[Z]$ and $C[Z']$ in set $\mathcal{C}^A$ and continue to the next iteration. Therefore, we assume from now on that the current step was not successful. In other words, the algorithm from Theorem 4.11 returned a cut$(Z, Z')$ with $|E_G(Z, Z')| \geq |E(C)|/\eta'$. Since this cut is within factor $\beta_{\text{ARV}}(m) = O(\sqrt{\log m})$ from the minimum 3/4-edge-balanced cut, we conclude that the value of the minimum 3/4-edge-balanced cut in $C$ is at least $\rho = \frac{|E(C)|}{\eta' \cdot \beta_{\text{ARV}}(m)}$.

From Lemma 4.12, if the maximum vertex degree $\Delta$ in graph $C$ is at most $|E(C)|/2^{40}$ and $\mathsf{OPT}_{\mathsf{cr}}(C) \leq |E(C)|^2/2^{40}$, then graph $C$ has a (3/4)-edge-balanced cut of value at most $\tilde{c} \cdot \sqrt{\mathsf{OPT}_{\mathsf{cr}}(C) + \Delta \cdot |E(C)|}$

where $\tilde{c} > 2^{40}$ is some universal constant. As the size of the minimum 3/4-balanced cut in $C$ is at least $\rho$, we conclude that either $\Delta \geq |E(C)|/2^{40}$, or $\mathsf{OPT}_{\mathsf{cr}}(C) > |E(C)|^2/2^{40}$, or $\sqrt{\mathsf{OPT}_{\mathsf{cr}}(C) + \Delta \cdot |E(C)|} \geq \rho/\tilde{c}$. The latter can only happen if either $\mathsf{OPT}_{\mathsf{cr}}(C) \geq \rho^2/\tilde{c}^2$, or $\Delta \geq \rho^2/(\tilde{c}^2 \cdot |E(C)|)$. Substituting the value of $\rho = \frac{|E(C)|}{\eta' \cdot \beta_{\mathrm{ARV}}(m)}$, we conclude that either $\mathsf{OPT}_{\mathsf{cr}}(C) \geq \frac{|E(C)|^2}{(\tilde{c}\eta'\beta_{\mathrm{ARV}}(m))^2}$, or $\Delta \geq \frac{|E(C)|}{(\tilde{c}\eta'\beta_{\mathrm{ARV}}(m))^2}$, or $\Delta \geq \frac{|E(C)|}{2^{40}}$. Note that we can check whether the last two conditions hold efficiently. If they do not hold, then we are guaranteed that $\mathsf{OPT}_{\mathsf{cr}}(C) \geq \frac{|E(C)|^2}{(\tilde{c}\eta'\beta_{\mathrm{ARV}}(m))^2} \geq \frac{|E(C)|^2}{\eta^2 \operatorname{poly}\log m}$. Recall that we are also guaranteed that $|E(C)| > (\eta')^4 \cdot |\delta_G(C)| = \Omega(\eta^4 |\delta^G(C)| \log^8 m)$. We then remove cluster $C$ from the set $\mathcal{C}^A$ of active clusters and add it to set $\mathcal{C}^I$ of inactive clusters, where it joins the set $\mathcal{C}_2^I$ of clusters. From now on we can assume that $|E(C)| > \Omega(\eta^4 |\delta^G(C)| \log^8 m)$, and that either $\Delta \geq \frac{|E(C)|}{(\tilde{c}\eta'\beta_{\mathrm{ARV}}(m))^2}$, or $\Delta \geq \frac{|E(C)|}{2^{40}}$ hold. Note that, since $\eta' = c\eta \log_{3/2} m \log_2 m$, and since we can set $c$ to be a large enough constant, we can ensure that $\frac{|E(C)|}{2^{40}} \geq \frac{|E(C)|}{\tilde{c}^2(\eta')^2\beta_{\mathrm{ARV}}(m)}$. Therefore, from now on we assume that there is some vertex $v^*$ in graph $C$, whose degree in $C$ is at least $\frac{|E(C)|}{(\tilde{c}\eta'\beta_{\mathrm{ARV}}(m))^2}$. Since we have assumed that $|E(C)| \geq (\eta')^4 |\delta_G(C)|$, we get that $\deg_G(v^*) \geq 8|\delta_G(C)|\eta'$.

**Step 3: routing to a vertex** We consider again the graph $C^+$ that we have defined in Step 1, with the corresponding set $T(C) = \{t_e \mid e \in \delta_G(C)\}$ of terminal vertices. Using standard Maximum Flow computation, we compute a maximum-cardinality set $\mathcal{P}$ of edge-disjoint paths, where each path connects a distinct vertex of $T(C)$ to to vertex $v^*$. We now consider two cases. In the first case, $|\mathcal{P}| = |\delta_G(C)|$. In this case, there is a set $\mathcal{Q}(C)$ of edge-disjoint paths in cluster $C$, routing edges of $\delta_e(C)$ to vertex $v^*$, which can be easily obtained from $\mathcal{P}$. We then remove cluster $C$ from the set $\mathcal{C}^A$ of active clusters and add it to set $\mathcal{C}^I$ of inactive clusters, where it joins cluster set $\mathcal{C}_3^I$.

Assume now that $|\mathcal{P}| < |\delta_G(C)|$. From the maximum flow / minimum cut theorem, there is a collection $E'$ of at most $|\delta_G(C)|$ edges in graph $C^+$, such that, in graph $C^+ \setminus E'$, there is no path connecting an edge of $\delta_G(C)$ to vertex $u^*$. Let $C'$ be the connected component of $C^+ \setminus E'$ containing $v^*$, so $C' \subseteq C$. Note that $\delta_G(C') \subseteq E'$, so there is a set $\mathcal{Q}(C')$ of edge-disjoint paths routing the edges of $\delta_G(C')$ to vertex $v^*$, with all inner vertices of the paths in $\mathcal{Q}(C')$ lying in $C'$; in other words, $\mathcal{Q}(C')$ is an internal $C'$-router. We add cluster $C'$ to set $\mathcal{C}^I$ and $\mathcal{C}_3^I$. Next, we delete cluster $C$ from $\mathcal{C}^A$, and we add every connected component of $C \setminus C'$ as a new cluster to set $\mathcal{C}^A$.

It now remains to prove that the total budget in the system does not increase as the result of these changes. Note that the budget of every vertex may only decrease, and the budget of every edge, except for the edges of $\delta_G(C) \cup E'$, may also only decrease. The increase in the budget of every edge in $\delta_G(C) \cup E'$ is bounded by $2\log_{3/2} m$. Therefore, the total increase in the budget is bounded by $(|\delta_G(C)| + |E'|) \cdot 2\log_{3/2} m \leq 4|\delta_G(C)| \cdot \log_{3/2} m$. Note that the budget of vertex $v^*$ was initially at least $\frac{c \deg_C(v) \log_{3/2} m \cdot \log_2(|E(C)|)}{8\eta'}$, and it becomes 0 at the end of this step. Therefore, the decrease in the budget is at least:

$$\frac{c \deg_C(v^*) \log_{3/2} m \cdot \log_2(|E(C)|)}{8\eta'} \geq 4|\delta_G(C)| \cdot \log_{3/2} m,$$

since $\deg_C(v^*) \geq 8|\delta_G(C)|\eta'$. Overall, the budget does not grow. The algorithm terminates once $\mathcal{C}^A$ becomes empty, and it returns the set $\mathcal{C}^I$ of clusters. From our invariants, it is immediate to verify that this set of clusters has all required properties. It remains to establish that the algorithm is efficient. In every iteration of the algorithm, we either add a cluster to set $\mathcal{C}^I$, or we split a single cluster of $\mathcal{C}^A$ into at least two clusters. Once a cluster is added to $\mathcal{C}^I$, it remains there until the end of the algorithm. It is then easy to verify that the number of iterations is bounded by $O(|V(G)|) \leq O(m)$, and every iteration can be executed efficiently.

## F.2 Proof of Observation 6.5

Let $\varphi^*$ be the optimal solution to instance $(\tilde{C}, \Sigma_{\tilde{C}})$. We can assume w.l.o.g. that every pair of edges cross at most once in $\varphi^*$. Denote by $\chi$ the collection of all pairs $e, e' \in E(\tilde{C})$ of edges, such that the images of $e$ and $e'$ cross in $\varphi^*$.

Assume for now that, for every cluster $W \in \mathcal{W}^{\text{light}}$, the routers $\mathcal{Q}(W)$ and $\hat{\mathcal{Q}}(W)$ are fixed, and so the rotation system $\hat{\Sigma}$ for graph $H$ is fixed as well. For every edge $e \in E(\tilde{C})$, we define an integer $N(e)$ as follows. If there is a cluster $W \in \mathcal{W}^{\text{light}}$ with $e \in E(W)$, then we let $N(e)$ be the number of paths in $\hat{\mathcal{Q}}(W)$ containing $e$. Therefore, $N(e) = \text{cong}_G(\hat{\mathcal{Q}}(W), e) \leq \text{cong}_G(\mathcal{Q}(W), e)$. Otherwise, we set $N(e) = 1$. We will prove the following observation:

**Observation F.1** *Suppose the routers $\hat{\mathcal{Q}}(W)$ for all clusters $W \in \mathcal{W}^{\text{light}}$ are fixed, and let $\hat{\Sigma}$ be the corresponding rotation system for $H$. Then:*

$$\mathsf{OPT}_{\mathsf{cnwrs}}(H, \hat{\Sigma}) \leq O\left(\sum_{(e,e') \in \chi} N(e) \cdot N(e')\right) + O\left(\sum_{e \in E(\tilde{C})} (N(e))^2\right)$$

.

The proof of Observation 6.5 immediately follows from Observation F.1. Indeed:

$$\begin{aligned}
\mathbf{E}\left[\mathsf{OPT}_{\mathsf{cnwrs}}(H, \hat{\Sigma})\right] &\leq \mathbf{E}\left[\sum_{(e,e') \in \chi} O(N(e) \cdot N(e')) + \sum_{e \in E(\tilde{C})} O((N(e))^2)\right] \\
&\leq O\left(\sum_{(e,e') \in \chi} \mathbf{E}\left[N(e) \cdot N(e')\right]\right) + O\left(\sum_{e \in E(\tilde{C})} \mathbf{E}\left[(N(e))^2\right]\right) \\
&\leq O\left(\sum_{(e,e') \in \chi} \left(\mathbf{E}\left[(N(e))^2\right] + \mathbf{E}\left[(N(e'))^2\right]\right)\right) + O\left(\sum_{e \in E(\tilde{C})} \mathbf{E}\left[N(e))^2\right]\right).
\end{aligned}$$

Consider now some edge $e \in E(\tilde{C})$. Assume first that there is some cluster $W \in \mathcal{W}^{\text{light}}$ with $e \in E(W)$. Then, as observed above, $N(e) \leq \text{cong}_G(\mathcal{Q}(W), e)$. Since $\mathcal{Q}(W)$ is a router of $\Lambda_G(W)$ that is drawn from the distribution $\mathcal{D}(W)$, and since cluster $W$ is $\beta_i$-light with respect to $\mathcal{D}(W)$, we get that:

$$\mathbf{E}_{\mathcal{Q}(W) \sim \mathcal{D}(W)}\left[(N(e))^2\right] \leq \mathbf{E}_{\mathcal{Q}(W) \sim \mathcal{D}(W)}\left[(\text{cong}_G(\mathcal{Q}(W), e))^2\right] \leq \beta_i.$$

Otherwise, $e \in E(\tilde{C}) \setminus \left(\bigcup_{W \in \mathcal{W}^{\text{light}}} E(W)\right)$, and so $N(e) = 1$ holds. Overall, for every edge $e \in E(\tilde{C})$, $\mathbf{E}\left[(N(e))^2\right] \leq \beta_i$. Therefore, we get that:

$$\mathbf{E}\left[\mathsf{OPT}_{\mathsf{cnwrs}}(H, \hat{\Sigma})\right] \leq |\chi| \cdot O(\beta_i) + |E(\tilde{C})| \cdot O(\beta_i) = O\left(\beta_i \cdot \left(\mathsf{OPT}_{\mathsf{cnwrs}}(\tilde{C}, \Sigma_{\tilde{C}}) + |E(\tilde{C})|\right)\right).$$

In order to complete the proof of Observation 6.5, it is now enough to prove Observation F.1.

**Proof of Observation F.1.** The proof uses arguments that are very similar to those used in the proof of Lemma 5.6, and more specifically in the proof of Claim E.1. Similar argument are used in several places throughout this paper, so we only provide a proof sketch here. We start with the optimal solution $\varphi^*$ to instance $(\tilde{C}, \Sigma_{\tilde{C}})$, and then gradually transform it to obtain a solution $\hat{\psi}$ to instance $(H, \hat{\Sigma})$.

Let $C^*$ be the graph that is obtained from $\tilde{C}$ as follows. We let $V(C^*) = V(\tilde{C})$. For every edge $e = (u, v) \in E(\tilde{C})$ with $N(e) > 0$, we add a set $J(e)$ of $N(e)$ parallel edges $(u, v)$ to graph $C^*$. We call the edges of $J(e)$ *copies of edge* $e$. We can modify the solution $\varphi^*$ to instance $\tilde{C}$ to obtain a drawing $\psi$ of graph $C^*$ in a natural way: the images of all vertices of $\tilde{C}$ remain unchanged. For every edge $e \in E(\tilde{C})$, we draw the images of the edges of $J(e)$ in $\psi$ in parallel very close to the original image of edge $e$ in $\varphi^*$. It is immediate to verify that the number of crossings in the resulting drawing $\psi$ of graph $C^*$ is bounded by $\sum_{(e,e') \in \chi} N(e) \cdot N(e')$.

Consider now some cluster $W \in \mathcal{W}^{\text{light}}$. We will now define, for every edge $e \in \delta_{\tilde{C}}(W)$, a curve $\gamma^*(e)$, that will serve as the image of $e$ in the solution $\hat{\psi}$ to instance $(H, \hat{\Sigma})$ that we construct. Note that for each edge $e \in \delta_{\tilde{C}}(W)$, $N(e) = 1$, and so set $J(e)$ of edges contains exactly one copy of edge $e$. We do not distinguis between edge $e$ and its unique copy in $J(e)$.

We use the internal $W$-router $\hat{\mathcal{Q}}(W)$ in graph $G$, in order to define a collection $\hat{\mathcal{Q}}'(W) = \left\{\hat{Q}'(e) \mid e \in \delta_{\tilde{C}}(W)\right\}$ of edge-disjoint paths in graph $C^* \cup \delta_{\tilde{C}}(W)$, such that, for every edge $e \in \delta_{\tilde{C}}(W)$, path $\hat{Q}'(e)$ originates at edge $e$, terminates at vertex $u(W)$ (the center vertex of the router $\hat{\mathcal{Q}}(W)$), and all internal vertices of $\hat{Q}'(e)$ lie in $V(W)$. In order to obtain the collection $\hat{\mathcal{Q}}'(W)$ of paths from the router $\hat{\mathcal{Q}}(W)$, for every edge $e \in E(W)$ with $N(e) > 0$, we assign every copy of $e$ in $J(e)$ to a distinct path of $\hat{\mathcal{Q}}(W)$ that contains the edge $e$.

Consider any edge $e \in \delta_{\tilde{C}}(W)$, and denote $e = (x_e, y_e)$, where $x_e \in V(W)$. Initially, we let $\gamma(e)$ be the image of the path $\hat{Q}'(e)$ in the drawing $\psi$ of $C^*$. Let $\Gamma(W) = \left\{\gamma(e) \mid e \in \delta_{\tilde{C}}(W)\right\}$ be the resulting collection of curves. Note that, for every edge $e \in \delta_{\tilde{C}}(W)$, curve $\gamma(e)$ connects the image of vertex $y_e$ in drawing $\varphi^*$ to the image of vertex $u(W)$ in the same drawing.

We are now ready to construct an initial drawing $\psi'$ of the graph $H$. For regular every vertex $v \in V(H) \cap V(\tilde{C})$, the image of $v$ in $\psi'$ remains the same as in $\varphi^*$ (and in $\psi$). For every edge $e$ of $H$ whose both endpoints are regular vertices, the image of $e$ remains the same as in $\varphi^*$ (and the same as in $\psi$). Consider now some cluster $W \in \mathcal{W}^{\text{light}}$. The image of the supernode $v_W$ in drawing $\psi'$ is the image of vertex $u(W)$ in $\varphi^*$ (and in $\psi$). For every edge $e \in \delta_{\tilde{C}}(W)$, the initial image of edge $e$ is the curve $\gamma(e)$. Lastly, since the degree of every terminal $t \in T$ is 1 in $H$, we can add these terminals and their adjacent edges to the current drawing without increasing the number of crossings. We note that the resulting drawing $\psi'$ of graph $H$ may not be a valid drawing. This is since, whenever there is a cluster $W \in \mathcal{W}^{\text{light}}$ and a vertex $x \in V(W)$ that lies on more than two paths of $\hat{\mathcal{Q}}(W)$, then point $p = \varphi^*(x)$ belongs to more than two curves of $\Gamma(W)$, and hence more than two edges cross at point $p$. In order to overcome this difficulty, for every cluster $W \in \mathcal{W}^{\text{light}}$, for every vertex $x \in V(W)$ that lies on at least two paths of $\hat{\mathcal{Q}}(W)$, we perform a nudging operatio of the curves in $\Gamma(W)$ in the vicinity of vertex $x$ (see Section 4.4.3). The curves are modified locally inside the tiny $x$-disc $D_{\varphi^*}(x)$ to ensure that every point of $D_{\varphi^*}(x)$ lies on at most two curves. Consider now a pair $e_1, e_2 \in \delta_{\tilde{C}}(W)$ of edges. Let $\hat{Q}(e_1), \hat{Q}(e_2)$ be the paths of $\hat{\mathcal{Q}}(W)$ that originate at $e_1$ and $e_2$, respectively, and assume that both paths contain vertex $x$. From the definition of the nudging procedure (see also Claim 4.34), and since the paths in $\hat{\mathcal{Q}}(W)$ are non-transversal with respect to $\Sigma_{\tilde{C}}$, the curves $\gamma(e_1), \gamma(e_2)$ that are obtained at the end of the nudging operation may only cross inside disc $D_{\varphi^*}(x)$ if some edge $e^* \in \delta_{\tilde{C}}(x)$ lies on both $\hat{Q}(e_1)$ and $\hat{Q}(e_2)$. We say that edge $e^*$ is *responsible* for this crossing of $\gamma(e_1)$ and $\gamma(e_2)$.

For every edge $e \in \delta_{\tilde{C}}(W)$, let $\gamma'(e)$ be the curve that is obtained after the nudging operation is

performed for every vertex $x \in V(W)$ that belongs to at least two paths of $\hat{\mathcal{Q}}(W)$, and denote $\Gamma'(W) = \left\{\gamma'(e) \mid e \in \delta_{\tilde{C}}(W)\right\}$. For every edge $e \in \delta_{\tilde{C}}(W)$, we replace the image of edge $e$ in the current drawing with the curve $\gamma'(e)$.

Consider the drawing $\psi'$ of graph $H$ that is obtained after all clusters $W \in \mathcal{W}^{\text{light}}$ are processed. It is now easy to verify that $\psi'$ is a valid drawing of graph $H$. We partition the crossigns of drawing $\psi'$ into two types: a crossing is of type 1 if it is present in drawing $\psi$, and it is of type 2 otherwise. Equivalently, a crossing $(e, e')_p$ of $\psi'$ is of type 2 if there is a cluster $W \in \mathcal{W}^{\text{light}}$, and a vertex $x \in V(W)$, such that the crossing point $p$ lies in $D_{\varphi^*}(x)$.

The number of type-1 crossings in $\psi'$ remains bounded by $\mathsf{cr}(\psi) \leq \sum_{(e,e')\in\chi} N(e) \cdot N(e')$. In order to bound the number of type-2 crossings, observe that for every cluster $W \in \mathcal{W}^{\text{light}}$, vertex $x \in V(W)$ and edge $e \in \delta_{\tilde{C}}(x)$, the number of type-2 crossings for which edge $e$ may be responsible is at most $(\text{cong}_{\tilde{C}}(\hat{\mathcal{Q}}(W), e))^2 = (N(e))^2$. Overall, we get that $\mathsf{cr}(\psi') \leq \sum_{(e,e')\in\chi} N(e) \cdot N(e') + \sum_{e\in E(\tilde{C})} (N(e))^2$.

Observe that for every regular vertex $x \in V(H) \cap V(\tilde{C})$, drawing $\psi'$ of $H$ obeys the rotation $\mathcal{O}_x \in \hat{\Sigma}$ (which is identical to the rotation of $x$ in $\Sigma_{\tilde{C}}$). However, it is possible that for some supernodes $v_W$, the rotation $\mathcal{O}_{v_W} \in \hat{\Sigma}$ is not obeyed by $\psi'$. We fix this issue by introducing at most $O\left(\sum_{e\in E(\tilde{C})} (N(e))^2\right)$ additional crossings, as follows.

Consider a cluster $W \in \mathcal{W}^{\text{light}}$, and denote $\delta_{\tilde{C}}(u(W)) = \{a_1, a_2, \ldots, a_r\}$, where the edges are indexed according to their order in the rotation $\mathcal{O}_{u(W)} \in \Sigma_{\tilde{C}}$. For all $1 \leq i \leq r$, let $\mathcal{Q}_i \subseteq \hat{\mathcal{Q}}(W)$ be the set of paths whose last edge is $a_i$. Denote by $A_i \subseteq \delta_{\tilde{C}}(W)$ the set of edges $e$, for which the unique path $\hat{Q}(e) \in \hat{\mathcal{Q}}(W)$ that originates at $e$ terminates at edge $a_i$; in other words, $\hat{Q}(e) \in \mathcal{Q}_i$. Denote by $\Gamma'_i \subseteq \Gamma'(W)$ the set of the images of the edges of $A_i$ in the current drawing $\psi'$. Let $\mathcal{O}'$ be that the circular order of the edges of $\delta_{\tilde{C}}(W) = A_1 \cup A_2 \cup \cdots \cup A_r$, in which their images in $\psi'$ enter the image of $v_W$. Then for all $1 \leq i \leq r$, the edges of $A_i$ appear consecutively in $\mathcal{O}'$ in some arbitrary order, while the edges lying in different groups of $A_1, A_2, \ldots, A_r$ appear in $\mathcal{O}'$ in the natural order of the indices of these groups.

Recall that the rotation $\mathcal{O}_{v_W} \in \hat{\Sigma}$ is a circular ordering of the edges of $\delta_{\tilde{C}}(W) = A_1 \cup A_2 \cup \cdots \cup A_r$ that is guided by the paths of $\hat{\mathcal{Q}}(W)$. In this ordering, for all $1 \leq i \leq r$, the edges of $A_i$ appear consecutively in some arbitrary order, while the edges lying in different groups of $A_1, A_2, \ldots, A_r$ appear in $\mathcal{O}_{v_W}$ in the natural order of the indices of these groups. In other words, the only difference between the orderings $\mathcal{O}'$ and $\mathcal{O}_{v_W}$ is that, for all $1 \leq i \leq r$, the edges of $A_i$ may appear in different order in the two orderings. Therefore, $\mathsf{dist}(\mathcal{O}', \mathcal{O}_{v_W}) \leq \sum_{i=1}^{r} |A_i|^2 = \sum_{i=1}^{r} (\text{cong}_{\tilde{C}}(\hat{\mathcal{Q}}(W), e_i))^2 \leq \sum_{i=1}^{r} (N(e_i))^2$. For all $1 \leq i \leq r$, we slightly modify the curves of $\Gamma'_i$ inside the tiny $v_W$-disc $D_{\psi'}(v_W)$ to ensure that the images of the edges of $\delta_{\tilde{C}}(W)$ enter the image of vertex $v_W$ in the circular order $\mathcal{O}_{v_W}$. From the above discussion, this can be done by introducing at most $\sum_{i=1}^{r} (N(e_i))^2$ new crossings. Once every cluster $W \in \mathcal{W}^{\text{light}}$ is processed, we obtain a valid solution $\hat{\psi}$ to instance $(H, \hat{\Sigma})$, with $\mathsf{cr}(\hat{\psi}) \leq O\left(\sum_{(e,e')\in\chi} N(e) \cdot N(e') + \sum_{e\in E(\tilde{C})} (N(e))^2\right)$.

□

# G Proofs Omitted from Section 7

## G.1 Proof of Claim 7.9

Consider any cluster $R \in \mathcal{L}$. Let $N(R)$ denote the number of clusters $C \in \mathcal{C}$ with $C \subseteq R$. Clearly, $N(R) \leq |\mathcal{C}| \leq m$ must hold.

Assume first that $R \neq G$, and vertex $v(R)$ is unmarked in the tree $\tau(\mathcal{L})$. Let $R'$ be the parent-cluster of $R$, that is $v(R')$ is the parent vertex of $v(R)$ in $\tau(\mathcal{L})$. In this case, the algorithm from Theorem 7.8,

when applied to the graph $G'$ corresponding to cluster $R'$, returned a type-2 legal clustering $\mathcal{R}$ of $G'$, with $R \in \mathcal{R}$, and moreover, $R$ is not the distinguished cluster $R^*$. Let $\mathcal{C}'$ denote the set of all clusters $C \in \mathcal{C}$ with $C \subseteq R'$, so that $N(R') = |\mathcal{C}'|$. Recall that, from the definition of type-2 legal clustering, the distinguished cluster $R^*$ must contain at least $\left\lfloor \left(1 - 1/2^{(\log m)^{3/4}}\right) |\mathcal{C}'| \right\rfloor$ clusters of $\mathcal{C}'$. Therefore, if $N(R') \geq 2^{(\log m)^{3/4}}$, then $R$ may contain at most $2N(R')/2^{(\log m)^{3/4}}$ clusters of $\mathcal{C}'$, that is, $N(R) \leq 2N(R')/2^{(\log m)^{3/4}}$. Otherwise, $N(R) \leq 1$ must hold. Consider now any root-to-leaf path $P$ in three $\tau(\mathcal{L})$, and assume that $R_1, R_2, \ldots, R_z$ is the sequence of unmarked clusters whose corresponding vertices appear on the path in this order. Then, for all $1 \leq i < z$, $N(R_{i+1}) \leq \left\lceil 2N(R_i)/2^{(\log m)^{3/4}} \right\rceil$, and so $z \leq O\left(\log^{3/4} m\right)$ must hold.

Next, we consider a cluster $R \in \mathcal{L} \setminus \{G\}$, whose corresponding vertex $v(R)$ in tree $\tau(\mathcal{L})$ is marked. Let $R'$ be the parent-cluster of $R$. Note that two cases are possible. The first case is that the algorithm from Theorem 7.8 was applied to the graph corresponding to $R'$, and it returned a type-1 legal clustering $\mathcal{R}$ with $R \in \mathcal{R}$. In this case, the theorem guarantees that $N(R) \leq \left\lfloor \left(1 - 1/2^{(\log m)^{3/4}}\right) N(R') \right\rfloor$. In the second case, there is a parent-cluster $R''$ of cluster $R'$, to which the algorithm from Theorem 7.8 was applied, and it returned a type-2 legal clustering $\mathcal{R}$, with $R' \in \mathcal{R}$, such that $R' = R^*$ is the distinguished cluster of the decomposition, and cluster $R$ lies in the type-1 legal clustering $\mathcal{R}'$ of the graph corresponding to cluster $R'$. In this latter case, from the definition of type-2 legal clustering, we are guaranteed that $N(R) \leq \left\lfloor \left(1 - 1/2^{(\log m)^{3/4}}\right) N(R'') \right\rfloor$. Therefore, if we consider any root-to-leaf path $P$ in the tree $\tau(\mathcal{L})$, and we let $R'_1, R'_2, \ldots, R'_y$ be the sequence of marked clusters whose corresponding vertices lie on $P$ in this order, then for all $1 < i < \lfloor y/2 \rfloor$, $N(R'_{2i}) \leq \left(1 - 1/2^{(\log m)^{3/4}}\right) N(R'_{2i-2})$. Since $N(G) \leq m$, we get that $y \leq O\left(2^{(\log m)^{3/4}} \cdot \log m\right) \leq 2^{O((\log m)^{3/4})}$.

## G.2 Proof of Claim 7.10

The proof is by induction on the distance from $v(R)$ to the root of the tree $\tau(\mathcal{L})$. Recall that, for $R = G$, we set $\mathcal{D}''(R)$ to assign probability 1 to an empty set of paths.

If $v(R)$ is the child vertex of $v(G)$, then consider the graph $G'$, that is obtained from $G$ by adding a new special vertex $v^*$ to it, that connects with an edge to some arbitrary vertex $v_0$. Recall that, when we applied the algorithm from Theorem 7.8 to this graph $G'$, it computed a legal clustering $\mathcal{R}$ of $G'$, with $R \in \mathcal{R}$, together with a distribution $\mathcal{D}'(R)$ over the sets of paths in $\Lambda'_{G'}(R)$, such that, for every edge $e \in E(G') \setminus E(R)$, $\mathbf{E}_{\mathcal{Q}'(R) \sim \mathcal{D}'(R)} \left[\mathrm{cong}_{G'}(\mathcal{Q}'(R), e)\right] \leq \beta$. Consider any external router $\mathcal{Q}'(R) \in \Lambda'_{G'}(R)$ that is assigned a non-zero probabilty. Let $u$ be the vertex at which every path of $\mathcal{Q}'(R)$ terminates. If $u \neq v^*$, then, since vertex $v^*$ has degree 1 in $G'$, no path in $\mathcal{Q}'(R)$ contains the vertex $v^*$, and so the paths of $\mathcal{Q}'(R)$ lie in $G$. Otherwise, by removing the last vertex from each path in $\mathcal{Q}'(R)$, we obtain a new set $\mathcal{Q}''(R)$ of paths in $\Lambda'_G(R)$, such that, for every edge $e \in E(G)$, $\mathrm{cong}_G(\mathcal{Q}''(R), e) \leq \mathrm{cong}_G(\mathcal{Q}'(R), e)$. Therefore, we can transform $\mathcal{D}'(R)$ into a distribution $\mathcal{D}''(R)$ over the family $\Lambda'_G(R)$ of external $R$-routers, such that, for every edge $e \in E(G) \setminus E(R)$, $\mathbf{E}_{\mathcal{Q}'(R) \sim \mathcal{D}''(R)} \left[\mathrm{cong}_G(\mathcal{Q}'(R), e)\right] \leq \beta$.

Next, we consider any cluster $R \in \mathcal{L}$, such that the distance from $v(R)$ to $v(G)$ in tree $\tau(\mathcal{L})$ is greater than 1. Let $R'$ be the parent-cluster of $R$, and let $G'$ be the graph obtained from $G$ by contracting all vertices of $V(G) \setminus V(R')$ into a special vertex $v^*$. Recall that, from Theorem 7.8, we have obtained a distribution $\mathcal{D}'(R)$ over external routers in $\Lambda'_{G'}(R)$, such that, for every edge $e \in E(G') \setminus E(R)$, $\mathbf{E}_{\mathcal{Q}'(R) \sim \mathcal{D}'(R)} \left[\mathrm{cong}_{G'}(\mathcal{Q}'(R), e)\right] \leq \beta$. Recall that, if $v(R)$ is a marked vertex, every router $\mathcal{Q}'(R) \in \Lambda'_{G'}(R)$, to which $\mathcal{D}'(R)$ assigns a non-zero probability, is careful with respect to $v^*$, that is, the paths in $\mathcal{Q}'(R)$ cause congestion at most 1 on every edge $e \in \delta_{G'}(v^*)$.

Let $i$ denote the number of unmarked vertices on the path connecting $v(R')$ to the root of $\tau(\mathcal{L})$. From the induction hypothesis, we have computed a distribution $\mathcal{D}''(R')$ over the external routers in $\Lambda'_G(R')$, such that, for every edge $e \in E(G) \setminus E(R')$, $\mathbf{E}_{\mathcal{Q}'(R')\sim\mathcal{D}''(R')}\left[\mathrm{cong}_G(\mathcal{Q}'(R'),e)\right] \leq \beta^{i+1}$.

We now compute the desired distribution a distribution $\mathcal{D}''(R)$ over the external routers in $\Lambda'_G(R)$, such that, for every edge $e \in E(G) \setminus E(R)$, $\mathbf{E}_{\mathcal{Q}'(R)\sim\mathcal{D}''(R)}\left[\mathrm{cong}_G(\mathcal{Q}'(R),e)\right] \leq \beta^{j+1}$, where $j = i$ if vertex $v(R)$ is marked, and $j = i+1$ otherwise. We provide the distribution implicitly, by providing an efficient algorithm for drawing a set $\tilde{\mathcal{Q}}'(R)$ of paths from the distribution. The algorithm for drawing an external router from the distribution $\mathcal{D}''(R)$ proceeds as follows.

First, the algorithm draws an external router $\mathcal{Q}'(R) \in \Lambda'_{G'}(R)$ for cluster $R$ in graph $G'$. If the paths in $\mathcal{Q}'(R)$ do not contain the vertex $v^*$, then this is the set of paths that we return. We now assume that at least one path in set $\mathcal{Q}'(R)$ contains vertex $v^*$. We denote by $u'$ the vertex of $G'$ that serves as the last vertex on every path in $\mathcal{Q}'(R)$.

Next, the algorithm draws a router $\mathcal{Q}'(R') \in \Lambda'_G(R')$ from the distribution $\mathcal{D}''(R')$ that we have constructed by the induction hypothesis. We denote by $u''$ the vertex of $G$ that serves as the last vertex on every path in $\mathcal{Q}'(R')$. The final set $\tilde{\mathcal{Q}}'(R)$ of paths that the algorithm returns is constructed by combining the sets $\mathcal{Q}'(R)$ and $\mathcal{Q}'(R')$ of paths, as follows.

We consider two cases. The first case is when $u' = v^*$. In this case, for every edge $e \in \delta_{G'}(R) = \delta_G(R)$, the unique path $Q(e) \in \mathcal{Q}'(R)$ that as $e$ as its first edge terminates at vertex $v^*$. We denote by $e'$ the last edge on path $Q(e)$. Note that edge $e'$, that is incident to vertex $v^*$ in graph $G'$, corresponds to an edge of $\delta_G(R')$ in graph $G$; we do not distinguish between the two edges. Therefore, there is some path $Q'(e) \in \mathcal{Q}'(R')$, whose first edge is $e'$, and last vertex is $u''$. By concatenating the paths $Q(e)$ and $Q'(e)$, we obtain path $Q^*(e)$ in graph $G$, connecting edge $e$ to vertex $u''$. We then let $\tilde{\mathcal{Q}}'(R) = \{Q^*(e) \mid e \in \delta_G(R)\}$ be the final set of paths that the algorithm outputs.

The second case is when $u' \neq v^*$. In this case, some paths in $\mathcal{Q}'(R)$ may contain vertex $v^*$ as an inner vertex. Consider any path $Q \in \mathcal{Q}'(R)$ that contains the vertex $v^*$, and let $e \in \delta_{G'}(v^*)$ be any edge that is incident to $v^*$ that the path contains. Recall that $e \in \delta_G(R)$, and so there is some path $Q'(e) \in \mathcal{Q}'(R')$, whose first edge is $e$, and last vertex is $u''$. We replace edge $e$ with path $Q'(e)$ on path $Q$. Note that originally path $Q$ must have contained two edges that are incident to $v^*$; denote them by $e$ and $e'$. We have replaced edge $e$ with a path connecting $e$ to vertex $u''$ in graph $G$, and we replace edge $e'$ with a path connecting $e'$ to $u''$ in graph $G$, but we reverse the direction of the path. In this way, we can glue the two paths to each other via the vertex $u''$. Once every path of $\mathcal{Q}'(R)$ containing $v^*$ is processed in this manner, we obtain the final set $\tilde{\mathcal{Q}}'(R)$ of paths in graph $G$.

This completes the definition of the distribution $\mathcal{D}''(R)$ over the set $\Lambda'_G(R)$ of external routers for $R$ in $G$. It now remains to analyze the expected congestion on each edge of $E(G) \setminus E(R)$. Fix any edge $e \in E(G) \setminus E(R)$. First, if edge $e$ lies in graph $R' \cup \delta_G(R')$, then $\mathrm{cong}_G(\tilde{\mathcal{Q}}'(R),e) = \mathrm{cong}_{G'}(\mathcal{Q}'(R),e)$, and so, from the definition of helpful clustering, $\mathbf{E}_{\tilde{\mathcal{Q}}'(R)\sim\mathcal{D}''(R)}\left[\mathrm{cong}_G(\tilde{\mathcal{Q}}'(R),e)\right] \leq \mathbf{E}_{\mathcal{Q}'(R)\sim\mathcal{D}'(R)}\left[\mathrm{cong}_{G'}(\mathcal{Q}'(R),e)\right] \leq \beta$.

Assume now that $e \in E(G) \setminus (E(R') \cup \delta_G(R'))$. Note that, if the set $\mathcal{Q}'(R) \in \Lambda'_{G'}(R)$ that was drawn from distribution $\mathcal{D}'(R)$ is careful with respect to vertex $v^*$, then every edge of $\delta_{G'}(v^*)$ may lie on at most one path of $\mathcal{Q}'(R) \in \Lambda'_{G'}(R)$, and so every path in the set $\mathcal{Q}'(R')$ that was drawn from distribution $\mathcal{D}''(R') \in \Lambda'_G(R')$ is used by at most one path in $\tilde{\mathcal{Q}}'(R)$. Therefore: $\mathbf{E}_{\tilde{\mathcal{Q}}'(R)\sim\mathcal{D}''(R)}\left[\mathrm{cong}_G(\tilde{\mathcal{Q}}'(R),e)\right] \leq \mathbf{E}_{\mathcal{Q}'(R')\sim\mathcal{D}''(R')}\left[\mathrm{cong}_G(\mathcal{Q}'(R'),e)\right] \leq \beta^{i+1}$ in this case.

Recall that, if vertex $v(R)$ is marked, then every router $\mathcal{Q}'(R)$ that has non-zero probability to be drawn from distribution $\mathcal{D}'(R)$ is careful with respect to $v^*$, while the number of unmarked vertices on the unique path connecting $v(R)$ to $v(G)$ in tree $\tau(\mathcal{L})$ is $i$.

It remains to consider the case where vertex $v(R)$ is unmarked. Consider the following two-step

process for drawing a router $\tilde{\mathcal{Q}}'(R) \in \Lambda_G(R)$ from the distribution $\mathcal{D}''(R)$, that is equivalent to the one described above. In the first step, we select a router $\mathcal{Q}'(R') \in \Lambda_G(R')$ from the distribution $\mathcal{D}''(R')$. Then, in the second step, we select a router $\mathcal{Q}'(R) \in \Lambda_{G'}(R)$ from distribution $\mathcal{D}'(R)$. Lastly, composing the two sets of paths as described above, we obain the final router $\tilde{\mathcal{Q}}'(R)$.

Fix an edge $e \in E(G) \setminus (E(R') \cup \delta_G(R'))$, and assume that the set of paths $\mathcal{Q}'(R') \in \Lambda_G(R')$ that was chosen from the distribution $\mathcal{D}''(R')$ causes congestion $z$ on edge $e$. We denote $\mathcal{Q}'(R') = \{Q(e') \mid e' \in \delta_G(R')\}$, where path $Q(e')$ originates at edge $e'$ and terminates at vertex $u''$. Let $E' \subseteq \delta_G(R')$ be the set of all edges $e'$, whose corresponding path $Q(e')$ contains the edge $e$, so $|E'| = z$. Denoting $E' = \{e_1, \ldots, e_z\}$, and assuming that the path set $\mathcal{Q}'(R')$ is fixed, we can now write:

$$\mathbf{E}_{\mathcal{Q}'(R) \sim \mathcal{D}'(R)} \left[ \mathrm{cong}_G(\tilde{\mathcal{Q}}'(R), e) \right] = \mathbf{E}_{\mathcal{Q}'(R) \sim \mathcal{D}'(R)} \left[ \sum_{i=1}^{z} \mathrm{cong}_{G'}(\mathcal{Q}'(R), e_i) \right] \leq \beta z.$$

Recall that $z$ is the congestion caused by the set $\mathcal{Q}'(R')$ of paths on edge $e$. Therefore, overall:

$$\mathbf{E}_{\tilde{\mathcal{Q}}'(R) \sim \mathcal{D}''(R)} \left[ \mathrm{cong}_G(\tilde{\mathcal{Q}}'(R), e) \right] \leq \beta \cdot \mathbf{E}_{\mathcal{Q}'(R') \sim \mathcal{D}''(R')} \left[ \mathrm{cong}_G(\mathcal{Q}'(R'), e) \right] \leq \beta^{i+2},$$

from the induction hypothesis. Since, in the case that $v(R)$ is unmarked, the number of unmarked vertices on the path connecting $v(R)$ to $v(G)$ in tree $\tau(\mathcal{L})$ is $i+1$, this completes the proof of the claim.

### G.3 Proof of Claim 7.11

Consider some cluster $R \in \mathcal{L}$. Let $\mathcal{R}$ be the set of child-clusters of $R$. Consider the graph $\tilde{R} = R \setminus \left(\bigcup_{R' \in \mathcal{R}} R'\right)$. Note that, from the definition of the laminar family, every edge of $E(G)$ may lie in at most one graph in the collection $\left\{\tilde{R} \mid R \in \mathcal{L}\right\}$.

Observe that collection $\mathcal{I}_1$ of instances can be defined as $\mathcal{I}_1 = \{I(R) \mid R \in \mathcal{L}\}$, where $I(R) = (G(R), \Sigma(R))$ is the instance associated with cluster $R$. Graph $G(R)$ is obtained from graph $G$, by first contracting the vertices of $V(G) \setminus V(R)$ into a supernode $v^*$, and then contracting each child cluster $R'$ of $R$ into a supernode $v(R')$. We partition the set of edges of $G(R)$ into two subsets: the first subset, that we call *internal edges*, and denote by $E_1(R)$, is the edge set $E(\tilde{R})$. The second subset, that we call *external edges*, and denote by $E_2(R)$, is the set of all edges that are incident to the supernodes of $G(R)$. From the above discussion, $\sum_{R \in \mathcal{L}} |E_1(R)| \leq |E(G)|$, as every edge may serve as an internal edge for at most one graph $G(R)$. It now remains to bound the total number of external edges in all graphs in $\{G(R) \mid R \in \mathcal{L}\}$.

For all $1 \leq i \leq \mathsf{dep}(\mathcal{L})$, we denote by $\mathcal{L}_i \subseteq \mathcal{L}$ the set of all clusters $R \in \mathcal{L}$, such that vertex $v(R)$ lies at distance exactly $i$ from the root of the tree $\tau(\mathcal{L})$. Note that for each cluster $R \in \mathcal{L}_i$, every external edge $e \in E_2(R)$ corresponds to some edge of the original graph $G$ that has at least one endpoint in cluster $R$. Moreover, since every basic cluster $C \in \mathcal{C}$ is either contained in $R$ or is disjoint from $R$, each such edge must lie in $E^{\mathsf{out}}(\mathcal{C})$. Since the clusters in set $\mathcal{L}_i$ are disjoint from each other, we get that $\sum_{R \in \mathcal{L}_i} |E_2(R)| \leq \sum_{C \in \mathcal{C}} |\delta_G(C)| \leq |E(G)|/\mu^{0.1}$, from the statement of Theorem 7.3.

Since, from Claim 7.9, $\mathsf{dep}(\mathcal{L}) \leq 2^{O((\log m)^{3/4})}$, while $\mu \geq 2^{c^*(\log m)^{7/8} \log\log m}$ for a large enough constant $c^*$, we get that, overall:

$$\sum_{R \in \mathcal{L}} |E_2(R)| \leq \mathsf{dep}(\mathcal{L}) \cdot \frac{|E(G)|}{\mu^{0.1}} \leq \frac{2^{O((\log m)^{3/4})} \cdot |E(G)|}{2^{0.1 \cdot c^* (\log m)^{7/8} \log\log m}} \leq |E(G)|.$$

## G.4 Proof of Observation 7.12

Let $u'$ be the parent-vertex of $u$ in $\tau$. Denote $U = V(\tau_u)$, and $U' = V(\hat{H}) \setminus U$. Recall that we have denoted by $S$ the cluster of $H$ that is defined by vertex set $U \subseteq V(\hat{H})$.

Let $\mathcal{R}' \subseteq \mathcal{R}$ contain all clusters $R$ with $v_R \in U$. Notice that, from Theorem 4.9, $(U, U')$ is the minimum cut in graph $\hat{H}$ separating $u$ from $u'$. Let $E' = E_{\hat{H}}(U, U')$. Observe that, equivalently, $E' = \delta_H(S)$. From the properties of minimum cut, there is a set $\mathcal{P}$ of edge-disjoint paths in graph $\hat{H}$, routing the edges of $E' = \delta_{\hat{H}}(U)$ to vertex $u$, such that all internal vertices on every path of $\mathcal{P}$ lie in $U$. Similarly, there is a set $\mathcal{P}'$ of edge-disjoint paths in graph $\hat{H}$, routing the edges of $E' = \delta_{\hat{H}}(U')$ to vertex $u'$, such that all internal vertices on every path of $\mathcal{P}'$ lie in $U'$.

The existence of the set $\mathcal{P}$ of paths in $\hat{H}[U]$ immediately implies that cluster $\hat{H}[U]$ has the 1-bandwidth property in graph $\hat{H}$. Since graph $\hat{H}[U]$ is precisely the contracted graph of $S$ with respect to cluster set $\mathcal{R}'$, that is, $\hat{H}[U] = S_{|\mathcal{R}'}$, and since every cluster in $\mathcal{R}'$ has the $\alpha$-bandwidth property, from Corollary 4.40, cluster $S$ has the $\alpha$-bandwidth property in graph $H$.

Next, we show an algorithm to construct the desired distribution $\mathcal{D}'(S)$ over the external routers in $\Lambda'_H(S)$. We start with the set $\mathcal{P}'$ of paths routing the edges of $E'$ to vertex $u'$ in graph $\hat{H}[U']$.

Assume first that $u'$ is not a supernode. Let $\hat{H}'$ be the graph obtained as follows: we first subdivide every edge $e \in E'$ with a terminal vertex $t_e$, and we let $T = \{t_e \mid e \in E'\}$ be the resulting set of terminals. We then let $\hat{H}'$ be the subgraph of the resulting graph induced by vertex set $U' \cup T$. Let $\mathcal{R}'' \subseteq \mathcal{R}$ be the set of all clusters $R$ with $v_R \in U'$. We apply the algorithm from Claim 4.41 to graph $\hat{H}'$, cluster set $\mathcal{R}''$, and set $\mathcal{P}'$ of paths, to obtain a set $\mathcal{P}''$ of paths in graph $H$, such that, for every edge $e \in \bigcup_{R \in \mathcal{R}''} E(R)$, the paths of $\mathcal{P}''$ cause congestion at most $\lceil 1/\alpha \rceil$, and for every edge $e \in E(H \setminus U) \setminus \left(\bigcup_{R \in \mathcal{R}''} E(R)\right)$, the paths of $\mathcal{P}''$ cause congestion at most 1. We are also guaranteed that the paths in $\mathcal{P}''$ route the edges of $\delta_H(S)$ to vertex $u'$, and all internal vertices on every path in $\mathcal{P}''$ are disjoint from $U$. Lastly, since the edges incident to the special vertex $v^*$ do not lie in the clusters of $\mathcal{R}''$, the set $\mathcal{P}''$ of paths is careful with respect to $v^*$. From the above discussion, $\mathcal{P}'' \in \Lambda'_H(S)$. We then let distribution $\mathcal{D}'(S)$ assign probability 1 to the set $\mathcal{P}''$ of paths.

Assume now that vertex $u'$ is a supernode, and $u' = v_R$ for some cluster $R \in \mathcal{R}$. We repeat the algorithm from above, except that, when we apply the algorithm from Claim 4.41 to graph $\hat{H}'$, we use cluster set $\mathcal{R}'' \setminus \{R\}$ instead of $\mathcal{R}''$. The resulting set of paths $\mathcal{P}''$ then routes the edges of $E'$ to the edges of $\delta_G(R)$, and they remain internally disjoint from cluster $S$. As before, the set $\mathcal{P}''$ of edges is careful with respect to $v^*$, and it causes edge-congestion at most $\lceil 1/\alpha \rceil$. Moreover, every edge of $\delta_G(R)$ participates in at most one path in $\mathcal{P}''$. We then use the algorithm from Lemma 4.27 in order to compute a distribution $\mathcal{D}(R)$ over the internal $R$-routers in $\Lambda_H(R)$, such that, for every edge $e \in E(R)$, $\mathbf{E}_{\mathcal{Q} \in \mathcal{D}(R)}\left[\mathrm{cong}(\mathcal{Q}, e)\right] \leq O(\log^4 m/\alpha)$ (in order to use the lemma we subdivide every edge in $\delta_G(R)$ with a terminal, and apply the lemma to the augmented cluster $R^+$ together with the resulting set of terminals). In order to define the distribution $\mathcal{D}'(S)$, we first select a set $\mathcal{Q} \in \Lambda_G(R)$ of paths from the distribution $\mathcal{D}(R)$, and then concatenate the paths in $\mathcal{P}''$ with the paths in $\mathcal{Q}$. It is immediate to verify that the resulting distribution $\mathcal{D}'(S)$ is supported over the external $S$-routers in $\Lambda'_H(S)$, which are careful with respect to $v^*$, and that $\mathbf{E}_{\mathcal{Q}'(S) \sim \mathcal{D}'(S)}\left[\mathrm{cong}_H(\mathcal{Q}'(S), e)\right] \leq O(\log^4 m/\alpha)$.

## G.5 Proof of Observation 7.17

Consider some cluster $J \in \mathcal{J}$, and let $u_0$ be the center node of cluster $J$. It is enough to show that there is a set $\mathcal{P}$ of paths (internal $J$-router) in graph $\hat{H}'$, routing all edges of $\delta_{\hat{H}'}(J)$ to vertex $u_0$, so that every inner vertex on every path in $\mathcal{P}$ lies in $J$, and the congestion of $\mathcal{P}$ is at most $O(\log m)$. Let $\mathcal{P}^*$ be the set of all paths $P$ in graph $\hat{H}'$, such that the first edge on $P$ lies in $\delta_{\hat{H}'}(J)$, the last vertex of $P$ is $u_0$, and all inner vertices of $P$ lie in $J$.

Recall that a flow $f$ defined over a set $\mathcal{P}'$ of (directed) paths is an assigment of a flow value $f(P)$ to every path $P \in \mathcal{P}'$. Given such a flow $f$, we say that an edge $e$ sends one flow unit iff the total amount of flow $f(P)$, for all paths $P \in \mathcal{P}'$ that originate at edge $e$, is 1. The congestion caused by flow $f$ is the maximum, over all edges $e'$, of $\sum_{\substack{P \in \mathcal{P}': \\ e \in P}} f(P)$.

We show below that there is a flow $f$, defined over the set $\mathcal{P}^*$ of paths, in which every edge $e \in \delta_{H'}(J)$ sends one flow unit, and the flow causes congestion at most $O(\log m)$. From the integrality of flow, it then follows that that there is a set $\mathcal{P}$ of paths routing the edges of $\delta_{\hat{H}'}(J)$ to vertex $u_0$, so that for every path of $\mathcal{P}$, every inner vertex on the path lies in $J$, and the congestion of $\mathcal{P}$ is at most $O(\log m)$, and so cluster $J$ has $\Omega(1/\log m)$-bandwidth property. From now on we focus on defining the flow $f$.

For $1 \leq i \leq h$, let $L'_i = L_i \cap V(J)$, and let $J_i$ be the subgraph of $J$ induced by vertex set $\{u_0\} \cup L'_1 \cup \cdots \cup L'_i$. We also let $J_0$ be the graph that consists of a single vertex – vertex $u_0$. For all $0 \leq i \leq h$, we denote $E_i = \delta_{\hat{H}'}(J_i)$, and we denote by $\mathcal{P}^*_i$ the set of all paths $P$ in graph $\hat{H}'$, such that the first edge of $P$ lies in $E_i$, the last vertex of $P$ is $u_0$, and all inner vertices of $P$ are contained in $J_i$. Additionally, we let $\tilde{E}_i \subseteq E_i$ be the set of all the edges $e \in E_i$, such that, for some vertex $v \in V(J_i) \setminus \{u_0\}$, $e \in \delta^{\text{up}}(v)$. We let $\tilde{\mathcal{P}}^*_i \subseteq \mathcal{P}^*_i$ be the set of all paths whose first edge lies in $\tilde{E}_i$. Note that any flow $f_i$ defined over the set $\mathcal{P}^*_i$ of paths immediately defines a flow $f'_i$ over the set $\tilde{\mathcal{P}}^*_i$ of paths, by setting, for every path $P \in \tilde{\mathcal{P}}^*_i$, $f'(P) = f(P)$, and setting the flow on all other paths to 0. We call $f'_i$ the *restriction of flow $f_i$ to the set $\tilde{\mathcal{P}}^*_i$ of paths*. We prove the following claim.

**Claim G.1** *For all $1 \leq i \leq h$, there is a flow $f_i$, defined over the set $\mathcal{P}^*_i$ of paths, in which every edge of $E_i$ sends one flow unit, and the total congestion is bounded by $2^{512} \cdot i$. Moreover, if we let $f'_i$ be the restriction of $f_i$ to the set $\tilde{\mathcal{P}}^*_i$ of paths, then the congestion caused by $f'_i$ is at most $\left(1 + \frac{256}{\log m}\right)^i$.*

Notice that the proof of Observation 7.17 immediately follows from Claim G.1, since $J_h = J$, and flow $f_h$ defines the desired flow in graph $J$, that causes congestion at most $O(h) \leq O(\log m)$. It now remains to prove Claim G.1.

**Proof of Claim G.1.** The proof is by induction on $i$. The base is when $i = 0$. In this case, $\delta_{\hat{H}'}(J_0) = \delta_{\hat{H}'}(u_0)$. The set $\mathcal{P}^*_0$ of paths contains, for every edge $e \in \delta_{\hat{H}'}(u_0)$, a path $P(e)$ that only consists of the edge $e$ itself. We obtain flow $f_0$ by sending one flow unit on each such path $P(e)$. Note that $\tilde{E}_0 = \emptyset$ in this case, and the resulting flow has congestion 1.

Assume now that the claim holds for some $0 \leq i < h$. We now prove it for $i + 1$.

We partition the set $E_{i+1}$ of edges into two subsets. The first subset, $E'_{i+1}$ is $E_{i+1} \cap E_i$. The second subset, $E''_{i+1}$ contains all remaining edges of $E_{i+1}$. It is easy to verify that, for every edge $e \in E''_{i+1}$, there is some vertex $v \in L'_{i+1}$, with $e \in \delta_{\hat{H}'}(v) \setminus E_i$.

For every edge $e \in E'_{i+1}$, the flow on the paths that originate from $e$ remains unchanged from $f_i$. In other words, for every path $P \in \mathcal{P}^*_i$ that starts with edge $e \in E'_{i+1}$ (and hence $P \in \mathcal{P}^*_{i+1}$), we set $f_{i+1}(P) = f_i(P)$.

Consider now some vertex $v \in L'_i$. Note that, when vertex $v$ was added to cluster $J$, the number of edges connecting $v$ to vertices that belonged to $J$ at that time was at least $|\delta_{\hat{H}'}(v)|/128$. From the definition of layered well-linked decomposition, $|\delta^{\text{up}}(v)| < |\delta^{\text{down}}(v)|/\log m \leq |\delta_{\hat{H}'}(v)|/\log m$. Therefore, at the time when $v$ was added to $J$, there were at least $|\delta_{\hat{H}'}(v)|/256$ edges in $\delta^{\text{down}}(v)$, that connected $v$ to the vertices of $J$. It is immediate to verify that each such edge must lie in $E_i$, and moreover, it must lie in $\tilde{E}_i$. To conclude, there is a set $E'(v) \subseteq \delta^{\text{down}}(v)$ of at least $|\delta_{\hat{H}'}(v)|/256$ edges, that lie in $\tilde{E}_i$. Note that none of these edges may lie in $E_{i+1}$. We now define the flow $f_{i+1}$ that originates at the edges of $\delta_{\hat{H}'}(v) \cap E''_{i+1}$.

Every edge $e \in \delta_{\hat{H}'}(v) \setminus E'(v)$ that lies in $E''_{i+1}$, spreads one unit of flow evenly among the edges of $E'(v)$, and then uses the flow that each of these edges sends in $f_i$, in order to reach $u_0$. In other words,

for every edge $e \in \delta_{\hat{H}'}(v) \cap E''_{i+1}$, for every edge $e' \in E'(v)$, and for every path $P \in \mathcal{P}^*_i$ whose first edge is $e'$, we consider the path $P' \in \mathcal{P}^*_{i+1}$, that is obtained by appending the edge $e$ at the beginning of path $P$, and we set $f_{i+1}(P') = f_i(P)/|E'(v)|$. Since each edge $e \in E'(v)$ sends one flow unit in $f_i$, each edge $e \in \delta_{\hat{H}'}(v) \cap E''_{i+1}$ now sends one flow unit in $f_{i+1}$. This completes the definition of the flow $f_{i+1}$. We now analyze the congestion caused by this flow.

First, the flow on the paths originating at the edges of $E'_{i+1}$ remains unchanged, and causes congestion at most $2^{512}i$. Next, we consider on flow on paths originating at edges of $E''_{i+1}$. Consider again some vertex $v \in L'_{i+1}$, and recall that $|E'(v)| \geq |\delta_{\hat{H}'}(v)|/256$. Observe that edges of $E'(v)$ must lie in edge set $\tilde{E}_i$. Since every edge $e \in \delta_{\hat{H}'}(v) \cap E''_{i+1}$ spreads one unit of flow evenly among the edges of $E'(v)$, each edge of $E'(v)$ is responsible for sending at most $\frac{|\delta_{\hat{H}'}(v)|}{|E'(v)|} \leq 256$ flow units. In other words, for each edge $e \in E'(v)$, the flow originating at $e$ in $f_i$ is scaled by at most factor 256 in order to obtain flow $f_{i+1}$. Therefore, the flow $f_{i+1}$ originating at edges of $E''_{i+1}$ causes congestion at most $256 \cdot \left(1 + \frac{256}{\log m}\right)^i$. Overall, flow $f_i$ causes congestion at most $2^{512}i + 256 \cdot \left(1 + \frac{256}{\log m}\right)^i \leq 2^{512} \cdot (i+1)$, since $i + 1 \leq h \leq \log m$.

Lastly, we bound the congestion of the flow $f'_{i+1}$, which is the restriction of the flow $f_{i+1}$ to the set $\tilde{\mathcal{P}}^*_i$ of paths. We partition the edges of $\tilde{E}_{i+1}$ into two subsets: $\tilde{E}'_{i+1} = \tilde{E}_i \cap \tilde{E}_{i+1}$, and $\tilde{E}''_{i+1}$ containing all remaining edges. Observe that for each edge $e \in \tilde{E}''_{i+1}$, there is some vertex $v \in L'_{i+1}$ with $e \in \delta^{\text{up}}(v)$. The flow $f'_{i+1}$ that originates at the edges of $\tilde{E}'_{i+1}$ remains unchanged from $f'_i$, and causes congestion at most $\left(1 + \frac{256}{\log m}\right)^i$. In order to bound the congestion caused by the flow $f'_{i+1}$ originating from edges of $\tilde{E}'_{i+1}$, consider some vertex $v \in L'_{i+1}$. Recall that $|E'(v)| \geq |\delta_{\hat{H}'}(v)|/256$, while $|\delta^{\text{up}}(v)| < |\delta_{\hat{H}'}(v)|/\log m \leq 256|E'(v)|/\log m$. Since every edge $e \in \delta^{\text{up}}(v) \setminus E'(v)$ spreads one unit of flow evenly among the edges of $E'(v)$, each edge of $E'(v)$ is responsible for sending at most $\frac{|\delta^{\text{up}}(v)|}{|E'(v)|} \leq \frac{256}{\log m}$ flow units. In other words, for each edge $e \in E'(v)$, the flow originating at $e$ in $f'_i$ is scaled by at most factor $256/\log m$ in order to obtain flow $f'_{i+1}$. Therefore, the flow $f'_{i+1}$ originating at edges of $\tilde{E}''_{i+1}$ causes congestion at most $\frac{256}{\log m} \cdot \left(1 + \frac{256}{\log m}\right)^i$. Overall, flow $f'_{i+1}$ causes congestion at most $\left(1 + \frac{256}{\log m}\right)^i + \frac{256}{\log m} \cdot \left(1 + \frac{256}{\log m}\right)^i \leq \left(1 + \frac{256}{\log m}\right)^{i+1}$. □

## G.6 Proof of Claim 7.21

We denote by $E' = \delta_{\check{H}}(S)$. We define a cluster $S'$ in graph $G$, corresponding to cluster $S$, as usual: First, we define vertex set $V(S')$, and then we let $S'$ be the subgraph of $G$ induced by $V(S')$. Vertex set $V(S')$ contains every regular vertex of $S$. Additionally, for every $R$-node $v_R \in S$, it contains all vertices of $R$, and for every $J$-node $v_{J'} \in S$, it contains all vertices of the cluster $J' \in \mathcal{J}'$. We start with the following simple observation.

**Observation G.2** *There is an efficient algorithm to compute a distribution $\mathcal{D}'(S')$ over the set $\Lambda'_G(S')$ of external $S'$-routers in $G$, such that, for every edge $e \in E(G) \setminus E(S')$, $\mathbf{E}_{\mathcal{Q}'(S') \sim \mathcal{D}'(S')}\left[\text{cong}(\mathcal{Q}'(S'), e)\right] \leq O(\log^{14.5} m)$.*

**Proof:** Recall that every cluster $R \in \mathcal{R}$ has the $\alpha_1 = 1/\log^6 m$-bandwidth property in graph $G$, and every cluster $J' \in \mathcal{J}'$ has the $\Omega(1/\log^{9.5} m)$-bandwidth property in $G$. Let $x$ be the vertex of graph $\check{H}$ that serves as the last vertex on every path of $\mathcal{P}(S)$. Assume first that $x$ is a regular vertex, that is, $x \in V(G)$. Then we can use the algorithm from Claim 4.41 (by first subdividing every edge of $E'$ with a terminal) to obtain a collection $\mathcal{P}(S')$ of paths in graph $G$, that is an external $S'$-router, such

that the paths in $\mathcal{P}(S')$ cause congestion $O\left(\log^{10.5} m\right)$. We define a distribution $\mathcal{D}'(S')$ over the set $\Lambda'_G(S')$ of external $S'$-routers, that assigns probability 1 to the router $\mathcal{P}(S')$.

Assume now that $x$ is a supernode, that corresponds to some cluster $A \in \mathcal{R}'' \cup \mathcal{J}'$. Again, applying the algorithm from Claim 4.41 (but this time replacing the graph $G$ with the graph that is obtained from $G$ by contracting cluster $A$ into a supernode $x$), we obtain a collection $\mathcal{P}(S')$ of paths in graph $G$, routing the edges of $E' = \delta_G(S')$ to the edges of $\delta_G(A)$, with congestion $O(\log^{10.5} m)$, such that the paths in $\mathcal{P}(S')$ are internally disjoint from both $S'$ and $A$. Moreover, from Claim 4.41 the paths in $\mathcal{P}(S')$ cause congestion at most $\beta' = O(\log m)$ on the edges of $\delta_G(A)$. Recall that cluster $A$ must have the $\Omega(1/\log^{9.5} m)$-bandwidth property in $G$. By applying the algorithm from Lemma 4.27 to cluster $A$, we obtain a distribution $\mathcal{D}(A)$ over the set $\Lambda_G(A)$ of internal $A$-routers, such that, for every edge $e \in E(A)$, $\mathbf{E}_{\mathcal{Q}(A)\sim\mathcal{D}(A)}\left[\mathrm{cong}(\mathcal{Q}(A), e)\right] \leq O((\log^4 m) \cdot (\log^{9.5} m)) = O(\log^{13.5} m)$. We now define the distribution $\mathcal{D}'(S')$ over the set $\Lambda'_G(S')$ of external $S'$-routers, by providing an algorithm to draw a set of paths from the distribution. In order to do so, we first choose a set $\mathcal{Q}(A) \in \Lambda_G(A)$ of paths from the distribution $\mathcal{D}(A)$. Denote $\mathcal{Q}(A) = \{Q(e) \mid e \in \delta_G(A)\}$, where path $Q(e)$ has $e$ as its first edge. Let $\mathcal{Q}'$ be a multi-set of paths, in which, for every edge $e \in \Lambda_G(A)$, the path $Q(e)$ is included $\mathrm{cong}_G(\mathcal{P}(S'), e) \leq O(\log m)$ times. We then concatenate the paths in $\mathcal{P}(S')$ with the paths in $\mathcal{Q}'$, obtaining an external $S'$-router $\mathcal{Q}'(S') \in \Lambda_G(S')$. From the above discussion, it is immediate to verify that, for every edge $e \in E(G) \setminus E(S')$, $\mathbf{E}_{\mathcal{Q}'(S')\sim\mathcal{D}'(S')}\left[\mathrm{cong}(\mathcal{Q}'(S'), e)\right] \leq O(\log^{14.5} m)$. □

We now consider three cases, depending on whether cluster $S$ contains any $J$-node, and whether the cluster $J' \in \mathcal{J}$ corresponding to the $J$-node has a cluster of $\mathcal{C}$ or of $\mathcal{W}'$ as its center cluster.

**Case 1.** The first case happens if there is at least one $J$-node $v_{J'} \in S$, such that the center-cluster of the cluster $J' \in \mathcal{J}$ is a cluster of $\mathcal{C}$, that we denote by $C^*$ (see Figure 44). Let $\mathcal{C}^* \subseteq \mathcal{C}$ be the set of all clusters $C \in \mathcal{C}$ with $C \subseteq S'$, and let $\mathcal{R}^* \subseteq \mathcal{R}$ be the set of all clusters $R \in \mathcal{R}$ with $R \subseteq S'$. Observe that cluster $C^*$ may not be contained in any cluster of $\mathcal{R}$, from the definition of cluster set $\mathcal{J}$. We will modify the set $\mathcal{R}$ of clusters, by deleting the clusters of $\mathcal{R}^*$ from it, and adding a new set $\mathcal{R}^{**}$ of clusters instead, so that the resulting cluster set $\tilde{\mathcal{R}} = (\mathcal{R} \setminus \mathcal{R}^*) \cup \mathcal{R}^{**}$ is a helpful clustering that is better than $\mathcal{R}$.

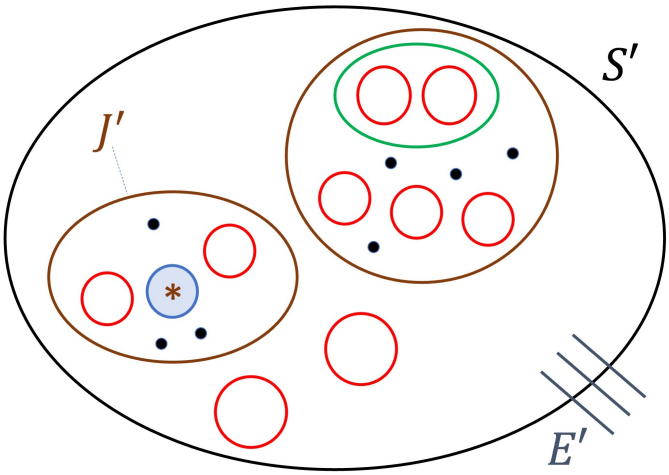



Figure 44: Case 1: $S'$ contains at least one $J'$-cluster (shown in brown) whose center cluster $C^* \in \mathcal{C}$ is marked with $*$. Clusters of $\mathcal{R}^*$ are shown in red.

In order to define the new set $\mathcal{R}^{**}$ of clusters, we start with the augmented cluster $X = (S')^+$; that is, we subdivide every edge $e \in E'$ in graph $G$ with a terminal $t_e$, and we let $T = \{t_e \mid e \in E'\}$ be the resulting set of terminals. We then let $X$ be the subgraph of the resulting graph induced by $T \cup V(S')$.

Next, we obtain a graph $X'$ from $X$, by contracting every basic cluster $C \in \mathcal{C}^*$ into a supernode $v_C$, so $X' = X_{|\mathcal{C}^*}$. We apply the algorithm from Theorem 4.19 to graph $X'$, its cluster $X' \setminus T$, and parameter $\alpha_0 = 1/\log^3 m$ (so the requirement that $\alpha_0 < \min\left\{\frac{1}{64\beta_{\mathrm{ARV}}(m)\cdot\log m}, \frac{1}{48\log^2 m}\right\}$ is satisfied). The algorithm computes a collection $\mathcal{Y}$ of clusters of $X' \setminus T$ (the well-linked decomosition), such that the vertex sets $\{V(Y)\}_{Y\in\mathcal{Y}}$ partition $V(X') \setminus T$, and every cluster $Y \in \mathcal{T}$ has the $\alpha_0$-bandwidth property, with $|\delta_{X'}(Y)| \leq |T| = |E'|$. We are also guaranteed that:

$$\sum_{Y\in\mathcal{Y}} |\delta_{X'}(Y)| \leq |T| \cdot \left(1 + O(\alpha_0 \cdot \log^{1.5} m)\right) = |E'| \cdot \left(1 + O(\alpha_0 \cdot \log^{1.5} m)\right). \tag{27}$$

Recall that, additionally, the algorithm computes, for every cluster $Y \in \mathcal{Y}$, a set $\mathcal{P}(Y) = \{P(e) \mid e \in \delta_{X'}(Y)\}$ of paths, such that, for every edge $e \in \delta_{X'}(Y)$, path $P(e)$ has $e$ as its first edge, and some terminal of $T$ as its last vertex, with all inner vertices of $P(e)$ lying in $V(X') \setminus V(Y)$. We are also guaranteed that, for every cluster $Y \in \mathcal{Y}$, the set $\mathcal{P}(Y)$ of paths causes edge-congestion at most 100 in $X'$.

We now define the set $\mathcal{R}^{**}$ of clusters, that will be added to $\mathcal{R}$ instead of the clusters of $\mathcal{R}^*$. For every cluster $Y \in \mathcal{Y}$, we let $R_Y$ be the cluster of graph $G$ corresponding to cluster $Y$. Intuitively, $R_Y$ is obtained from $Y$ by uncontracting all basic clusters whose corresponding supernode lies in $Y$. Formally, we denote by $\mathcal{C}(Y) \subseteq \mathcal{C}$ the set of all basic clusters $C$ whose corresponding supernode $v_C$ lies in $Y$. We define vertex set $V(R_Y)$ to contain all regular vertices lying in $Y$ (that is, vertices of $V(Y) \cap V(G)$), and all vertices of $\bigcup_{C\in\mathcal{C}(Y)} V(C)$. We then let $R_Y$ be the subgraph of $G$ induced by $V(R_Y)$. Note that $Y = (R_Y)_{|\mathcal{C}(Y)}$. Since every cluster in $\mathcal{C}$ has the $\alpha_0$-bandwidth property, and every cluster $Y \in \mathcal{Y}$ has the $\alpha_0$-bandwidth property, from Corollary 4.40, cluster $R_Y$ has the $\alpha_0^2 = \alpha_1$-bandwidth property. Consider now the set $\mathcal{P}(Y)$ of paths in graph $X'$, routing the edges of $\delta_{X'}(Y)$ to the vertices of $T$, with congestion at most 100, and recall that the paths of $\mathcal{P}(Y)$ are internally disjoint from $Y$. Since $Y = (R_Y)_{|\mathcal{C}(Y)}$, and every cluster in $\mathcal{C}$ has the $\alpha_0$-bandwidth property, we can use the algorithm from Claim 4.41, to obtain a collection $\mathcal{P}(R_Y)$ of paths in graph $G$, routing the edges of $\delta_G(R_Y)$ to the edges of $E'$, with congestion at most $200/\alpha_0$, such that the paths in $\mathcal{P}(R_Y)$ are internally disjoint from $R_Y$, and they cause congestion at most 100 on edges of $E'$. We are now ready to define a distribution $\mathcal{D}'(R_Y)$ over the set $\Lambda'_G(R_Y)$ of external $R_Y$-routers in $G$, by providing an algorithm to draw an external router $\mathcal{Q}'(R_Y)$ from the distribution. In order to draw a set $\mathcal{Q}'(R_Y)$ of paths from distribution $\mathcal{D}'(R_Y)$, we start by drawing an external $S'$-router $\mathcal{Q}'(S') \in \Lambda_G(S')$ from the distribution $\mathcal{D}'(S')$ given by Observation G.2. Recall that paths in $\mathcal{Q}'(S')$ route the edges of $E'$ to some vertex $u \notin S'$, they are internally disjoint from $S'$, and $\mathbf{E}_{\mathcal{Q}'(S')\sim\mathcal{D}'(S')}\left[\mathrm{cong}(\mathcal{Q}'(S'), e)\right] \leq O(\log^{14.5} m)$. We denote $\mathcal{Q}'(S) = \{Q(e) \mid e \in \delta_G(S')\}$, where for all $e \in \delta_G(S')$, path $Q(e)$ has $e$ as its first edge. We let $\mathcal{Q}''$ be a multi-set of paths obtained by including, for each edge $e \in \delta_G(S')$, exactly $\mathrm{cong}_G(\mathcal{P}(R_Y), e) \leq 100$ copies of the path $Q(e)$. By concatenating the paths in $\mathcal{P}(R_Y)$ with the paths in $\mathcal{Q}''$, we obtain a set $\mathcal{Q}'(R_Y)$ of paths (an external $R_Y$-router), routing the edges of $\delta_G(R_Y)$ to vertex $u$ in $G$, such that the paths in the set are inernally disjoint from $R_Y$. For every edge in $E(S')$, the congestion caused by the paths in $\mathcal{Q}'(R_Y)$ is at most $200/\alpha_0 \leq O(\log^3 m)$, while for every edge $e \in E(G \setminus S')$, $\mathrm{cong}_G(\mathcal{Q}'(R_Y), e) \leq 100\,\mathrm{cong}_G(\mathcal{Q}'(S'), e)$. This completes the definition of the distribution $\mathcal{D}'(R_Y)$ over the set $\Lambda'_G(R_Y)$ of external $R_Y$-routers. From the above discussion, for every edge $e \in E(G) \setminus E(R_Y)$, $\mathbf{E}_{\mathcal{Q}'(R_Y)\sim\mathcal{D}'(R_Y)}\left[\mathrm{cong}(\mathcal{Q}'(R_Y), e)\right] \leq O(\log^{15.5} m) \leq \beta$, since $\beta = \log^{18} m$. We set $\mathcal{R}^{**} = \{R_Y \mid Y \in \mathcal{Y}\}$, and we define a new clustering $\tilde{\mathcal{R}} = (\mathcal{R} \setminus \mathcal{R}^*) \cup \mathcal{R}^{**}$. It is immediate to verify that all clusters in $\tilde{\mathcal{R}}$ are disjoint from each other and every cluster $R \in \tilde{\mathcal{R}}$ has $\alpha_1$-bandwidth property. From the construction of the graph $X'$, we are guaranteed that, for every basic cluster $C \in \mathcal{C}$, and for every cluster $R \in \mathcal{R}$, either $C \subseteq R$, or $C \cap R = \emptyset$ must hold. We have also defined, for every cluster $R \in \mathcal{R}$, a distribution $\mathcal{D}'(R)$ over external $R$-routers in $\Lambda'_G(R)$, such that, for every edge $e \in E(G) \setminus E(R)$, $\mathbf{E}_{\mathcal{Q}'(R)\sim\mathcal{D}'(R)}\left[\mathrm{cong}_G(\mathcal{Q}'(R), e)\right] \leq \beta$. Since we have ensured that

vertices $v^*, u^*$ do not lie in set $S$, we are guaranteed that $R^* \in \mathcal{R} \setminus \mathcal{R}^*$, and so $R^* \in \tilde{\mathcal{R}}$. Similarly, the special vertex $v^*$ does not lie in any cluster of $\tilde{\mathcal{R}}$. Therefore, $(\tilde{\mathcal{R}}, \left\{\mathcal{D}'(R)_{R\in\tilde{\mathcal{R}}}\right\})$ is a helpful clustering of $G$ with respect to $v^*$ and $\mathcal{C}$, with $R^* \in \tilde{\mathcal{R}}$. It now only remains to show that it is better than the original clustering $\mathcal{R}$.

Observe that the only basic clusters that may be contained in the clusters of $\mathcal{R}^*$ are the clusters of $\mathcal{C}^*$. From the definition of the set $\mathcal{J}$ of clusters, since cluster $C^* \in \mathcal{C}^*$ is a center-cluster of some cluster $J' \in \mathcal{J}'$, we are guaraneed that cluster $C^*$ is not contained in any cluster of $\mathcal{R}$. But, since the vertices of $\{V(Y)\}_{Y\in\mathcal{Y}}$ partition vertex set $V(X') \setminus T$, we are guaranteed that every cluster of $\mathcal{C}^*$ is contained in some cluster of $\mathcal{R}^{**}$. Therefore, the number of basic clusters of $\mathcal{C}$ contained in $G \setminus \left(\bigcup_{R\in\mathcal{R}} R\right)$ is strictly greater than the number of basic clusters of $\mathcal{C}$ contained in $G \setminus \left(\bigcup_{R\in\tilde{\mathcal{R}}} R\right)$. We conclude that clustering $\tilde{\mathcal{R}}$ is a better clustering than $\mathcal{R}$.

**Case 2.** We now consider the second case, where cluster $S$ does not contain any $J$-node, so every vertex of $S$ is either an $R$-node or a regular vertex. In this case, we proceed exactly as before: we define the augmented cluster $X = (S')^+$ and its contracted version $X' = X_{|\mathcal{C}^*}$. We then compute a well-linked decomposition $\mathcal{Y}$ of $X' \setminus T$, and the corresponding set $\mathcal{R}^{**} = \{R_Y \mid Y \in \mathcal{Y}\}$ of clusters of $G$ exactly as before. We also define a new clustering $\tilde{\mathcal{R}}$ and the distributions $\mathcal{D}'(R_Y)$ over sets of $R_Y$-routers for clusters $R_Y \in \mathcal{R}^{**}$ exactly as before. Using the same reasoning as in Case 1, the final clustering $(\tilde{\mathcal{R}}, \left\{\mathcal{D}'(R)_{R\in\tilde{\mathcal{R}}}\right\})$ is a helpful clustering of $G$ with respect to $v^*$ and $\mathcal{C}$, with $R^* \in \tilde{\mathcal{R}}$. However, since we are no longer guaranteed that $S$ contains a $J$-node (which in turn contains a cluster of $\mathcal{C}$ as a center cluster), we need to employ a different argument in order to prove that $\tilde{\mathcal{R}}$ is a better clustering than $\mathcal{R}$. Since $S$ does not contain any $J$-node, from the definition of a simplifying cluster, $|E_{\check{H}}(S)| \geq |\delta_{\check{H}}(S)|/\log m = |E'|/\log m$ must hold. Notice that every edge of $E_{\check{H}}(S)$ corresponds to an edge of $S'_{|\mathcal{R}^*}$, and so $|E(S'_{|\mathcal{R}^*})| \geq |E'|/\log m$. On the other hand, from Equation (27), $|E(S'_{|\mathcal{R}^{**}})| \leq \sum_{Y\in\mathcal{Y}} |\delta_{X'}(Y)| - |E'| \leq O(|E'|\alpha_0 \cdot \log^{1.5} m) < |E'|/\log m$, since $\alpha_0 = 1/\log^3 m$. From our definition of the set $\tilde{\mathcal{R}}$ of clusters, $|E(G_{|\tilde{\mathcal{R}}})| = |E(G_{|\mathcal{R}})| - |E(S'_{|\mathcal{R}^*})| + |E(S'_{|\mathcal{R}^{**}})| < |E(G_{|\mathcal{R}})|$. We conclude that the new helpful clustering $\tilde{\mathcal{R}}$ is better than $\mathcal{R}$.

**Case 3.** It remains to consider the third case, where $S$ contains at least one $J$-node, and for each such $J$-node $v_{J'}$, the center-cluster of the cluster $J' \in \mathcal{J}'$ lies in $\mathcal{W}'$. We fix any $J$-node $v_{J'} \in V(S)$, and we denote by $W' \in \mathcal{W}'$ the center-cluster of $J'$.

There is a difficulty with following the approach used in Cases 1 and 2 in this case: it is possible that every cluster of $\mathcal{C}^*$ is contained in some cluster of $\mathcal{R}^*$, and, additionally, it is possible that the total number of edges in $S'_{|\mathcal{R}^*}$ is quite small compared to $|E'|$. While we could still obtain a new helpful clustering $\tilde{\mathcal{R}}$ in the same way as in Cases 1 and 2, it may no longer be the case that $\tilde{\mathcal{R}}$ is a better clustering than $\mathcal{R}$. In order to overcome this difficulty, we will replace cluster $S'$ of $G$ with a different cluster $\tilde{S}' \subseteq S'$ that has similar properties to cluster $S'$, except that, if we denote by $\tilde{\mathcal{R}}^*$ the set of all clusters $R \in \mathcal{R}^*$ with $R \subseteq \tilde{S}'$, then $|E(S'_{|\tilde{\mathcal{R}}^*})|$ is sufficiently large compared to $|E_G(\tilde{S}')|$. We construct the cluster $\tilde{S}'$ using the following claim.

**Claim G.3** *There is an efficient algorithm, that, if Case 3 happenned, computes a cluster $\tilde{S}' \subseteq S'$ in graph $G$, such that for every cluster $R \in \mathcal{R}^*$, either $R \subseteq \tilde{S}'$ or $R \cap \tilde{S}' = \emptyset$ holds, and similarly, for every cluster $C \in \mathcal{C}^*$, either $C \subseteq \tilde{S}'$ or $C \cap \tilde{S}' = \emptyset$ holds. Moreover, if we denote by $\tilde{\mathcal{R}}^*$ the set of all clusters $R \in \mathcal{R}^*$ with $R \subseteq \tilde{S}'$, then $|E(\tilde{S}'_{|\tilde{\mathcal{R}}^*})| \geq |\delta_G(\tilde{S}')|/(64\log m)$. Additionally, the algorithm computes a distribution $\mathcal{D}'(\tilde{S}')$ over the set $\Lambda'_G(\tilde{S}')$ of external $\tilde{S}'$-routers in graph $G$, such that, for every edge $e \in E(G \setminus \tilde{S}')$, $\mathbf{E}_{\mathcal{Q}'(\tilde{S}')\sim\mathcal{D}'(\tilde{S}')}\left[\mathrm{cong}(\mathcal{Q}'(\tilde{S}'), e)\right] \leq O(\log^{14.5} m)$.*

We provide the proof of Claim G.3 below, after completing the proof of Claim 7.21 using it. We employ the algorithm from Cases 1 and 2, except that we apply it to cluster $\tilde{S}'$ of $G$ instead of $S'$, and we replace the set $\mathcal{R}^*$ of clusters with $\tilde{\mathcal{R}}^*$. Let $\tilde{\mathcal{C}}^* \subseteq \mathcal{C}^*$ be the set of all basic clusters $C \in \mathcal{C}^*$ with $C \subseteq \tilde{S}'$. Let $\tilde{\mathcal{R}}^{**}$ be the set of clusters that the algorithm from Cases 1 and 2 computes (that were denoted by $\mathcal{R}^{**}$ before), when applied to cluster $\tilde{S}'$. For every cluster $R \in \tilde{\mathcal{R}}^{**}$, the algorithm obtains a distribution $\mathcal{D}'(R)$ over the set $\Lambda'_G(R)$ of external $R$-routers. Let $\tilde{\mathcal{R}} = (\mathcal{R} \setminus \tilde{\mathcal{R}}^*) \cup \tilde{\mathcal{R}}^{**}$. Using the same arguments as in Cases 1 and 2, $(\tilde{\mathcal{R}}, \{\mathcal{D}'(R)\}_{R \in \tilde{\mathcal{R}}})$ is a helpful clustering in $G$ with respect to $v^*$ and $\mathcal{C}$, with $R^* \in \tilde{\mathcal{R}}$. It now only remains to show that $\tilde{\mathcal{R}}$ is a better clustering than $\mathcal{R}$.

As in Case 2, the only basic clusters of $\mathcal{C}$ that the clusters of $\tilde{\mathcal{R}}^*$ may contain are the clusters of $\tilde{\mathcal{C}}^*$. As in Case 2, each such cluster is guaranteed to be contained in some cluster of $\tilde{\mathcal{R}}^{**}$. Therefore, the number of clusters of $\mathcal{C}$ that are contained in $G \setminus (\bigcup_{R \in \mathcal{R}} R)$ is greater than or equal to the number of clusters of $\mathcal{C}$ that are contained in $G \setminus (\bigcup_{R \in \tilde{\mathcal{R}}} R)$. In order to prove that $\tilde{\mathcal{R}}$ is a better clustering than $\mathcal{R}$, it is now enough to prove that $|E(G_{|\tilde{\mathcal{R}}})| < |E(G_{|\mathcal{R}})|$.

We denote $\delta_G(\tilde{S}')$ by $E''$. On the one hand, from Claim G.3, $|E(\tilde{S}'_{|\tilde{\mathcal{R}}^*})| \geq |E''|/(64 \log m)$. On the other hand, from Equation (27), $|E(\tilde{S}'_{|\tilde{\mathcal{R}}^{**}})| \leq \sum_{Y \in \mathcal{Y}} |\delta_{X'}(Y)| - |E''| \leq |E''| \cdot O(\alpha_0 \cdot \log^{1.5} m)) < |E''|/(64 \log m)$, since $\alpha_0 = 1/\log^3 m$. As in Case 2, $|E(G_{|\tilde{\mathcal{R}}})| = |E(G_{|\mathcal{R}})| - |E(S'_{|\tilde{\mathcal{R}}^*})| + |E(S'_{|\tilde{\mathcal{R}}^{**}})| < |E(G_{|\mathcal{R}})|$. Therefore, $\tilde{\mathcal{R}}$ is a better clustering than $\mathcal{R}$. It now remains to complete the proof of Claim G.3, which we do next.

**Proof of Claim G.3.** Recall that we have fixed a $J$-node $v_{J'} \in V(S)$, and a center-cluster $W' \in \mathcal{W}'$ of the cluster $J' \in \mathcal{J}'$. We let $\tilde{S}'$ be a vertex-induced subgraph of $S'$ with the following properties:

- $W' \subseteq \tilde{S}'$;
- for every cluster $R \in \mathcal{R}^*$, either $R \subseteq \tilde{S}'$, or $R \cap \tilde{S}' = \emptyset$;
- for every cluster $C \in \mathcal{C}^*$, either $C \subseteq \tilde{S}'$ or $C \cap \tilde{S}' = \emptyset$; and
- $|E(\tilde{S}')|$ is minimized among all graphs $\tilde{S}'$ for which the above conditions hold.

Such a graph $\tilde{S}'$ can be computed via standard minimum cut computation: we start with graph $G$, and we let $G_1$ be the graph obtained from $G$ by contracting the cluster $W'$ into a source $s$, and contracting $G \setminus S'$ into a destination $t$. Next, we obtain a graph $G_2$ from $G$, by contracting every cluster $R \in \mathcal{R}$ with $R \subseteq S' \setminus W'$ into a supernode $v_R$, and similarly contracting every basic cluster $C \in \mathcal{C}'$ with $C \subseteq S' \setminus W'$ into a supernode $v_C$. Let $(Z, Z')$ be the minimum $s$-$t$ cut in $G_2$, and denote $E'' = E_{G_2}(Z, Z')$. Observe that, from the max-flow / min-cut theorem, there is a collection $\mathcal{Q}$ of edge-disjoint path in graph $G_2$, routing the edges of $E''$ to $t$, such that all paths in $\mathcal{Q}$ are internally disjoint from $Z$. We let $\tilde{S}' \subseteq S'$ be the subgraph of $\tilde{S}'$ that is defined by $Z$ (that is, we un-contract every cluster of $\mathcal{R} \cup \mathcal{C}$ whose corresponding supernode lies in $Z$). Note that $\delta_G(\tilde{S}') = E''$, and $W' \subseteq \tilde{S}'$. Let $W \in \mathcal{W}$ be the $W$-cluster in graph $\hat{H} = G_{|\mathcal{C}' \cup \mathcal{R}}$ that corresponds to $W'$; in other words, cluster $W'$ was obtained from $W$ by un-contracting its supernodes. Let $\tilde{\mathcal{R}}^*$ the set of all clusters $R \in \mathcal{R}^*$ with $R \subseteq \tilde{S}'$. Clearly, every edge of $W$ is an edge of $S'_{|\tilde{\mathcal{R}}^*}$. From the definition of a valid set of $W$-clusters, $|E(S'_{|\tilde{\mathcal{R}}^*})| \geq |E_{\hat{H}}(W)| \geq |\delta_{\hat{H}}(W)|/(64 \log m) \geq |E''|/(64 \log m)$ (we have used the fact that $|E(Z, Z')| \leq |\delta_G(W')| = |\delta_{\hat{H}}(W)|$ must hold, from the definition of minimum cut).

It now remains to define the distribution $\mathcal{D}'(\tilde{S}')$ over the set $\Lambda'_G(\tilde{S}')$ of external $\tilde{S}'$-routers in graph $G$. As before, we will provide an efficient algorithm to draw a set $\mathcal{Q}'(S')$ of paths from the distribution. As observed already, there is a set $\mathcal{Q}$ of edge-disjoint paths in graph $G_2$ routing the edges of $\delta_{G_2}(Z)$ to vertex $t$, so that the paths are internally disjoint from cluster $Z$. Note that every edge $e \in E''$

has exactly one path $Q(e) \in \mathcal{Q}$ whose first edge is $e$, and the last edge of $Q(e)$ lies in $E'$. Since every cluster of $\mathcal{R}$ has the $\alpha_1 = 1/\log^6 m$-bandwidth property, and every cluster $C \in \mathcal{C}$ has the $\alpha_0 = 1/\log^3 m$-bandwidth property, we can use the algorithm from Claim 4.41 to compute a collection $\mathcal{P} = \{P(e) \mid e \in E''\}$ of paths in graph $G_1$, where for every edge $e \in E''$, path $P(e)$ has $e$ as its first edge and some edge of $E'$ as its last edge. Moreover, the paths in $\mathcal{P}$ cause edge-congestion at most $2/\alpha_1 \leq 2/\log^6 m$ in graph $G_1$, and congestion at most 1 on edges of $E'$, and they are internally disjoint from $\tilde{S}'$. Note that the paths of $\mathcal{P}$ are also contained in the original graph $G$. We are now ready to define the distribution $\mathcal{D}'(\tilde{S}')$. In order to draw an external $\tilde{S}'$-router $\mathcal{Q}'(\tilde{S}') \in \Lambda'_G(\tilde{S}')$ from the distribution, we first draw a set $\mathcal{Q}'(S') \in \Lambda_G(S')$ of paths from the distribution $\mathcal{D}'(S')$, given by Observation G.2. Recall that paths in $\mathcal{Q}'(S')$ route the edges of $E'$ to some vertex $u \notin S'$, they are internally disjoint from $S'$, and $\mathbf{E}_{\mathcal{Q}'(S')\sim\mathcal{D}'(S')}\left[\mathrm{cong}(\mathcal{Q}'(S'), e)\right] \leq O(\log^{14.5} m)$. By concatenating the paths in $\mathcal{P}$ with the paths in $\mathcal{Q}'(S')$, we obtain a set $\mathcal{Q}'(\tilde{S}')$ of paths, routing the edges of $\delta_G(\tilde{S}')$ to vertex $u$ in $G$, such that the paths in the set are inernally disjoint from $\tilde{S}'$. Moreover, for every edge of $S' \cup E'$, the congestion caused by the paths in $\mathcal{Q}'(\tilde{S}')$ is at most $O(\log^6 m)$, while for every edge of $G \setminus S'$, $\mathrm{cong}_G(\mathcal{Q}'(\tilde{S}'), e) \leq \mathrm{cong}_G(\mathcal{Q}'(S'), e)$. This completes the definition of the distribution $\mathcal{D}'(\tilde{S}')$ over the set $\Lambda'_G(\tilde{S}')$ of external $\tilde{S}'$-routers. Clearly, for every edge $e \in E(G \setminus \tilde{S}')$, $\mathbf{E}_{\mathcal{Q}'(\tilde{S}')\sim\mathcal{D}'(\tilde{S}')}\left[\mathrm{cong}(\mathcal{Q}'(\tilde{S}'), e)\right] \leq O(\log^{14.5} m)$. □

## G.7 Proof of Observation 7.24

Consider any vertex $v \in V(\check{H}) \setminus (\bigcup_{i=1}^{r} S'_i)$, and assume that it lies in $S_i \setminus S'_i$, for some $1 \leq i \leq r$.

Recall that $|\delta^{\mathrm{up}}(v)| \leq |\delta_{\check{H}}(v)|/\log m$ must hold, and, since $v$ was not added to $S'_i$, $|\delta^{\mathrm{down,straight''}}(v)| \leq |\delta_{\check{H}}(v)|/128$. Therefore, $|\delta^{\mathrm{down,right}}(v)| + |\delta^{\mathrm{down,left}}(v)| + |\delta^{\mathrm{down,straight'}}(v)| \geq 63|\delta_{\check{H}}(v)|/64$ holds.

Assume now that $i > 1$, and consider the Gomory-Hu tree $\tau$ of the graph $\check{H}$, and the two connected components of the graph obtained from $\tau$ after the edge $(u_{i-1}, u_i)$ is removed from $\tau$. Denote by $A$ the set of all vertices lying in the connected component containing $u_{i-1}$, and by $B$ the set of all vertices lying in the other connected component. From the definition of cluster $S_i$, $v \in B$, and moreover, $(A, B)$ is the minimum $u_{i-1}$-$u_i$ cut in graph $\check{H}$. Therefore, if we let $A' = A \cup \{v\}$ and $B' = B \setminus \{v\}$, then $|E_{\check{H}}(A', B')| \geq |E_{\check{H}}(A, B)|$ must hold. Observe that the only difference between the edge sets $E_{\check{H}}(A', B')$ and $E_{\check{H}}(A, B)$ is that the edges of $\delta^{\mathrm{down,left}}(v)$ contribute to $E_{\check{H}}(A, B)$ but not to $E_{\check{H}}(A', B')$; the edges of $\delta^{\mathrm{down,right}}(v) \cup \delta^{\mathrm{down,straight'}}(v) \cup \delta^{\mathrm{down,straight''}}(v)$ contribute to $E_{\check{H}}(A', B')$ but not to $E_{\check{H}}(A, B)$; and the edges of $\delta^{\mathrm{up}}(v)$ may contribute to either cut (see Figure 45). Therefore, it must be the case that:

$$|\delta^{\mathrm{down,left}}(v)| \leq |\delta^{\mathrm{down,right}}(v)| + |\delta^{\mathrm{down,straight'}}(v)| + |\delta^{\mathrm{down,straight''}}(v)| + |\delta^{\mathrm{up}}(v)|.$$

From the definition of the layers $L_1, \ldots, L_h$, $|\delta^{\mathrm{up}}(v)| \leq |\delta(v)|/\log m$, and, since $v$ was not added to $S'_i$, $|\delta^{\mathrm{down,straight''}}(v)| \leq |\delta(v)|/128$. Therefore:

$$|\delta^{\mathrm{down,left}}(v)| \leq |\delta^{\mathrm{down,right}}(v)| + |\delta^{\mathrm{down,straight'}}(v)| + |\delta(v)|/64.$$

If we assume, for contradiction, that $|\delta^{\mathrm{down,left}}(v)| > 2(|\delta^{\mathrm{down,right}}(v)| + |\delta^{\mathrm{down,straight'}}(v)|)$, then, combining this with the above inequality, we will get that $|\delta^{\mathrm{down,right}}(v)| + |\delta^{\mathrm{down,straight'}}(v)| < |\delta(v)|/64$, and therefore, $|\delta^{\mathrm{down,left}}(v)| < |\delta(v)|/32$, contradicting the fact that $|\delta^{\mathrm{down,left}}(v)| + |\delta^{\mathrm{down,right}}(v)| + |\delta^{\mathrm{down,straight'}}(v)| \geq 63|\delta(v)|/64$. We conclude that $|\delta^{\mathrm{down,left}}(v)| \leq 2(|\delta^{\mathrm{down,right}}(v)| + |\delta^{\mathrm{down,straight'}}(v)|)$ must hold.

We now show that $S_r = S'_r$. Assume for contradiction that this is not the case, and let $v \in S_r \setminus S'_r$ be a vertex that minimizes the index $j$ for which $v \in L'_j$. Notice that, from the defini-

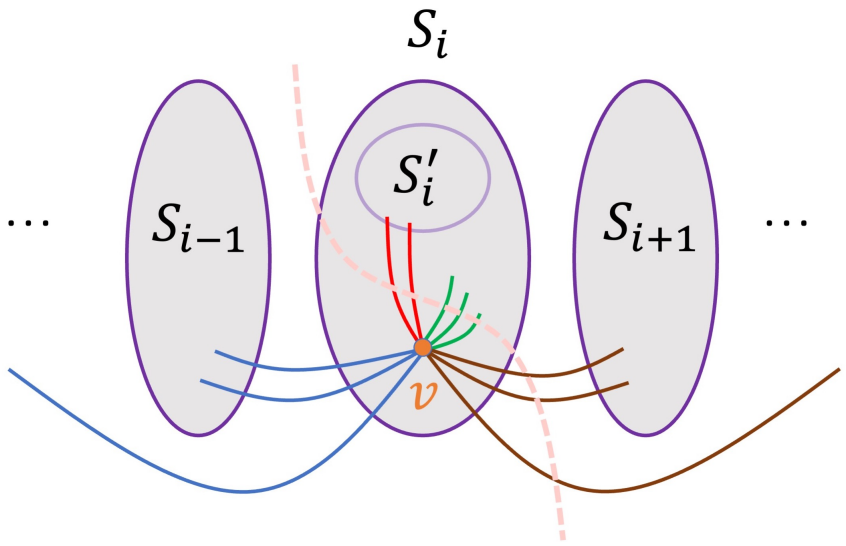



Figure 45: Illustration for the proof of Observation 7.24. The edges of $\delta^{\text{down,left}}(v)$ are shown in blue, the edges of $\delta^{\text{down,straight}''}(v)$ are shown in red, the edges of $\delta^{\text{down,straight}'}(v)$ are shown in green, and the edges of $\delta^{\text{down,right}}(v)$ are shown in brown. The blue edges lie in $E(A,B) \setminus E(A',B')$. The edges crossing the pink dashed line lie in $E(A',B') \setminus E(A,B)$. Additionally, edges of $\delta^{\text{up}}(v)$ may lie in set $E(A',B') \setminus E(A,B)$.

tion, $\delta^{\text{down,right}}(v)$, $\delta^{\text{down,straight}'}(v) = \emptyset$ must hold. Since we have established that $|\delta^{\text{down,left}}(v)| \leq 2(|\delta^{\text{down,right}}(v)| + |\delta^{\text{down,straight}'}(v)|)$, we get that $\delta^{\text{down,left}} = \emptyset$ as well. Altogether, we get that $\delta^{\text{down}}(v) = \delta^{\text{down,straight}''}(v)$. Since $|\delta^{\text{up}}(v)| \leq |\delta_{\check{H}}(v)|/\log m$, we get that the number of edges connecting $v$ to vertices of $S'_r$, $|\delta^{\text{down,straight}''}(v)| > |\delta_{\check{H}}(v)|/2$, and so $v$ should have been added to set $S'_r$. Therefore, $S_r = S'_r$ must hold.

The proof that $|\delta^{\text{down,right}}(v)| \leq 2(|\delta^{\text{down,left}}(v)| + |\delta^{\text{down,straight}'}(v)|)$ is similar, except that now we need to consider the cut obtained by deleting the edge $(u_i, u_{i+1})$ from the tree $\tau$. The proof that $S_1 = S'_1$ is symmetric to the proof that $S_r = S'_r$ (in fact, if we reverse the order of the vertices $u_1, \ldots, u_r$ on path $P^*$, by rooting the tree $\tau$ at vertex $u_r = u^*$, then the definitions of sets $S_i, S'_i$ will remain unchanged, and we can simply repeat the proof from above).

## G.8 Proof of Observation 7.25

Consider any vertex $v \in V(\check{H}) \setminus (\bigcup_{i=1}^{r} S'_i)$. From Observation 7.24, $|\delta^{\text{down,left}}(v)| \leq 2(|\delta^{\text{down,right}}(v)| + |\delta^{\text{down,straight}'}(v)|)$. Therefore:

$$|\delta^{\text{down,left}}(v)| + |\delta^{\text{down,right}}(v)| + |\delta^{\text{down,straight}'}(v)| \leq 3(|\delta^{\text{down,right}}(v)| + |\delta^{\text{down,straight}'}(v)|).$$

On the other hand, since $|\delta^{\text{down,straight}''}(v)| \leq |\delta_{\check{H}}(v)|/128$, and $|\delta^{\text{up}}(v)| \leq |\delta_{\check{H}}(v)|/\log m$, we get that:

$$|\delta^{\text{down,left}}(v)| + |\delta^{\text{down,right}}(v)| + |\delta^{\text{down,straight}'}(v)| \geq \left(\frac{127}{128} - \frac{1}{\log m}\right) \cdot |\delta_{\check{H}}(v)|.$$

By combining the two inequalities, we get that:

$$|\delta^{\text{down,right}}(v)| + |\delta^{\text{down,straight}'}(v)| \geq \left(\frac{127}{384} - \frac{1}{3\log m}\right) \cdot |\delta_{\check{H}}(v)| > |\delta^{\text{down,straight}''}(v)| + |\delta^{\text{up}}(v)|.$$

Therefore, we can define a mapping $f^{\text{right}}(v)$, that maps every edge of $\delta^{\text{down,straight}''}(v) \cup \delta^{\text{up}}(v)$ to a distinct edge of $\delta^{\text{down,right}}(v) \cup \delta^{\text{down,straight}'}(v)$.

Using exactly the same reasoning, we get that:

$$|\delta^{\text{down,left}}(v)| + |\delta^{\text{down,straight}'}(v)| > |\delta^{\text{down,straight}''}(v)| + |\delta^{\text{up}}(v)|.$$

Therefore, we can define a mapping $f^{\text{left}}(v)$, that maps every edge of $\delta^{\text{down,straight}''}(v) \cup \delta^{\text{up}}(v)$ to a distinct edge of $\delta^{\text{down,left}}(v) \cup \delta^{\text{down,straight}'}(v)$.

## G.9 Constructing the Monotone Paths – proof of Lemma 7.27

For all $1 \leq j \leq h$, we define two sets of edges, $E_j^{\text{left}}$ and $E_j^{\text{right}}$, as follows. Start with $E_j^{\text{left}} = E_j^{\text{right}} = \emptyset$. For all $1 \leq i \leq r$, for every vertex $v \in U_{i,j} \setminus \{S'_i\}$, we add all edges of $\delta^{\text{down,left}}(v) \cup \delta^{\text{down,straight}'}(v)$ to $E_j^{\text{left}}$, and similarly, we add all edges of $\delta^{\text{down,right}}(v) \cup \delta^{\text{down,straight}'}(v)$ to $E_j^{\text{right}}$. We prove the following claim.

**Claim G.4** *There is an efficient algorithm to construct, for all $1 \leq j \leq h$, a set $\mathcal{P}_j^{\text{left}} = \left\{P^{\text{left}}(e) \mid e \in E_j^{\text{left}}\right\}$ of edge-disjoint left-monotone paths, and a set $\mathcal{P}_j^{\text{right}} = \left\{P^{\text{right}}(e) \mid e \in E_j^{\text{right}}\right\}$ of edge-disjoint right-monotone paths, such that, for every edge $e \in E_j^{\text{left}}$, path $P^{\text{left}}(e)$ starts with edge $e$, and similarly, for every edge $e' \in E_j^{\text{right}}$, path $P^{\text{right}}(e')$ starts with edge $e'$.*

**Proof:** The proof is by induction on $j$. The base is when $j = 1$. Consider some vertex $v \in U_{i,1} \setminus \{S'_i\}$, for some $1 \leq i \leq r$. Note that every edge in $\delta^{\text{down}}(v)$ must connect $v$ to a vertex of $L'_0$, and every vertex of $L'_0$ lies in $\{u_1, \ldots, u_r\}$. Consider now some edge $e = (v, u)$ that is incident to $v$, that lies in $E_1^{\text{left}}$. Note that $e \notin \delta^{\text{down,straight}'}(v)$, as all edges connecting $u$ to $u_i$ lie in $\delta^{\text{down,straight}''}(v)$, and no edge of $\delta^{\text{down}}(v)$ may connect $v$ to a vertex outside $\{u_1, \ldots, u_r\}$. Therefore, $e \in \delta^{\text{left}}(v)$ must hold. We then let $P^{\text{left}}(e) = (e)$. It is immediate to verify that it is a left-monotone path. We define paths $P^{\text{right}}(e)$ for every edge $e \in \delta_{\hat{H}(v)} \cap E_1^{\text{right}}$ similarly.

We assume now that the claim holds for some index $1 \leq j < h$, and we prove it for index $j + 1$. Consider some vertex $v \in U_{i,j+1} \setminus \{S'_i\}$, for some $1 \leq i \leq r$, and let $e = (v, u)$ be any edge that is incident to $v$ and lies in $E_j^{\text{left}}$. In this case, $e \in \delta^{\text{down,left}}(v) \cup \delta^{\text{down,straight}'}(v)$ must hold. If we denote by $1 \leq i' \leq r$, $1 \leq j' \leq h$ the indices for which $u \in U_{i',j'}$, then $i' \leq i$ and $j' \leq j$ must hold.

Assume first that $u \in S'_1 \cup \cdots \cup S'_r$. In this case, since $e \notin \delta^{\text{down,straight}''}(v)$, $e \in \delta^{\text{down,left}}(v)$ must hold, and $i' < i$. In this case, we let $P^{\text{left}}(e) = (e)$. It is easy to see that this path is a valid left-monotone path. Otherwise, $u \notin S'_1 \cup \cdots \cup S'_r$, and $e \in \delta^{\text{up}}(u)$. We then consider the edge $e' \in \delta^{\text{down,left}}(u) \cup \delta^{\text{down,straight}'}(u)$ to which edge $e$ is mapped by $f^{\text{left}}(u)$. We then let $P^{\text{left}}(e)$ be the path obtained by concatenating the edge $e$ and the path $P^{\text{left}}(e') \in \mathcal{P}_{j'}^{\text{left}}$. Since path $P^{\text{left}}(e')$ is left-monotone, and since $j' \leq j$ and $i \leq i$, path $P^{\text{left}}(e)$ is also left-monotone. We note that edge $e$, by the definition, may not lie on any path in $\mathcal{P}_1^{\text{left}} \cup \cdots \cup \mathcal{P}_j^{\text{left}}$. Moreover, edge $e$ is the only edge that is mapped to edge $e'$ by map $f^{\text{left}}(u)$. This ensures that all paths in the resulting set $\mathcal{P}_j^{\text{left}} = \left\{P^{\text{left}}(e'') \mid e'' \in E_j^{\text{left}}\right\}$ are edge-disjoint.

The construction of the set $\mathcal{P}_{j+1}^{\text{right}}$ of paths is symmetric. □

We are now ready to complete the proof of Lemma 7.27. Consider an edge $e = (u, v) \in \hat{E}$, with $u \in S_i$, $v \in S_{i'}$, and $i < i'$. We describe the construction of path $P(e, u)$; the construction of path $P(e, v)$ is symmetric. If $u \in S'_i$, then path $P(e, u)$ consists only of the vertex $u$. It is a left-monotone path by definition. Therefore, we assume now that $u \notin S'_i$. We assume that $u$ lies in layer $L'_j$, for some

$1 \leq j \leq h$, and $v$ lies in layer $L'_{j'}$, for some $0 \leq j' \geq h$ (vertex $u$ may not lie in $L'_0$, since we have assumed that $u \notin S'_i$).

We now consider two cases. The first case is when $j \leq j'$. In this case, $e \in \delta^{\text{up}}(u)$. We let $e' \in \delta^{\text{down,left}}(u) \cup \delta^{\text{down,straight}'}(u)$ be the edge to which $e$ is mapped by $f^{\text{left}}(u)$. Since $e' \in E^{\text{left}}_j$, there is a left-monotone path $P^{\text{left}}(e') \in \mathcal{P}^{\text{left}}_j$. We then let $P(e,u)$ be the path obtained by concatenating edge $e$ with path $P^{\text{left}}(e')$. It is easy to verify that path $P(e,u)$ is left-monotone.

The second case is when $j > j'$. In this case, $e \in \delta^{\text{right}}(u)$. Recall that, from Observation 7.24, $|\delta^{\text{down,right}}(u)| \leq 2(|\delta^{\text{down,left}}(u)| + |\delta^{\text{down,straight}'}(u)|)$. Therefore, we can define another mapping $g^{\text{left}}(u)$, that maps the edges of $\delta^{\text{down,right}}(u)$ to edges of $\delta^{\text{down,left}}(u) \cup \delta^{\text{down,straight}'}(u)$, such that at most two edges are mapped to every edge of $\delta^{\text{down,left}}(u) \cup \delta^{\text{down,straight}'}(u)$. We then let $e'$ be the edge to which $e$ is mapped by $g^{\text{left}}(u)$. As before, $e' \in E^{\text{left}}_j$ must hold, and so there is a left-monotone path $P^{\text{left}}(e') \in \mathcal{P}^{\text{left}}_j$. We then let $P(e,u)$ be the path obtained by concatenating edge $e$ with path $P^{\text{left}}(e')$ as before. This finishes the definition of the path $P(e,u)$. Path $P(e,v)$ is defined symmetrically. Since the paths in $\left(\bigcup_{j=1}^{h} \mathcal{P}^{\text{right}}_j\right) \cup \left(\bigcup_{j=1}^{h} \mathcal{P}^{\text{right}}_j\right)$ cause congestion $O(\log m)$, it is easy to verify that the set $\left\{P(e,v), P(e,u) \mid e = (u,v) \in \hat{E}\right\}$ of paths causes congestion $O(\log m)$.

## G.10 Proof of Observation 7.29

Fix an index $1 < i < r$. Consider the two connected components of the Gomory-Hu tree $\tau$ that are obtained after the edge $(u_{i-1}, u_i)$ is deleted from it. Denote the sets of vertices of the two components by $A$ and $B$, where $u_{i-1} \in A$. Recall that $(A,B)$ is the minimum cut separating $u_{i-1}$ from $u_i$ in $\check{H}$. Observe that $A = V(S_1) \cup \cdots \cup V(S_{i-1})$, while $B = V(S_i) \cup \cdots \cup V(S_r)$. Note also that:

$$|E(A,B)| \geq |E'_{i-1}| + |\tilde{E}^{\text{left}}_i| + |\tilde{E}^{\text{left}}_{i+1}| + |\tilde{E}^{\text{over}}_i|$$

(see Figure 46).

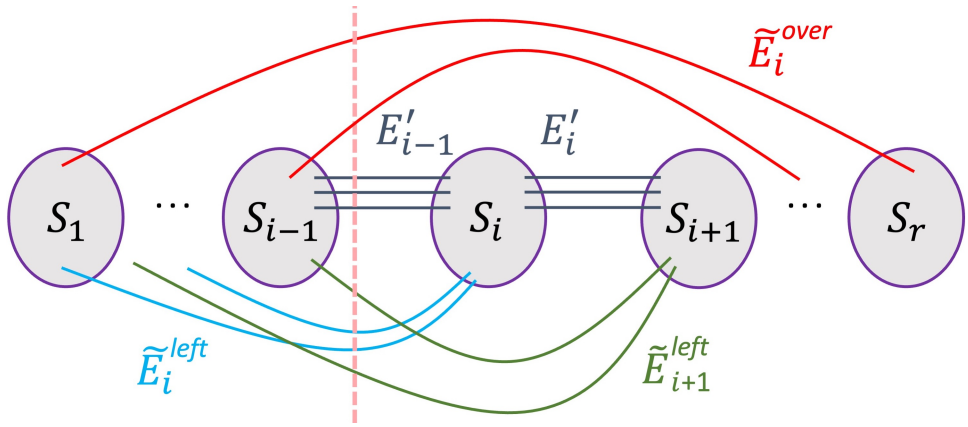



Figure 46: Illustration for the Proof of Observation 7.29. Cut $(A,B)$ is shown in a pink dashed line.

Next, we consider another $u_{i-1}$–$u_i$ cut $(X,Y)$ in $\check{H}$, where $Y = V(S_i)$, and $X = V(\check{H}) \setminus Y$. Observe that:

$$|E(X,Y)| = |E'_{i-1}| + |\tilde{E}^{\text{left}}_i| + |\tilde{E}^{\text{right}}_i| + |E'_i|$$

(see Figure 47).

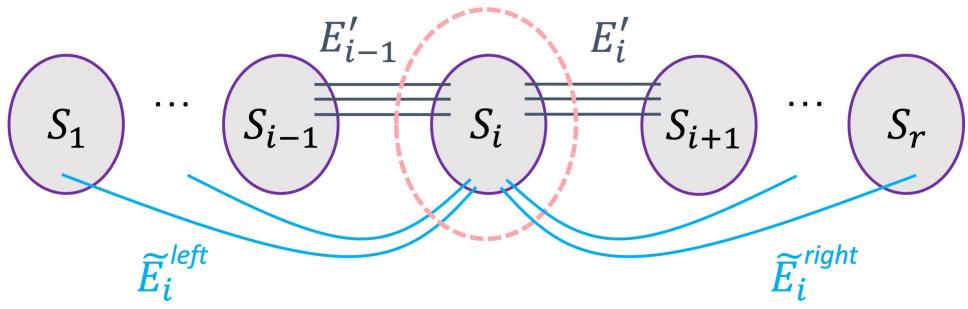



Figure 47: Illustration for the Proof of Observation 7.29. Cut $(X, Y)$ is shown in a pink dashed line.

Since $|E(A, B)| \leq |E(X, Y)|$ must hold, we conclude that:

$$|\tilde{E}_{i+1}^{\text{left}}| + |\tilde{E}_i^{\text{over}}| \leq |\tilde{E}_i^{\text{right}}| + |E'_i|. \tag{28}$$

We now repeat the same reasoning with cuts separating $u_{i+1}$ from $u_{i+2}$. Consider the two connected components of the Gomory-Hu tree $\tau$ that are obtained after the edge $(u_{i+1}, u_{i+2})$ is deleted from it. Denote the sets of vertices of the two components by $A'$ and, $B'$, where $u_{i+1} \in A'$. Recall that $(A', B')$ is the minimum cut separating $u_{i+1}$ from $u_{i+2}$ in $\check{H}$. Note that $A' = V(S_1) \cup \cdots \cup V(S_{i+1})$, and:

$$|E(A', B')| \geq |E'_{i+1}| + |\tilde{E}_{i+1}^{\text{right}}| + |\tilde{E}_i^{\text{right}}| + |\tilde{E}_i^{\text{over}}|$$

(see Figure 48).

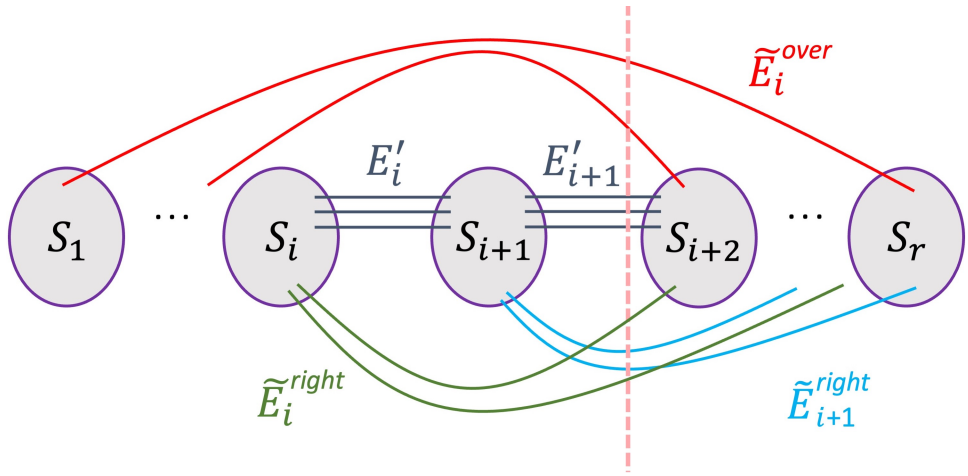



Figure 48: Illustration for the proof of Observation 7.29, with cut $(A, B)$ shown in a pink dashed line.

We now consider another $u_{i+1}$–$u_{i+2}$ cut $(X', Y')$ in $\check{H}$, where $X' = V(S_{i+1})$, and $Y' = V(\check{H}) \setminus X'$. Observe that:

$$|E(X', Y') = |E'_{i+1}| + |\tilde{E}_{i+1}^{\text{right}}| + |E'_i| + |\tilde{E}_{i+1}^{\text{left}}|$$

(see Figure 49).

Since $|E(A', B')| \leq |E(X', Y')|$ must hold, we conclude that:

$$|\tilde{E}_i^{\text{right}}| + |\tilde{E}_i^{\text{over}}| \leq |E'_i| + |\tilde{E}_{i+1}^{\text{left}}|. \tag{29}$$

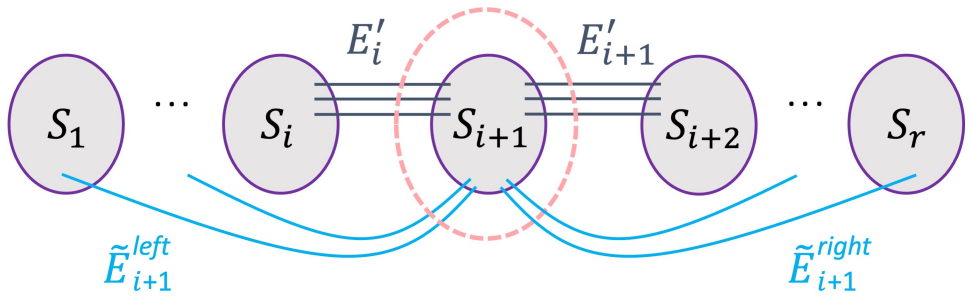



Figure 49: Illustration for the proof of Observation 7.29, with cut $(X, Y)$ shown in a pink dashed line.

By adding Equation (28) and Equation (29), we conclude that $|\tilde{E}_i^{\text{over}}| \leq |E'_i|$. We also immediately get that $|\tilde{E}_{i+1}^{\text{left}}| \leq |E'_i| + |\tilde{E}_i^{\text{right}}|$ from Equation (28) and $|\tilde{E}_i^{\text{right}}| \leq |E'_i| + |\tilde{E}_{i+1}^{\text{left}}|$ from Equation (29).

## G.11 Proof of Claim 7.30

Fix an index $1 < i < r$. We prove that $|\delta^{\text{down}}(S''_i)| \leq 0.1|\delta^{\text{right}}(S''_i)|$, and show the existence of the set $\mathcal{P}^{\text{right}}$ of paths. The proof that $|\delta^{\text{down}}(S''_i)| \leq 0.1|\delta^{\text{left}}(S''_i)|$ and the proof of the existence of the set $\mathcal{P}^{\text{left}}$ of paths is symmetric.

Let $\mathcal{P}$ be the set of all paths $P$ in graph $\check{H}$, such that the first edge of $P$ lies in $\delta^{\text{down}}(S''_i)$, the last edge lies in $\delta^{\text{right}}(S''_i)$, and all inner vertices of $P$ lies in $S''_i$. We show a flow $f$, defined over the set $\mathcal{P}$ of paths, in which every edge in $\delta^{\text{down}}(S''_i)$ sends one flow unit, every edge in $\delta^{\text{right}}(S''_i)$ receives at most $1/10$ flow unit, and each edge of $E(S''_i)$ carries at most one flow unit. The existence of such a flow will then prove that $|\delta^{\text{down}}(S''_i)| \leq 0.1|\delta^{\text{right}}(S''_i)|$, and will imply the existence of the path set $\mathcal{P}^{\text{right}}$ with the required properties from the integrality of flow.

In order to define the flow $f$, we consider the vertices of $S''_i$ in the decreasing order of their levels $L'_h, \ldots, L'_1$. When we consider level $L'_j$, for $1 \leq j \leq h$, we assume that, for every vertex $v \in L'_j \cap S''_i$, for every edge $e \in \delta^{\text{up}}(v)$, connecting $v$ to another vertex of $S''_i$, the flow on edge $e$ is fixed already, and that the only edges that are incident to $v$ that may carry non-zero flow are edges of $\delta^{\text{up}}(v) \cup \delta^{\text{down,straight}''}(v)$. Initially, for every edge $e \in \delta^{\text{down}}(S''_i)$, we set $f(e) = 1$. Note that for every edge $e \in \delta^{\text{down}}(S''_i)$, if $v$ is the endpoint of $e$ lying in $S''_i$, then $e \in \delta^{\text{down,straight}''}(v)$.

We start with the level $L'_h$. Consider any vertex $v \in L'_h$. From the definition, $\delta^{\text{up}}(v) = \emptyset$. Recall that, from Observation 7.24, $|\delta^{\text{down,right}}(v)| + |\delta^{\text{down,left}}(v)| + |\delta^{\text{down,straight}'}(v)| \geq 63|\delta(v)|/64$, and additionally, $|\delta^{\text{down,left}}(v)| \leq 2(|\delta^{\text{down,right}}(v)| + |\delta^{\text{down,straight}'}(v)|)$. Therefore, $|\delta^{\text{down,right}}(v)| + |\delta^{\text{down,straight}'}(v)| \geq 21|\delta(v)|/64$ must hold. At the same time, $|\delta^{\text{down,straight}''}(v)| \leq |\delta(v)|/128$. We now consider two cases. The first case is when $|\delta^{\text{down,straight}'}(v)| \geq |\delta(v)|/64$. In this case, $|\delta^{\text{down,straight}''}(v)| \leq |\delta^{\text{down,straight}'}(v)|$ holds. We assign, to each edge $e \in \delta^{\text{down,straight}''}(v)$, a distinct edge $e' \in \delta^{\text{down,straight}'}(v)$, and we set $f(e') = f(e)$. Consider now the second case, where $|\delta^{\text{down,right}}(v)| \geq |\delta(v)|/8$ must hold. In this case, $|\delta^{\text{down,straight}''}(v)| \leq |\delta^{\text{down,straight}'}(v)|/10$. We can now assign, to each edge $e \in \delta^{\text{down,straight}''}(v)$ ten distinct edges $e_1, \ldots, e_{10} \in \delta^{\text{down,right}}(v)$, such that each edge of $\delta^{\text{down,right}}(v)$ is assigned to at most one edge of $\delta^{\text{down,straight}''}(v)$. If edge $e' \in \delta^{\text{down,right}}(v)$ is assigned to edge $e \in \delta^{\text{down,straight}''}(v)$, then we set $f(e') = f(e)/10$.

Assume now that we have processed levels $L'_h, \ldots, L'_{j+1}$, and consider some level $L'_j$. Let $v \in L'_j \cap S''_i$ be any vertex. The processing of vertex $v$ is similar to the one above, except that now some edges of $\delta^{\text{up}}(v)$ may carry flow, and we need to forward this flow to edges in $\delta^{\text{down}}(v)$. As before, from Observation 7.24, $|\delta^{\text{down,right}}(v)| + |\delta^{\text{down,left}}(v)| + |\delta^{\text{down,straight}'}(v)| \geq 63|\delta(v)|/64$, and additionally, $|\delta^{\text{down,left}}(v)| \leq 2(|\delta^{\text{down,right}}(v)| + |\delta^{\text{down,straight}'}(v)|)$. Therefore, $|\delta^{\text{down,right}}(v)| + |\delta^{\text{down,straight}'}(v)| \geq$

$21|\delta(v)|/64$ must hold. At the same time, $|\delta^{\text{down,straight}''}(v)| \leq |\delta(v)|/128$, and $\delta^{\text{up}}(v) \leq \delta^{\text{down}}(v)/\log m$. We again consider two cases. The first case is when $|\delta^{\text{down,straight}'}(v)| \geq |\delta(v)|/64$. In this case, $|\delta^{\text{down}''}(v)| + |\delta^{\text{up}}(v)| \leq |\delta^{\text{down,straight}'}(v)|$ holds. We assign, to each edge $e \in \delta^{\text{down,straight}''}(v) \cup \delta^{\text{up}}(v)$, a distinct edge $e' \in \delta^{\text{down,straight}'}(v)$, and we set $f(e') = f(e)$. Consider now the second case, where $|\delta^{\text{down,right}}(v)| \geq |\delta(v)|/8$ must hold. In this case, $|\delta^{\text{down,straight}''}(v)| + |\delta^{\text{up}}(v)| \leq |\delta^{\text{down,straight}'}(v)|/10$. As before, we can assign, to each edge $e \in \delta^{\text{down,straight}''}(v) \cup \delta^{\text{up}}(v)$ ten distinct edges $e_1, \ldots, e_{10} \in \delta^{\text{down,right}}(v)$, such that each edge of $\delta^{\text{down,right}}(v)$ is assigned to at most one edge of $\delta^{\text{down,straight}''}(v) \cup \delta^{\text{up}}(v)$. If edge $e' \in \delta^{\text{down,right}}(v)$ is assigned to edge $e \in \delta^{\text{down,straight}''}(v) \cup \delta^{\text{up}}(v)$, then we set $f(e') = f(e)/10$.

Notice that, when vertices $v \in L'_1$ are processed, we are guaranteed that $\delta^{\text{down}'}(v) = \emptyset$, and so eventually all flow originating at the edges of $\delta^{\text{down}}(S''_i)$ reaches the edges of $\delta^{\text{right}}(S''_i)$. This completes the definition of the flow $f$. It is immediate to verify that $f$ is defined over the set $\mathcal{P}$ of paths; every edge in $\delta^{\text{down}}(S''_i)$ sends one flow unit; every edge in $\delta^{\text{right}}(S''_i)$ receives at most $1/10$ flow units; and each edge of $E(S''_i)$ carries at most one flow unit.

## G.12 Proof of Claim 7.31

We prove that $|\delta^{\text{left}}(S''_i)| \leq 1.1|\delta^{\text{right}}(S''_i)|$; the proof that $|\delta^{\text{right}}(S''_i)| \leq 1.1|\delta^{\text{left}}(S''_i)|$ is symmetric. Consider the cut $(A, B)$ in graph $\check{H}$, where $A = V(S_1) \cup \cdots \cup V(S_{i-1})$, and $B = V(S_i) \cup \cdots \cup V(S_r)$ (see Figure 50).

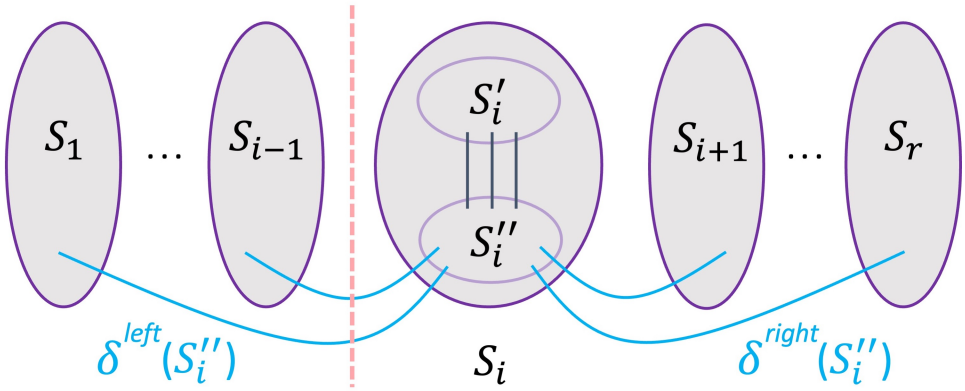



Figure 50: Illustration for the proof of Claim 7.31, with cut $(A, B)$ shown in a pink dashed line.

Note that $A$ and $B$ are precisely the sets of vertices of the two connected components of the graph $\tau \setminus \{(u_{i-1}, u_i)\}$, where $\tau$ is the Gomory-Hu tree of graph $\check{H}$. Therefore, $(A, B)$ is the minimum $u_{i-1}$-$u_i$ cut in $\check{H}$. We now consider another $u_{i-1}$-$u_i$ cut $(A', B')$ in $\check{H}$, where $A' = A \cup V(S''_i)$, and $B' = B \setminus V(S''_i)$ (see Figure 51).

Observe that edges of $\delta^{\text{left}}(S''_i)$ contribute to $E(A, B)$ but not to $E(A', B')$, while edges of $\delta^{\text{right}}(S''_i) \cup \delta^{\text{down}}(S''_i)$ contribute to $E(A', B')$ but not to $E(A, B)$, and this is the only difference between the two edge sets. Since $(A, B)$ is the minimum $u_{i-1}$-$u_i$ cut, we get that $|\delta^{\text{left}}(S''_i)| \leq |\delta^{\text{right}}(S''_i)| + |\delta^{\text{down}}(S''_i)|$ must hold. Since, from Claim 7.30, $|\delta^{\text{down}}(S''_i)| \leq 0.1|\delta^{\text{right}}(S''_i)|$, we conclude that $|\delta^{\text{left}}(S''_i)| \leq 1.1|\delta^{\text{right}}(S''_i)|$.

## G.13 Proof of Claim 7.32

Fix an index $1 < i < r$. We start by proving that $|\delta^{\text{right}}(S''_i)| \leq 1.3|E_i| + 1.3|\delta^{\text{right}}(S'_i)|$.

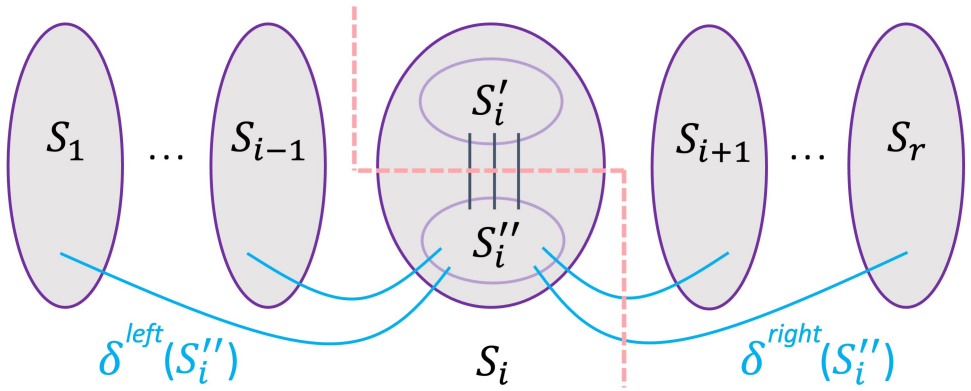



Figure 51: Illustration for the proof of Claim 7.31, with cut $(A', B')$ shown in a pink dashed line. Edges of $\delta^{\text{down}}(S''_i)$ are shown in black.

As before, we consider the cut $(A, B)$ in graph $\check{H}$, where $A = V(S_1) \cup \cdots \cup V(S_{i-1})$, and $B = V(S_i) \cup \cdots \cup V(S_r)$ (see Figure 52). As before, $A$ and $B$ are precisely the sets of vertices of the two connected components of the graph $\tau \setminus \{(u_{i-1}, u_i)\}$, where $\tau$ is the Gomory-Hu tree of graph $\check{H}$, and so $(A, B)$ is the minimum $u_{i-1}$-$u_i$ cut in $\check{H}$. We now consider another $u_{i-1}$-$u_i$ cut $(A', B')$ in $\check{H}$, where $B' = V(S'_i)$, and $A' = V(\check{H}) \setminus V(S'_i)$ (see Figure 53).

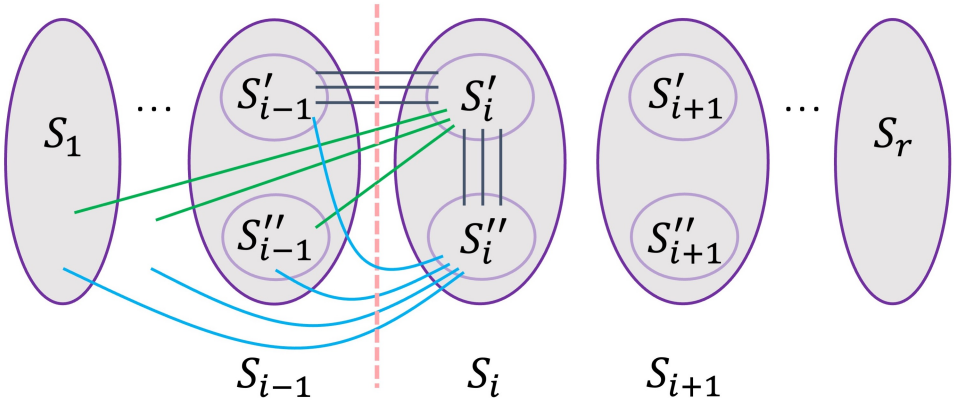



Figure 52: Illustration for the proof of Claim 7.32, with cut $(A, B)$ shown in a pink dashed line. Edges of $\delta^{\text{left}}(S'_i)$ are shown in light green, and edges of $\delta^{\text{left}}(S''_i)$ are shown in blue.

Note that $|E(A, B)| \geq |E_{i-1}| + |\delta^{\text{left}}(S'_i)| + |\delta^{\text{left}}(S''_i)|$, while $|E(A', B')| = |E_{i-1}| + |\delta^{\text{left}}(S'_i)| + |\delta^{\text{down}}(S''_i)| + |E_i| + |\delta^{\text{right}}(S'_i)|$. From the fact that $(A, B)$ is the minimum $u_{i-1}$-$u_i$ cut, it must be the case that:

$$|\delta^{\text{left}}(S''_i)| \leq |\delta^{\text{down}}(S''_i)| + |E_i| + |\delta^{\text{right}}(S'_i)|$$

Since, from Claim 7.31, $|\delta^{\text{right}}(S''_i)| \leq 1.1|\delta^{\text{left}}(S''_i)|$, and, from Claim 7.30, $|\delta^{\text{down}}(S''_i)| \leq 0.1|\delta^{\text{right}}(S''_i)|$, we get that:

$$\begin{aligned}
|\delta^{\text{right}}(S''_i)| &\leq 1.1|\delta^{\text{left}}(S''_i)| \\
&\leq 1.1|\delta^{\text{down}}(S''_i)| + 1.1|E_i| + 1.1|\delta^{\text{right}}(S'_i)| \\
&\leq 0.11|\delta^{\text{right}}(S''_i)| + 1.1|E_i| + 1.1|\delta^{\text{right}}(S'_i)|,
\end{aligned}$$

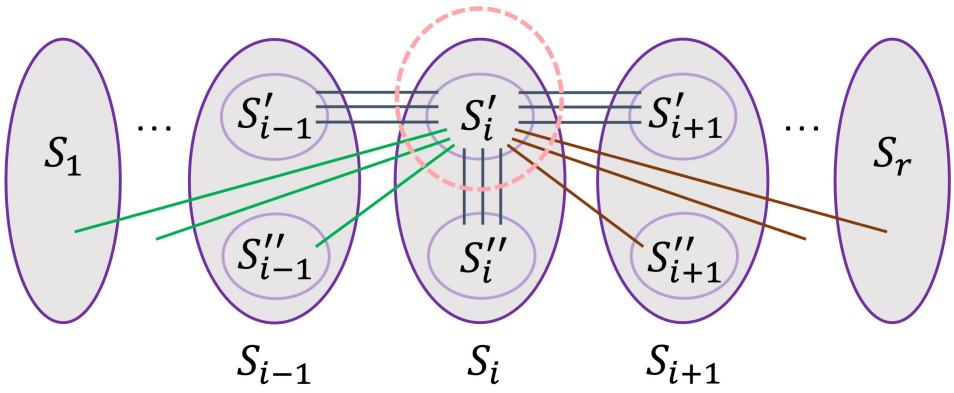



Figure 53: Illustration for the proof of Claim 7.32, with cut $(A', B')$ shown in a pink dashed line. Edges of $\delta^{\text{right}}(S'_i)$ are shown in brown, and edges of $\delta^{\text{left}}(S'_i)$ are shown in green.

and so $|\delta^{\text{right}}(S''_i)| \leq 1.3|E_i| + 1.3|\delta^{\text{right}}(S'_i)|$.

The proof that $|\delta^{\text{left}}(S''_{i+1})| \leq 1.3|E_i| + 1.3|\delta^{\text{left}}(S'_{i+1})|$ is symmetric. We consider the cut $(X, Y)$ in graph $\check{H}$, where $X = V(S_1) \cup \cdots \cup V(S_{i+1})$, and $Y = V(S_{i+2}) \cup \cdots \cup V(S_r)$ (see Figure 54). Note that $X$ and $Y$ are precisely the sets of vertices of the two connected components of the graph $\tau \setminus \{(u_{i+1}, u_{i+2})\}$, where $\tau$ is the Gomory-Hu tree of graph $\check{H}$, and so $(X, Y)$ is the minimum $u_{i+1}$-$u_{i+2}$ cut in $\check{H}$. We now consider another $u_{i+1}$-$u_{i+2}$ cut $(X', Y')$ in $\check{H}$, where $X' = V(S'_{i+1})$, and $Y' = V(\check{H}) \setminus V(S'_{i+1})$ (see Figure 55).

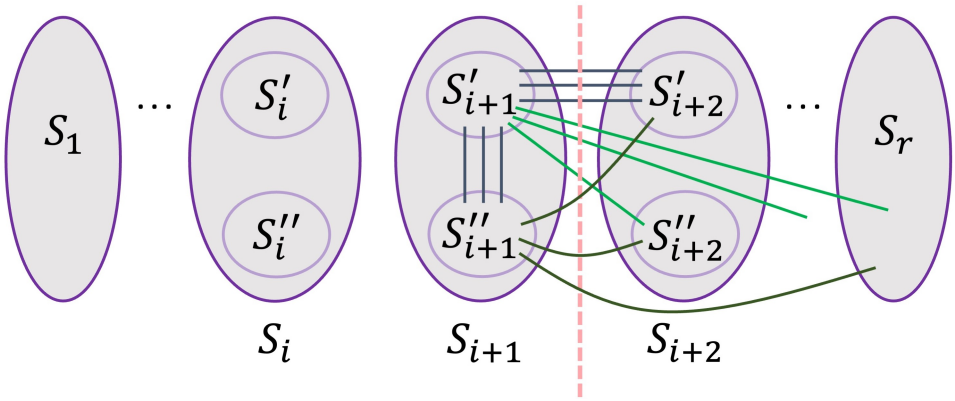



Figure 54: Illustration for the proof of Claim 7.32, with cut $(X, Y)$ shown in a pink dashed line. Edges of $\delta^{\text{right}}(S'_{i+1})$ are shown in light green, and edges of $\delta^{\text{right}}(S''_{i+1})$ are shown in dark green.

Note that $|E(X, Y)| \geq |E_{i+1}| + |\delta^{\text{right}}(S'_{i+1})| + |\delta^{\text{right}}(S''_{i+1})|$, while $|E(X', Y')| = |E_{i+1}| + |\delta^{\text{right}}(S'_{i+1})| + |\delta^{\text{left}}(S'_{i+1})| + |\delta^{\text{down}}(S''_{i+1})| + |E_i|$. From the fact that $(X, Y)$ is the minimum $u_i$-$u_{i+1}$ cut, it must be the case that:

$$|\delta^{\text{right}}(S''_{i+1})| \leq |\delta^{\text{left}}(S'_{i+1})| + |\delta^{\text{down}}(S''_{i+1})| + |E_i|$$

Since, from Claim 7.31, $|\delta^{\text{left}}(S''_{i+1})| \leq 1.1|\delta^{\text{right}}(S''_{i+1})|$, and, from Claim 7.30, $|\delta^{\text{down}}(S''_{i+1})| \leq 0.1|\delta^{\text{left}}(S''_{i+1})|$, we get that:

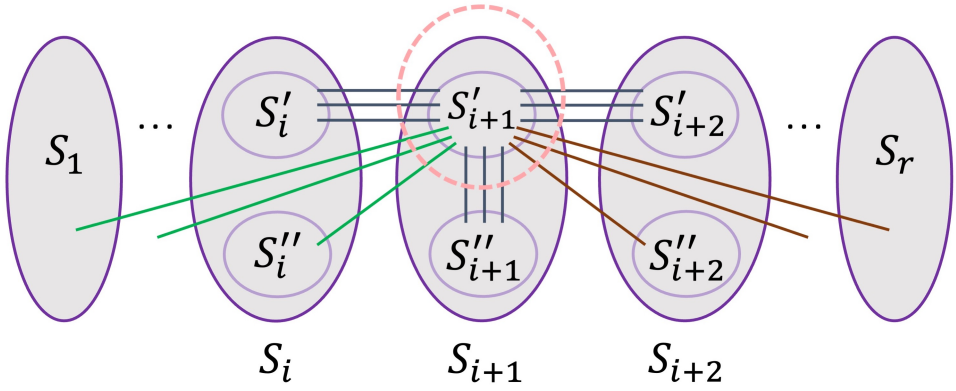



Figure 55: Illustration for the proof of Claim 7.32, with cut $(X', Y')$ shown in a pink dashed line. Edges of $\delta^{\text{right}}(S'_{i+1})$ are shown in brown, and edges of $\delta^{\text{left}}(S'_{i+1})$ are shown in green.

$$\begin{aligned}|\delta^{\text{left}}(S''_{i+1})| &\leq 1.1|\delta^{\text{right}}(S''_{i+1})| \\ &\leq 1.1|\delta^{\text{down}}(S''_{i+1})| + 1.1|E_i| + 1.1|\delta^{\text{left}}(S'_{i+1})| \\ &\leq 0.11|\delta^{\text{left}}(S''_{i+1})| + 1.1|E_i| + 1.1|\delta^{\text{left}}(S'_i)|,\end{aligned}$$

and so $|\delta^{\text{left}}(S''_{i+1})| \leq 1.3|E_i| + 1.3|\delta^{\text{left}}(S'_{i+1})|$.

### G.14 Proof of Claim 7.33

Fix an index $1 \leq i < r$. As before, we consider the cut $(A, B)$ in graph $\check{H}$, where $A = V(S_1) \cup \cdots \cup V(S_{i-1})$, and $B = V(S_i) \cup \cdots \cup V(S_r)$. As before, $A$ and $B$ are precisely the sets of vertices of the two connected components of the graph $\tau \setminus \{(u_{i-1}, u_i)\}$, where $\tau$ is the Gomory-Hu tree of graph $\check{H}$, and so $(A, B)$ is the minimum $u_{i-1}$-$u_i$ cut in $\check{H}$ (see Figure 56).

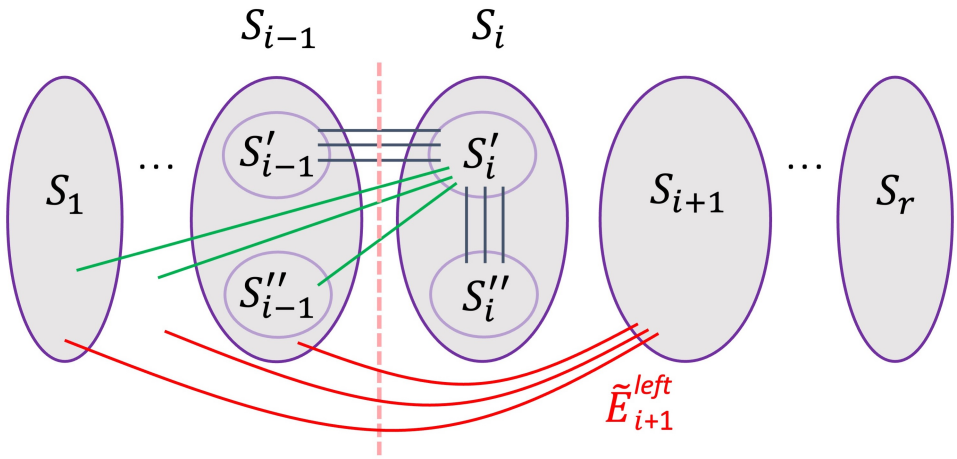



Figure 56: Illustration for the proof of Claim 7.33, with cut $(A, B)$ shown in pink dashed line. Edges of $\delta^{\text{left}}(S'_i)$ are shown in green.

We now consider another $u_{i-1}$-$u_i$ cut $(A', B')$ in $\check{H}$, where $B' = V(S'_i)$, and $A' = V(\check{H}) \setminus V(S'_i)$ (see Figure 57).

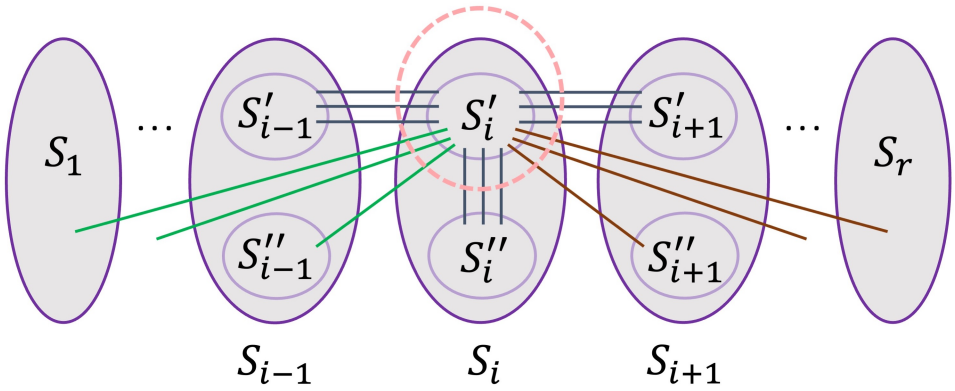



Figure 57: Illustration for the proof of Claim 7.33, with cut $(A', B')$ shown in pink dashed line. Edges of $\delta^{\text{left}}(S'_i)$ are shown in green, and edges of $\delta^{\text{right}}(S'_i)$ are shown in brown.

Note that $|E(A,B)| \geq |E_{i-1}| + |\delta^{\text{left}}(S'_i)| + |\tilde{E}^{\text{left}}_{i+1}|$, while $|E(A',B')| = |E_{i-1}| + |\delta^{\text{left}}(S'_i)| + |\delta^{\text{down}}(S''_i)| + |E_i| + |\delta^{\text{right}}(S'_i)|$. From the fact that $(A,B)$ is the minimum $u_{i-1}$-$u_i$ cut, it must be the case that:

$$|\tilde{E}^{\text{left}}_{i+1}| \leq |\delta^{\text{down}}(S''_i)| + |E_i| + |\delta^{\text{right}}(S'_i)|$$

Recall that, from Claim 7.30, $|\delta^{\text{down}}(S''_i)| \leq 0.1|\delta^{\text{right}}(S''_i)|$, and, from Claim 7.32, $|\delta^{\text{right}}(S''_i)| \leq 1.3|E_i| + 1.3|\delta^{\text{right}}(S'_i)|$. Therefore, $\delta^{\text{down}}(S''_i) \leq 0.13|E_i| + 0.13|\delta^{\text{right}}(S'_i)|$, and:

$$|\tilde{E}^{\text{left}}_{i+1}| \leq 1.13|E_i| + 1.13|\delta^{\text{right}}(S'_i)|. \tag{30}$$

Consider now the set $\delta^{\text{left}}(S'_{i+1})$ of edges (see Figure 58). Recall that this set contains every edge $e = (u,v)$ with $u \in V(S'_{i+1})$, and $v$ either lying in $V(S_1) \cup \cdots \cup V(S_{i-1})$, or in $V(S''_i)$. In the former case, $e \in \tilde{E}^{\text{left}}_{i+1}$, while in the latter case, $e \in \delta^{\text{right}}(S''_i)$. Therefore, $|\delta^{\text{left}}(S'_{i+1})| \leq |\tilde{E}^{\text{left}}_{i+1}| + |\delta^{\text{right}}(S''_i)|$. From Claim 7.32: $|\delta^{\text{right}}(S''_i)| \leq 1.3|E_i| + 1.3|\delta^{\text{right}}(S'_i)|$. Combining this with Equation (30), we get that $|\delta^{\text{left}}(S'_{i+1})| \leq |\tilde{E}^{\text{left}}_{i+1}| + |\delta^{\text{right}}(S''_i)| \leq 2.5|E_i| + 2.5|\delta^{\text{right}}(S'_i)|$, as required.

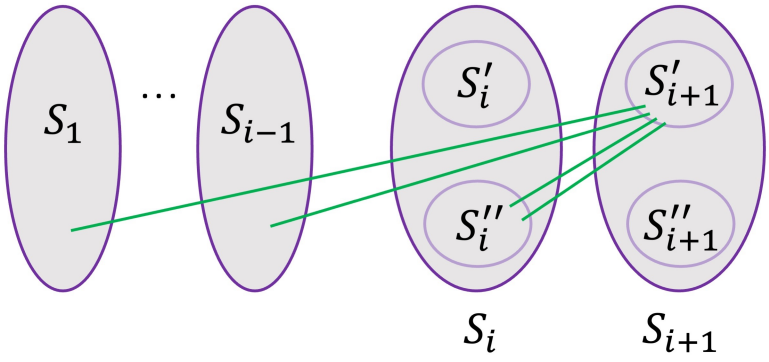



Figure 58: Set $\delta^{\text{left}}(S'_{i+1})$ of edges (shown in green).

We now employ a symmetric argument in order to bound $|\delta^{\text{right}}(S'_i)|$: consider the cut $(X,Y)$ in graph $\check{H}$, where $X = V(S_1) \cup \cdots \cup V(S_{i+1})$, and $Y = V(S_{i+2}) \cup \cdots \cup V(S_r)$ (see Figure 59).

As before, $X$ and $Y$ are precisely the sets of vertices of the two connected components of the graph $\tau \setminus \{(u_{i+1}, u_{i+2})\}$, where $\tau$ is the Gomory-Hu tree of graph $\check{H}$, and so $(X,Y)$ is the minimum $u_{i+1}$-

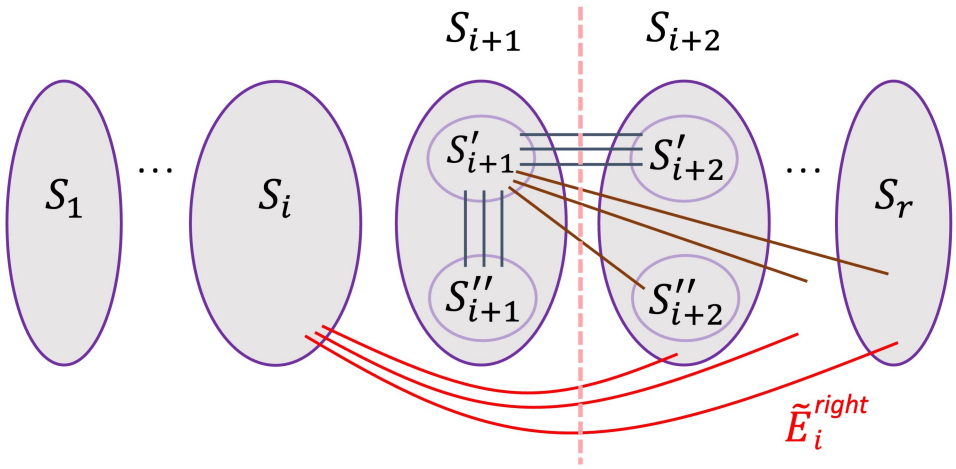



Figure 59: Illustration for the proof of Claim 7.33, with cut $(X, Y)$ shown in pink dashed line. Edges of $\delta^{\text{right}}(S'_{i+1})$ are shown in brown.

$u_{i+2}$ cut in $\check{H}$. We now consider another $u_{i+1}$-$u_{i+2}$ cut $(X', Y')$ in $\check{H}$, where $X' = V(S'_{i+1})$, and $Y' = V(\check{H}) \setminus V(S'_{i+1})$ (see Figure 60). Note that $|E(X,Y)| \geq |E_{i+1}| + |\delta^{\text{right}}(S'_{i+1})| + |\tilde{E}_i^{\text{right}}|$, while $|E(X',Y')| = |E_{i+1}| + |\delta^{\text{right}}(S'_{i+1})| + |E_i| + |\delta^{\text{left}}(S'_{i+1})| + |\delta^{\text{down}}(S''_{i+1})|$. From the fact that $(X,Y)$ is the minimum $u_{i+1}$-$u_{i+2}$ cut, it must be the case that:

$$|\tilde{E}_i^{\text{right}}| \leq |E_i| + |\delta^{\text{left}}(S'_{i+1})| + |\delta^{\text{down}}(S''_{i+1})|.$$

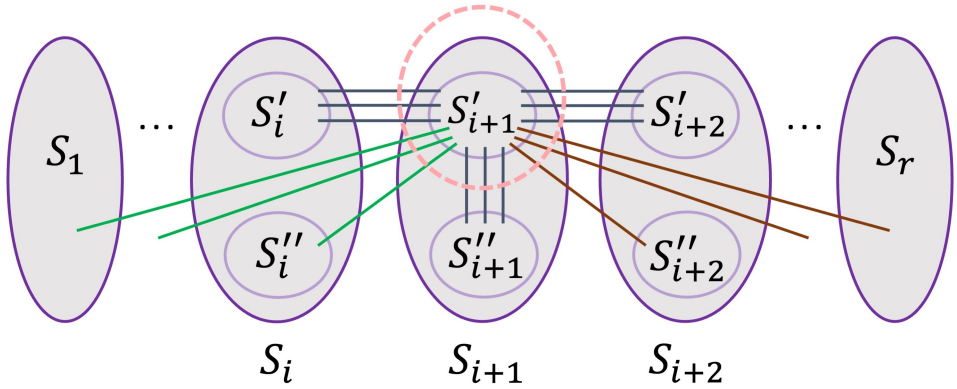



Figure 60: Illustration for the proof of Claim 7.33, with cut $(X', Y')$ shown in pink dashed line. Edges of $\delta^{\text{left}}(S'_{i+1})$ are shown in green, and edges of $\delta^{\text{right}}(S'_{i+1})$ are shown in brown.

As before, from Claim 7.30, $|\delta^{\text{down}}(S''_{i+1})| \leq 0.1 \cdot |\delta^{\text{left}}(S''_{i+1})|$, and, from Claim 7.32, $|\delta^{\text{left}}(S''_{i+1})| \leq 1.3|E_i| + 1.3|\delta^{\text{left}}(S'_{i+1})|$. Therefore, $|\delta^{\text{down}}(S''_i)| \leq 0.13|E_i| + 0.13|\delta^{\text{left}}(S'_{i+1})|$, and:

$$|\tilde{E}_i^{\text{right}}| \leq 1.13|E_i| + 1.13|\delta^{\text{left}}(S'_{i+1})|. \tag{31}$$

Consider now edge set $\delta^{\text{right}}(S'_i)$. Recall that it contains every edge $e = (u,v)$ with $u \in V(S_i)$, such that either $v \in V(S_{i+2}) \cup \cdots \cup V(S_r)$, or $v \in V(S''_{i+1})$. In the former case, $e \in \tilde{E}_i^{\text{right}}$, and in the latter case, $v \in \delta^{\text{left}}(S''_{i+1})$ holds. Therefore, $|\delta^{\text{right}}(S'_i)| \leq |\tilde{E}_i^{\text{right}}| + |\delta^{\text{left}}(S''_{i+1})|$. From Claim 7.32: $|\delta^{\text{left}}(S''_{i+1})| \leq 1.3|E_i| + 1.3|\delta^{\text{left}}(S'_{i+1})|$. Therefore, altogether, $|\delta^{\text{right}}(S'_i)| \leq |\tilde{E}_i^{\text{right}}| + |\delta^{\text{left}}(S''_{i+1})| \leq 2.5|E_i| + 2.5|\delta^{\text{left}}(S'_{i+1})|$.

## G.15 Proof of Claim 7.34

Fix some index $1 < i < r$, and recall how graph $S'_i \subseteq S_i$ was constructed. Initially, we set $V(S'_i) = \{u_i\}$. As long as there was any vertex $x \in S_i \setminus S'_i$, such that the number of edges connecting $x$ to vertices of $V(S'_i)$ was at least $|\delta(x)|/128$, we added $x$ to $V(S'_i)$. Recall that $|\delta^{\mathrm{up}}(x)| \leq |\delta(x)|/\log m$. Therefore, at least $|\delta(x)|\left(\frac{1}{128} - \frac{1}{\log m}\right)$ edges that connect $x$ to vertices of $V(S'_i)$ lie in $\delta^{\mathrm{down}}(x)$. We then let $S'_i$ be the subgraph of $\check{H}$ induced by $V(S'_i)$.

Denote $V(S'_i) = \{u_i = x_0, x_1, \ldots, x_z\}$, where the vertices $x_1, \ldots, x_z$ were added to $S'_i$ in the order of their indices. Notice that for all $1 \leq a \leq z$, if $x_a \in L'_{j_a}$, then, from the above discussion, at least one vertex in $\{x_0, \ldots, x_{a-1}\}$ must lie in $L'_1 \cup L'_2 \cup \cdots \cup L'_{j_a-1}$. It is then easy to see (by induction on $a$), that, if $u_i \in L'_j$, for some $1 \leq j \leq h$, then every vertex of $S'_i \setminus \{u_i\}$ lies in $L'_{j+1} \cup \cdots \cup L'_h$.

For all $0 \leq a \leq z$, we consider the vertex set $X_a = \{x_0, \ldots, x_a\}$, and we define a weight $w_a(e)$ of every edge $e \in E(\check{H})$ with respect to $X_a$ as follows. First, we let $\delta^{\mathrm{up}}(X_a)$ be the set of all edges $e = (x, y)$ with $x \in X_a$, $y \notin X_a$, such that $e \in \delta^{\mathrm{up}}(x)$. Every edge $e \in \delta^{\mathrm{up}}(X_a)$ is assigned weight $w_a(e) = 130$. Every edge $e \in \left(\bigcup_{v \in X_a} \delta(v)\right) \setminus \delta^{\mathrm{up}}(X_a)$ is assigned weight $w_a(e) = 1$. All other edges $e \in E(\check{H})$ are assigned weight $w_a(e) = 0$. We denote $W_a = \sum_{e \in E(\check{H})} w_a(e)$. Observe that $W_0 = |\delta^{\mathrm{down}}(u_i)| + 130|\delta^{\mathrm{up}}(u_i)| \leq |\delta(u_i)| \cdot \left(1 + \frac{130}{\log m}\right)$. Additionally, $W_z \geq |\bigcup_{v \in X_z} \delta(v)| = |\bigcup_{v \in S'_i} \delta(v)|$. We now show that for all $1 \leq a \leq z$, $W_a \leq W_{a-1}$. Notice that, if this is the case, then $|\delta(S'_i)| \leq W_z \leq W_0 \leq |\delta(u_i)| \cdot \left(1 + \frac{130}{\log m}\right)$. Therefore, in order to complete the proof of Claim 7.34, it is enough to prove the following observation.

**Observation G.5** *For all $1 \leq a \leq z$, $W_a \leq W_{a-1}$.*

**Proof:** Consider some index $1 \leq a \leq z$. Recall that $X_a = X_{a-1} \cup \{x_a\}$. Recall that, as observed above, at least $|\delta(x_a)|\left(\frac{1}{128} - \frac{1}{\log m}\right)$ edges that connect $x_a$ to vertices of $X_{a-1}$ lie in $\delta^{\mathrm{down}}(x_a)$. Denote this edge set by $E^* \subseteq \delta^{\mathrm{up}}(X_a)$. Each edge $e \in E^*$ has $w_{a-1}(e) = 130$, and $w_a(e) = 1$. Additionally, every edge $e \in \delta^{\mathrm{down}}(x_a) \setminus E^*$ has $w_{a-1}(e) = 0$ and $w_a(e) = 1$. Finally, each edge $e \in \delta^{\mathrm{up}}(x_a)$ has $w_{a-1}(e) = 0$ and $w_a(e) = 130$. For all other edges $e' \in E(\check{H})$, $w_{a-1}(e') = w_a(e')$. Therefore, altogether, we get that:

$$\begin{aligned}
W_a &= W_{a-1} - 129|E^*| + |\delta^{\mathrm{down}}(x_a) \setminus E^*| + 130|\delta^{\mathrm{up}}(x_a)| \\
&= W_{a-1} - 130|E^*| + |\delta^{\mathrm{down}}(x_a)| + 130|\delta^{\mathrm{up}}(x_a)| \\
&\leq W_{a-1} - 130 \cdot |\delta(x_a)| \left(\frac{1}{128} - \frac{1}{\log m}\right) + |\delta(x_a)| \cdot \left(1 + \frac{130}{\log m}\right) \\
&\leq W_{a-1} - \frac{|\delta(x_a)|}{64} + \frac{|\delta(x_a)| \cdot 260}{\log m} \\
&\leq W_{a-1}.
\end{aligned}$$

(we have used the fact that $|E^*| \geq |\delta(x_a)|\left(\frac{1}{128} - \frac{1}{\log m}\right)$ and that $m$ is sufficiently large). □

## G.16 Proof of Claim 7.36

Assume that there is an index $1 \leq a \leq r$, such that at least $|\delta(u_{i^*})|/16$ edges connect $u_{i^*}$ to vertices of $S''_a$. We start by proving that $|\delta^{\mathrm{right}}(S''_a)| \geq |\delta(u_{i^*})| \cdot \frac{\log m}{256}$. Denote by $E'$ the set of all edges connecting $u_{i^*}$ to vertices of $S''_a$, so $|E'| \geq |\delta(u_{i^*})|/16$. We denote by $E'' = E' \cap \delta^{\mathrm{down}}(u_{i^*})$. Since $|\delta^{\mathrm{up}}(u_{i^*})| \leq |\delta(u_{i^*})|/\log m$, $|E''| \geq |\delta(u_{i^*})|/32$.

Consider now a set $\mathcal{P}$ of paths, defined as follows: $\mathcal{P}$ contains every path $P$ of $\check{H}$, whose first edge lies in $E''$, last edge lies in $\delta^{\text{right}}(S''_a)$, and all inner vertices lie in $S''_a$. We will show a flow $f$ defined over the paths in $\mathcal{P}$, in which every edge in $E''$ sends $\frac{\log m}{8}$ flow units, every edge in $\delta^{\text{right}}(S''_a)$ receives at most one flow unit, and every edge $e \in E(S''_a)$ carries at most one flow unit. Clearly, this will prove that $|\delta^{\text{right}}(S''_a)| \geq |E''| \cdot \frac{\log m}{8} \geq |\delta(u_{i^*})| \cdot \frac{\log m}{256}$.

We now focus on defining the flow $f$. Initially, we set the flow on every edge $e \in E''$ to $\frac{\log m}{8}$, and for every edge of $e' \in E(\check{H}) \setminus E''$, we set $f(e') = 0$. Note that, if $e = (u_{i^*}, v) \in E''$, then $e \in \delta^{\text{up}}(v)$ must hold, since $e \in \delta^{\text{down}}(u_{i^*})$ from the definition of $E''$. Next, we consider indices $j = h, h-1, \ldots, 1$ one by one. We assume that, when index $j$ is considered, for every vertex $v \in L'_j \cap S''_a$, for every edge $e \in \delta^{\text{up}}(v)$, the flow $f(e)$ is fixed, and $f(e) \leq \frac{\log m}{8}$. During the iteration when index $j$ is processed we will finalize the flow values $f(e')$ for every edge $e' \in \left\{\delta^{\text{down}}(v) \mid v \in L'_j \cap S''_a\right\}$.

We now describe the iteration when index $j$ is processed. Consider any vertex $v \in L'_j \cap S''_a$. Recall that $|\delta^{\text{up}}(v)| \leq |\delta(v)|/\log m$, and so the total flow that the edges of $\delta^{\text{up}}(v)$ carry is bounded by $\frac{|\delta(v)|}{\log m} \cdot \frac{\log m}{8} \leq \frac{|\delta(v)|}{8}$. Recall that, from Observation 7.24, $|\delta^{\text{down,right}}(v)| + |\delta^{\text{down,left}}(v)| + |\delta^{\text{down,straight}'}(v)| \geq 63|\delta(v)|/64$, and $|\delta^{\text{down,left}}(v)| \leq 2(|\delta^{\text{down,right}}(v)| + |\delta^{\text{down,straight}'}(v)|)$. By combining the two inequalities, we get that $|\delta^{\text{down,right}}(v)| + |\delta^{\text{down,straight}'}(v)| \geq 21|\delta(v)|/64$. We now define the flow on every edge $e' \in \delta^{\text{down}}(v)$, as follows. If $e' \in \delta^{\text{down,left}}(v) \cup \delta^{\text{down,straight}''}(v)$, then we set $f(e') = 0$. Otherwise, $e' \in \delta^{\text{down,right}}(v) \cup \delta^{\text{down,straight}'}(v)$. We then set $f(e') = \frac{\sum_{e \in \delta^{\text{up}}(v)} f(e)}{|\delta^{\text{down,right}}(v)| + |\delta^{\text{down,straight}'}(v)|}$. From the above discussion, we are guaranteed that for every edge $e \in \delta^{\text{down}}(v)$, $f(e) \leq 1$. This completes the description of the interation where index $j$ is processed. Once al indices $j = r, r-1, \ldots, 1$ are processed, we obtain a final flow $f$. From the construction of the flow $f$, flow conservation constraints hold for every vertex $v \in S''_a$. The only edges that carry non-zero flow are edges of $E'' \cup E(S''_a) \cup \delta^{\text{right}}(S''_a)$. Moreover, each edge in $E''$ carries $\frac{\log m}{8}$ flow units, and every edge in $\delta^{\text{right}}(S''_a)$ carries at most one flow unit. We can then apply standard flow-paths decomposition of the flow $f$, to obtain a flow that is defined over the set $\mathcal{P}$ of paths, where every edge of $E''$ sends $\frac{\log m}{8}$ flow units, and every edge in $\delta^{\text{right}}(S''_a)$ receives at most one flow unit. We conclude that $|\delta^{\text{right}}(S''_a)| \geq |E''| \cdot \frac{\log m}{8} \geq |\delta(u_{i^*})| \cdot \frac{\log m}{256}$. The proof that $|\delta^{\text{left}}(S''_a)| \geq |\delta(u_{i^*})| \cdot \frac{\log m}{256}$ is symmetric.

Lastly, we prove that $u_a$ is a $J$-node. Observe first that, since $S''_a \neq \emptyset$, from Observation 7.24, $a \notin \{1, r\}$. Conisder a cut $(A, B)$ in graph $\check{H}$, where $A = S_1 \cup \cdots \cup S_{a-1}$, and $B = S_a \cup S_{a+1} \cup \cdots \cup S_r$ (see Figure 61). From the construction of the Gomory-Hu tree $\tau$, $(A, B)$ is a minimum $u_{a-1}$-$u_a$ cut in graph $\check{H}$. Since $\delta^{\text{left}}(S''_a) \subseteq E(A, B)$, we get that $|E(A, B)| \geq |\delta^{\text{left}}(S''_a)| \geq |\delta(u_{i^*})| \cdot \frac{\log m}{256}$.

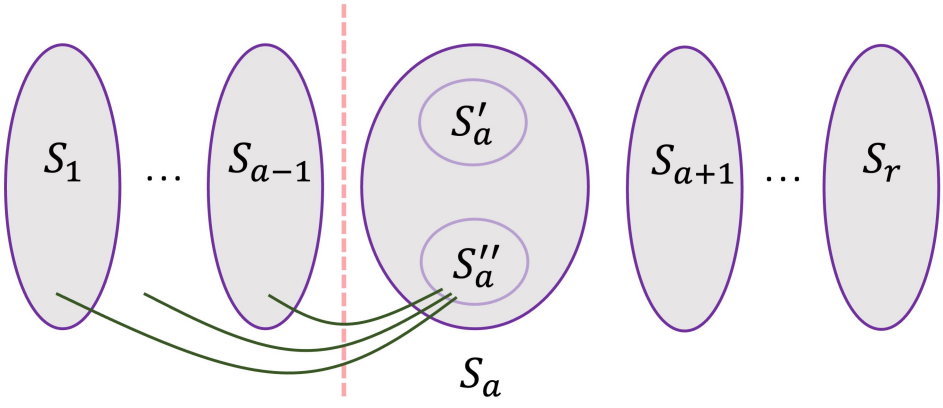



Figure 61: Illustration for the Proof of Claim 7.36, with cut $(A, B)$ shown in pink dashed line. Edges of $\delta^{\text{left}}(S''_a)$ are shown in green.

Consider another $u_{a-1}$-$u_a$ cut $(A', B')$ in graph $\check{H}$, where $B' = S'_a$ and $A' = V(\check{H}) \setminus S'_a$ (see Figure 62). Then $|E(A', B')| \geq |E(A, B)| \geq |\delta(u_{i^*})| \cdot \frac{\log m}{256}$.

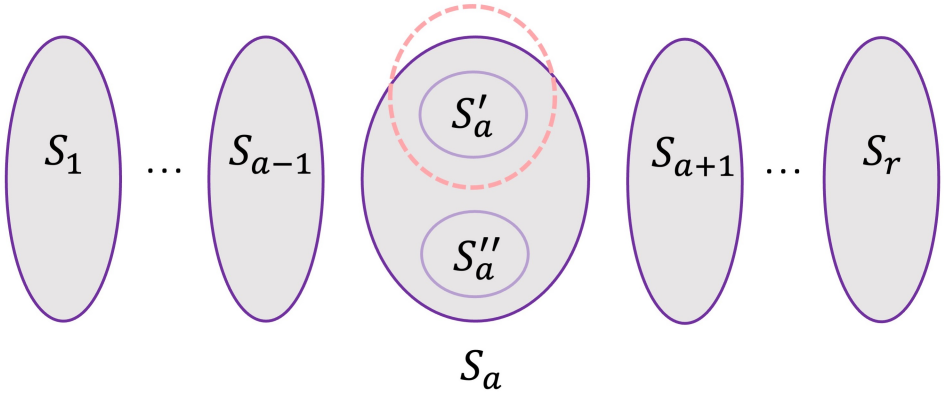



Figure 62: Illustration for the Proof of Claim 7.36, with cut $(A', B')$ shown in pink dashed line.

Assume now for contradiction that $u_a$ is not a $J$-node. Then from Claim 7.34, $|E(A', B')| = |\delta(S'_a)| \leq \left(1 + \frac{130}{\log m}\right) |\delta(u_a)| \leq 2|\delta(u_a)|$. Therefore, we conclude that $|\delta(u_a)| \geq \frac{|E(A', B')|}{2} \geq |\delta(u_{i^*})| \cdot \frac{\log m}{512} > |\delta(u_{i^*})|$, contradicting the choice of the index $i^*$ (we have used the fact that $m$ is sufficiently large).

## G.17 Proof of Claim 7.38

We consider two cuts in graph $\check{H}$. The first cut, $(A_1, B_1)$ is defined as follows: $A_1 = V(S_1) \cup \cdots \cup V(S_i)$, $B_1 = V(S_{i+1}) \cup \cdots \cup V(S_r)$. From the defintion of the Gomory-Hu tree $\tau$, and from Corollary 4.10, $(A_1, B_1)$ is a minimum $u_i$–$u_{i+1}$ cut in graph $\check{H}$, and moreover, there is a set $\mathcal{P}_1 = \{P(e) \mid e \in E(A_1, B_1)\}$ of edge-disjoint paths in graph $\check{H}$, where, for each edge $e \in E(A_1, B_1)$, path $P(e)$ has $e$ as its first edge, vertex $u_i$ as its last vertex, and is internally disjoint from $B_1$, and hence from $S^*$. We can define another $u_i$–$u_{i+1}$ cut $(A'_1, B'_1)$ in graph $\check{H}$, where $B'_1 = V(S'_{i+1})$, and $A'_1 = V(\check{H}) \setminus B'_1$. Since $(A_1, B_1)$ is a minimum $u_i$–$u_{i+1}$ cut, we get that:

$$|E(A_1, B_1)| \leq |E(A'_1, B'_1)| = |\delta(S'_{i+1})| \leq 2|\delta(u_{i+1})| \leq 2|\delta(u_i)|$$

(we have used Claim 7.34 for the penultimate inequality, and the definition of the index $i = i^*$ for the last inequality).

Similarly, we consider a second cut $(A_2, B_2)$, that is defined as follows: $A_2 = V(S_1) \cup \cdots \cup V(S_{i+2})$, $B_2 = V(S_{i+3}) \cup \cdots \cup V(S_r)$. As before, from the defintion of the Gomory-Hu tree $\tau$, and from Corollary 4.10, $(A_2, B_2)$ is a minimum $u_{i+2}$–$u_{i+3}$ cut in graph $\check{H}$, and moreover, there is a set $\mathcal{P}_2 = \{P'(e) \mid e \in E(A_2, B_2)\}$ of edge-disjoint paths in graph $\check{H}$, where, for each edge $e \in E(A_2, B_2)$, path $P'(e)$ has $e$ as its first edge, vertex $u_{i+3}$ as its last vertex, and is internally disjoint from $A_2$, and hence from $S^*$. As before, we can define another $u_{i+2}$–$u_{i+3}$ cut $(A'_2, B'_2)$ in graph $\check{H}$, setting $A'_2 = V(S'_{i+2})$, and $B'_2 = V(\check{H}) \setminus A'_2$. Since $(A_2, B_2)$ is a minimum $u_{i+2}$–$u_{i+3}$ cut, we get that:

$$|E(A_2, B_2)| \leq |E(A'_2, B'_2)| = |\delta(S'_{i+2})| \leq 2|\delta(u_{i+2})| \leq 2|\delta(u_i)|.$$

Observe that $\delta(S^*) = E(A_1, B_1) \cup E(A_2, B_2)$ (see Figure 63). Therefore, from the above discussion, $|\delta(S^*)| \leq 4|\delta(u_i)|$. On the other hand, all edges connecting $u_i$ to vertices of $\bigcup_{a>i+2} S_a$ lie in set $\tilde{E}^{\text{over}}_{i+1}$ (see Figure 63), and so, from Observation 7.29, $|E'_{i+1}| \geq |\tilde{E}^{\text{over}}_{i+1}| \geq |\delta(u_i)|/16$. Recall that

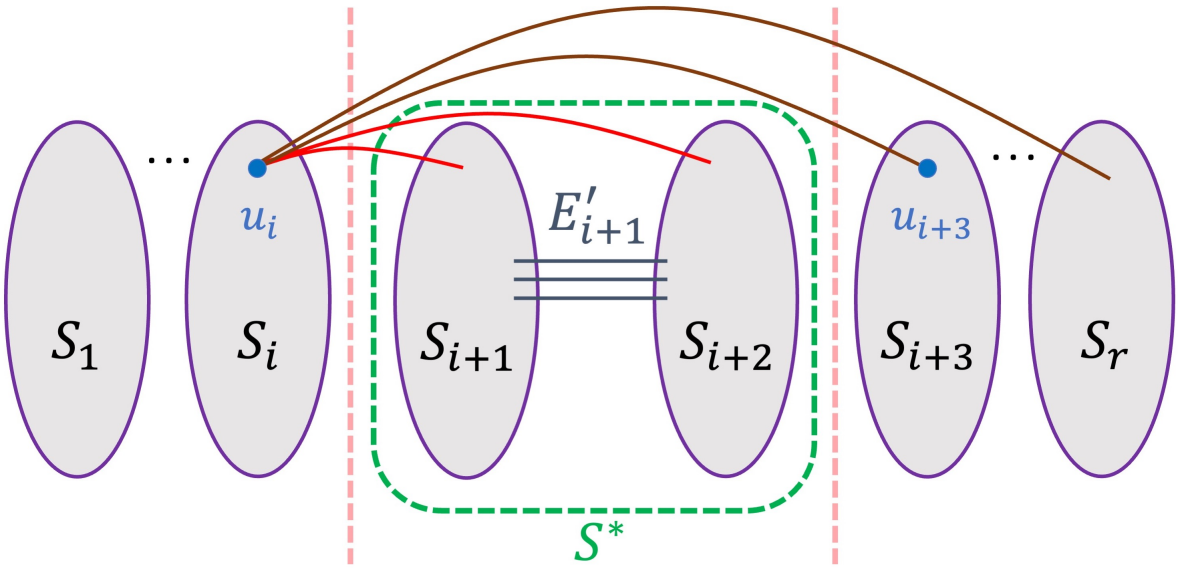



Figure 63: Illustration for the proof of Claim 7.38. The edges of $E^*$ are shown in red and brown. At least $|\delta(u_i)|/16$ edges of $E^*$ (shown in brown) have an endpoint in $S_{i+1} \cup \ldots, \cup S_r$, and these edges belong to set $E(A_2, B_2)$.

$E'_{i+1} = E(S_{i+1}, S_{i+2})$, and so in particular, $E'_{i+1} \subseteq E(S^*)$. We conclude that $|E(S^*)| \geq |E'_{i+1}| \geq |\delta(u_i)|/16 \geq |\delta(S^*)|/64$.

In order to prove that $S^*$ is a simplifying cluster, it is now enough to show a collection $\mathcal{P}^* = \{P^*(e) \mid e \in \delta(S^*)\}$ of paths in graph $\check{H}$, that cause congestion at most $\beta' = O(\log m)$, such that, for every edge $e \in \delta(S^*)$, path $P^*(e)$ has $e$ as its first edge, vertex $u_{i+3}$ as its last vertex, and it is internally disjoint from $S^*$.

Recall that we have defined a set $\mathcal{P}_2 = \{P'(e) \mid e \in E(A_2, B_2)\}$ of edge-disjoint paths in graph $\check{H}$, where, for each edge $e \in E(A_2, B_2)$, path $P'(e)$ has $e$ as its first edge, vertex $u_{i+3}$ as its last vertex, and is internally $S^*$. For each edge $e \in E(A_2, B_2)$, we set $P^*(e) = P(e)$. Since $\delta(S^*) = E(A_1, B_1) \cup E(A_2, B_2)$, it remains to define the paths $P^*(e)$ for edges $e \in E(A_1, B_1)$.

As observed above, $|E(A_1, B_1)| \leq 2|\delta(u_i)|$. From Observation 7.37, at least $|\delta(u_i)|/16$ edges connect $u_i$ to vertices of $\bigcup_{a>i+2} S_a$. Denote this set of edges by $\tilde{E}^*$. Clearly, $\tilde{E}^* \subseteq E(A_2, B_2)$ (see Figure 63). Since $|E(A_1, B_1)| \leq 2|\delta(u_i)| \leq 32|\tilde{E}^*|$, we can define a mapping $M$ from the edges of $E(A_1, B_1)$ to edges of $\tilde{E}^*$, such that, for every edge $e' \in \tilde{E}^*$, at most 32 edges of $E(A_1, B_1)$ are mapped to it. Consider now some edge $e \in E(A_1, B_1)$. The final path $P^*(e)$ is a concatenation of two paths. The first path is $P(e) \in \mathcal{P}_1$, that originates at $e$, terminates at $u_i$, and it is internally disjoint from $S^*$. Let $e' = M(e)$ be the edge of $\tilde{E}^*$ to which edge $e$ is mapped (recall that $e'$ must be incident to $u_i$). The second path is $P'(e') \in \mathcal{P}_2$, that starts at edge $e'$ and terminates at vertex $u_{i+3}$. This completes the definition of the set $\mathcal{P}^* = \{P^*(e) \mid e \in \delta(S^*)\}$ of paths. From the construction, for every edge $e \in \delta(S^*)$, path $P^*(e)$ has $e$ as its first edge, vertex $u_{i+3}$ as its last vertex, and it is internally disjoint from $S^*$. It now remains to bound the congestion of the path set $\mathcal{P}^*$. Recall that each of the path sets $\mathcal{P}_1, \mathcal{P}_2$ causes congestion 1. Each path in $\mathcal{P}_1$ is used once, and each path in $\mathcal{P}_2$ may be used by up to 33 paths. Therefore, the total congestion caused by paths in $\mathcal{P}^*$ is at most 34. We conclude that $S^*$ is a simplfying cluster.

## G.18 Proof of Claim 7.39

Since $u_{i+1}$ is a $J$-node, in order to prove that $S^*$ is a simplifying cluster, it is enough to show a collection $\mathcal{P}^* = \{P^*(e) \mid e \in \delta(u_{i+1})\}$ of paths in graph $\check{H}$, causing congestion at most $\beta' = O(\log m)$, where for each edge $e \in \delta(u_{i+1})$, path $P^*(e)$ has $e$ as its first edge, vertex $u_{i+2}$ as its last vertex, and is internally disjoint from $S^*$. As before, we define two cuts in graph $\check{H}$: cut $(A_1, B_1)$, with $A_1 = V(S_1) \cup \cdots \cup V(S_i)$ and $B_1 = V(\check{H}) \setminus A_1$, and cut $(A_2, B_2)$, with $A_2 = V(S_1) \cup \cdots \cup V(S_{i+1})$ and $B_2 = V(\check{H}) \setminus A_2$. As before, from our construction, $(A_1, B_1)$ is a minimum $u_i$–$u_{i+1}$ cut in $\check{H}$, and so there is a set $\mathcal{P}_1 = \{P(e) \mid e \in E(A_1, B_1)\}$ of edge-disjoint paths, that are internally disjoint from $B_1$, where for each edge $e \in E(A_1, B_1)$, path $P(e)$ has $e$ as its first edge and vertex $u_i$ as its last vertex. Similarly, $(A_2, B_2)$ is a minimum $u_{i+1}$–$u_{i+2}$ cut in $\check{H}$, and so there is a set $\mathcal{P}_2 = \{P'(e) \mid e \in E(A_2, B_2)\}$ of edge-disjoint paths, that are internally disjoint from $A_2$, where for each edge $e \in E(A_2, B_2)$, path $P'(e)$ has $e$ as its first edge and vertex $u_{i+2}$ as its last vertex.

We partition the edge set $\delta(S^*)$ into three subsets (see Figure 64). The first subset, that we denote by $\delta_1$, contains all edges of $\delta(S^*)$ that lie in the set $E(A_2, B_2)$. The second set, that we denote by $\delta_2$, contains all edges of $\delta(S^*)$ that lie in $\delta^{\text{down}}(S''_{i+1})$ – that is, they connect $u_{i+1}$ to vertices of $S''_{i+1}$. The third set $\delta_3$ contains all remaining edges. Note that every edge of $\delta_3$ must lie in $E_i \cup \delta^{\text{left}}(S'_{i+1}) \subseteq E(A_1, B_1)$. We now consider each of the three sets of edges in turn.

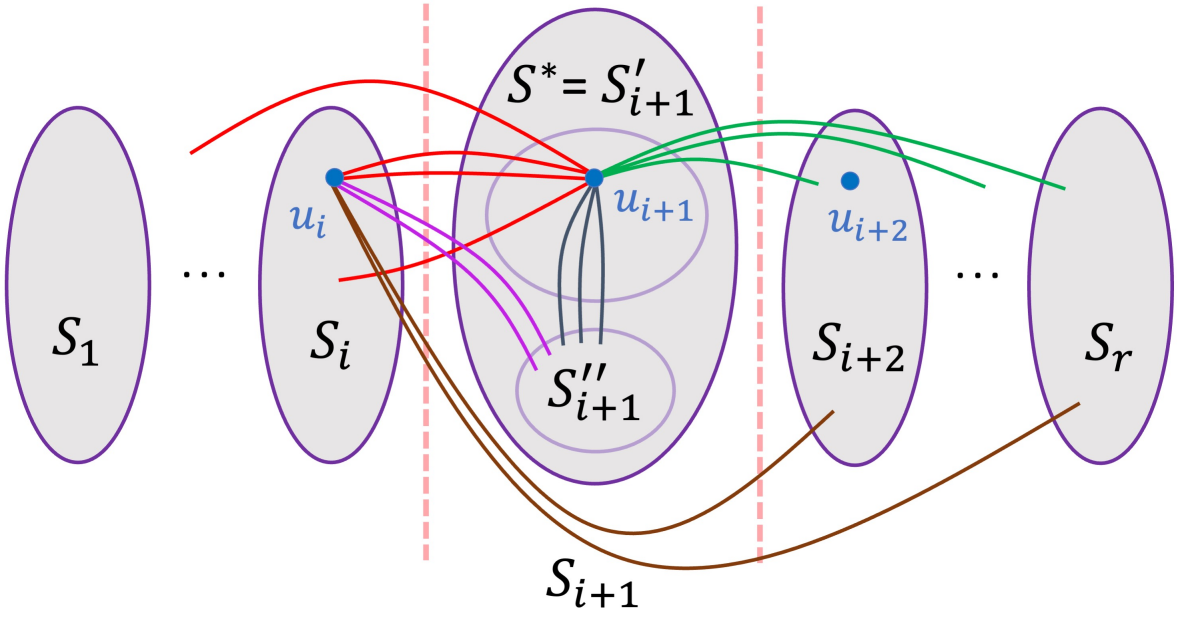



Figure 64: Partition of the set $\delta(S^*)$ of edges into three subsets: set $\delta_1$ (shown in green), set $\delta_2$ (shown in black), and set $\delta_3$ (shown in red). The pink dashed line on the left shows cut $(A_1, B_1)$, and the pink dashed line on the right shows cut $(A_2, B_2)$.

First, for every edge $e \in \delta_1$, we let $P^*(e) = P'(e)$, where $P'(e)$ is the path of $\mathcal{P}_2$, that has $e$ as its first edge, vertex $u_{i+2}$ as its last vertex, and is internally disjoint from $S^*$. We set $\mathcal{P}^*_1 = \{P^*(e) \mid e \in \delta_1\}$. Clearly, $\mathcal{P}^*_1 \subseteq \mathcal{P}_2$, and the paths in $\mathcal{P}^*_1$ are edge-disjoint.

Next, we consider the edges of $\delta_2 = \delta^{\text{down}}(S''_{i+1})$. Recall that, from Claim 7.30, there is a set $\mathcal{P}^{\text{right}} = \left\{P^{\text{right}}(e) \mid e \in \delta_2\right\}$ of edge-disjoint paths in $H$, where, for each edge $e \in \delta_2$, path $P^{\text{right}}(e)$ has $e$ as its first edge, some edge of $\delta^{\text{right}}(S''_{i+1})$ as its last edge, and all inner vertices of $P^{\text{right}}(e)$ are contained in $S''_{i+1}$, so that the paths of $\mathcal{P}^{\text{right}}$ are internally disjoint from $S^*$. Consider an edge $e \in \delta_2$, and let

$e' \in \delta^{\text{right}}(S''_{i+1})$ be the last edge on the path $P^{\text{right}}(e)$. Then $e' \in E(A_2, B_2)$. We let $P^*(e)$ be the path obtained by concatenating the path $P^{\text{right}}(e)$ with the path $P'(e') \in \mathcal{P}_2$. Clearly, path $P^*(e)$ has $e$ as its first edge, vertex $u_{i+2}$ as its last vertex, and it is internally disjoint from $S^*$. We set $\mathcal{P}_2^* = \{P^*(e) \mid e \in \delta_2\}$. It is easy to verify that the paths of $\mathcal{P}_2^*$ are edge-disjoint.

Lastly, we consider the edges of $\delta_3 \subseteq E(A_1, B_1)$. Observe that a cut $(A'_1, B'_1)$, where $A'_1 = \{u_i\}$, and $B'_1 = V(\breve{H}) \setminus A'_1$ is a $u_i$–$u_{i+1}$-cut in graph $\breve{H}$, and so $|\delta_3| \leq |E(A_1, B_1)| \leq |E(A'_1, B'_1)| \leq |\delta(u_i)|$. We denote by $\tilde{E}'$ the set of all edges connecting $u_i$ to vertices of $S''_{i+1}$, and we denote by $\tilde{E}''$ the set of all edges connecitng $u_i$ to vertices of $S_{i+2} \cup \cdots \cup S_r$.

Recall that set $E^*$ of edges contains all edges connecting $u_i$ to vertices of $\bigcup_{a>i} S_a$. Let $\breve{E}_i \subseteq E_i$ be the set of edges connecting $u_i$ to $u_{i+1}$. Since $S'_{i+1} = \{u_{i+1}\}$, $E^* = \tilde{E}' \cup \tilde{E}'' \cup \breve{E}_i$. From our assumtion, $|E^*| \geq |\delta(u_i)|/4$. Furthermore, since vertex $u_i$ was not added to the $J$-cluster corresponding to vertex $u_{i+1}$, $|\breve{E}_i| \leq |\delta(u_i)|/128$. Therefore, either $|\tilde{E}'| \geq |\delta(u_i)|/16$, or $|\tilde{E}''| \geq |\delta(u_i)|/16$ must hold.

We assume first that it is the latter. Since $|\delta_3| \leq |\delta(u_i)|$, we can define a mapping $M$ from the edges of $\delta_3$ to edges of $\tilde{E}''$, where, for each edge $e' \in \tilde{E}''$, at most 16 edges of $\delta_3$ are mapped to $e'$. Observe that $\tilde{E}'' \subseteq E(A_2, B_2)$. Consider now some edge $e \in \delta_3$. We obtain the path $P^*(e)$ by concatenating two paths: path $P(e) \in \mathcal{P}_1$, connecting $e$ to vertex $u_i$, and the path $P'(e') \in \mathcal{P}_2$, where $e'$ is the edge of $\tilde{E}''$ to which $e$ is mapped. Recall that $P'(e')$ has $e'$ as its first edge and $u_{i+2}$ as its last vertex; edge $e'$ is incident to $u_i$. Therefore, path $P^*(e)$ has $e$ as its first edge, vertex $u_{i+2}$ as its last vertex, and it is internally disjoint from $S^*$. We then set $\mathcal{P}_3^* = \{P^*(e) \mid e \in \delta_3\}$. It is easy to verify that the paths of $\mathcal{P}_3^*$ cause edge-congestion at most 17.

Lastly, we assume that $|\tilde{E}'| \geq |\delta(u_i)|/16$. As before, we define a mapping $M$ from edges of $\delta_3$ to edges of $\tilde{E}'$, where, for each edge $e' \in \tilde{E}'$, at most 16 edges of $\delta_3$ are mapped to $e'$. Consider now some edge $e' = (u_i, v) \in \tilde{E}'$, and recall that $v \in S''_{i+1}$. Recall that the algorithm from Lemma 7.27 provided a construction of a right-monotone path $P(e', v)$. This path starts with edge $e'$, and it must terminate at some vertex of $V(S'_{i+2}) \cup V(S'_{i+3}) \cup \cdots \cup V(S'_r)$. All inner vertices on path $P(e', v)$ must lie in $V(S''_{i+1}) \cup V(S''_{i+2}) \cup \cdots \cup V(S''_r)$. Therefore, if $e''$ is the first edge of $P(e', v)$ that is not contained in $S''_{i+1}$, then $e'' \in E(A_2, B_2)$. We denote by $\tilde{P}(e')$ the subpath of $P(e', v)$ that starts with edge $e'$ and ends with edge $e''$. From Lemma 7.27, we are guaranteed that all paths in set $\left\{\tilde{P}(e') \mid e' \in \tilde{E}'\right\}$ cause congestion $O(\log m)$. Consider now some edge $e \in \delta_3$. We let $P^*(e)$ be a path that is a concatenation of three paths. The first path is the path $P(e) \in \mathcal{P}_1$, that connects $e$ to $u_i$. Let $e' \in \tilde{E}'$ be the edge to which $e$ is mapped by $M$. The second path is $\tilde{P}(e')$, connecting $e'$ to some edge $e'' \in E(A_2, B_2)$. The third path is the path $P'(e'') \in \mathcal{P}_2$, connecting $e''$ to vertex $u_{i+2}$. It is immediate to verify that the resulting path $P^*(e)$ has $e$ as its first edge, $u_{i+2}$ as its last vertex, and it is internally disjoint from $S^*$. We then set $\mathcal{P}_3^* = \{P^*(e) \mid e \in \delta_3\}$. It is easy to verify that the paths of $\mathcal{P}_3^*$ cause congestion at most $O(\log m)$.

Finally, we set $\mathcal{P}^* = \mathcal{P}_1^* \cup \mathcal{P}_2^* \cup \mathcal{P}_3^*$. From our construction, the set $\mathcal{P}^*$ of paths routes the edges of $\delta(S^*)$ to vertex $u_{i+2}$, the paths of $\mathcal{P}^*$ are internally disjoint from $S^*$, and they cause edge-congestion $O(\log m)$ as required. We conclude that $S^*$ is a simplifying cluster.

## G.19 Proof of Claim 7.41

Since $u_{i+2}$ is a $J$-node, in order to prove that $S^*$ is a simplifying cluster, it is enough to show a collection $\mathcal{P}^* = \{P^*(e) \mid e \in \delta(u_{i+2})\}$ of paths in graph $\breve{H}$, where for each edge $e \in \delta(u_{i+2})$, path $P^*(e)$ has $e$ as its first edge, vertex $u_{i+3}$ as its last vertex, and is internally disjoint from $S^*$. The construction of the set $\mathcal{P}^*$ of paths is almost identical to that from Case 2.

As before, we define two cuts in graph $\breve{H}$: cut $(A_1, B_1)$, with $A_1 = V(S_1) \cup \cdots \cup V(S_{i+1})$ and $B_1 = V(\breve{H}) \setminus A_1$, and cut $(A_2, B_2)$, with $A_2 = V(S_1) \cup \cdots \cup V(S_{i+2})$ and $B_2 = V(\breve{H}) \setminus A_2$ (see

Figure 65). As before, from our construction, $(A_1, B_1)$ is a minimum $u_{i+1}$–$u_{i+2}$ cut in $\check{H}$, and so there is a set $\mathcal{P}_1 = \{P(e) \mid e \in E(A_1, B_1)\}$ of edge-disjoint paths, that are internally disjoint from $B_1$, where for each edge $e \in E(A_1, B_1)$, path $P(e)$ has $e$ as its first edge and vertex $u_{i+1}$ as its last vertex. Similarly, $(A_2, B_2)$ is a minimum $u_{i+2}$–$u_{i+3}$ cut in $\check{H}$, and so there is a set $\mathcal{P}_2 = \{P'(e) \mid e \in E(A_2, B_2)\}$ of edge-disjoint paths, that are internally disjoint from $A_2$, where for each edge $e \in E(A_2, B_2)$, path $P'(e)$ has $e$ as its first edge and vertex $u_{i+3}$ as its last vertex.

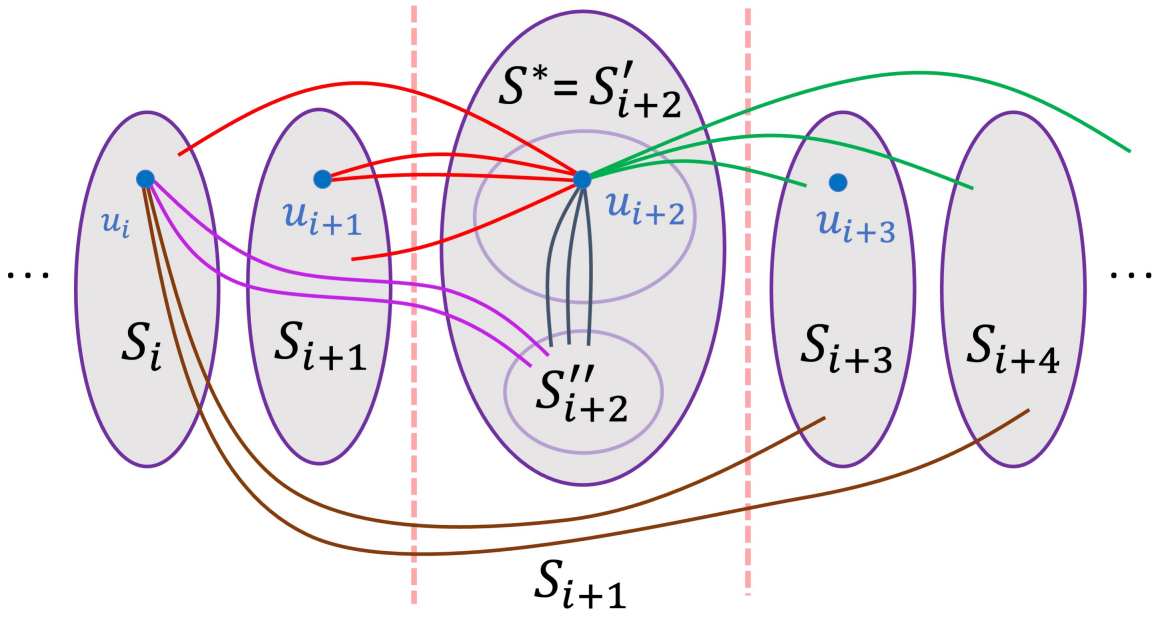



Figure 65: Illustration for the proof of Claim 7.41. Edges of $\delta_1$ are shown in green, edges of $\delta_2$ are shown in black, and edges of $\delta_3$ are shown in red. The pink dashed line on the left shows cut $(A_1, B_1)$ and the pink dashed line on the right shows cut $(A_2, B_2)$.

As before, we partition the edge set $\delta(S^*)$ into three subsets. The first subset, that we denote by $\delta_1$, contains all edges of $\delta(S^*)$ that lie in the set $E(A_2, B_2)$. The second set, that we denote by $\delta_2$, contains all edges of $\delta(S^*)$ that lie in $\delta^{\text{down}}(S''_{i+2})$ – that is, they connect $u_{i+2}$ to vertices of $S''_{i+2}$. The third set $\delta_3$ contains all remaining edges. As before, $\delta_3$ must lie in $E_{i+1} \cup \delta^{\text{left}}(S'_{i+2}) \subseteq E(A_1, B_1)$. We consider each of the three sets of edges in turn.

The constructions of the path sets $\mathcal{P}^*_1 = \{P^*(e) \mid e \in \delta_1\}$ and $\mathcal{P}^*_2 = \{P^*(e) \mid e \in \delta_2\}$ remain exactly the same as in Case 2. We now focus on constructing the set $\mathcal{P}^*_3 = \{P^*(e) \mid e \in \delta_3\}$ of paths.

Consider the edges of $\delta_3 \subseteq E(A_1, B_1)$. Recall that we have assumed that $u_{i+1}$ is not a $J$-node. From the choice of the index $i^* = i$, $|\delta(u_{i+1})| \leq |\delta(u_i)|$. As before, we consider another $u_{i+1}$–$u_{i+2}$ cut $(A'_1, B'_1)$, where $A'_1 = \{u_{i+1}\}$, and $B'_1 = V(\check{H}) \setminus A'_1$. As before, $|\delta_3| \leq |E(A_1, B_1)| \leq |E(A'_1, B'_1)| \leq |\delta(u_{i+1})| \leq |\delta(u_i)|$.

We denote by $\tilde{E}'$ the set of all edges connecting $u_i$ to vertices of $S''_{i+2}$, and by $\tilde{E}''$ the set of all edges connecitng $u_i$ to vertices of $S_{i+3} \cup \cdots \cup S_r$. Since $u_{i+2}$ is a $J$-node, $S'_{i+2} = \{u_{i+2}\}$. As before, since vertex $u_i$ was not added to the $J$-cluster corresponding to vertex $u_{i+2}$, the number of edges connecting $u_i$ to $u_{i+2}$ is bounded by $|\delta(u_i)|/128$. Recall that we have established that at least $|\delta(u_i)|/8$ edges connect $u_i$ to vertices of $\bigcup_{a>i+1} V(S_a)$. Each such edge either lies in $\tilde{E}' \cup \tilde{E}''$, or it connects $u_i$ to $u_{i+2}$. Therefore, either $|\tilde{E}'| \geq |\delta(u_i)|/32$, or $|\tilde{E}''| \geq |\delta(u_i)|/32$ must hold. The remainder of the construction of the paths in $\mathcal{P}^*_3$ is very similar to that for Case 2.

We assume first that $|\tilde{E}''| \geq |\delta(u_i)|/32$. Since $|\delta_3| \leq |\delta(u_i)|$, we can define a mapping $M$ from the edges of $\delta_3$ to edges of $\tilde{E}''$, where, for each edge $e' \in \tilde{E}''$, at most 32 edges of $\delta_3$ are mapped to $e'$. Observe that $\tilde{E}'' \subseteq E(A_1, B_1)$ and $\tilde{E}'' \subseteq E(A_2, B_2)$. Consider now some edge $e \in \delta_3$. We obtain the path $P^*(e)$ by concatenating three paths. The first path is path $P(e) \in \mathcal{P}_1$, connecting $e$ to vertex $u_{i+1}$. Denote by $e'$ the edge of $\tilde{E}''$ to which edge $e$ is mapped. The second path is path $P(e') \in \mathcal{P}_1$ (which we reverse), connecting vertex $u_{i+1}$ to edge $e'$. The third path is path $P'(e') \in \mathcal{P}_2$, connecting edge $e'$ to vertex $u_{i+3}$. Clearly, path $P^*(e)$ has $e$ as its first edge, vertex $u_{i+3}$ as its last vertex, and it is internally disjoint from $S^*$. We then set $\mathcal{P}_3^* = \{P^*(e) \mid e \in \delta_3\}$. It is easy to verify that the paths of $\mathcal{P}_3^*$ cause edge-congestion at most $O(1)$.

Finally, we assume that $|\tilde{E}'| \geq |\delta(u_i)|/32$. Recall that the edges of $\tilde{E}'$ connect vertex $u_i$ to vertices of $S''_{i+2}$, and in particular $\tilde{E}' \subseteq E(A_1, B_1)$. As before, we define a mapping $M$ from edges of $\delta_3$ to edges of $\tilde{E}'$, where, for each edge $e' \in \tilde{E}'$, at most 32 edges of $\delta_3$ are mapped to $e'$.

Consider now some edge $e' = (u_i, v) \in \tilde{E}'$, and recall that $v \in S''_{i+1}$. Recall that the algorithm from Lemma 7.27 provides a construction of a right-monotone path $P(e', v)$. This path starts with edge $e'$, and it must terminate at some vertex of $V(S'_{i+3}) \cup V(S'_{i+4}) \cup \cdots \cup V(S'_r)$. All inner vertices on path $P(e', v)$ must lie in $V(S''_{i+2}) \cup V(S''_{i+3}) \cup \cdots \cup V(S''_r)$. Therefore, if $e''$ is the first edge of $P(e', v)$ that is not contained in $S''_{i+2}$, then $e'' \in E(A_2, B_2)$. We denote by $\tilde{P}(e')$ the subpath of $P(e', v)$ that starts with edge $e'$ and ends with edge $e''$. From Lemma 7.27, we are guaranteed that all paths in set $\left\{\tilde{P}(e') \mid e' \in \tilde{E}'\right\}$ cause congestion $O(\log m)$. Consider now some edge $e \in \delta_3$. We let $P^*(e)$ be a path that is a concatenation of four paths. The first path is the path $P(e) \in \mathcal{P}_1$, that connects $e$ to $u_{i+1}$. Let $e' \in \tilde{E}'$ be the edge to which $e$ is mapped by $M$, and recall that $e' \in E(A_1, B_1)$. The second path is $P(e') \in \mathcal{P}_1$, which we reverse, so the path now connects vertex $u_{i+1}$ to edge $e'$. The third path is $\tilde{P}(e')$, connecting $e'$ to some edge $e'' \in E(A_2, B_2)$. The fourth and the last path is the path $P'(e'') \in \mathcal{P}_2$, connecting $e''$ to vertex $u_{i+3}$. It is immediate to verify that the resulting path $P^*(e)$ has $e$ as its first edge, $u_{i+3}$ as its last vertex, and it is internally disjoint from $S^*$. We then set $\mathcal{P}_3^* = \{P^*(e) \mid e \in \delta_3\}$. From the above discussion, the paths of $\mathcal{P}_3^*$ cause congestion at most $O(\log m)$.

Lastly, we set $\mathcal{P}^* = \mathcal{P}_1^* \cup \mathcal{P}_2^* \cup \mathcal{P}_3^*$. It is easy to verify that the set $\mathcal{P}^*$ of paths routes the edges of $\delta(S^*)$ to vertex $u_{i+3}$, the paths are internally disjoint from $S^*$, and they cause congestion $O(\log m)$ as required. We conclude that $S^*$ is a simplifying cluster.

## G.20 Proof of Claim 7.42

Since $u_{i+1}$ is a $J$-node, it is enough to show a collection $\mathcal{P}^* = \{P^*(e) \mid e \in \delta(u_{i+1})\}$ of paths in graph $\check{H}$, where for each edge $e \in \delta(u_{i+1})$, path $P^*(e)$ has $e$ as its first edge, vertex $u_{i+2}$ as its last vertex, and is internally disjoint from $S^*$. As before, we define two cuts in graph $\check{H}$: cut $(A_1, B_1)$, with $A_1 = V(S_1) \cup \cdots \cup V(S_i)$ and $B_1 = V(\check{H}) \setminus A_1$, and cut $(A_2, B_2)$, with $A_2 = V(S_1) \cup \cdots \cup V(S_{i+1})$ and $B_2 = V(\check{H}) \setminus A_2$. As before, from our construction, $(A_1, B_1)$ is a minimum $u_i$–$u_{i+1}$ cut in $\check{H}$, and so there is a set $\mathcal{P}_1 = \{P(e) \mid e \in E(A_1, B_1)\}$ of edge-disjoint paths, that are internally disjoint from $B_1$, where for each edge $e \in E(A_1, B_1)$, path $P(e)$ has $e$ as its first edge and vertex $u_i$ as its last vertex. Similarly, $(A_2, B_2)$ is a minimum $u_{i+1}$–$u_{i+2}$ cut in $\check{H}$, and so there is a set $\mathcal{P}_2 = \{P'(e) \mid e \in E(A_2, B_2)\}$ of edge-disjoint paths, that are internally disjoint from $A_2$, where for each edge $e \in E(A_2, B_2)$, path $P'(e)$ has $e$ as its first edge and vertex $u_{i+2}$ as its last vertex.

As before, we partition the set $\delta(S^*)$ of edges into three subsets (see Figure 66). The first subset, that we denote by $\delta_1$, contains all edges of $\delta(S^*)$ that lie in the set $E(A_2, B_2)$. The second set, that we denote by $\delta_2$, contains all edges of $\delta(S^*)$ that lie in $\delta^{\text{down}}(S''_{i+1})$ – that is, they connect $u_{i+1}$ to vertices of $S''_{i+1}$. The third set, $\delta_3$, contains all remaining edges. As before, very edge of $\delta_3$ must lie in $E_i \cup \delta^{\text{left}}(S'_{i+1}) \subseteq E(A_1, B_1)$. We now consider each of the three sets of edges in turn.

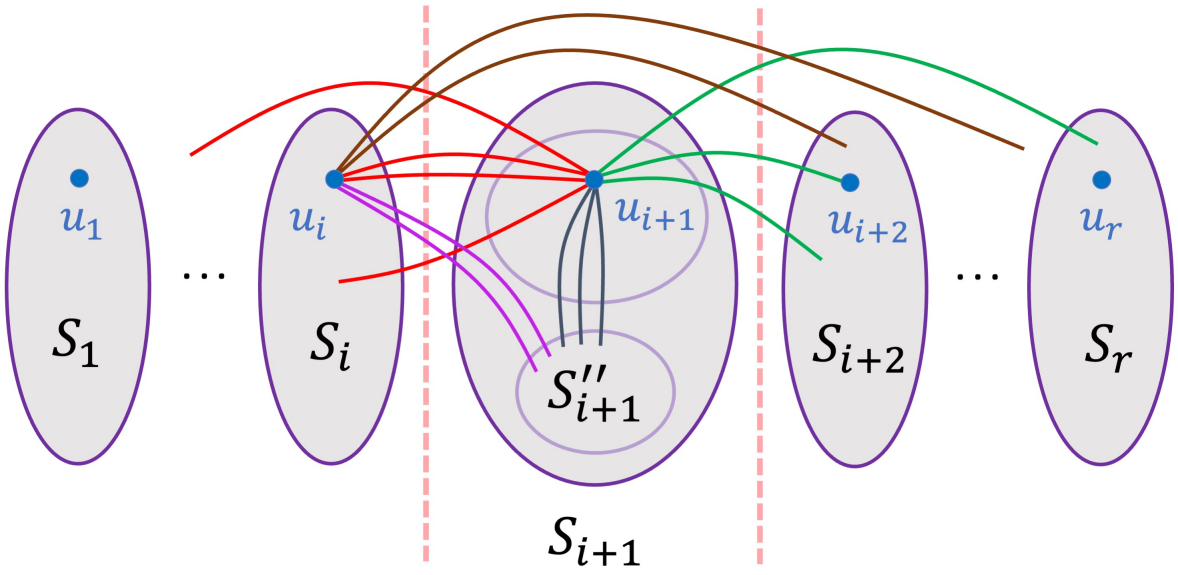



Figure 66: Illustration for the proof of Claim 7.42. Edges of $\delta_1$ are shown in green, edges of $\delta_2$ are shown in black, and edges of $\delta_3$ are shown in red. Additionally, edges of $\tilde{E}'$ are shown in purple and edges of $\tilde{E}''$ are shown in brown. The pink dashed line on the left shows cut $(A_1, B_1)$ and the pink dashed line on the right shows cut $(A_2, B_2)$.

First, for every edge $e \in \delta_1$, we let $P^*(e) = P'(e)$, where $P'(e)$ is the path of $\mathcal{P}_2$, that has $e$ as its first edge, vertex $u_{i+2}$ as its last vertex, and is internally disjoint from $S^*$. We set $\mathcal{P}_1^* = \{P^*(e) \mid e \in \delta_1\}$. Clearly, $\mathcal{P}_1^* \subseteq \mathcal{P}_2$, and the paths in $\mathcal{P}_1^*$ are edge-disjoint.

Next, we consider the edges of $\delta_2 = \delta^{\text{down}}(S''_{i+1})$. Recall that, from Claim 7.30, there is a set $\mathcal{P}^{\text{right}} = \left\{P^{\text{right}}(e) \mid e \in \delta_2\right\}$ of edge-disjoint paths in $H$, where, for each edge $e \in \delta_2$, path $P^{\text{right}}(e)$ has $e$ as its first edge, some edge of $\delta^{\text{right}}(S''_{i+1})$ as its last edge, and all inner vertices of $P^{\text{right}}(e)$ are contained in $S''_{i+1}$, so that the paths of $\mathcal{P}^{\text{right}}$ are internally disjoint from $S^*$. Consider an edge $e \in \delta_2$, and let $e' \in \delta^{\text{right}}(S''_{i+1})$ be the last edge on the path $P^{\text{right}}(e)$. Then $e' \in E(A_2, B_2)$. We let $P^*(e)$ be the path obtained by concatenating the path $P^{\text{right}}(e)$ with the path $P'(e') \in \mathcal{P}_2$. Clearly, path $P^*(e)$ has $e$ as its first edge, vertex $u_{i+2}$ as its last vertex, and it is internally disjoint from $S^*$. We set $\mathcal{P}_2^* = \{P^*(e) \mid e \in \delta_2\}$. It is easy to verify that the paths of $\mathcal{P}_2^*$ are edge-disjoint.

Lastly, we consider the edges of $\delta_3 \subseteq E(A_1, B_1)$. Clearly, $|\delta_3| \leq |E_i| + |\hat{E}_i| \leq 8|E_i| + 7|\delta^{\text{right}}(S'_i)| + 7|\delta^{\text{left}}(S'_{i+1})|$, from Equation (6). Note that, if $i = 1$, then, from Observation 7.24, $S'_1 = S_1$, and so $\delta^{\text{left}}(S'_2) = \emptyset$. Otherwise, from Claim 7.33, $|\delta^{\text{left}}(S'_{i+1})| \leq 2.5|E_i| + 2.5|\delta^{\text{right}}(S'_i)|$. In either case, we get that:

$$|\delta_3| \leq 30|E_i| + 29|\delta^{\text{right}}(S'_i)| \leq 30|\delta^{\text{right}}(S'_i)|,$$

since have assumed that $|\delta^{\text{right}}(S'_i)| > 64|E_i|$. We partition the set $\delta^{\text{right}}(S'_i)$ into two subsets: set $\tilde{E}'$, containing all edges $(u_i, v)$, with $v \in S''_{i+1}$, and set $\tilde{E}''$, containing all remaining edges. Note that, for each edge $(u_i, v) \in \tilde{E}''$, $v \in S_{i+2} \cup \cdots \cup S_r$ must hold. The remainder of the construction of the set $\mathcal{P}_3^* = \{P^*(e) \mid e \in \delta_3\}$ of paths is very similar to the analysis of Case 2 in the proof of Lemma 7.35. Since $|\delta_3| \leq 30|\delta^{\text{right}}(S'_i)|$, either $|\tilde{E}'| \geq |\delta_3|/60$ or $|\tilde{E}''| \geq |\delta_3|/60$ must hold. Assume first that it is the latter. Then we can define a map $M$ from the edges of $\delta_3$ to edges of $\tilde{E}''$, where, for each edge

$e' \in \tilde{E}''$, at most 60 edges of $\delta_3$ are mapped to $e'$. Observe that $\tilde{E}'' \subseteq E(A_2, B_2)$. Consider now some edge $e \in \delta_3$. We obtain the path $P^*(e)$ by concatenating two paths: path $P(e) \in \mathcal{P}_1$, connecting $e$ to vertex $u_i$, and the path $P'(e') \in \mathcal{P}_2$, where $e'$ is the edge of $\tilde{E}''$ to which $e$ is mapped. Recall that path $P'(e')$ has $e'$ as its first edge and $u_{i+2}$ as its last vertex, and that edge $e'$ is incident to $u_i$. Therefore, path $P^*(e)$ has $e$ as its first edge, vertex $u_{i+2}$ as its last vertex, and it is internally disjoint from $S^*$. We then set $\mathcal{P}_3^* = \{P^*(e) \mid e \in \delta_3\}$. It is easy to verify that the paths of $\mathcal{P}_3^*$ cause edge-congestion at most 60.

Lastly, we assume that $|\tilde{E}'| \geq |\delta_3|/60$. As before, we define a mapping $M$ from edges of $\delta_3$ to edges of $\tilde{E}'$, where, for each edge $e' \in \tilde{E}'$, at most 60 edges of $\delta_3$ are mapped to $e'$. Consider now some edge $e' = (u_i, v) \in \tilde{E}'$, and recall that $v \in S''_{i+1}$. Recall that the algorithm from Lemma 7.27 provided a construction of a right-monotone path $P(e', v)$. This path starts with edge $e'$, and it must terminate in some vertex of $V(S'_{i+2}) \cup V(S'_{i+3}) \cup \cdots \cup V(S'_r)$. All inner vertices on path $P(e', v)$ must lie in $V(S''_{i+1}) \cup V(S''_{i+2}) \cup \cdots \cup V(S''_r)$. Therefore, if $e''$ is the first edge of $P(e', v)$ that is not contained in $S''_{i+1}$, then $e'' \in E(A_2, V_2)$. We denote by $\tilde{P}(e')$ the subpath of $P(e', v)$ that starts with edge $e'$ and ends with edge $e''$. From Lemma 7.27, we are guaranteed that all paths in set $\left\{\tilde{P}(e') \mid e' \in \tilde{E}'\right\}$ cause congestion $O(\log m)$. Consider now some edge $e \in \delta_3$. We let $P^*(e)$ be a path that is a concatenation of three paths. The first path is the path $P(e) \in \mathcal{P}_1$, that connects $e$ to $u_i$. Let $e' \in \tilde{E}'$ be the edge to which $e$ is mapped by $M$. The second path is $\tilde{P}(e')$, connecting $e'$ to some edge $e'' \in E(A_2, B_2)$. The third path is the path $P'(e'') \in \mathcal{P}_2$, connecting $e''$ to vertex $u_{i+2}$. It is immediate to verify that the resulting path $P^*(e)$ has $e$ as its first edge, $u_{i+2}$ as its last vertex, and it is internally disjoint from $S^*$. We then set $\mathcal{P}_3^* = \{P^*(e) \mid e \in \delta_3\}$. From the above discussion, the paths of $\mathcal{P}_3^*$ cause congestion at most $O(\log m)$.

Finally, we set $\mathcal{P}^* = \mathcal{P}_1^* \cup \mathcal{P}_2^* \cup \mathcal{P}_3^*$. It is easy to verify that the set $\mathcal{P}^*$ of paths routes the edges of $\delta(S^*)$ to vertex $u_{i+2}$ with congestion $O(\log m)$, and all paths of $\mathcal{P}^*$ are internally disjoint from $S^*$. We conclude that $S^*$ is a simplifying cluster, a contradiction.

## G.21 Proof of Claim 7.44

We provide the construction of the set $\mathcal{P}^{\mathsf{out,left}}$ of paths; the construction of the set $\mathcal{P}^{\mathsf{out,right}}$ of paths is symmetric.

We maintain a set $\mathcal{R} = \left\{R(e) \mid e \in \hat{E}\right\}$ of paths, that we gradually modify over the course of the algorithm. We will ensure that, throughout the algorithm, $\mathcal{R}$ is a collection of simple paths that contains, for each edge $e \in \hat{E}$, a path $R(e)$ that originates at $e$ and is a left-monotone path. Additionally, for every vertex $v \in V'$, the number of paths of $\mathcal{R}$ terminating at $v$ is $n_1(v)$ throughout the algorithm. Initially, for each edge $e \in \hat{E}$, we let $R(e)$ be the path obtained by appending edge $e$ at the beginning of the path $P^1(e) \in \hat{\mathcal{P}}(e)$, and we set $\mathcal{R} = \left\{R(e) \mid e \in \hat{E}\right\}$. Clearly, all invariants hold for the initial set $\mathcal{R}$ of paths.

We perform the algorithm as long as there are two paths $R(e), R(e') \in \mathcal{R}$, and some vertex $v$ that lies on both paths, such that the intersection of $R(e)$ and $R(e')$ at $v$ is transversal. Note that $v$ must be an inner vertex on both $R(e)$ and $R(e')$. Assume that path $R(e)$ terminates at some vertex $u \in V'$, while path $R(e')$ terminates at vertex $u' \in V'$. We perform splicing of paths $R(e), R(e')$ at vertex $v$ (see Section 4.1.4), obtaining two new paths: path $\tilde{R}(e)$, whose first edge is $e$ and last vertex is $u'$; and path $\tilde{R}(e')$, whose first edge is $e'$ and last vertex is $u$. If any of the resulting paths $\tilde{R}(e), \tilde{R}(e')$ is non-simple, we remove cycles from it, until it becomes a simple path. We then update the set $\mathcal{R}$ of paths by replacing $R(e)$ and $R(e')$ with paths $\tilde{R}(e)$ and $\tilde{R}(e')$, respectively. It is easy to verify that, if $R(e), R(e')$ were both left-monotone paths, then so are paths $\tilde{R}(e)$ and $\tilde{R}(e')$. It is also immediate to verify that all other invariants hold. Clearly, when the algorithm terminates, the final set $\mathcal{P}^{\mathsf{out,left}} = \mathcal{R}$

of paths has all required properties.

It now remains to show that the algorithm is efficient. From Observation 4.6, after every iteration, either (i) $\sum_{R\in\mathcal{R}}|E(R)|$ decreases, or (ii) $|\Pi^T(\mathcal{R})|$ decreases, and $\sum_{R\in\mathcal{R}}|E(R)|$ remains fixed. It is then immediate to verify that the algorithm terminates after poly($|E(G)|$) iterations.

## G.22 Proof of Claim 7.46

In order to prove the claim, we use the following observation.

**Observation G.6** *Let $\mathcal{I}$ be a collection of $k$ intervals of non-zero length, where for each interval $I\in\mathcal{I}$, $I\subseteq[0,r]$, interval $I$ is closed on the left and open on the right, and the endpoints of $I$ are integers in $\{0,\ldots,r\}$. Let $\mathcal{I}'$ be another collection of $k$ intervals of non-zero length, where for each interval $I'\in\mathcal{I}'$, $I'\subseteq[0,r]$, interval $I'$ is closed on the left and open on the right, and the endpoints of $I'$ are integers in $\{0,\ldots,r\}$. Assume further that for every integer $0\leq i\leq r$, the number of intervals of $\mathcal{I}$ for which $i$ serves as the left endpoint is equal to the number of intervals of $\mathcal{I}'$ for which $i$ serves as the left endpoint, and similarly, the number of intervals of $\mathcal{I}$ for which $i$ serves as the right endpoint is equal to the number of intervals of $\mathcal{I}'$ for which $i$ serves as the right endpoint. Then for every integer $p\in[0,r)$, the total number of intervals in $\mathcal{I}$ containing $p$ is equal to the total number of intervals in $\mathcal{I}'$ containing $p$.*

**Proof:** Let $p$ be any integer in $[0,r)$. Consider any interval $I=(a,b)\in\mathcal{I}\cup\mathcal{I}'$. Clearly, $p\in I$ iff $a\leq p<b$.

Let $\mathcal{I}_1\subseteq\mathcal{I}$ be the set of all intervals $I=[a,b)\in\mathcal{I}$ with $a\leq p$, and let $\mathcal{I}_2\subseteq\mathcal{I}$ be the set of all intervals $I=[a,b)\in\mathcal{I}$ with $b\leq p$. Clearly, $\mathcal{I}_2\subseteq\mathcal{I}_1$, and an interval $I\in\mathcal{I}$ contains the point $p$ iff $I\in\mathcal{I}_1\setminus\mathcal{I}_2$. We define subsets $\mathcal{I}'_1,\mathcal{I}'_2$ of intervals of $\mathcal{I}'$ similarly. As before, an interval $I'\in\mathcal{I}'$ contains the point $p$ iff $I'\in\mathcal{I}'_1\setminus\mathcal{I}'_2$. Since, for every integer $0\leq i\leq p$, the number of intervals in $\mathcal{I}$ whose left endpoint is $i$ is equal to the number of intervals in $\mathcal{I}'$ whose left endpoint is $i$, we get that $|\mathcal{I}_1|=|\mathcal{I}'_1|$. Similarly, since, for every integer $0\leq i\leq p$, the number of intervals in $\mathcal{I}$ whose right endpoint is $i$ is equal to the number of intervals in $\mathcal{I}'$ whose right endpoint is $i$, we get that $|\mathcal{I}_2|=|\mathcal{I}'_2|$. Therefore, $|\mathcal{I}_1\setminus\mathcal{I}_2|=|\mathcal{I}'_2\setminus\mathcal{I}'_2|$. Since $\mathcal{I}_1\setminus\mathcal{I}_2$ is precisely the set of all intervals $I\in\mathcal{I}$ that contain $p$, and $\mathcal{I}'_1\setminus\mathcal{I}'_2$ is precisely the set of all intervals $I'\in\mathcal{I}'$ that contiain $p$, the observation follows. □

We construct two collections of intervals, $\mathcal{I}=\left\{I(e)\mid e\in\hat{E}\right\}$, and $\mathcal{I}'=\left\{I'(e)\mid e\in\hat{E}\right\}$, as follows. Consider an edge $e\in\hat{E}$. Assume that $\mathrm{span}'(e)=\{i',i'+1,\ldots,j'-1\}$, and that $\mathrm{span}''(e)=\{i'',i''+1,\ldots,j''-1\}$. We then let $I(e)=[i',j')$ and $I'(e)=[i'',j'')$.

Let $L$ be the multiset of integers, that serve as the left endpoint of every interval in $\mathcal{I}$. Consider an integer $1\leq i\leq r$. The number of times that $i$ appears in set $L$ is equal to the number of edges $e\in\hat{E}$, such that $i$ is the first element of $\mathrm{span}'(e)$; equivalently, the prefix path $P^1(e)$ must terminate at a vertex of $S_i$. Therefore, the number of times that integer $i$ appears in $L$ is $\sum_{v\in V(S_i)}n_1(v)$. Note that, if $i$ is the left endpoint of some interval $I'(e)\in\mathcal{I}'$, then path $P^{\mathsf{out}}(e)$ must originate at a vertex of $S_i$. Therefore, path $P^{\mathsf{out,left}}(e)$ that was constructed by the algorithm from Claim 7.44 must terminate at a vertex of $S_i$. From Claim 7.44, for every vertex $v\in V'$, the number of paths of $\mathcal{P}^{\mathsf{out,left}}$ terminating at $v$ is exactly $n_1(v)$. Therefore, the number of intervals of $\mathcal{I}'$ for which $i$ serves as the left endpoint is equal to $\sum_{v\in V(S_i)}n_1(v)$, which is exactly the number of intervals of $\mathcal{I}$, for which $i$ serves as the left endpoint.

Using similar reasoning, for every integer $1\leq i\leq r$, the number of intervals of $\mathcal{I}'$ for which $i$ serves as the right endpoint is equal to the number of intervals of $\mathcal{I}$, for which $i$ serves as the right endpoint.

Note that for each integer $1\leq t\leq r$, the number of intervals of $\mathcal{I}$ containing $t$ is precisely $N_t$, while the number of intervals of $\mathcal{I}'$ containing $t$ is precisely $N'_t$. We conclude that $N_t=N'_t$.

In order to prove the second assertion, observe that, for every index $1 \leq t < r$, for every edge $e \in \hat{E}$ with $t \in \mathrm{span}'(e)$, the mid-segment $P^2(e)$ of the nice guiding path $P(e) \in \hat{\mathcal{P}}$ must contain some edge of $E_t$. Since the paths in $\hat{\mathcal{P}}$ cause congestion at most $O(\log^{18} m)$, we get that $N_t \leq O(\log^{18} m) \cdot |E_t|$.

## G.23 Proof of Observation 7.57

Consider an edge $e \in E(G)$. Recall that, if $e$ is a primary edge for an index $1 \leq z \leq r$ (that is, $e \in E(\tilde{S}_z) \cup \delta_G(\tilde{S}_z)$), then $N'_z(e) = 1$. Otherwise, $N'_z(e)$ is the number of paths in set:

$$\left\{Q(e') \mid e' \in E_{z-1} \cup E_z^{\text{left}}\right\} \cup \left\{Q'(e') \mid e' \in E_z \cup E_z^{\text{right}}\right\}$$

that contain $e$. Here, for an edge $e' \in E_{z-1} \cup E_z^{\text{left}}$, $Q(e')$ is the unique path of the internal router $\mathcal{Q}(U_{z-1})$ that originates at $e'$, and for an edge $e' \in E_z \cup E_z^{\text{right}}$, $Q'(e')$ is the unique path of the external router $\mathcal{Q}'(U_z)$ that originates at $e'$. If a secondary edge $e \in E(S_{z-1}) \cup E(S_{z+1})$, then, from Observation 7.55, $\mathbf{E}\left[N'_z(e)\right] \leq \mathbf{E}\left[N_z(e)\right] \leq \hat{\eta}$. Otherwise, $N'_z(e)$ is the total number of auxiliary cycles in set $\left\{W(e') \mid e' \in E_z^{\text{left}} \cup E_z^{\text{right}}\right\}$ that contain the edge $e$. Consider now some edge $e' \in \hat{E}$, and assume that $\mathrm{span}(e') = \{i, \ldots, j-1\}$. Then $e'$ may lie in set $E_z^{\text{left}} \cup E_z^{\text{right}}$ only for $z \in \{i, j\}$. It may also be a primary edge only for indices $z \in \{i, j\}$. Since, from Observation 7.47, edge $e$ may appear on at most $O(\log^{34} m)$ cycles in $\mathcal{W}$, and since there are at most $O(1)$ indices $z$, for which $e \in E_{z-1} \cup E_z \cup E_z^{\text{left}} \cup E_z^{\text{right}}$, or $e$ is a primary edge, we get that, overall,

$$\mathbf{E}\left[\sum_{z=1}^{r} N'_z(e)\right] \leq O(\hat{\eta}) + O(\log^{34} m) \leq O(\hat{\eta}).$$

## G.24 Proof of Observation 7.58

Recall that, for $1 \leq z \leq r$, $\Pi_z^T$ is the set of all triples $(e, e', v)$, where $e \in E_z^{\text{right}}$, $e' \in \hat{E}_z$, and $v$ is a vertex that lies on both $W(e)$ and $W(e')$, such that cycles $W(e)$ and $W(e')$ have a transversal intersection at $v$. Note that $E_z^{\text{right}} \subseteq \hat{E}_z$. Recall that, from Observation 7.48, for every pair $e, e' \in \hat{E}_z$ of edges, there is at most one vertex $v$, such that $W(e)$ and $W(e')$ have a transversal intersection at vertex $v$. We say that a triple $(e, e', v) \in \Pi_z^T$ is a *type-1 triple* if the cycles $W(e), W(e')$ share an edge. Let $(e, e', v)$ be a type-1 triple of $\Pi_z^T$, and let $e^*$ be an arbitrary edge shared by $W(e)$ and $W(e')$. We say that edge $e^*$ is *responsible* for the triple $(e, e', v)$. If the cycles $W(e), W(e')$ do not share edges, then we say that triple $(e, e', v)$ is a type-2 triple.

We now bound the total number of type-1 triples in $\bigcup_{z=1}^{r} \Pi_z^T$. Consider an edge $e^* \in E(G)$. From Observation 7.47, edge $e^*$ may appear on at most $O(\log^{34} m)$ cycles in $\mathcal{W}$. Consider now any such pair $W(e), W(e')$ of cycles. Assume that $\mathrm{span}(e) = \{i, \ldots, j-1\}$, and $\mathrm{span}(e') = \{i', \ldots, j'-1\}$. Recall that a triple $(e, e', v)$ may only lie in a set $\Pi_z^T$ if $e \in E_z^{\text{right}}$, so $z = i$ must hold. Therefore, every pair $e, e' \in \hat{E}$ of edges, for which $e^* \in E(W(e)) \cap E(W(e'))$, contributes at most $O(1)$ triples to set $\bigcup_{z=1}^{r} \Pi_z^T$. Overall, edge $e^*$ may be responsible for at most $O(\log^{68} m)$ triples in $\bigcup_{z=1}^{r} \Pi_z^T$, and the total number of type-1 triples in $\bigcup_{z=1}^{r} \Pi_z^T$ is at most $|E(G)| \cdot O(\log^{68} m)$.

Next, we consider a type-2 triple $(e, e', v) \in \Pi_z^T$. Observe that $v$ is the only vertex at which $W(e)$ and $W(e')$ have a transversal intersection, and cycles $W(e), W(e')$ do not share any edges. It is then easy to see that, in the drawing $\varphi^*$ of graph $G$, there must be a crossing between an edge of $W(e)$ and an edge of $W(e')$. We say that this crossing is responsible for the triple $(e, e', v)$.

Consider now some crossing $(e_1, e_2) \in \chi^*$. As before, from Observation 7.47 edge $e_1$ lies on at most $O(\log^{34} m)$ cycles of $\mathcal{W}$, and the same bound holds for edge $e_2$. Therefore, there are at most $O(\log^{68} m)$

pairs $(e, e') \in \hat{E}$ of edges, with $e_1 \in W(e)$ and $e_2 \in W(e')$. As before, there is at most one index $z$ for which $e_1 \in E_z^{\text{right}}$, and at most one index $z$, for which $e_2 \in E_z^{\text{right}}$. Therefore, crossing $(e_1, e_2)$ may be responsible for at most $O(\log^{68} m)$ triples in $\bigcup_{z=1}^{r} \Pi_z^T$, and the total number of type-2 triples in $\bigcup_{z=1}^{r} \Pi_z^T$ is at most $|\chi^*| \cdot O(\log^{68} m)$.

## G.25 Proof of Observation 7.68

From now on, we assume for contradiction that we assume that $E(H(e_1)) \cap E(H(e_2)) = \emptyset$, and, additionally, there is no pair of edges $\tilde{e}_1 \in H(e_1)$, $\tilde{e}_2 \in H(e_2) \cup W''(e_2)$, whose images cross in $\varphi^*$, and similarly, there is no pair of edges $\tilde{e}'_1 \in H(e_1) \cup W''(e_1)$, $\tilde{e}'_2 \in H(e_2)$, whose images cross in $\varphi^*$. From Observation 7.61, edges $\hat{a}'_{e_1}, \hat{a}'_{e_2}, a'_{e_1}, a'_{e_2}$ appear in this order in the rotation $\mathcal{O}_{u_{z-1}} \in \Sigma$.

We now define two points in the drawing $\varphi^*$ of $G$: point $p$, that is an internal point on the image of edge $\hat{a}_{e_2}$, that is very close to the image of the vertex $\hat{x}_{e_2}$, such that the segment of the image of edge $a_{e_2}$ between $p$ and the image of $\hat{x}_{e_2}$ does not participate in any crossings. The second point, $p'$, is defined similarly on the image of edge $a_{e_2}$, very close to the image of vertex $x_{e_2}$ (see Figure 67).

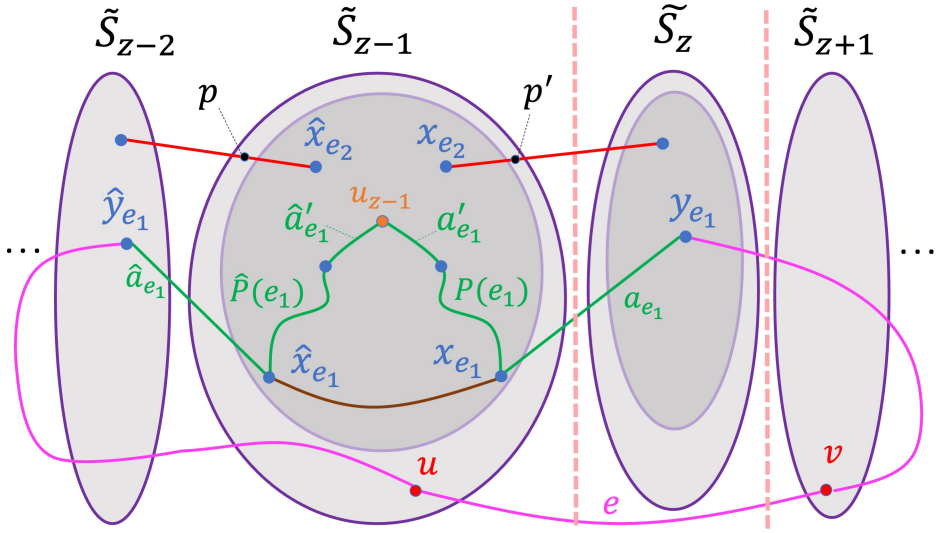



Figure 67: Illustration for the proof of Observation 7.68. Edges $\hat{a}_{e_1}$ and $a_{e_2}$ are shown in red, with points $p$ and $p'$ marked. Path $W'(e_1)$ is the concatenation of the brown path and edges $a_{e_1}, \hat{a}_{e_1}$. Path $W''(e_1)$ is shown in pink.

Next, we consider three curves in the drawing $\varphi^*$ of $G$. The first curve, $\gamma_1$, is the union of the images of paths $\hat{P}(e_1)$ and $P(e_1)$ in $\varphi^*$. The second curve, $\gamma_2$, is the image of the path $W'(e_1)$, and the third curve, $\gamma_3$, is the image of the path $W''(e_1)$ in $\varphi^*$ (see Figure 68). Observe that the endpoints of each of the three curves are the images of vertices $y_{e_1}$ and $\hat{y}_{e_1}$, and that points $p$ and $p'$ may not lie on any of these curves, as we have assumed that $E(H(e_1)) \cap E(H(e_2)) = \emptyset$. We will next show that the closed curve obtained by the union of curves $\gamma_1$ and $\gamma_2$ may not separate points $p$ and $p'$; the closed curve obtained by the union of the curves $\gamma_1$ and $\gamma_3$ must separate points $p$ and $p'$; and the closed curve obtained by the union of the curves $\gamma_2$ and $\gamma_3$ may not separate points $p$ and $p'$. We will then show that this is impossible, reaching a contradiction.

**Observation G.7** *Let $\tilde{\gamma}_1$ be the closed curve obtained by the union of the curves $\gamma_2$ and $\gamma_3$. The points $p$ and $p'$ are not separated by $\tilde{\gamma}_1$. In other words, if we consider the open regions into which curve $\tilde{\gamma}_1$ partitions the sphere, then points $p, p'$ lie in the same region.*

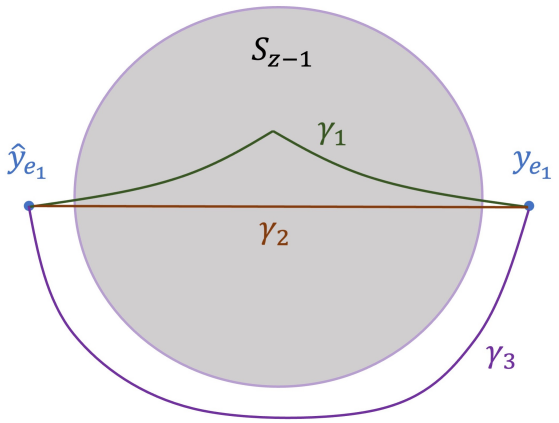



Figure 68: Curves $\gamma_1, \gamma_2$, and $\gamma_3$.

**Proof:** Assume otherwise. Consider another curve $\gamma^*$, which is obtained from the image of the path $W'(e_2)$, by truncating it so it connects point $p$ to point $p'$. Notice that path $W'(e_2)$ may not share any vertices with $W''(e_1)$ (except possibly for the endpoints of path $W'(e_2)$, whose images do not appear on curve $\gamma^*$). Moreover, cycles $W(e_1)$ and $W(e_2)$ may not have transversal intersections at a vertex of $V(S_{z-1})$ (from Observation 7.48). Since we have assumed that no edge of $W'(e_2) \subseteq H(e_2)$ may cross an edge of $H(e_1) \cup W''(e_1)$, we get that curve $\gamma^*$ may not cross curve $\tilde{\gamma}_1$, and so points $p$ and $p'$ may not be separated by curve $\tilde{\gamma}_1$. □

From Observation G.7, we can define a simple curve $\zeta$, whose endpoints are $p$ and $p'$, such that neither of the curves $\gamma_2, \gamma_3$ crosses $\zeta$.

**Observation G.8** *Let $\tilde{\gamma}_2$ be the closed curve obtained by the union of the curves $\gamma_1$ and $\gamma_2$. Then points $p$ and $p'$ are not separated by $\tilde{\gamma}_2$. In other words, if we consider the open regions into which curve $\tilde{\gamma}_2$ partitions the sphere, then points $p, p'$ lie in the same region.*

**Proof:** Assume otherwise. Consider another curve $\gamma^*$, which is obtained from the image of the path $W''(e_2)$, by appending to it the image of edge $\hat{a}_{e_2}$ between the image of vertex $\hat{y}_{e_2}$ and point $p$, and the image of edge $a_{e_2}$ between the image of vertex $y_{e_2}$ and point $p'$.

Notice that path $W''(e_2)$ may not share any vertices with paths $W'(e_1)$, $P(e_1)$, and $P(e_2)$ (except for possibly the endpoints of $W''(e_2)$). Observe also that paths $W'(e_1), P(e_1)$ both contain the edge $a_{e_1} = (x_{e_1}, y_{e_1})$, so, even if vertex $y_{e_1} \in W''(e_2)$, curve $\gamma^*$ may not cross curve $\tilde{\gamma}_2$ at the image of vertex $y_{e_1}$. Similarly, paths $W'(e_1), \hat{P}(e_1)$ both contain the edge $\hat{a}_{e_1} = (\hat{x}_{e_1}, \hat{y}_{e_1})$. Therefore, even if vertex $\hat{y}_{e_1} \in W''(e_2)$, curve $\gamma^*$ may not cross curve $\tilde{\gamma}_2$ at the image of vertex $\hat{y}_{e_1}$. Since we have assumed that no edge of $W''(e_2)$ may cross an edge of $H(e_1)$, we conclude that curve $\gamma^*$ may not cross curve $\gamma_2$. Therefore, points $p$ and $p'$ must lie in the same region defined by $\tilde{\gamma}_2$. □

From Observation G.8, we can now define a simple curve $\zeta'$, whose endpoints are $p$ and $p'$, such that neither of the curves $\gamma_1, \gamma_2$ crosses $\zeta'$. Let $\zeta^*$ be the curve obtained from the union of the curves $\zeta, \zeta'$. Recall that curve $\gamma_2$, that connects the images of the vertices $y_{e_1}$ and $\hat{y}_{e_1}$, may not cross the curve $\zeta^*$. Therefore, if we denote by $q$ and $q'$ the images of the vertices $y_{e_1}$ and $\hat{y}_{e_1}$, respectively, then points $q$ and $q'$ do not lie on curve $\zeta^*$, and they are not separated by curve $\zeta^*$. Lastly, we need the following observation.

**Observation G.9** *Let $\tilde{\gamma}_3$ be the closed curve obtained by the union of the curves $\gamma_1$ and $\gamma_3$. Then points $p$ and $p'$ are separated by $\tilde{\gamma}_3$. In other words, if we consider the set $\mathcal{F}$ open regions into which curve $\tilde{\gamma}_3$ partitions the sphere, then points $p, p'$ lie in different regions.*

**Proof:** Recall that we have assumed that $E(H(e_1)) \cap E(H(e_2)) = \emptyset$, and that no edge of $H(e_1) \cup W''(e_1)$ may cross an edge of $P(e_2) \cup \hat{P}(e_2) \subseteq H(e_2)$. Additionally, path $W''(e_1)$ is internally disjoint from paths $P(e_2)$ and $\hat{P}(e_2)$, and, since paths in $\mathcal{Q}(S_{z-1})$ are non-transversal with respect to $\Sigma$, paths $P(e_1), \hat{P}(e_1), P(e_2), \hat{P}(e_2)$ do not have transversal intersections. Therefore, the image of path $P(e_2)$ beween point $p'$ and the image of vertex $u_{z-1}$ may not cross the curve $\tilde{\gamma}_3$, and it must be contained in a single region of $\mathcal{F}$, that we denote by $F$. Similarly, the image of path $\hat{P}(e_2)$ beween point $p$ and the image of vertex $u_{z-1}$ may not cross the curve $\tilde{\gamma}_3$, and it must be contained in a single region of $\mathcal{F}$, that we denote by $F'$. Lastly, recall that we have established that the images of edges $\hat{a}'_{e_1}, \hat{a}'_{e_2}, a'_{e_1}, a'_{e_2}$ enter the image of edge $u_{z-1}$ in this circular order. Recall that edges $\hat{a}'_{e_1}, a'_{e_1}$ lie on paths $\hat{P}(e_1), P(e_1)$, respectively, while edges $\hat{a}'_{e_2}, a'_{e_2}$ lie on paths $\hat{P}(e_2), P(e_2)$, respectively. Since curve $\tilde{\gamma}_3$ is a closed curve, it must be the case that $F \neq F'$. □

To summarize, so far we have defined two points $p, p'$, and two curves $\zeta, \zeta'$ connecting them. We have also denoted by $\zeta^*$ the closed curve obtained by taking the union of these two curves. We also defined two points $q$ and $q'$, and two curves $\gamma_1, \gamma_3$ connecting them, and we denoted by $\tilde{\gamma}_3$ the closed curve obtained by taking the union of these two curves. We have established that curve $\zeta^*$ may not separate $q$ and $q'$, while curve $\tilde{\gamma}_3$ must separate $p$ and $p'$. Lastly, from the definition, curve $\gamma_1$ may not cross $\zeta'$, while curve $\gamma_3$ may not cross $\zeta$. We now show that this is impossible to achieve. We denote by $D$ a disc in the plane, whose boundary lies on curve $\zeta^*$, and whose interior contains the points $q$ and $q'$, and does not contain any point of $\zeta^*$. Such a disc must exist, since points $q$ and $q'$ are not separated by $\zeta^*$. Note that points $p$ and $p'$ do not lie in the interior of the disc $D$, and yet they are separated by curve $\tilde{\gamma}_3$. this may only happen if curve $\tilde{\gamma}_3$ intersects both $\zeta$ and $\zeta'$.

Consider curve $\tilde{\gamma}$ that is obtained from $\tilde{\gamma}_3$ by deleting the point $q$ from it. On this curve, we can mark two points $a$ and $b$, such that $a$ lies on $\zeta$, $b$ lies on $\zeta'$, and no point of $\tilde{\gamma}$ that lies between $a$ and $b$ belongs to $\zeta^*$. Denote by $\tilde{\gamma}'$ the segment of $\tilde{\gamma}$ between $a$ and $b$. Note that this segment is disjoint from the interior of $D$, so it may not contain the point $q'$. Therefore, either $\tilde{\gamma} \subseteq \gamma_1$, or $\tilde{\gamma} \subseteq \gamma_3$. In the former case, we get that $\gamma_1$ crosses $\zeta'$, while in the latter case, we get that $\gamma_3$ crosses $\zeta$, a contradiction.

# H Proofs Omitted from Section 9

## H.1 Proof of Claim 9.11

Consider the $\mathcal{J}$-contracted instance $\hat{I} = (\hat{G}, \hat{\Sigma})$. Since it is a wide instance, there is a high-degree vertex $v^* \in V(\hat{G})$, a partition $(E_1, E_2)$ of the edges of $\delta_{\hat{G}}(v^*)$, such that the edges of $E_1$ appear consequently in the rotation $\mathcal{O}_{v^*} \in \hat{\Sigma}$, and a collection $\mathcal{R}$ of at least $\left\lfloor |E(\hat{G})|/\mu^{50} \right\rfloor = \left\lfloor \hat{m}(I)/\mu^{50} \right\rfloor$ simple edge-disjoint cycles in $\hat{G}$, such that every cycle $P \in \mathcal{R}$ contains one edge of $E_1$ and one edge of $E_2$. Note that we can compute the vertex $v^*$, the partition $(E_1, E_2)$ of the edges of $\delta_{\hat{G}}(v^*)$, and the set $\mathcal{R}$ of cycles with the above properties efficiently.

Assume first that $v^* = v_J$. In this case, $\delta_{\hat{G}}(v^*) = \delta_G(J)$, so $(E_1, E_2)$ is also a partition of the edges of $\delta_G(J)$. From the definition of a $\mathcal{J}$-contracted instance, the rotation $\mathcal{O}_{v^*} \in \hat{\Sigma}$ is identical to the ordering $\mathcal{O}(J)$ of the edges of $\delta_{\hat{G}}(v^*) = \delta_G(J)$. Therefore, edges of $E_1$ appear consecutively in the ordering $\mathcal{O}(J)$. The set $\mathcal{R}$ of cycles in graph $\hat{G}$ then naturally defines a set $\mathcal{P}$ of simple paths in graph $G$, where every path $P' \in \mathcal{P}$ has an edge of $E_1$ as its first edge, an edge of $E_2$ as its last edge, and it is internally disjoint from $J$. We discard arbitrary paths from $\mathcal{P}$ until $|\mathcal{P}| = \left\lfloor \frac{\hat{m}(I)}{\mu^{50}} \right\rfloor$ holds, obtaining the desired set of promising paths.

From now on we assume that $v^* \neq v_J$, that is, $v^*$ is a vertex of $G$. From the definition of a high-degree vertex, $\deg_G(v) \geq \frac{\hat{m}(I)}{\mu^4}$. Therefore, there is a collection $\mathcal{Q}$ of at least $\left\lceil \frac{2\hat{m}(I)}{\mu^{50}} \right\rceil$ edge-disjoint paths in $G$

connecting $v^*$ to vertices of $J$, and we can compute such a collection of paths efficiently using standard Maximum $s$-$t$ Flow. We can assume w.l.o.g. that every path in $\mathcal{Q}$ is internally disjoint from $J$, and we view the paths in $\mathcal{Q}$ as being directed away from $v^*$. We can then compute a partition $(E_1', E_2')$ of the edges of $\delta_G(J)$, such that the edges of $E_1'$ appear consecutively in the ordering $\mathcal{O}(J)$, and there are two subsets $\mathcal{Q}_1, \mathcal{Q}_2 \subseteq \mathcal{Q}$ of paths of cardinality $\left\lfloor \frac{\hat{m}(I)}{\mu^{50}} \right\rfloor$ each, such that the last edge of every path in $\mathcal{Q}_1$ lies in $E_1'$, and the last edge of every path in $\mathcal{Q}_2$ lies in $E_2'$. By arbitrarily matching the paths in $\mathcal{Q}_1$ to the paths in $\mathcal{Q}_2$ and concatenating the pairs of matched paths, we obtain a collection $\mathcal{P}$ of edge-disjoint paths, each of which has an edge of $E_1'$ as its first edge, an edge of $E_2'$ as its last edge, and is internally disjoint from $J$. We discard paths from $\mathcal{Q}$ until $|\mathcal{Q}| = \left\lfloor \frac{\hat{m}(I)}{\mu^{50}} \right\rfloor$ holds, obtaining the desired promising set of paths.

## H.2 Proof of Observation 9.13

We denote the input instances by $I = (G, \Sigma)$, $I_1 = (G_1, \Sigma_1)$, and $I_2 = (G_2, \Sigma_2)$. We denote the core structure by $\mathcal{J} = (J, \{b_u\}_{u \in V(J)}, \rho_J, F^*(\rho_J))$, and the $\mathcal{J}$-enhancement structure by $\mathcal{A} = \left(P, \{b_u\}_{u \in V(J')}, \rho'\right)$, where $J' = J \cup P$. We let $(\mathcal{J}_1, \mathcal{J}_2)$ be the split of the core structure $\mathcal{J}$ via the enhancement structure $\mathcal{A}$, and we denote $\mathcal{J}_1 = (J_1, \{b_u\}_{u \in V(J_1)}, \rho_{J_1}, F_1)$ and $\mathcal{J}_2 = (J_2, \{b_u\}_{u \in V(J_2)}, \rho_{J_2}, F_2)$.

Let $E^{\mathsf{del}} = E(G) \setminus (E(G_1) \cup E(G_2))$, let $G' = G \setminus E^{\mathsf{del}}$, and let $\Sigma'$ be the rotation system for $G'$ that is induced by $\Sigma$. We first compute a $\mathcal{J}$-clean solution $\varphi'$ to instance $I' = (G', \Sigma')$ of $\mathsf{MCNwRS}$ with $\mathsf{cr}(\varphi') \leq \mathsf{cr}(\varphi_1) + \mathsf{cr}(\varphi_2)$, and then insert the edges of $E^{\mathsf{del}}$ into this drawing, to obtain the final solution $\varphi$ to instance $I$.

We now describe the construction of the solution $\varphi'$ to instance $I'$. Consider the drawing $\rho'$ of graph $J'$, and the faces $F_1, F_2$ of this drawing that were used to define the split $(\mathcal{J}_1, \mathcal{J}_2)$ of the core structure $\mathcal{J}$. Recall that the drawing of graph $J_1$ induced by $\rho'$ is precisely $\rho_{J_1}$, and the drawing of graph $J_2$ induced by $\rho'$ is precisely $\rho_{J_2}$.

Consider now the $\mathcal{J}_1$-clean drawing $\varphi_1$ of graph $G_1$ on the sphere. The drawing of $J_1$ induced by $\varphi_1$ is identical to $\rho_{J_1}$, and the images of all edges and vertices of $G_1$ are contained in region $F_1$ of this drawing. We plant the drawing $\varphi_1$ of $G_1$ into the face $F_1$ of drawing $\rho'$, so that the images of the edges and the vertices of $J_1$ in both drawings coincide, and the images of all vertices and edges of $G_1$ appear in face $F_1$. We similarly plant drawing $\varphi_2$ of $G_2$ inside face $F_2$ of $\rho'$, obtaining a solution $\varphi'$ to instance $I'$, whose cost is bounded by $\mathsf{cr}(\varphi_1) + \mathsf{cr}(\varphi_2)$. Since the images of all edges and vertices of $G'$ are contained in region $F^*_{\rho_J} = F_1 \cup F_2$, this drawing is $\mathcal{J}$-clean.

In order to complete the construction of the solution $\varphi$ to instance $I$, it remains to "insert" the images of the edges of $E^{\mathsf{del}}$ into $\varphi'$. We do so using Lemma 2.9. There is, however, one subtlety in using this lemma directly in order to insert the edges of $E^{\mathsf{del}}$ into the drawing $\varphi'$: we need to ensure that the drawing remains $\mathcal{J}$-clean, so the images of the newly inserted edges may not cross the images of the edges of $J$. This is easy to achieve, for example, by first contracting core $J$ into a supernode and modifying the drawing $\varphi$ to obtain a drawing of the resulting graph in a natural way. We then insert the edges of $E^{\mathsf{del}}$ into this drawing of the contracted instance using the algorithm from Lemma 2.9, and then un-contract the supernode $v_J$. We obtain a $\mathcal{J}$-clean solution $\varphi$ to instance $I$, whose number of crossings is bounded by $\mathsf{cr}(\varphi_1) + \mathsf{cr}(\varphi_2) + |E^{\mathsf{del}}| \cdot |E(G)|$.

## H.3 Proof of Claim 9.14

The proof of Claim 9.14 is similar to the proof of Claim 9.9 in [CMT20]. For all $1 \leq i \leq 4k+2$, we denote by $\gamma_i$ the image of path $P_i$ in $\varphi$, that is, $\gamma_i$ is the concatenation of the images of the edges of $P_i$. Clearly, all curves in the resulting set $\Gamma = \{\gamma_1, \ldots, \gamma_{4k+2}\}$ connect $\varphi(u)$ to $\varphi(v)$, and they enter

the image of $u$ in $\varphi$ in the order of their indices. Notice that the curves $\gamma_i \in \Gamma$ are not necessarily simple: if a pair of edges lying on path $P_i$ cross at some point $p$, then curve $\gamma_i$ crosses itself at point $p$. In such a case, point $p$ may not lie on any other curve in $\Gamma$.

Next, we will slightly modify the curves in $\Gamma$ by "nudging" them in the vicinity of their common vertices. In order to do so, we consider every vertex $x \in V(G) \setminus \{u, v\}$ that belongs to at least two paths of $\mathcal{P}$ one by one.

We now describe an iteration when a vertex $x \in V(G) \setminus \{u, v\}$ is processed. Let $\mathcal{P}^x \subseteq \mathcal{P}$ be the set of all paths containing vertex $x$. Note that $x$ must be an inner vertex on each such path. For convenience, we denote $\mathcal{Q}^x = \{P_{i_1}, \ldots, P_{i_z}\}$. Consider the tiny $x$-disc $D(x) = D_\varphi(x)$. For all $1 \leq j \leq z$, denote by $s_j$ and $t_j$ the two points on the curve $\gamma_{i_j}$ that lie on the boundary of disc $D(x)$. We use the algorithm from Claim 4.34 to compute a collection $\{\sigma_1, \ldots, \sigma_z\}$ of curves, such that, for all $1 \leq j \leq z$, curve $\sigma_j$ connects $s_j$ to $t_j$, and the interior of the curve is contained in the interior of $D(x)$. Recall that every pair of resulting curves crosses at most once, and every point in the interior of $D(x)$ may be contained in at most two curves. Moreover, a pair $\sigma_r, \sigma_q$ of such curves may only cross if the two pairs $(s_r, t_r), (s_q, t_q)$ of points on the boundary of $D(x)$ cross. This, in turn, may only happen if paths $P_{i_r}, P_{i_q}$ have a transversal intersection at vertex $x$, which is impossible. Therefore, the curves $\sigma_1, \ldots, \sigma_z$ do not cross each other. For all $1 \leq j \leq z$, we modify the curve $\gamma_{i_j}$, by replacing the segment of the curve that is contained in disc $D(x)$ with $\sigma_j$.

Once every vertex $x \in V(G) \setminus \{u, v\}$ is processed, we obtain the final set $\Gamma' = \left\{\gamma'_1, \ldots, \gamma'_{4k+2}\right\}$ of curves. Notice that for any pair $1 \leq j < j' \leq 4k+2$ of indices, curves $\gamma'_j, \gamma'_{j'}$ cross if and only if there is a crossing $(e, e')_p$ in $\varphi$ with $e \in E(P_j)$ and $e' \in E(P_{j'})$.

It is now enough to prove that curves $\gamma'_1$ and $\gamma'_{2k+1}$ do not cross. Assume for contradiction that the two curves cross. Among all crossing points between these two curves, let $p'$ be the point that is closest to $\varphi(u)$ on $\gamma'_1$. Let $\lambda$ be the segment of $\gamma'_1$ from $\varphi(u)$ to $p'$, and let $\lambda'$ be defined similarly for $\gamma'_{2k+1}$. We modify curves $\lambda$ and $\lambda'$ to remove all their self-loops, so the curves become simple. Note that curves $\lambda$ and $\lambda'$ both originate at $\varphi(u)$ and terminate at $p'$, and they do not cross. Let $\lambda^*$ be the simple closed curve obtained from the union of $\lambda$ and $\lambda'$. Curve $\lambda^*$ partitions the sphere into two internally disjoint discs, that we denote by $D$ and $D'$. Since the curves $\gamma'_1, \ldots, \gamma'_{4k+2}$ enter the image of $u$ in $\varphi$ in the order of their indices, either (i) for all $1 < i < 2k+1$, the intersection of $\gamma'_i$ with the tiny $u$-disc $D_\varphi(u)$ lies in $D$, and for all $2k+1 < i \leq 4k+2$, the intersection of $\gamma'_i$ with $D_\varphi(u)$ lies in $D'$, or (ii) the opposite is true. We assumme without loss of generality that it is the former. Note that point $\varphi(v)$ must lie in the interior of one of these discs – we assume without loss of generality that it is $D'$.

Consider now some index $1 < i < 2k+1$. Recall that the segment of $\gamma'_i$ that is contained in $D_\varphi(u)$ is contained in $D$, while point $\varphi(v)$, that is an endpoint of $\gamma'_i$, lies in the interior of $D'$. Therefore, there must be some point $r$ that lies on $\gamma_i$ and on $\lambda^*$, such that the two curves have a transversal intersection at $r$. This point may not be $p'$, since curves $\gamma'_1$ and $\gamma'_{2k+1}$ have a crossing at $p'$, and this crossing corresponds to a crossing between an edge of $P_1$ and an edge of $P_{2k+1}$ in $\varphi$. Therefore, $\gamma'_i$ has a transversal crossing with either $\lambda$ or $\lambda'$. In the former case, there is a crossing between an edge of $P_1$ and an edge of $P_i$, while in the latter case there is a crossing between an edge of $P_{2k+1}$ and an edge of $P_i$. We conclude that for all $1 < i < 2k+1$, some edge of $P_i$ must cross an image of an edge of $P_1$ or of $P_{2k+1}$ in $\varphi$. Since the edges of $P_1$ participate in at most $k$ crossings, and so do the edges of $P_{2k+1}$, and since we have assumed that an edge of $P_1$ crosses an edge of $P_{2k+1}$, this is impossible.

## H.4 Proof of Claim 9.21

Assume for contradiction that there is a vertex $x \in V(G) \setminus V(J)$, and a set $\mathcal{Q} \subseteq \mathcal{P}^*$ of $\left\lceil \frac{512\mu^{13b}\mathsf{cr}(\varphi)}{m} \right\rceil$ good paths that are unlucky with respect to $x$. We denote $\mathcal{Q} = \{Q_1, \ldots, Q_\lambda\}$, where $\lambda = \left\lceil \frac{512\mu^{13b}\mathsf{cr}(\varphi)}{m} \right\rceil$.

Let $\lambda' = 4 \cdot \left\lceil \frac{4\mu^{13b}\mathsf{cr}(\varphi)}{m} \right\rceil$, so $\lambda/\lambda' \geq 16$. For all $1 \leq i \leq \lambda$, we denote by $\hat{e}_i$ the first edge on path $Q_i$ that is incident to vertex $x$, and by $\hat{e}'_i$ the edge following $\hat{e}_i$ on path $Q_i$. We assume that the paths in $\mathcal{Q}$ are indexed so that the edges $\hat{e}_1, \ldots, \hat{e}_\lambda$ appear in the rotation $\mathcal{O}_x \in \Sigma$ in the order of their indices.

Recall that we are given a partition $(E_1, E_2)$ of the edges of $\delta_G(J)$, such that the edges of $E_1$ appear consecutively in the ordering $\mathcal{O}(J)$, and every path in $\mathcal{P}$ has an edge of $E_1$ as its first edge, and an edge of $E_2$ as its last edge. From our construction, every path in $\mathcal{P}^*$, and hence in $\mathcal{Q}$, has an edge of $E_1$ as its first edge, and an edge of $E_2$ as its last edge. Consider the solution $\varphi$ to instance $I$, that is $\mathcal{J}$-valid. Let $\varphi'$ be the drawing that is obtained from $\varphi$ after we delete all edges and vertices from it, except for those lying in $J$, and on the paths of $\mathcal{Q}$. Since all paths in $\mathcal{Q}$ are good, there are no crossings in $\varphi'$ in which the edges of $J$ are involved. Let $D(J)$ be the disc associated with core $J$ in $\varphi$. This disc contains the image of $J$ in $\varphi'$ in its interior, and its boundary follows closely the drawing of $J$. Notice that the image of every path $Q \in \mathcal{Q}$ in $\varphi$ must intersect the interior of the region $F^*$, from the definition of a valid core structure (see Definition 9.3), and since each such path contains edges incident to vertices of $J$. Therefore, the image of every path $Q \in \mathcal{Q}$ in $\varphi'$ is contained in the region $F^*$. We can then ensure that no crossing points of $\varphi'$ are contained in disc $D(J)$; the only vertices and edges whose images in $\varphi'$ are contained in $D(J)$ are the vertices and edges of $J$; and the only other edges whose images intersect $D(J)$ in $\varphi'$ are the edges of $\delta_G(J)$ that lie on the paths of $\mathcal{Q}$. Moreover, for each such edge $e$, the intersection of $\varphi'(e)$ and $D(J)$ is a simple curve.

We partition the boundary of disc $D(J)$ into two segment $\sigma$ and $\sigma'$, such that $\sigma$ contains all intersection points of the boundary of $D(J)$ with the images of the edges in $E_1$ in $\varphi$, while $\sigma'$ contains all intersection points of the boundary of $D(J)$ with the images of the edges in $E_2$ in $\varphi$ (see Figure 69).

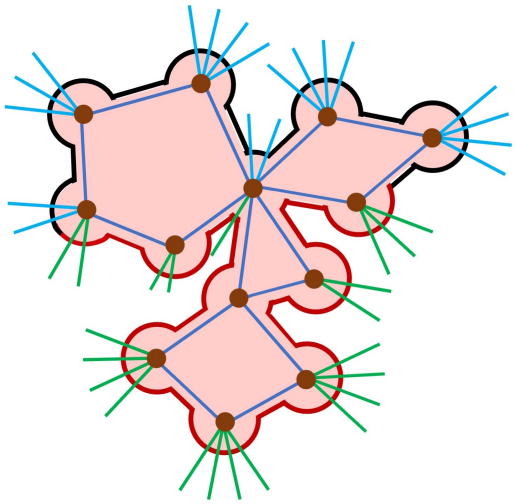


Figure 69: Partitioning the boundary of disc $D(J)$ into segments $\sigma$ (black) and $\sigma'$ (red). Edges of $E_1$ are shown in light blue and edges of $E_2$ are shown in green.

Let $\varphi''$ be the drawing that is obtained from $\varphi'$ by deleting the images of the vertices and the edges of $J$ from it, and deleting the segment of the image of every edge $e \in \delta_G(J)$ that is contained in the interior of $D(J)$. We then contract segment $\sigma$ into a point $p$, and segment $\sigma'$ into a point $q$, so that the image of every path $Q \in \mathcal{Q}$ in the resulting drawing connects $p$ to $q$, and for every edge $e \in \bigcup_{Q \in \mathcal{Q}} E(Q)$, the number of crossings in which $e$ participates in $\varphi''$ is bounded by the number of crossings in which $e$ participates in $\varphi'$.

For all $1 \leq i \leq \lambda$, we let $R_i$ be the subpath of $Q_i$ from its first vertex to $x$, so $R_i$ contains an edge of $E_1$, and let $\mathcal{R} = \{R_i \mid 1 \leq i \leq \lambda\}$. We also denote $\mathcal{Q}' = \{Q_{i \cdot \lambda'} \mid 1 \leq i \leq 16\}$, and, for all $1 \leq i \leq 16$, we

let $\tilde{R}_i = R_{i\cdot\lambda'}$ – the subpath of path $Q_{i\lambda'}$, from its first endpoint to vertex $x$. Let $\tilde{\mathcal{R}} = \left\{\tilde{R}_1, \ldots, \tilde{R}_{16}\right\}$.

Recall that all paths in $\mathcal{Q}$ are good, and so each such path participates in at most $\frac{\mathsf{cr}(\varphi)\cdot\mu^{12b}}{m}$ crossings. Since for every pair $P, P' \in \mathcal{P}^*$ of paths, for every vertex $v \in V(P) \cap V(P')$ with $v \notin V(J)$, the intersection of $P$ and $P'$ at $v$ is non-transversal with respect to $\Sigma$, the paths in set $\mathcal{R}$ are non-transversal with respect to $\Sigma$, so are the paths in $\tilde{\mathcal{R}}$. Therefore, from Claim 9.14, for any pair $2 \cdot \left\lceil \frac{\mathsf{cr}(\varphi)\cdot\mu^{12b}}{m} \right\rceil < i < j \leq \lambda$ of indices with $j - i > 2 \cdot \left\lceil \frac{\mathsf{cr}(\varphi)\cdot\mu^{12b}}{m} \right\rceil$, there is no crossing in $\varphi''$ between an edge of $R_i$ and an edge of $R_j$. In particular, since $\lambda' = 4 \cdot \left\lceil \frac{4\mu^{13b}\mathsf{cr}(\varphi)}{m} \right\rceil$, there are no crossings in $\varphi''$ between pairs of edges lying in distinct paths of $\tilde{\mathcal{R}}$.

For $1 \leq i \leq 16$, we let $\gamma_i$ be the curve that is obtained from the image of path $\tilde{R}_i$ in $\varphi''$, after removing all self-loops. Consider the resulting collection $\Gamma = \{\gamma_1, \ldots, \gamma_{16}\}$ of curves. All curves in $\Gamma$ originate at point $p$ and terminate at the image of $x$ in $\varphi''$. The curves do not cross themselves or each other, and they enter the image of $x$ in the order of their indices. The curves in $\Gamma$ partition the sphere into 16 regions, that we denote by $\tilde{F}_1, \ldots, \tilde{F}_{16}$. Region $\tilde{F}_1$ has curves $\gamma_1$ and $\gamma_{16}$ as its boundaries, and for $1 < i \leq 16$, region $\tilde{F}_i$ has curves $\gamma_{i-1}$ and $\gamma_i$ as its boundaries. Note that point $q$ must be contained in the interior of one of these regions. We assume without loss of generality that it is $\tilde{F}_1$ (as otherwise we could re-index the paths of $\mathcal{Q}$ accordingly).

Next, we consider the path $Q^* = Q_{8\lambda'+1}$, and we prove that path $Q^*$ is not unlucky for vertex $x$, reaching a contradiction.

**Observation H.1** *Path $Q^*$ is not unlucky for vertex $x$.*

**Proof:** Let $e^*$, $e^{**}$ be the edges of $Q^*$ that are incident to $x$, with $e^*$ lying before $e^{**}$ on the path. Let $\hat{E}_1(x) \subseteq \delta_G(x)$ be the set of edges $\hat{e} \in \delta_G(x)$, such that $\hat{e}$ lies between $e^*$ and $e^{**}$ in the rotation $\mathcal{O}_x \in \Sigma$ (in clock-wise orientation), and $\hat{e}$ lies on some good path of $\mathcal{P}^*$. Let $\hat{E}_2(x) \subseteq \delta_G(x)$ be the set of edges $\hat{e} \in \delta_G(x)$, such that $\hat{e}$ lies between $e^{**}$ and $e^*$ in the rotation $\mathcal{O}_x \in \Sigma$ (in clock-wise orientation), and $\hat{e}$ lies on some good path of $\mathcal{P}^*$. It is enough to prove that $|\hat{E}_1(x)|, |\hat{E}_2(x)| \geq \frac{\mathsf{cr}(\varphi)\mu^{13b}}{m}$.

For all $2 \leq i \leq 16$, let $\mathcal{Q}_i = \{Q_j \mid (i-1)\lambda' < j < i\lambda'\}$. Recall that $\lambda' = 4 \cdot \left\lceil \frac{4\mu^{13b}\mathsf{cr}(\varphi)}{m} \right\rceil$, and every path $\tilde{R}_i \in \tilde{\mathcal{R}}$ participates in at most $\frac{\mathsf{cr}(\varphi)\cdot\mu^{12b}}{m}$ crossings in $\varphi''$, since the paths in $\mathcal{Q}$ are good. Therefore, there must be a subset $\mathcal{Q}'_i \subseteq \mathcal{Q}_i$ of at least $2 \cdot \left\lceil \frac{4\mu^{13b}\mathsf{cr}(\varphi)}{m} \right\rceil$ paths, such that for every path $Q_j \in \mathcal{Q}'_i$, the image of $Q_j$ in $\varphi$ does not cross the curves $\gamma_{i-1}$ and $\gamma_i$. In particular, the image of every path in set $\{R_j \mid Q_j \in \mathcal{Q}'_i\}$ is contained in region $\tilde{F}_i$ in $\varphi''$.

Recall that we have defined a set $\mathcal{Q}'_4 \subseteq \mathcal{Q}_4$ of at least $2 \cdot \left\lceil \frac{4\mu^{13b}\mathsf{cr}(\varphi)}{m} \right\rceil$ paths, where for each path $Q_j \in \mathcal{Q}'_4$, the image of the corresponding path $R_j$ is contained in $\tilde{F}_4$. Recall that, for every path $Q_j \in \mathcal{Q}$, we denoted by $\hat{e}_j$ the first edge on path $Q_j$ that is incident to $x$. We denote by $E^L = \{\hat{e}_j \mid Q_j \in \mathcal{Q}'_4\}$. Clearly, for every edge $\hat{e}_j \in E^L$, the image of $\hat{e}_j$ in $\varphi''$ lies in region $\tilde{F}_4$.

Similarly, we have defined a set $\mathcal{Q}'_{14} \subseteq \mathcal{Q}_{14}$ of at least $2 \cdot \left\lceil \frac{4\mu^{13b}\mathsf{cr}(\varphi)}{m} \right\rceil$ paths, where for each path $Q_j \in \mathcal{Q}'_{14}$, the image of the corresponding path $R_j$ is contained in $\tilde{F}_{14}$. We denote by $E^R = \{\hat{e}_j \mid Q_j \in \mathcal{Q}'_{14}\}$. Clearly, for every edge $\hat{e}_j \in E^R$, the image of $\hat{e}_j$ in $\varphi''$ lies in region $\tilde{F}_{14}$.

It is also easy to see that the imgage of edge $e^*$ must be contained in $\tilde{F}_7 \cup \tilde{F}_8 \cup \tilde{F}_9 \cup \tilde{F}_{10}$ (as otherwise path $Q^*$ would need to cross more than $\frac{\mathsf{cr}(\varphi)\cdot\mu^{12b}}{m}$ other paths in $\mathcal{Q}$, for example, the paths of $\mathcal{Q}'_7$, or the paths of $\mathcal{Q}'_{10}$).

Lastly, we show that the intersection of the image of edge $e^{**}$ and the tiny $x$-disc $D_{\varphi''}(x)$, that we denote by $\sigma(e^{**})$, must be contained in $\tilde{F}_2 \cup \tilde{F}_1 \cup \tilde{F}_{16}$. Note that, if this is the case, then either

(i) $E^L \subseteq \hat{E}_1(x)$ and $E^R \subseteq \hat{E}_2(x)$; or (ii) $E^R \subseteq \hat{E}_1(x)$ and $E^L \subseteq \hat{E}_2(x)$ hold. In either case, since $|E^R|, |E^L| \geq 2 \cdot \left\lceil \frac{4\mu^{13b}\mathsf{cr}(\varphi)}{m} \right\rceil$, path $Q^*$ is not unlucky for $x$.

It now remains to prove that $\sigma(e^{**})$ is contained in $\tilde{F}_2 \cup \tilde{F}_1 \cup \tilde{F}_{16}$. Assume otherwise. From the definition of tiny $x$-disc, $\sigma(e^{**})$ must be contained in some region $\tilde{F}_i$, for $3 \leq i \leq 15$. Recall that we have defined a subset $\mathcal{Q}'_2 \subseteq \mathcal{Q}_2$ of at least $2 \cdot \left\lceil \frac{4\mu^{13b}\mathsf{cr}(\varphi)}{m} \right\rceil$ paths, such that, for every path $Q_j \in \mathcal{Q}'_2$, the image of the corresponding path $R_j \in \mathcal{R}$ is contained in region $\tilde{F}_2$. We have also defined a collection $\mathcal{Q}'_{16} \subseteq \mathcal{Q}_{16}$ of at least $2 \cdot \left\lceil \frac{4\mu^{13b}\mathsf{cr}(\varphi)}{m} \right\rceil$ paths, such that, for every path $Q_j \in \mathcal{Q}'_{16}$, the image of the corresponding path $R_j \in \mathcal{R}$ is contained in region $\tilde{F}_{16}$.

Consider now the segment $\gamma^*$ of the image of path $Q^*$ in $\varphi''$ from vertex $x$ to point $q$. Since $\sigma(e^{**})$ is contained in $\bigcup_{3 \leq i \leq 15} \tilde{F}_i$, while point $q$ is contained in $\tilde{F}_1$, curve $\gamma^*$ has to either cross the image of every path in $\{R_j \mid Q_j \in \mathcal{Q}'_2\}$, or it has to cross the image of every path in $\{R_j \mid Q_j \in \mathcal{Q}'_{16}\}$. In either case, since the paths of $\mathcal{Q}$ are non-transversal with respect to $\Sigma$, the edges of path $Q^*$ must participate in at least $2 \cdot \left\lceil \frac{4\mu^{13b}\mathsf{cr}(\varphi)}{m} \right\rceil$ crossings, contradicting the fact that $Q^*$ is a good path. □

## H.5 Proof of Claim 9.23

We start with $\varphi'$ being the drawing of graph $G'$ that is induced by $\varphi$. Since bad event $\mathcal{E}_1$ did not happen, drawing $\varphi'$ does not contain crossings between edges of $E(P^*)$ and edges of $E(J)$, but it may contain crossings between pairs of edges in $E(P^*)$. We next show how to modify this drawing in order to eliminate all such crossings. Let $\gamma$ be the image of the path $P^*$ in $\varphi'$. Notice that $\gamma$ is either a closed or an open curve, that may cross itself in a number of points. Recall that path $P^*$ is internally disjont from $J$, and there are no crossings in $\varphi'$ between edges of $P^*$ and edges of $J$. Moreover, from the definition of valid core structure (see Definition 9.3), and since $\varphi$ is a $\mathcal{J}$-valid drawing of $G$, $\gamma$ must intersect the interior of the region $F^*$. Therefore, $\gamma \subseteq F^*$ must hold. In order to obtain the desired final drawing of graph $G'$, we will only modify the images of the edges and vertices that lie on $P^*$, with the new images contained in $F^*$, so that the resulting drawing of $G'$ is $\varphi$-compatible.

We can partition the curve $\gamma$ into a collection $\Gamma$ of curves, for which the following hold. First, there is a single special curve $\gamma^* \in \Gamma$, which is either a simple closed or a simple open curve. In the former case, $\gamma^*$ contains the image of exactly one vertex of $J$, and in the latter case, the endpoints of $\gamma^*$ are images of two distinct vertices of $J$. All other curves in $\Gamma$ are simple closed curves. For every pair $\gamma_1, \gamma_2 \in \Gamma$ of distinct curves, $\gamma_1$ and $\gamma_2$ may share at most one point, and that point must be a crossing point between a pair of edges of $E(P^*)$ in $\varphi'$. We ensure that every point of $\gamma$ lies on at least one curve of $\Gamma$. Since all curves in $\Gamma$ are simple, no curve in $\Gamma$ may cross itself. We need the following observation.

**Observation H.2** *If neither of the Events $\mathcal{E}_1, \mathcal{E}_3$ happenned, then for every curve $\gamma' \in \Gamma \setminus \{\gamma^*\}$, every vertex $x$ with $\varphi'(x) \in \gamma'$ is a light vertex.*

**Proof:** Assume otherwise. Let $\gamma' \in \Gamma \setminus \{\gamma^*\}$ be a curve, and $x$ a vertex with $\varphi'(x) \in \gamma'$, such that $x$ is a heavy vertex. Denote $\varphi'(x)$ by $p$, and notice that point $p$ may not lie on any other curves in $\Gamma$ (since every point shared by a pair of curves in $\Gamma$ is a crossing point between a pair of edges from $E(P^*)$.)

Let $D = D_{\varphi'}(x)$ be a tiny $x$-disc. For every edge $\hat{e}$ that is incident to $x$, we denote by $\sigma(\hat{e})$ the intersection of $\varphi'(\hat{e})$ with $D$. Let $e, e'$ be the two edges of $P^*$ that are incident to $x$. Denote by $\hat{E}_1(x) \subseteq \delta_G(x)$ the set of edges $\hat{e} \in \delta_G(x)$, such that $\hat{e}$ lies strictly between $e$ and $e'$ in the rotation $\mathcal{O}_x \in \Sigma$ (in clock-wise orientation), and $\hat{e}$ lies on some good path of $\mathcal{P}^*$. Let $\hat{E}_2(x) \subseteq \delta_G(x)$ be the set of edges $\hat{e} \in \delta_G(x)$, such that $\hat{e}$ lies strictly between $e'$ and $e$ in the rotation $\mathcal{O}_x \in \Sigma$ (in clock-wise

orientation), and $\hat{e}$ lies on some good path of $\mathcal{P}^*$. Since Event $\mathcal{E}_3$ did not happen, path $P$ may not be unlucky with respect to $x$, so $|\hat{E}_1(x)|, |\hat{E}_2(x)| \geq \frac{\mathsf{cr}(\varphi)\mu^{13b}}{m}$ holds.

Curve $\gamma'$ partitions the sphere into two regions, that we denote by $\tilde{F}$ and $\tilde{F}'$. Since the curves $\sigma(e), \sigma(e')$ are contained in $\gamma'$, it must be the case that either (i) for every edge $\hat{e} \in \hat{E}_1(x)$, $\sigma(\hat{e}) \subseteq \tilde{F}$, and for every edge $\hat{e} \in \hat{E}_2(x)$, $\sigma(\hat{e}) \subseteq \tilde{F}'$, or (ii) the opposite is true. We assume w.l.o.g. that it is the former. Note that, since $\gamma$ is disjoint from the image of the core $J$ in $\varphi'$ (except for its endpoints), the image of $J$ in $\varphi'$ must be contained either in the interior of $\tilde{F}$, or in the interior of $\tilde{F}'$; we assume w.l.o.g. that it is the former. Consider now some edge $\hat{e} \in \hat{E}_2(x)$, and let $\hat{P} \in \mathcal{P}^*$ be a path that contains $\hat{e}$. Observe that $\sigma(\hat{e}) \subseteq \tilde{F}'$, while both endpoints of $\hat{P}$ belong to $J$, whose image lies in the interior of $\tilde{F}$. Therefore, the image of path $\hat{P}$ in $\varphi'$ must cross the curve $\gamma'$. Let $q$ be a point of $\gamma'$ that lies on the image of $\hat{P}$, such that the image of $\hat{P}$ and $\gamma'$ have a transversal intersection at $q$. Note that $q$ may not be the image of a vertex of $G$, since for every vertex $v \in V(G) \setminus V(J)$, for every pair $P, P' \in \mathcal{P}^*$ of paths containing $v$, the intersection of $P$ and $P'$ at $v$ is non-transversal with respect to $\Sigma$. Therefore, $q$ is a crossing point between an edge of $\hat{P}$ and an edge of $P^*$. We conclude that the edges of $P^*$ participate in at least $|\hat{E}_2(x)| \geq \frac{\mathsf{cr}(\varphi)\mu^{13b}}{m}$ crossings. But since we have assumed that Event $\mathcal{E}_1$ did not happen, path $P^*$ must be good, a contradiction. (Note that it is possible that, for some good path $\hat{P} \in \mathcal{P}^*$, both edges of $\hat{P}$ that are incident to $x$ lie in $\hat{E}_2(x)$. But in that case, the image of $\hat{P}$ must cross $\gamma$ twice, since both endpoints of $\hat{P}$ lie in $J$). □

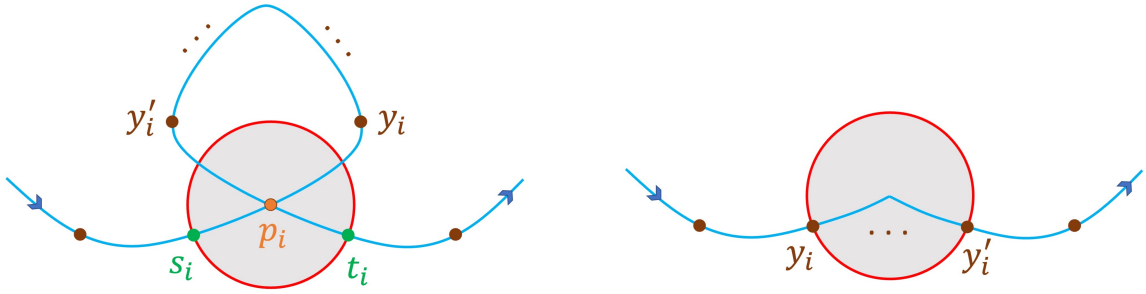
Let $p_1, \ldots, p_z$ denote the points on the curve $\gamma^*$ that correspond to crossing points between pairs of edges of $P^*$, and assume that these points appear on $\gamma^*$ in this order. For all $1 \leq i \leq z$, let $e_i, e_i'$ be the pair of edges of $P^*$ that cross at point $p_i$, with edge $e_i$ appearing before edge $e_i'$ on path $P^*$. For all $1 \leq i \leq z$, let $Q_i$ be the subpath of path $P^*$ from edge $e_i$ to edge $e_i'$, and we denote by $y_i, y_i'$ the second and the penultimate vertices of $Q_i$, respectively. Let $Q_i'$ be the subpath of $Q_i$ connecting $y_i$ to $y_i'$. From Observation H.2, if Events $\mathcal{E}_1, \mathcal{E}_3$ did not happen, the every vertex of $Q_i'$ is a light vertex. Since we have deleted all edges of $E'$ from $G$ to obtain graph $G'$, every vertex of $Q_i'$ is incident to exactly two edges in $G'$, and these edges lie on path $P^*$.

We let $D_i$ be a tiny $p_i$-disc in the drawing $\varphi'$. Let $s_i$ be the point on the image of edge $e_i$ lying on the boundary of $D_i$, and let $t_i$ be the point on the image of edge $e_i'$ lying on the boundary of $D_i$. We modify the drawing $\varphi'$ in order to "straighten" the loop corresponding to the image of the path $Q_i'$, as follows. First, we truncate the images of the edges $e_i$ and $e_i'$, by deleting the segment of $\varphi'(e_i)$ between $s_i$ and $\varphi'(y_i)$, and similarly deleting the segment of $\varphi'(e_i')$ between $t_i$ and $\varphi'(y_i')$. We then delete the images of all vertices and edges of $Q_i'$ from $\varphi'$. We place the new image of $y_i$ at point $s_i$, and the new image of $y_i'$ at point $t_i$. We then add an image of the path $Q_i'$ as a simple curve with endpoints $s_i$ and $t_i$, that is contained in $D_i$, so that the image of $Q_i'$ is contained in $\gamma^* \cap D_i$ (see Figure 70).

Clearly, this modification does not increase the number of crossings, and it is local to region $F^*$. Once every point $p_1, \ldots, p_z$ is processed, we obtain the final solution $\varphi'$ to instance $I'$ that is compatible with $\varphi$, with $\mathsf{cr}(\varphi') \leq \mathsf{cr}(\varphi)$. Note that our transformation step does not introduce any new crossings. Therefore, if $(e_1, e_2)_p$ is a crossing in drawing $\varphi'$, then there is a crossing between edges $e_1$ and $e_2$ at point $p$ in drawing $\varphi$.

## H.6 Proof of Claim 9.27

Assume for contradiction that Event $\mathcal{E}$ did not happen, but $|E''| > \frac{2\mathsf{cr}(\varphi)\cdot\mu^{12b}}{m} + |\chi^{\mathsf{dirty}}(\varphi)|$. From the Maximum Flow - Minimum Cut Theorem, there is a collection $\mathcal{Q}$ of $\left\lceil \frac{2\mathsf{cr}(\varphi)\cdot\mu^{12b}}{m} \right\rceil + |\chi^{\mathsf{dirty}}(\varphi)|$ edge-disjont paths connecting $s$ to $t$ in $H$. Note that every edge in $H$ corresponds to some distinct edge in graph $G$. We do not distinguish between these edges. We show that, for every path $Q \in \mathcal{Q}$, there is a crossing between an edge of $Q$ and an edge of $E(P^*) \cup E(J)$ in $\varphi'$. From Claim 9.23, and since the




(a) Before: tiny $p_i$-disc $D_i$ is shown in gray, and the original image of path $Q_i$ is shown in blue.

(b) After: the new images of vertices $y_i, y_i'$ are shown in brown, and the new image of path $Q_i$ is shown in blue.

Figure 70: Modifying the image of path $Q_i$.

number of crossings in which the edges of $J$ may participate is bounded by $|\chi^{\mathsf{dirty}}(\varphi)|$, it then follows that the edges of $P^*$ participate in at least $\left\lceil \frac{2\mathsf{cr}(\varphi)\cdot\mu^{12b}}{m} \right\rceil$ crossings in $\varphi$. But, since we have assumed that Event $\mathcal{E}_1$ did not happen, path $P^*$ is good, so its edges may particpate in at most $\frac{\mathsf{cr}(\varphi)\cdot\mu^{12b}}{m}$ crossings, a contradiction. It now remains to prove that, for every path $Q \in \mathcal{Q}$, there is a crossing between an edge of $Q$ and an edge of $E(P^*) \cup E(J)$ in $\varphi'$.

Consider any path $Q \in \mathcal{Q}$. This path naturally defines a path $Q'$ in graph $G$, whose first edge, denoted by $e(Q)$, lies in $\tilde{E}_1$, and last edge, denoted by $e'(Q)$, lies in $\tilde{E}_2$. Then the image of edge $e(Q)$ in $\varphi'$ must intersect the interior of region $F_1$, while the image of edge $e'(Q)$ in $\varphi'$ must intersect the interior of region $F_2$. Therefore, the image of the path $Q'$ in $\varphi'$ must cross the boundary of the face $F_1$. Since path $Q'$ is internally disjoint from $V(J')$, the image of some edge on path $Q'$ must cross the image of some edge of $E(J') = E(P^*) \cup E(J)$ in $\varphi'$.

## H.7 Proof of Observation 9.28

We start by recalling how the enhancement path $P^*$ was selected. Recall that initial set $\mathcal{P}^*$ of paths had cardinality $k \geq \frac{15m}{16\mu^b}$. We denoted by $E_1^* = \{e_1, \ldots, e_k\} \subseteq E_1$ the subset of edges that belong to the paths of $\mathcal{P}^*$, where the edges are indexed so that $e_1, \ldots, e_k$ appear consecutively, in the order of their indices in the ordering $\mathcal{O}(J)$. For all $1 \leq j \leq k$, we denote by $P_j \in \mathcal{P}^*$ the unique path originating at the edge $e_j$. We then selected an index $\lfloor k/3 \rfloor < j^* < \lceil 2k/3 \rceil$ uniformly at random, and we let $P^* = P_{j^*}$. Let $e' \in E_2$ be the edge of $P^*$ lying in $E_2$. Let $\tilde{E}_1$ be the set of edges lying between $e_{j^*}$ and $e'$ in $\mathcal{O}(J)$, and let $\tilde{E}_2$ be the set of edges lying between $e'$ and $e_{j^*}$ in $\mathcal{O}(J)$. Then one of the sets $\tilde{E}_1, \tilde{E}_2$ of edges must contain all edges in $\left\{e_1, \ldots, e_{\lfloor k/3 \rfloor - 1}\right\}$, while the other must contain all edges in $\left\{e_{\lceil 2k/3 \rceil + 1}, \ldots, e_k\right\}$. Therefore, $|\tilde{E}_1|, |\tilde{E}_2| \geq \frac{k}{6} \geq \frac{m}{12\mu^b}$.

Notice that graph $G_1$ may only contain edges from one of the sets $\tilde{E}_1, \tilde{E}_2$, and so $|E(G_1)| \leq m - \frac{m}{32\mu^b}$. Using similar reasoning, $|E(G_2)| \leq m - \frac{m}{32\mu^b}$.

## H.8 Proof of Theorem 9.48

Our algorithm consists of a number of phases. For all $j \geq 1$, the input to phase $j$ consists of a collection $\mathcal{R}_j$ of disjoint clusters of $S$, and, for every cluster $R \in \mathcal{R}_j$, two sets $\mathcal{P}_1(R), \mathcal{P}_2(R)$ of paths in graph $G$. We require that $\mathcal{P}_1(R) = \{P_1(e) \mid e \in \delta_G(R)\}$, where for every edge $e \in \delta_G(R)$, path $P(e)$ has $e$ as its first edge and some edge of $\delta_G(S)$ as its last edge, and all inner vertices of $P(e)$ lie in $V(S) \setminus V(R)$.

Additionally, $\mathrm{cong}_G(\mathcal{P}_1(R)) \leq 400/\alpha$. We also require that there is a subset $\hat{E}_R \subseteq \delta_G(R)$ of at least $\lfloor |\delta_G(R)|/64 \rfloor$ edges, such that $\mathcal{P}_2(R) = \left\{P_2(e) \mid e \in \hat{E}_R\right\}$, where for every edge $e \in \hat{E}_R$, path $P(e)$ has $e$ as its first edge and some edge of $\delta_G(S)$ as its last edge, and all inner vertices of $P(e)$ lie in $V(S) \setminus V(R)$. We denote by $S_j$ the subgraph of $G$ induced by the set $V(S) \setminus \left(\bigcup_{R \in \mathcal{R}_j} V(R)\right)$ of vertices. We will ensure that the following ivariants hold:

P1. for every cluster $R \in \mathcal{R}_j$, $|\delta_G(R)| \leq |\delta_G(S)|$;

P2. every cluster $R \in \mathcal{R}_j$ has the $\alpha$-bandwidth property in graph $G$;

P3. $\sum_{R \in \mathcal{R}_j} |\delta_G(R)| \leq 2|\delta_G(S)| \cdot \sum_{j'=0}^{j-1} \frac{1}{2^{j'}}$;

P4. the congestion caused by the set $\bigcup_{R \in \mathcal{R}_j} \mathcal{P}_2(R)$ of paths is at most $400j/\alpha$;

P5. $|\delta_G(S_j)| \leq |\delta_G(S)|/16^{j-1}$, and there is a set $\mathcal{Q}_j$ of paths in graph $G$, routing the edges of $\delta_G(S_j)$ to edges of $\delta_G(S)$, such that for every path in $\mathcal{Q}_j$, all inner vertices on the path lie in $V(S) \setminus V(S_j)$, and the paths in $\mathcal{Q}_j$ cause congestion at most $2/\alpha$.

The algorithm terminates once $\bigcup_{R \in \mathcal{R}_j} V(R) = V(S)$. Notice that, if we ensure that the above properties hold after each phase, the number of phases of the algorithm is $z \leq \lceil \log m \rceil$ (since $|\delta_G(S_z)| \geq 1$ must hold). Once the algorithm terminates, we return the final set $\mathcal{R}_z$ of clusters. It is then easy to verify that this set of clusters has all required properties.

The input to the first phase is $\mathcal{R}_1 = \emptyset$, so $S_1 = S$. The set $\mathcal{Q}_1$ of paths contains, for every edge $e \in \delta_G(S)$, a path $Q(e)$ consisting of the edge $e$ only. It is easy to verify that all invariants hold for this input.

We now assume that we are given an input $\mathcal{R}_j$ to phase $j$, for which Properties P1 – P5 hold. We now describe the algorithm for executing the $j$th phase.

The algorithm consists of two steps. In the first step, we apply the algorithm from Theorem 4.19 to graphs $G$ and $S_j$ (if $S_j$ is not connected, then we apply the algorithm to every connected component of $S_j$). We let $\mathcal{R}'$ be the set of clusters that the algorithm returns. We start by setting $\mathcal{R}_{j+1} = \mathcal{R}_j \cup \mathcal{R}'$ (but eventually we will discard some clusters from $\mathcal{R}_{j+1}$ in the second step). Before we continue to the second step, we verify that Invariants P1 – P3 hold for the current set $\mathcal{R}_{j+1}$ of clusters.

Recall that, from Invariant P5, $|\delta_G(S_j)| \leq \delta_G(S)/16^{j-1}$. The algorithm from Theorem 4.19 ensures that, for every cluster $R \in \mathcal{R}'$, $|\delta_G(R)| \leq |\delta_G(S_j)| \leq |\delta_G(S)|$. It also ensures that every cluster $R \in \mathcal{R}'$ has the $\alpha$-bandwidth property in $G$, and that:

$$\sum_{R \in \mathcal{R}'} |\delta_G(R)| \leq |\delta_G(S_j)| \cdot \left(1 + O(\alpha \cdot \log^{1.5} m)\right) \leq 2|\delta_G(S_j)| \leq 2|\delta_G(S)|/16^{j-1}.$$

(we have used the fact that $\alpha < \frac{1}{c \log^2 m}$ for a large enough constant $c$). Therefore, Invariants P1 – P3 hold for the current set $\mathcal{R}_{j+1}$ of clusters. Notice that currently $V(S) = \bigcup_{R \in \mathcal{R}_{j+1}} V(R)$ holds.

We now describe the second step of the algorithm. Our goal is to discard some clusters from $\mathcal{R}_{j+1}$, and to define the sets $\mathcal{P}_1(R)$, $\mathcal{P}_2(R)$ of paths for each cluster that remains in $\mathcal{R}_{j+1}$, so that Invariants P4 and P5 hold.

In order to do so, we construct a flow network $H$, as follows. We start from graph $G$, and contract all vertices of $V(G) \setminus V(S)$ into a destination vertex $t$. We also contract every cluster $R \in \mathcal{R}_{j+1}$ into a vertex $u(R)$. Additionally, we add a source vertex $s$. For every cluster $R \in \mathcal{R}'$, we connect $s$ to vertex $u(R)$ with an edge of capacity $|\delta_G(R)|$. All remaining edges of $H$ have capacity 64. This completes the definition of the flow network $H$.

Next, we compute a minimum $s$-$t$ cut $(X,Y)$ in $H$. We partition the edges of $E_H(X,Y)$ into two subsets: set $E'$ containing all edges incident to $s$, and set $E''$ containing all remaining edges. Recall that the capacity of every edge in $E''$ is 64. Clearly, the value of the minimum $s$-$t$ cut in $H$ is at most:

$$\sum_{R\in\mathcal{R}'}|\delta_G(R)|\leq 2|\delta_G(S)|/16^{j-1},$$

as we could set $X=\{s\}$ and $Y=V(G)\setminus X$. Therefore, the total capacity of all edges in $E''$ is at most $2|\delta_G(S)|/16^{j-1}$, and $|E''|\leq\frac{2|\delta_G(S)|}{16^{j-1}\cdot 64}\leq\frac{|\delta_G(S)|}{16^j}$.

We discard from set $\mathcal{R}_{j+1}$ all clusters $R$ with $u(R)\in X$, obtaining the final set $\mathcal{R}_{j+1}$ of clusters. Clearly, Invariants P1 – P3 continue to hold for this final set of clusters. Let $S_{j+1}$ the subgraph of $S$ induced by $V(S)\setminus\left(\bigcup_{R\in\mathcal{R}_{j+1}}V(R)\right)$. Then $\delta_G(S_{j+1})=E''$, and so $|\delta_G(S_{j+1})|\leq|\delta_G(S)|/16^j$.

We now show that there exists a set $\mathcal{Q}_{j+1}$ of paths in graph $G$, routing the edges of $\delta_G(S_{j+1})$ to edges of $\delta_G(S)$, such that all inner vertices on every path lie in $V(S)\setminus V(S_{j+1})$, and the paths of $\mathcal{Q}_{j+1}$ cause congestion at most $2/\alpha$. In order to do so, we first consider the flow network $H$. Let $\mathcal{P}^*$ be the set of all paths $P$ in $H$, such that the first edge on $P$ lies in $E''$, the last vertex of $P$ is $t$, and all inner vertices of $P$ lie in $Y$. From the maximum flow / minimum cut theorem, there is a flow $f$ in $H$, defined over the set $\mathcal{P}^*$ of paths, where every edge of $E''$ sends 64 flow units. Note that the edges of $E'$ (and in particular, all edges incident to $s$) do not carry any flow in $f$. Let $H'$ be the graph obtained from $H$ after we contract all vertices of $X$ into a supernode $s^*$, and delete all edges incident to the orginal source $s$ from this graph. From the above discussion, there is an $s^*$-$t$ flow in the resulting graph, in which every edge of $E''$ carries one flow unit, and all other edges of $H'$ carry at most one flow unit each (the flow is obtained by scaling flow $f$ by factor 64). Therefore, there is a collection $\mathcal{Q}$ of $|E''|$ edge-disjoint paths in graph $H'$, routing the edges of $E''$ to vertex $t$. Let $\mathcal{R}^*$ be the set of all clusters $R$ whose corresponding supernode $u(R)$ lies in $Y$. Since every cluster in $\mathcal{R}^*$ has $\alpha$-bandwidth property, from Claim 4.41, there is a collection $\mathcal{Q}_{j+1}$ of paths in graph $G$, routing the edges of $E''=\delta_G(S_{j+1})$ to edges of $\delta_G(S)$, with congestion at most $2/\alpha$, such that all inner vertices on every path in $\mathcal{Q}_{j+1}$ lie in $V(S)\setminus V(S_{j+1})$. This establishes Property P5.

For every cluster $R\in\mathcal{R}_{j+1}\cap\mathcal{R}_j$, we leave the sets $\mathcal{P}_1(R)$ and $\mathcal{P}_2(R)$ of paths unchanged. This ensures that the congestion caused by the set $\mathcal{P}_1(R)$ of paths is at most $400/\alpha$, and that the total congestion caused by the set $\bigcup_{R\in\mathcal{R}_{j+1}\cap\mathcal{R}_j}\mathcal{P}_2(R)$ of paths is at most $400j/\alpha$. Next, we define the sets $\mathcal{P}_1(R)$ and $\mathcal{P}_2(R)$ of paths for clusters $R\in\mathcal{R}_{j+1}\setminus\mathcal{R}_j$.

Consider some cluster $R\in\mathcal{R}_{j+1}\setminus\mathcal{R}_j$, and recal that $\mathcal{R}_{j+1}\setminus\mathcal{R}_j\subseteq\mathcal{R}'$. Recall that the algorithm from Theorem 4.19 returned a set $\mathcal{P}'(R)=\{P'(e)\mid e\in\delta_G(R)\}$ of paths in graph $G$ with $\mathrm{cong}_G(\mathcal{P}'(R))\leq 100$, such that, for every edge $e\in\delta_G(R)$, path $P'(e)$ has $e$ as its first edge and some edge of $\delta_G(S_j)$ as its last edge, and all inner vertices of $P'(e)$ lie in $V(S_j)\setminus V(R)$. We combine these paths with the set $\mathcal{Q}_j$ of paths given by Invariant P5 to obtain the desired set $\mathcal{P}_1(R)=\{P_1(e)\mid e\in\delta_G(R)\}$ of paths, where for every edge $e\in\delta_G(R)$, path $P(e)$ has $e$ as its first edge and some edge of $\delta_G(S)$ as its last edge, and all inner vertices of $P(e)$ lie in $V(S)\setminus V(R)$. Since $\mathrm{cong}_G(\mathcal{P}'(R))\leq 100$, while $\mathrm{cong}_G(\mathcal{Q}_j)\leq 2/\alpha$, it is easy to verify that $\mathrm{cong}_G(\mathcal{P}_1(R))\leq 400/\alpha$.

It now remains to define the sets $\mathcal{P}_2(R)$ of paths for all clusters $R\in\mathcal{R}_{j+1}\setminus\mathcal{R}_j$. In order to do so, we consider again the flow network $H$. Recall that for every cluster $R\in\mathcal{R}_{j+1}\setminus\mathcal{R}_j$, $u(R)\in Y$ holds, and moreover, there is an edge $(s,u(R))$ of capacity $|\delta_G(R)|$, that belongs to $E'\subseteq E_H(X,Y)$. Let $\mathcal{P}^{**}$ be the set of all paths $P$ that connect $s$ to $t$ in $H$, such that the first edge on $P$ lies in $E'$. From the maximum flow/minimum cut theorem, there is a flow $f$ in $H$ over the set $\mathcal{P}^{**}$ of paths, in which every edge $e=(s,u(R))\in E'$ sends $|\delta_G(R)|$ flow units (the capacity of the edge $e$). Scaling this flow down by factor 64, using the integrality of flow, and deleting the first edge from every flow-path, we obtain a collection $\mathcal{Q}'$ of edge-disjoint paths in graph $H$, such that, for every cluster $R\in\mathcal{R}_{j+1}\setminus\mathcal{R}_j$, at least

$\lfloor |\delta_G(R)|/64 \rfloor$ paths in $\mathcal{Q}'$ originate at edges of $\delta_H(u(R))$, and all paths in $\mathcal{Q}'$ terminate at vertex $t$. As before, we use the algorithm from Claim 4.41 in order to obtain a collection $\mathcal{Q}''$ of paths in graph $G$, such that, for every cluster $R \in \mathcal{R}_{j+1} \setminus \mathcal{R}_j$, there is a subset $\mathcal{Q}''(R) \subseteq \mathcal{Q}''$ of at least $\lfloor |\delta_G(R)|/64 \rfloor$ paths that originate at edges of $\delta_G(R)$, and all paths in $\mathcal{Q}''(R)$ terminate at edges of $\delta_G(S)$. The algorithm ensures that, for every edge $e \in \bigcup_{R \in \mathcal{R}_{j+1} \setminus \mathcal{R}_j} \delta_G(R)$, at most one path of $\mathcal{Q}''$ uses $e$, and the total congestion caused by the paths of $\mathcal{Q}''$ is at most $2/\alpha$. Consider now a cluster $R \in \mathcal{R}_{j+1} \setminus \mathcal{R}_j$, and the corresponding set $\mathcal{Q}''(R)$ of paths. Let $Q \in \mathcal{Q}''(R)$ be any such path. Observe that path $Q$ may not be internally disjoint from $R$. We let $e$ be the last edge on $Q$ that belongs to $\delta_G(R)$, and we truncate path $Q$, so that it now originates at edge $e$ and terminates at some edge of $\delta_G(S)$. This ensures that path $Q$ is internally disjoint from $R$. We let $\mathcal{P}_2(R)$ be the resulting set of paths, obtained after every path of $\mathcal{Q}''(R)$ was processed. From the above discussion, the set $\mathcal{P}_2(R)$ routes a subset $\hat{E}_R \subseteq \delta_G(R)$ of at least $\lfloor |\delta_G(R)|/64 \rfloor$ edges to edges of $\delta_G(S)$; all paths in $\mathcal{P}_2(R)$ are internally disjoint from $R$; and the total congestion caused by the paths in $\bigcup_{R \in \mathcal{R}_{j+1} \setminus \mathcal{R}_j} \mathcal{P}_2(R)$ is at most $2/\alpha$. Altogether, the paths in $\bigcup_{R \in \mathcal{R}_{j+1}} \mathcal{P}_2(R)$ cause congestion at most $400(j+1)/\alpha$, establising Property P4.

## H.9 Proof of Claim 9.54

Consider a pair of indices $0 \leq j \leq r$ and $0 \leq a < 2^{r-j}$, and recall that there is a level-$j$ curve $\lambda_{j,a} \in \Lambda_j$ connecting point $p_{a \cdot 2^j}$ to point $p_{(a+1) \cdot 2^j}$. Recall that we have defined a segment $\sigma_{j,a}$ of the boundary of disc $D$, whose endpoints are $p_{a \cdot 2^j}$ and $p_{(a+1) \cdot 2^j}$, where $\sigma_{j,a}$ does not contain the point $p_{(a+1) \cdot 2^j + 1}$. It will be convenient for us to view the segment $\sigma_{j,a}$ as closed on one side and open on another side, that is, $p_{a \cdot 2^j} \in \sigma_{j,a}$, and $p_{(a+1) \cdot 2^j} \notin \sigma_{j,a}$. We let $T_a^j$ be the set of all anchor vertices whose images lie on the curve $\sigma_{j,a}$.

Notice that, for each level $j$, the collections $\left\{ T_a^j \mid 0 \leq a < 2^{r-j} \right\}$ of vertices are disjoint from each other. For every pair $0 \leq j < j' \leq r$ of levels and indices $0 \leq a < 2^{r-j}$ and $0 \leq a' < 2^{r-j'}$, either $\sigma_{j,a} \cap \sigma_{j',a'} = \emptyset$, or $\sigma_{j,a} \subseteq \sigma_{j',a'}$. In the former case, $T_a^j \cap T_{a'}^{j'} = \emptyset$, while in the latter case, $T_a^j \subseteq T_{a'}^{j'}$.

For all $0 \leq j \leq r$ and $0 \leq a < 2^{r-j}$, we let $X_a^j$ be the subset of vertices of $G'$ with the following properties:

- $X_a^j \cap A = T_a^j$;
- $|\delta_{G'}(X_a^j)|$ is minimized among all sets $X_a^j$ with the above property; and
- $|X_a^j|$ is minimized among all sets $X_a^j$ with the above two properties.

In other words, we let $(X_a^j, V(G') \setminus X_a^j)$ be a minimum cut separating vertices of $T_a^j$ from the remaining vertices of $A$, that minimizes the number of vertices in $X_a^j$. Note that, from Observation 9.52, $|\delta_{G'}(X_a^j)| \leq 4\check{m}'/\mu^{2b}$.

The following simple observation follows immediately from submodularity of cuts.

**Observation H.3** *For every pair $0 \leq j \leq j' \leq r$ of levels and indices $0 \leq a < 2^{r-j}$ and $0 \leq a' < 2^{r-j'}$, if $T_a^j \cap T_{a'}^{j'} = \emptyset$ then $X_a^j \cap X_{a'}^{j'} = \emptyset$, and if $T_a^j \subseteq T_{a'}^{j'}$, then $X_a^j \subseteq X_{a'}^{j'}$.*

**Proof:** Consider a pair $0 \leq j \leq j' \leq r$ of levels, and indices $0 \leq a < 2^{r-j}$ and $0 \leq a' < 2^{r-j'}$. Assume first that $T_a^j \cap T_{a'}^{j'} = \emptyset$, but $X_a^j \cap X_{a'}^{j'} \neq \emptyset$. Let $Y = X_a^j \setminus X_{a'}^{j'}$ and $Y' = X_{a'}^{j'} \setminus X_a^j$. Since $X_a^j \cap A = T_a^j$ and $X_{a'}^{j'} \cap A = T_{a'}^{j'}$, we get that $Y \cap A = T_a^j$ and $Y' \cap A = T_{a'}^{j'}$. Since $X_a^j \cap X_{a'}^{j'} \neq \emptyset$, $|Y| < |X_a^j|$ and $|Y'| < |X_{a'}^{j'}|$ holds. Lastly, from submodularity of cuts:

$$|\delta_{G'}(Y)| + |\delta_{G'}(Y')| \leq |\delta_{G'}(X_a^j)| + |\delta_{G'}(X_{a'}^{j'})|.$$

Since $|\delta_{G'}(X_a^j)|$ minimizes the number of edges in a cut separating the vertices of $T_a^j$ from the remaining vertices of $A$, and similarly $|\delta_{G'}(X_{a'}^{j'})|$ minimizes the number of edges in a cut separating the vertices of $T_{a'}^{j'}$ from the remaining vertices of $A$, $|\delta_{G'}(Y)| = |\delta_{G'}(X_a^j)|$ and $|\delta_{G'}(Y')| \leq |\delta_{G'}(X_{a'}^{j'})|$ must hold, a contradiction.

Assume now that $T_a^j \subseteq T_{a'}^{j'}$, but $X_a^j \not\subseteq X_{a'}^{j'}$. Let $Y = X_a^j \cap X_{a'}^{j'}$, and let $Y' = X_{a'}^{j'} \cup X_a^j$. It is immediate to verify that $Y \cap A = T_a^j$, $Y' \cap A = T_{a'}^{j'}$, and $|Y| < |X_a^j|$. From submodularity of cuts:

$$|\delta_{G'}(Y)| + |\delta_{G'}(Y')| \leq |\delta_{G'}(X_a^j)| + |\delta_{G'}(X_{a'}^{j'})|.$$

Using the same argument as before, $|\delta_{G'}(Y)| = |\delta_{G'}(X_a^j)|$ and $|\delta_{G'}(Y')| = |\delta_{G'}(X_{a'}^{j'})|$ must hold. This contradicts the minimality of the cut $X_a^j$, as $|Y| < |X_a^j|$. □

We denote, for all $0 \leq j \leq r$, $\mathcal{X}^j = \left\{X_a^j \mid 0 \leq a < 2^{r-j}\right\}$, and $\mathcal{X} = \bigcup_{j=0}^{r} \mathcal{X}^j$. For simplicity, we will refer to the sets of vertices in $\mathcal{X}$ as *clusters* (each such vertex set $X_a^j$ indeed naturally defines a cluster $G'[X_a^j]$ of graph $G'$). Note that the set $\mathcal{X}$ of clusters is laminar.

We can naturally associate a partitioning tree $\tau$ with the set $\mathcal{X}$ of clusters. The set of vertices of the tree $\tau$ is $\{u(X) \mid X \in \mathcal{X} \cup \{V(G')\}\}$. The root of the tree is vertex $u(X)$ where $X = V(G')$. This vertex has one child vertex – $u(X_0^r)$, corresponding to the unique cluster in $\mathcal{X}^r$. For every non-root vertex $u(X_a^j)$, there are exactly two level-$(j-1)$ clusters that are contained in $X_a^j$: clusters $X_{a'}^{j-1}$ and $X_{a''}^{j-1}$, where $a' = 2a$ and $a'' = 2a+1$. Vertices $u(X_{a'}^{j-1})$ and $u(X_{a''}^{j-1})$ become child-vertices of $u(X_a^j)$ in the tree; we refer to clusters $X_{a'}^{j-1}$ and $X_{a''}^{j-1}$ as *child-clusters* of $X_a^j$, where $X_{a'}^{j-1}$ is the left child and $X_{a''}^{j-1}$ is the right child. We also say that $X_a^j$ is a *parent-cluster* of $X_{a'}^{j-1}$ and $X_{a''}^{j-1}$. The leaves of the tree $\tau$ are vertices in set $\left\{u(X) \mid X \in \mathcal{X}^0\right\}$.

It will be convenient for us to subdivide some of the edges of $G'$, in order to ensure the following two properties:

- P1. for every cluster $X \in \mathcal{X}$, if $e = (x, y) \in \delta_{G'}(X)$, with $x \in X$, then vertex $y$ lies in the parent-cluster of $X$, and neither $x$ nor $y$ are anchor vertices;
- P2. for every cluster $X \in \mathcal{X}$, if $X'$ and $X''$ are the two child-clusters of $X$, and we denote $Y = X \setminus (X' \cup X'')$, then for every pair $e, e' \in \delta_{G'}(Y)$ of edges, the two edges $e, e'$ do not share endpoints.

In order to achieve the above properties, we will subdivide some edges of $G'$, and we will update the clusters in $\mathcal{X}$ accordingly. We will ensure that the clusters remain laminar, and that, for all $0 \leq j \leq r$ and $0 \leq a < 2^{r-j}$, $X_a^j$ remains the smallest cut separating the vertices of $T_a^j$ from the remaining vertices of $A$, with $|\delta_{G'}(X_a^j)|$ remaining unchanged.

In order to perform this transformation, we process the clusters of $\mathcal{X}$ in sets $\mathcal{X}^0, \mathcal{X}^1, \ldots, \mathcal{X}^r$ in this order of the sets.

Consider an iteration when some cluster $X_a^j$ is processed, for $0 \leq j \leq r$ and $0 \leq a < 2^{r-j}$. Consider any edge $e = (x, y) \in \delta_{G'}(X_a^j)$, and assume that $x \in X_a^j$. We subdivide edge $e$ with two new vertices, replacing it with a path $(x, t_e, t'_e, y)$. We add vertex $t_e$ to both $X_a^j$ and all its ancestor clusters, and we add vertex $t'_e$ to all ancestor clusters of $X_a^j$ (but not to $X_a^j$). Notice that, after this subdivision step, the updated set $\mathcal{X}$ of clusters remains laminar, and for every cluster $X \in \mathcal{X}$, $|\delta_{G'}(X)|$ does not grow. Once we process every edge $e \in \delta_{G'}(X_a^j)$, we complete the processing of cluster $X_a^j$. Once every cluster in $\mathcal{X}$ is processed, we obtain an updated graph $G'$, with the updated family $\mathcal{X}$ of clusters, for which properties P1 and P2 hold. We update the input drawing $\varphi$ of graph $G'$ by subdiving the images of its edges appropriately, to obtain a drawing of the current graph $G'$, that we also denote by $\varphi$. Note that

for all $0 \leq j \leq r$ and $0 \leq a < 2^{r-j}$, every edge $e$ in the current set $\delta_{G'}(X_a^j)$ is obtained by subdiving some edge in the original graph $G'$, and both endpoints of $e$ are new vertices that were used for the subdivision. We can then place the images of the endpoints of $e$ close enough to each other, so that the image of the edge $e$ does not participate in any crossings in the new drawing $\varphi$. We will assume from now on that for all $0 \leq j \leq r$ and $0 \leq a < 2^{r-j}$, the edges of $\delta_{G'}(X_a^j)$ do not participate in crossings in $\psi$.

For all $0 \leq j \leq r$ and $0 \leq a < 2^{r-j}$, we define a graph $H_{j,a}$ associated with cluster $X_a^j$, as follows. If $j = 0$, then $H_{j,a} = G'[X_a^j]$. Otherwise, let $X_{a'}^{j-1}, X_{a''}^{j-1}$ be the two child clusters of cluster $X_a^j$, where $X_{a'}^{j-1}$ is the left child cluster. We let $H_{j,a}$ be the subgraph of $G'$ induced by vertex set $X_a^j \setminus \left(X_{a'}^{j-1} \cup X_{a''}^{j-1}\right)$. We also define three subsets of vertices of $H_{j,a}$: set $T_{j,a}^{\mathsf{parent}}$ contains all vertices of $H_{j,a}$ that serve as endpoints of the edges of $\delta_{G'}(X_a^j)$; set $T_{j,a}^{\mathsf{lchild}}$ contains all vertices of $H_{j,a}$ that serve as endpoints of the edges of $\delta_{G'}(X_{a'}^{j-1})$; and set $T_{j,a}^{\mathsf{rchild}}$ contains all vertices of $H_{j,a}$ that serve as endpoints of the edges of $\delta_{G'}(X_{a''}^{j-1})$. From Property P2, these three sets of vertices are mutually disjoint. We denote by $T(H_{j,a}) = T_{j,a}^{\mathsf{parent}} \cup T_{j,a}^{\mathsf{lchild}} \cup T_{j,a}^{\mathsf{rchild}}$.

For $j = 0$, for all $0 \leq a < 2^r$, we define the set $T_{j,a}^{\mathsf{parent}}$ of vertices of graph $H_{j,a}$ similarly. We do not define the sets $T_{j,a}^{\mathsf{lchild}}, T_{j,a}^{\mathsf{rchild}}$ of vertices, but instead we use the set $T_a^j$ of anchor vertices that we defined already. From Property P1, vertex sets $T_{j,a}^{\mathsf{parent}}$ and $T_a^j$ are disjoint. We denote $T(H_{j,a}) = T_{j,a}^{\mathsf{parent}} \cup T_a^j$.

Lastly, we define a graph $H^*$ – a subgraph of $G'$ induced by vertex set $V(G') \setminus X_0^r$. We let $T^*$ be the set of all anchor vertices in $A \setminus T_0^r$, and we let $T^{**}$ be the set of all vertices of $H^*$ that serve as endpoints of the edges of $\delta_{G'}(X_0^r)$. We denote $T(H^*) = T^* \cup T^{**}$.

Let $\mathcal{H} = \{H^*\} \cup \left\{H_{j,a} \mid 0 \leq j \leq r, 0 \leq a < 2^{r-j}\right\}$ be the resulting collection of subgraphs of $G'$. Note that the graphs in $\mathcal{H}$ are mutually disjoint from each other and $\bigcup_{H \in \mathcal{H}} V(H) = V(G')$.

Next, for every graph $H \in \mathcal{H}$, we will define a disc $D(H)$, and we will also define an ordering of the vertices in $T(H)$. We will then modify the current drawing $\varphi$ of graph $G'$, so that, for every graph $H \in \mathcal{H}$, the image of $H$ lies in disc $D(H)$, with the vertices of $T(H)$ lying on the disc boundary, in the pre-specified order. We will ensure that, for all $0 \leq j \leq r$ and $0 \leq a < 2^{r-j}$, the only edges whose images cross the curve $\lambda_a^j$ are the edges of $\delta_{G'}(X_a^j)$. Since, as observed above, $|\delta_{G'}(X_a^j)| \leq 4\check{m}'/\mu^{2b}$, this will ensure that at most $4\check{m}'/\mu^{2b}$ edges cross each curve $\lambda \in \Lambda$ in the final drawing. We now consider every graph $H \in \mathcal{H}$ in turn, define the corresponding disc $D(H)$, and the ordering of the vertices of $T(H)$.

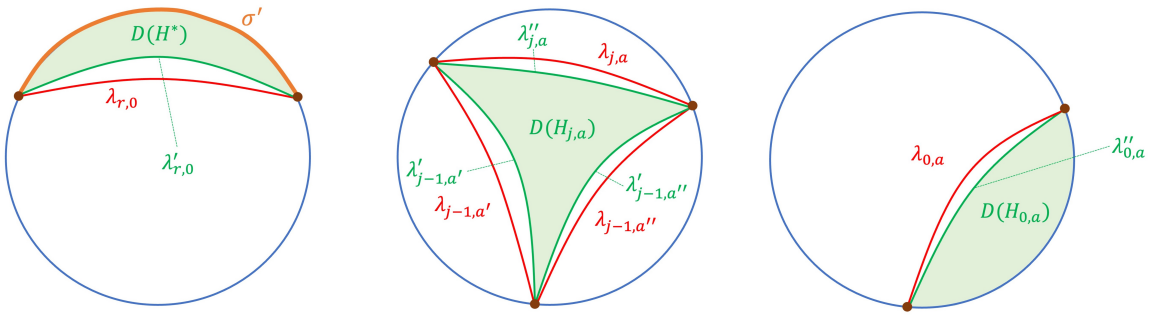
Consider some pair of indices $0 \leq j < r$ and $0 \leq a < 2^{r-j}$. Recall that $(X_a^j, V(G') \setminus X_a^j)$ is a minimum cut in the current graph $G'$ separating vertices of $T_a^j$ from the remaining vertices of $A$. Therefore, there is a set $\mathcal{Q}_{j,a} = \left\{Q_{j,a}(e) \mid e \in \delta_{G'}(X_a^j)\right\}$ of edge-disjoint paths, that are internally disjoint from $X_a^j$, such that, for every edge $e \in \delta_{G'}(X_a^j)$, path $Q_{j,a}(e)$ originates at edge $e$, and terminates at some vertex of $A \setminus T_a^j$. Similarly, there is a set $\mathcal{Q}'_{j,a} = \left\{Q'_{j,a}(e) \mid e \in \delta_{G'}(X_a^j)\right\}$ of edge-disjoint paths, whose inner vertices are contained in $X_a^j$, such that, for every edge $e \in \delta_{G'}(X_a^j)$, path $Q'_{j,a}(e)$ originates at edge $e$, and terminates at some vertex of $T_a^j$. From Lemma 4.7, we can assume w.l.o.g. that the paths in set $\mathcal{Q}_{j,a}$ are non-transversal with respect to the rotation system $\Sigma'$, and the same is true regarding the paths in $\mathcal{Q}'_{j,a}$. We define an oriented ordering $\mathcal{O}_{j,a}$ of the edges in set $\delta_{G'}(X_a^j)$, as follows. For every edge $e \in \delta_{G'}(X_a^j)$, let $v_e$ be the last vertex on path $Q'_{j,a}(e)$, that must lie in $T_a^j$. From our construction, it is easy to verify that every vertex of $A$ has degree 1 in $G'$, so all vertices in set $\left\{v_e \mid e \in \delta_{G'}(X_a^j)\right\}$ are distinct. We define the oriented ordering $\mathcal{O}_{j,a}$ of the edges of $\delta_{G'}(X_a^j)$ to be the order in which their corresponding vertices $v_e$ are encountered along the boundary of the disc $D$, as we traverse it in counter-clock-wise direction. We use this ordering in order to define an oriented ordering $\mathcal{O}(T_{j,a}^{\mathsf{parent}})$ of

the set $T^{\mathsf{parent}}_{j,a}$ of vertices of graph $H^j_a$: recall that the vertices of $T^{\mathsf{parent}}_{j,a}$ are the endpoints of the edges of $\delta_{G'}(X^j_a)$ that lie in $X^j_a$, and every edge in $\delta_{G'}(X^j_a)$ is incident to a distinct vertex of $T^{\mathsf{parent}}_{j,a}$. We let the oriented ordering $\mathcal{O}(T^{\mathsf{parent}}_{j,a})$ of the vertices of $T^{\mathsf{parent}}_{j,a}$ be identical to the oriented ordering $\mathcal{O}_{j,a}$ of the edges of $\delta_{G'}(X^j_a)$, except that we reverse the orientation. In other words, in order to obtain the ordering $\mathcal{O}(T^{\mathsf{parent}}_{j,a})$, we replace, in ordering $\mathcal{O}_{j,a}$ every edge $e \in \delta_{G'}(X^j_a)$ with its endpoint that lies in $T^{\mathsf{parent}}_{j,a}$, and then reverse the orientation of the resulting ordering.

Assume now that cluster $X^j_a$ is the child cluster of some other cluster $X^{j'}_{a'}$. We assume w.l.o.g. that it is the left child cluster; the other case is dealt with similarly. Recall that the set $T^{\mathsf{lchild}}_{j',a'}$ of vertices contains all endpoints of the edges of $\delta_{G'}(X^j_a)$ that lie in $H_{j',a'}$. We define an ordering $\mathcal{O}(T^{\mathsf{lchild}}_{j',a'})$ of the vertices of $T^{\mathsf{lchild}}_{j',a'}$ to be identical to the oriented ordering $\mathcal{O}_{j,a}$ of the edges of $\delta_{G'}(X^j_a)$. In other words, in order to obtain the ordering $\mathcal{O}(T^{\mathsf{lchild}}_{j',a'})$, we replace, in ordering $\mathcal{O}_{j,a}$ every edge $e \in \delta_{G'}(X^j_a)$ with its endpoint that lies in $T^{\mathsf{lchild}}_{j',a'}$. If $j = r$ and $a = 0$, then cluster $X^r_0$ is the child cluster of $V(G')$. The latter cluster, in turn, corresponds to graph $H^*$. In this case, the set $T^{**}$ of vertices of $H^*$ contains all endpoints of the edges of $\delta_{G'}(H_{r,0})$ that lie in $V(H^*)$. We define an ordering $\mathcal{O}(T^{**})$ of the vertices of $T^{**}$ similarly: it is identical to the ordering $\mathcal{O}_{r,0}$ of the edges of $\delta_{G'}(X^r_0)$.

Next, we define, for every graph $H \in \mathcal{H}$, a corresponding disc $D(H)$. Consider first the graph $H^*$. Let $\lambda'_{r,0}$ be a curve that has the same endpoints as $\lambda_{r,0}$, is internally disjoint from $\lambda_{r,0}$, and follows the curve $\lambda_{r,0}$ closely outside the disc $D^r_0$. Let $\sigma'$ be the segment of the boundary of disc $D$ that connects the two endpoints of $\lambda_{r,0}$, and is internally disjoint from the boundary of disc $D^r_0$. We let $D(H^*)$ be the disc that is contained in $D$, whose boundary is the concatenation of the curves $\lambda'_{r,0}$ and $\sigma'$ (see Figure 71(a)).

Consider now indices $0 < j \leq r$ and $0 \leq a < 2^{r-j}$, and the graph $H_{j,a}$. Let $X_{j-1,a'}$ and $X_{j-1,a''}$ be the left and the right child clusters of $X_{j,a}$, respectively. We let $\lambda''_{j,a}$ be a curve whose endpoints are the same as those of $\lambda_{j,a}$, so that $\lambda''_{j,a}$ follows the curve $\lambda_{j,a}$ closely inside disc $D^j_a$. We let $\lambda'_{j-1,a'}$ be a curve whose endpoints are the same as those of $\lambda_{j-1,a'}$, so that $\lambda'_{j-1,a'}$ follows the curve $\lambda_{j-1,a'}$ closely, and is internally disjoint from disc $D^{j-1}_{a'}$. Similarly, we let $\lambda'_{j-1,a''}$ be a curve whose endpoints are the same as those of $\lambda_{j-1,a''}$, so that $\lambda'_{j-1,a''}$ follows the curve $\lambda_{j-1,a''}$ closely, and is internally disjoint from disc $D^{j-1}_{a''}$. The concatenation of the curves $\lambda''_{j,a}, \lambda'_{j-1,a'}$ and $\lambda'_{j-1,a''}$ is a simple closed curve that is contained in disc $D^j_a$. We let $D(H^j_a)$ be the disc that is contained in $D$, whose boundary is the concatenation of $\lambda''_{j,a}, \lambda'_{j-1,a'}$ and $\lambda'_{j-1,a''}$. (see Figure 71(b)).




(a) Disc $D(H*)$. (b) Disc $D(H_{j,a})$ for $0 < j \leq r$. (c) Disc $D(H_{0,a})$.

Figure 71: Discs $D(H)$ for graphs $H \in \mathcal{H}$

Lastly, we consider the index $j = 0$, and any index $0 \leq a < 2^r$. We let $\lambda''_{0,a}$ be a curve whose endpoints are the same as those of $\lambda_{0,a}$, so that $\lambda''_{0,a}$ follows the curve $\lambda_{0,a}$ closely inside disc $D^0_a$. Recall that

$\sigma_a^0$ is a segment of the boundary of disc $D$, whose endpoints are the same as those of $\lambda_{0,a}$, with point $p_{a+1}$ not lying on $\sigma_a^0$. We let $D(H_{0,a})$ be the disc that is contained in $D$, whose boundary is the concatenation of $\lambda''_{0,a}$ and $\sigma_a^0$ (see Figure 71(c)).

We note that every anchor vertex in $A$ must belong to one of the graphs in $\{H^*\} \cup \{H_{0,a} \mid 0 \leq a < 2^r\}$. For every graph $H \in \mathcal{H}$, we define a set $\chi(H)$ of crossings as follows. For $H = H^*$, $\chi(H)$ contains all crossings in the drawing $\varphi$. For a pair of indices $0 \leq j \leq r$ and $0 \leq a < 2^{r-j}$, $\chi(H_{j,a})$ is the set of all crossings in $\varphi$ in which the edges of $G'[X_a^j]$ participate. The following claim is central to the proof of Claim 9.54.

**Claim H.4** *Consider any graph $H \in \mathcal{H}$, let $\Sigma_H$ be the rotation system for $H$ induced by $\Sigma'$, and let $I_H = (H, \Sigma_H)$ be the resulting instance of* MCNwRS. *There is a solution $\psi(H)$ to instance $I_H$ with $\mathsf{cr}(\psi(H)) \leq O(|\chi(H)|)$, where the image of the graph $H$ is contained in disc $D(H)$. Moreover, the following hold:*

- *If $H = H^*$, then the images of the vertices of $T^* = V(H^*) \cap A$ appear on segment $\sigma'$ of the boundary of $D(H^*)$, in the same locations as in $\varphi$, and the images of the vertices of $T^{**}$ appear on segment $\lambda'_{r,0}$ of the boundary of $D(H^*)$, in the same order as in $\mathcal{O}(T^{**})$, incuding the orientation (that is defined with respect to disc $D(H^*)$).*
- *If $H = H_{j,a}$ for $j = 0$, then the images of the vertices of $T_a^j = V(H_{j,a}) \cap A$ appear on the segment $\sigma_a^j$ of the boundary of disc $D(H_{j,a})$, in the same locations as in $\varphi$, and the images of the vertices of $T_{j,a}^{\mathsf{parent}}$ appear on the segment $\lambda''_{j,a}$ of the boundary of disc $D(H_{j,a})$, in the same order as in $\mathcal{O}(T_{j,a}^{\mathsf{parent}})$, including the orientation (that is defined with respect disc $D(H_{j,a})$).*
- *If $H = H_{j,a}$ for $j > 0$, then the images of the vertices of $T_{j,a}^{\mathsf{parent}}$ appear on the segment $\lambda''_{j,a}$ of the boundary of disc $D(H_{j,a})$, in the same order as in $\mathcal{O}(T_{j,a}^{\mathsf{parent}})$, including the orientation, and similarly, the images of the vertices in sets $T_{j,a}^{\mathsf{lchild}}$ and $T_{j,a}^{\mathsf{rchild}}$ appear on the segments $\lambda'_{j-1,a'}$ and $\lambda'_{j-1,a''}$ of the boundary of the disc $D(H_{j,a})$, respectively, where $X_{a'}^{j-1}$ is the left child of $X_a^j$ and $X_{a''}^{j-1}$ is its right child. The ordering of the images of the vertices of $T_{j,a}^{\mathsf{lchild}}$ on $\lambda'_{j-1,a'}$ is identical to $\mathcal{O}(T_{j,a}^{\mathsf{lchild}})$, including orientation, and the ordering of the images of the vertices of $T_{j,a}^{\mathsf{rchild}}$ on $\lambda'_{j-1,a''}$ is identical to $\mathcal{O}(T_{j,a}^{\mathsf{rchild}})$, including orientation. The orientations of all orderings are with respect to disc $D(H_{j,a})$.*

We provide the proof of Claim H.4 in the following subsection, after we complete the proof of Claim 9.54 using it.

In order to construct the solution $\psi'$ to instance $I'$ we start by planting, for every graph $H \in \mathcal{H}$, the image $\psi(H)$ of $H$ into the disc $D(H)$. From Claim H.4, and since every vertex of $A$ lies in either $T^*$ or in $\bigcup_{a=0}^{2^r-1} T_a^0$, the images of the vertices of $A$ remain the same as in $\varphi$. In order to complete the drawing of graph $G'$, we need to insert the edges of $\delta_{G'}(X_a^j)$ for all $0 \leq j \leq r$ and $0 \leq a < 2^{r-j}$ into the current drawing. Observe that the endpoints of all such edges have degree 2 in $G'$ from our construction of graph $G'$.

We now fix an index $0 \leq j < r$ and $0 \leq a < 2^{r-j}$. Assume that cluster $X_a^j$ is a child cluster of some cluster $X_{a'}^{j+1}$, and assume w.l.o.g. that it is a left child cluster (the other case is dealt with similarly). Consider the set $E' = \delta_{G'}(X_a^j)$ of edges. Recall that these edges define a perfect matching between the sets $T_{j+1,a'}^{\mathsf{lchild}}$ and $T_{j,a}^{\mathsf{parent}}$ of vertices. The images of the vertices of $T_{j+1,a'}^{\mathsf{lchild}}$ appear on curve $\lambda'_{j,a}$, while the images of the vertices of $T_{j,a}^{\mathsf{parent}}$ appear on curve $\lambda''_{j,a}$. Let $D_{j,a}^*$ be the disc that is contained in $D$, whose boundary is the concatenation of the curves $\lambda'_{j,a}$ and $\lambda''_{j,a}$. Denote $E' = \{e_1, e_2, \ldots, e_q\}$, where the edges are indexed according to the ordering $\mathcal{O}_{j,a}$. For all $1 \leq i \leq q$, let $e_i = (x_i, y_i)$, where

$x_i \in T^{\mathsf{lchild}}_{j+1,a'}$ and $y_i \in T^{\mathsf{parent}}_{j,a}$. Then the images of vertices $x_1, \ldots, x_q$ appear on curve $\lambda'_{j,a}$ in the order of their indices, and the images of vertices $y_1, \ldots, y_q$ appear on curve $\lambda''_{j,a}$ in the order of their indices, but the orientations of the two orderings are different (the orientation of the first ordering is with respect to $D(H_{j+1,a'})$, while the orientation of the second ordering is with respect to $D(H_{j,a})$). Therefore, the images of vertices $x_1, x_2, \ldots, x_q, y_q, \ldots, y_1$ appear on the boundary of disc $D^*_{j,a}$ in this circular order. We can then define, for all $1 \leq i \leq q$, a curve $\gamma_i$ that is contained in disc $D^*_{j,a}$ connecting the image of $x_i$ to the image of $y_i$. We can ensure that no two such curves cross each other, and each curve crosses $\lambda_{j,a}$ exactly once. We then let, for all $1 \leq i \leq q$, curve $\gamma_i$ be the image of edge $e_i$. Since, as observed above, $|E'| = |\delta_{G'}(X^j_a)| \leq 4\check{m}'/\mu^{2b}$, we introduce at most $4\check{m}'/\mu^{2b}$ crossings between images of edges of $G'$ and curve $\lambda_{j,a}$. It now only remains to take care of edge set $\delta_{G'}(X^r_0)$. We insert these edges exactly as before. The only difference is that vertex set $T^{\mathsf{lchild}}_{j+1,a}$ is replaced with $T^{**}$.

We have now obtained a solution $\psi'$ to instance $I'$, in which the images of all vertices and edges of $G'$ lie in disc $D$, and the images of all anchor vertices remain the same as in $\varphi$. For every curve $\lambda \in \Lambda$, for every vertex $v \in V(G')$, the image of $v$ in $\psi'$ does not lie on an inner point of $\lambda$, and for every edge $e \in E(G')$, the image of $e$ in $\psi'$ may intersect $\lambda$ in at most one point. For every curve $\lambda \in \Lambda$, the total number of edges in $E(G')$ whose images intersect $\lambda$ is at most $4\check{m}'/\mu^{2b}$.

It now remains to bound the number of crossings in $\psi'$. Since the insertion of the edges of $\delta_{G'}(X^j_a)$ for all $0 \leq j \leq r$ and $0 \leq a < 2^{r-j}$ did not introduce any crossings, from Claim H.4, $\mathsf{cr}(\psi') \leq \sum_{H \in \mathcal{H}} O(|\chi(H)|)$. Recall $|\chi(H^*)| = \mathsf{cr}(\varphi)$, and, for all $0 \leq j \leq r$ and $0 \leq a < 2^{r-j}$, $|\chi(H^j_a)|$ is number of crossings in $\varphi$ in which the edges of $G'[X^j_a]$ participate. Observe that vertex sets in $\left\{X^j_a \mid 0 \leq j \leq r, 0 \leq a < 2^{r-j}\right\}$ define a laminar family of depth $O(\log \check{m}')$. Therefore, every edge $e$ may belong to at most $O(\log \check{m}')$ graphs in set $\left\{G'[X^j_a] \mid 0 \leq j \leq r, 0 \leq a < 2^{r-j}\right\}$. We conclude that every crossing $(e, e')_p$ of $\varphi$ may belong to at most $O(\log \check{m}')$ sets $\{\chi(H) \mid H \in \mathcal{H}\}$. Therefore, overall, $\mathsf{cr}(\psi') \leq \sum_{H \in \mathcal{H}} O(|\chi(H)|) \leq \mathsf{cr}(\varphi) \cdot O(\log \check{m}')$. In order to complete the proof of Claim 9.54, it remains to prove Claim H.4, which we do next.

## H.10 Proof of Claim H.4

Fix a pair of indices $0 \leq j \leq r$ and $0 \leq a < 2^{r-j}$. Recall that we have defined two sets of paths associated with the edges of $\delta_{G'}(X^j_a)$. The first set of paths is a set $\mathcal{Q}_{j,a} = \left\{Q_{j,a}(e) \mid e \in \delta_{G'}(X^j_a)\right\}$ of edge-disjoint paths, that are internally disjoint from $X^j_a$, such that, for every edge $e \in \delta_{G'}(X^j_a)$, path $Q_{j,a}(e)$ originates at edge $e$, and terminates at some vertex of $A \setminus T^j_a$. The second set of paths is a set $\mathcal{Q}'_{j,a} = \left\{Q'_{j,a}(e) \mid e \in \delta_{G'}(X^j_a)\right\}$ of edge-disjoint paths, that are contained in $G'[X^j_a]$, such that, for every edge $e \in \delta_{G'}(X^j_a)$, path $Q'_{j,a}(e)$ originates at edge $e$, and terminates at some vertex of $T^j_a$. Both sets of paths are non-transversal with respect to $\Sigma'$. We now define two sets of curves: $\Gamma_{j,a}$ (corresponding to the paths in $\mathcal{Q}_{j,a}$), and $\Gamma'_{j,a}$ (corresponding to the paths in $\mathcal{Q}'_{j,a}$) that will be used in constructing the new drawings for graphs in $\mathcal{H}$. We denote by $\sigma'_{j,a}$ the segment of the boundary of disc $D$ that is the complement of $\sigma_{j,a}$. In other words, if the boundary of $D$ is denoted by $\beta$, then $\sigma'_{j,a} = \beta \setminus \sigma_{j,a}$. For every edge $e \in \delta_{G'}(X^j_a)$, we denote $e = (x_e, y_e)$, where $x_e \in X^j_a$.

We start with the set $\Gamma'_{j,a}$ of curves. Initially, for every edge $e \in \delta_{G'}(X^j_a)$, we let $\gamma'_{j,a}(e)$ be the image of the path $Q'_{j,a}(e)$. Note that curve $\gamma'_{j,a}(e)$ connects the image of $y_e$ to some point on $\sigma_{j,a}$, and it contains $\varphi(e)$. Let $\Gamma'_{j,a} = \left\{\gamma'_{j,a}(e) \mid e \in \delta_{G'}(X^j_a)\right\}$. From the definition of the ordering $\mathcal{O}_{j,a}$, the oriented ordering $\mathcal{O}_{j,a}$ of the edges in $\delta_{G'}(X^j_a)$ is identical to the order of the endpoints of their corresponding curves $\gamma'_{j,a}(e)$ along the boundary of the disc $D$. We assume w.l.o.g. that the orientation of the ordering $\mathcal{O}_{j,a}$ is counter-clock-wise. The curves of $\Gamma'_{j,a}$ may not be in general position: if a vertex

of $V(G') \setminus X^j_a$ lies on more than two paths in $\mathcal{Q}'_{j,a}$, then its image belongs to more than two curves of $\Gamma'_{j,a}$. We perform a nudging procedure by modifying the curves in $\Gamma'_{j,a}$ locally within tiny discs $D_\varphi(v)$ of vertices $v \in X^j_a$ that belong to at least two paths of $\mathcal{Q}'_{j,a}$, using the algorithm from Claim 4.34 (see also Section 4.4.3 for the definition of a nuding procedure). Since that paths in $\mathcal{Q}'_{j,a}$ are non-transversal with respect to $\Sigma'$, this nudging procedure does not introduce any new crossings between the curves in $\Gamma'_{j,a}$. We summarize the properties of the resulting set $\Gamma'_{j,a}$ of curves, that are immediate form our definitions and construction, and the fact that the vertices of $A$ have degree 1 in $G'$, in the following observation.

**Observation H.5** *Consider the final set $\Gamma'_{j,a} = \left\{\gamma'_{j,a}(e) \mid e \in \delta_{G'}(X^j_a)\right\}$ of curves. For every edge $e \in \delta_{G'}(X^j_a)$, curve $\gamma'_{j,a}(e)$ connects the image of vertex $y_e$ in $\varphi$ to some point on $\sigma_{j,a}$, and it contains $\varphi(e)$. The number of crossings between the curves in $\Gamma'_{j,a}$ is at most $|\chi(H_{j,a})|$, and the number of crossings between the curves in $\Gamma'_{j,a}$ and the edges of $G' \setminus X^j_a$ is at most $|\chi(H_{j,a})|$. The oriented ordering $\mathcal{O}_{j,a}$ of the edges of $\delta_{G'}(X^j_a)$ is identical to the oriented ordering of the endpoints of the corresponding curves $\gamma'_{j,a}(e)$ on the boundary of disc $D$; we assume that the orientation of the ordering is counter-clock-wise.*

The construction of the set $\Gamma_{j,a}$ of curves is very similar, except that we also perform a type-2 uncrossing for them. In order to obtain the set $\Gamma_{j,a}$ of curves, we simply apply the algorithm from Theorem 4.37 to perform a type-2 uncrossing of the set $\mathcal{Q}_{j,a}$ of paths. We denote the resulting set of curves by $\Gamma_{j,a} = \left\{\gamma_{j,a}(e) \mid e \in \delta_{G'}(X^j_a)\right\}$. We summarize the properties of the resulting set of curves in the following observation, that follows immediately from the discussion so far and Theorem 4.37.

**Observation H.6** *Consider the set $\Gamma_{j,a} = \left\{\gamma_{j,a}(e) \mid e \in \delta_{G'}(X^j_a)\right\}$ of curves. For every edge $e \in \delta_{G'}(X^j_a)$, curve $\gamma_{j,a}(e)$ connects the image of vertex $y_e$ in $\varphi$ to some point on $\sigma'_{j,a}$. There are no crossings between the curves in $\Gamma'_{j,a}$, and the number of crossings between the curves in $\Gamma_{j,a}$ and the edges of $G'[X^j_a]$ is at most $|\chi(H_{j,a})|$. The number of crossings between the curves in $\Gamma_{j,a}$ and the curves in $\Gamma'_{j,a}$ is bounded by $|\chi(H_{j,a})|$.*

(The last assertion follows from Observation H.5 and the fact that the curves in $\Gamma_{j,a}$ are aligned with the graph $\bigcup_{Q \in \mathcal{Q}_{j,a}} Q$.)

We let $\mathcal{O}'_{j,a}$ be the oriented ordering of the edges in $\delta_{G'}(X^j_a)$ defined by the oriented ordering of the endpoints of the corresponding curves in $\left\{\gamma_{j,a}(e) \mid e \in \delta_{G'}(X^j_a)\right\}$ on the boundary of disc $D$. We assume w.l.o.g. that the orientation of the ordering is counter-clock-wise. We need the following obervation:

**Observation H.7** $\mathsf{dist}(\mathcal{O}_{j,a}, \mathcal{O}'_{j,a}) \leq O(|\chi(H_{j,a})|)$.

Note that the above observation bounds the distance between two **oriented** orderings.

**Proof:** For every edge $e \in \delta_{G'}(X^j_a)$, let $\gamma^*(e)$ be the curve obtained by concatenating curves $\gamma_{j,a}(e)$ and $\gamma'_{j,a}(e)$. Let $\Gamma^* = \left\{\gamma^*(e) \mid e \in \delta_{G'}(X^j_a)\right\}$ be the resulting set of curves. It is immediate to verify that $\Gamma^*$ is a valid reordering set of curves for the oriented orderings $\mathcal{O}_{j,a}, \mathcal{O}'_{j,a}$. From Observations H.5 and H.6, the number of crossings between the curves of $\Gamma$ is at most $O(|\chi(H_{j,a})|)$. □

We are now ready to define a solution $\psi(H)$ for every instance $I_H$ with $H \in \mathcal{H}$. We start with a graph $H = H_{j,a}$, where $0 < j \leq r$ and $0 \leq a < 2^{r-j}$. Assume that $X^{j-1}_{a'}$ and $X^{j-1}_{a''}$ are the left and the right child clusters of $X^j_a$ respectively. Recall that the boundary of the disc $D(H_{j,a})$ is the concatenation

of the curves $\lambda''_{j,a}, \lambda'_{j-1,a'}$ and $\lambda'_{j-1,a''}$. We let $\lambda^*_{j,a}$ be a curve with the same endpoints as $\lambda''_{j,a}$, that is internally disjoint from $\lambda''_{j,a}$, and is contained in disc $D(H_{j,a})$. We let $D'(H_{j,a})$ be the disc that is contained in $D(H_{j,a})$, whose boundary is the concatenation of the curves $\lambda^*_{j,a}, \lambda'_{j-1,a'}$ and $\lambda'_{j-1,a''}$.

We obtain an initial drawing $\psi(H_{j,a})$ of graph $H_{j,a}$ as follows. We start with the drawing of the graph $H_{j,a}$ that is induced by $\varphi$. Conside now some vertex $t \in T^{\mathsf{parent}}_{j,a}$, and let $e_t$ be the unique edge of $H_{j,a}$ incident to $t$. We replace the current image of $e_t$ with the concatenation of $\varphi(e_t)$ and the curve $\gamma_{j,a}(e_t) \in \Gamma_{j,a}$, and we move the image of $t$ to the endpoint of this curve that lies on segment $\sigma'_{j,a}$ of the boundary of disc $D$. Consider now some vertex $t \in T^{\mathsf{lchild}}_{j,a}$, and let $e_t$ be the unique edge of $H_{j,a}$ incident to $t$. We replace the current image of $e_t$ with the curve $\gamma'_{j-1,a'}(e_t) \in \Gamma'_{j-1,a'}$, and we place the image of $t$ on the endpoint of the curve that lies in the segment $\sigma'_{j-1,a'}$ of the boundary of $D$. Lastly, we consider vertices $t \in T^{\mathsf{rchild}}_{j,a}$, and for each such vertex, we let $e_t$ be the unique edge of $H_{j,a}$ incident to $t$. We replace the current image of $e_t$ with the curve $\gamma'_{j-1,a''}(e_t) \in \Gamma'_{j-1,a''}$, and we place the image of $t$ on the endpoint of the curve that lies in the segment $\sigma'_{j-1,a'}$ of the boundary of $D$. We plant the resulting drawing of graph $H_{j,a}$ into the disc $D'(H_{j,a})$, so that the segments $\sigma'_{j,a}, \sigma_{j-1,a'}$ and $\sigma_{j-1,a''}$ of the boundary of $D$ coincide with segments $\lambda^*_{j,a}, \lambda'_{j-1,a'}$ and $\lambda'_{j-1,a''}$ of the boundary of $D'(H_{j,a})$, respectively. We denote the resulting drawing of $H_{j,a}$ by $\psi'_{j,a}$. It is easy to verify that it is a valid solution to instance $I_{H_{j,a}}$. Observe that, from our construction, the images of the vertices of $T^{\mathsf{parent}}_{j,a}$ appear on curve $\lambda^*_{j,a}$ in drawing $\psi'_{j,a}$, and their (oriented) ordering on this curve (with respect to disc $D'(H_{j,a})$) is identical to the ordering $\mathcal{O}'_{j,a}$ of their corresponding edges in $\delta_{G'}(X_{j,a})$ that we have defined above. The vertices of $T^{\mathsf{lchild}}(j,a)$ appear on curve $\lambda'_{j-1,a'}$, and their (oriented) ordering on this curve (with respect to disc $D'(H_{j,a})$) is identical to $\mathcal{O}(T^{\mathsf{lchild}}_{j,a})$. Similarly, the vertices of $T^{\mathsf{rchild}}(j,a)$ appear on curve $\lambda'_{j-1,a''}$, and their (oriented) ordering on this curve (with respect to disc $D'(H_{j,a})$) is identical to $\mathcal{O}(T^{\mathsf{rchild}}_{j,a})$.

We now bound the number of crossings in drawing $\psi'_{j,i}$. For convenience, denote $H' = H_{j,a} \setminus \left(T^{\mathsf{parent}}_{j,a} \cup T^{\mathsf{lchild}}_{j,a} \cup T^{\mathsf{rchild}}_{j,a}\right)$. Recall that $\chi(H_{j,a})$ is the set of all crossings in drawing $\psi'$ of $G'$ in which the edges of $G'[X^j_a]$ participate. First, the total number of crossings between the edges of $E(H')$ is clearly bounded by $|\chi(H_{j,a})|$. There are no crossings between the curves in $\Gamma_{j,a}$. The number of crossings between the curves in $\Gamma_{j,a}$ and the edges of $E(H')$ is bounded by $|\chi(H_{j,a})|$ from Observation H.6. The number of crossings between the curves in $\Gamma'_{j-1,a'}$ and the edges of $E(H')$ is bounded by $|\chi(H_{j,a})|$, and the number of crossings between the curves in $\Gamma'_{j-1,a'}$ is also bounded by $|\chi(H_{j,a})|$ from Observation H.5. Similarly, the number of crossings between the curves in $\Gamma'_{j-1,a''}$, and the number of crossings between the curves in $\Gamma'_{j-1,a''}$ and the edges of $E(H')$ is bounded by $|\chi(H_{j,a})|$. It now remains to bound the number of crossings between the curves of $\Gamma'_{j-1,a'}$ and the curves of $\Gamma'_{j-1,a''}$. Notice that each such crossing corresponds to a unique crossing between an edge of $G'[X^{j-1}_{a'}]$ and an edge of $G'[X^{j-1}_{a''}]$. Since $G'[X^{j-1}_{a'}], G'[X^{j-1}_{a''}] \subseteq G'[X^j_a]$, the number of crossings between the curves of $\Gamma'_{j-1,a'}$ and the curves of $\Gamma'_{j-1,a''}$ is also bounded by $|\chi(H_{j,a})|$. Overall, the total number of crossings in $\varphi_{j,a}$ is bounded by $O(|\chi(H_{j,a})|)$.

Finally, we need to "fix" the current drawing of graph $H_{j,a}$ by reordering the images of the vertices of $T^{\mathsf{parent}}_{j,a}$. Recall that every vertex $t \in T^{\mathsf{parent}}_{j,a}$ is an endpoint of a distinct edge in $\delta_{G'}(X^j_a)$. We have defined two oriented orderings of the edges of $\delta_{G'}(X^j_a)$: ordering $\mathcal{O}_{j,a}$ and ordering $\mathcal{O}'_{j,a}$. Each of these orderings naturally defines an oriented ordering of the vertices of $T^{\mathsf{parent}}_{j,a}$: we have denoted by $\mathcal{O}(T^{\mathsf{parent}}_{j,a})$ the ordering of the vertices of $T^{\mathsf{parent}}_{j,a}$ defined by $\mathcal{O}_{j,a}$, after we reverse the ordering. We denote the oriented ordering of the vertices of $T^{\mathsf{parent}}_{j,a}$ corresponding to $\mathcal{O}'_{j,a}$ by $\mathcal{O}'(T^{\mathsf{parent}}_{j,a})$. Note that the images of the vertices of $T^{\mathsf{parent}}_{j,a}$ appear on the curve $\lambda^*_{j,a}$ in the ordering $\mathcal{O}'(T^{\mathsf{parent}}_{j,a})$, as we traverse the boundary of the disc $D'(H_{j,a})$ in the clockwise direction. But we are required to ensure that the imges of the vertices of $T^{\mathsf{parent}}_{j,a}$ appear on the curve $\lambda''_{j,a}$ in the ordering $\mathcal{O}(T^{\mathsf{parent}}_{j,a})$, as we traverse

the boundary of the disc $D'(H_{j,a})$ in the clockwise direction. Let $D^*$ be the disc that is contained in $D(H_{j,a})$, whose boundary is the concatenation of the curves $\lambda''_{j,a}$ and $\lambda^*_{j,a}$.

We place the images of the vertices of $T^{\mathsf{parent}}_{j,a}$ on curve $\lambda''_{j,a}$, so that they are encountered in the order $\mathcal{O}(T^{\mathsf{parent}}_{j,a})$, as we traverse the boundary of $D^*$ in the clockwise direction. Notice that the previous images of the vertices of $T^{\mathsf{parent}}_{j,a}$ appeared on curve $\lambda^*_{j,a}$, and they are encountered on that curve in the order $\mathcal{O}'(T^{\mathsf{parent}}_{j,a})$, as we traverse the boundary of $D^*$ in the counter-clockwise direction. Recall that, from Observation H.7, the distance between the oriented orderings $\mathcal{O}(T^{\mathsf{parent}}_{j,a})$ and $\mathcal{O}'(T^{\mathsf{parent}}_{j,a})$ is $\mathsf{dist}(\mathcal{O}_{j,a},\mathcal{O}'_{j,a}) \leq O(|\chi(H_{j,a})|)$. Therefore, we can define, for every vertex $t \in T^{\mathsf{parent}}_{j,a}$ a curve $\gamma^*_t$, that is contained in disc $D^*$, and connects the original image of $t$ to the new image of $t$. The total number of crossings between the curves in $\left\{\gamma^*_t \mid t \in T^{\mathsf{parent}}_{j,a}\right\}$ is bounded by $O(|\chi(H_{j,a})|)$. In order to obtain the final solution $\psi(H_{j,a})$ to instance $I_{H_{j,a}}$, we start with solution $\psi'_{j,a}$ to the same instance. For every vertex $t \in T^{\mathsf{parent}}_{j,a}$, we consider the unique edge $e_t$ of $H_{j,a}$ that is incident to $t$. We extend the image of $e_t$ by appending the curve $\gamma^*_t$ to it, and we move the image of $t$ to its new location on curve $\lambda''_{j,a}$. This completes the construction of solution $\psi(H_{j,a})$ to instance $I_{H_{j,a}}$, for $j > 0$.

Next, we consider a graph $H_{j,a} \in \mathcal{H}$ with $j = 0$. We first define a curve $\lambda^*_{j,a}$ exactly as before. We then let $D'(H_{j,a})$ be the disc that is contained in $D(H_{j,a})$, whose boundary is the concatenation of the curves $\lambda^*_{j,a}$ and $\sigma_{j,a}$. In order to construct the initial solution $\psi'_{j,a}$ to instance $H_{j,a}$, we start with the drawing of graph $H_{j,a}$ that is induced by $\varphi$. We then process every vertex $t \in T^{\mathsf{parent}}_{j,a}$ exactly as before, replacing the image of the unique edge $e_t$ incident to $t$ with the curve $\gamma_{j,a}(e_t)$, and moving the image of $t$ to segment $\sigma'_{j,a}$ of the boundary of $D$. We plant the resulting drawing of graph $H_{j,a}$ inside disc $D'(H_{j,a})$, so that the segments $\sigma_{j,a}$ in disc $D$ and $D'(H_{j,a})$ coincide, and the images of the anchor vertices in set $T_{j,a}$ remain unchanged. We also ensure that segment $\sigma'_{j,a}$ on the bondary of disc $D$ coincides with segment $\lambda^*_{j,a}$ on the boundary of $D'(H_{j,a})$. We then modify the images of the edges incident to the vertices $T^{\mathsf{parent}}_{j,a}$, and update the images of the vertices of $T^{\mathsf{parent}}_{j,a}$ exactly as before.

The solution $\psi(H^*)$ to the instance $I_{H^*}$ associated with graph $H^*$ is computed very similarly. The main difference is that this time we do not need to define the curve $\lambda^*_{j,a}$, and instead we can plant the initial image of $H^*$ directly into the disc $D(H^*)$, making sure that the images of the anchor vertices that belong to $H^*$ (the vertices of $T^*$) remain unchanged. We no longer need to take care of the set $T^{\mathsf{parent}}_{j,a}$ of vertices, and the vertices of $T^{**}$ are treated like the vertices of $T^{\mathsf{lchild}}_{j,a}$ or $T^{\mathsf{rchild}}_{j,a}$ in the case where $j > 0$.

## H.11 Proof of Observation 9.55

We start with the following simple observation.

**Observation H.8** *Let $p_i, p_{i'}$ be a pair of distinct points of $\Pi$, and assume that, for some integers $0 \leq j \leq r$ and $0 \leq a \leq 2^{r-j}$, $i' = a \cdot 2^j$. Assume further that $|i' - i| \leq 2^j$. Then there is a tunnel of length at most $j+1$ connecting $p_i$ to $p_{i'}$.*

**Proof:** We assume w.l.o.g. that $i' < i$; the other case is symmetric. The proof is by induction on $j$. If $j = 0$, then $i = i' + 1$, and we can let tunnel $L$ consist of a single level-0 curve $\lambda_{0,i'}$, that connects $p_{i'}$ to $p_i$.

Consider now some integer $j > 0$, and assume that the claim holds for all integers $0 \leq \tilde{j} < j$. We prove that the claim holds for $j$. If $i' - i = 2^j$, then there is a single level-$j$ curve $\lambda_{j,a}$ that connects $p_{i'}$ to $p_i$, and we let the tunnel $L$ consist of this one curve. We assume from now on that $i' - i < 2^j$. Let $j'$ be the largest integer so that $2^{j'} \leq i' - i$, so $0 \leq j' < j$ holds. Then there must be a level-$j'$ curve $\lambda_{j',a'}$, whose endpoints are $p_{i'}$ and $p_{i''}$, where $i'' = i' + 2^{j'}$. Notice that $i' < i'' \leq i$ must hold,

and moreover, $|i-i''| \leq 2^{j'} < 2^j$ must hold. If $p_{i''} = p_i$, then we let the tunnel $L$ consist of the curve $\lambda_{j',a'}$. Otherwise, from the induction hypothesis, there is a tunnel $L'$, of length at most $j' < j$, that connects $p_{i''}$ to $p_i$. We let $L$ be a tunnel that is obtained by appending curve $\lambda_{j',a'}$ at the beginning of tunnel $L'$. □

We are now ready to complete the proof of Observation 9.55. Consider a pair $p_i, p_{i'}$ of distinct points of $\Pi$, and assume w.l.o.g. that $i' < i$. Let $j$ be the largest integer, so that at least two points lie in $\left\{p_{a\cdot 2^j} \mid 0 \leq a < 2^{r-j}\right\} \cap \{p_{i''} \mid i' \leq i'' \leq i\}$. Then there must be a pair of points $p_x, p_y$, with $i' \leq x < y \leq i$, such that $x = 2^j \cdot a$ for some integer $a$, and $y = 2^j \cdot (a+1)$. Note that there is a level-$j$ curve $\lambda_{j,a}$ in $\Lambda$ connecting $p_x$ and $p_y$. Moreover, $i - y \leq 2^j$ and $x - i' \leq 2^j$. From Observation H.8, there is a tunnel $L_1$ of length at most $j$ connecting $p_{i'}$ to $p_x$, and a tunnel $L_2$ of length at most $j$ connecting $p_y$ to $p_i$. We then let $L$ be a tunnel obtained by concatenating tunnel $L_1$, curve $\lambda_{j,a}$, and tunnel $L_2$. Note that tunnel $L$ connects $p_{i'}$ to $p_i$, and its length is at most $2j+3 \leq 2r+3 \leq O(\log \check{m}')$.

# I Proofs Omitted from Section 10

## I.1 Proof of Lemma 10.6

We use a parameter $\tau' = c\tau \log_{3/2} m \log_2 m$, were $c$ is a large enough constant, whose value we set later.

The algorithm maintains a collection $\mathcal{C}$ of disjoint clusters of $H \setminus T$, with $\bigcup_{C\in\mathcal{C}} V(C) = V(H) \setminus T$. Set $\mathcal{C}$ of clusters is partitioned into two subsets: set $\mathcal{C}^A$ of active clusters and set $\mathcal{C}^I$ of inactive clusters. We will ensure that every cluster $C \in \mathcal{C}^I$ has the $\alpha'$-bandwidth property, and $|E(C)| \leq m/\tau$. We start with $\mathcal{C}^I = \emptyset$, and $\mathcal{C}^A$ containing all connected components of $H \setminus T$. The algorithm terminates once $\mathcal{C}^A = \emptyset$, and, when this happens, we return $\mathcal{C}^I$ as the algorithm's outcome.

In order to bound the number of edges in $|\bigcup_{C\in\mathcal{C}} \delta_H(C)|$, we use edge budgets and vertex budgets, that we define next.

**Edge budgets.** Consider a cluster $C \in \mathcal{C}$ and an edge $e \in \delta_H(C)$. If $C \in \mathcal{C}^I$, we set the budget $B_C(e) = 1$, and otherwise we set it to be $B_C(e) = \log_{3/2}(2|\delta_H(C)|)$. If cluster $C$ is the unique cluster with $e \in \delta_H(C)$, then we set the budget of the edge $e$ to be $B(e) = B_C(e)$. If there are two clusters $C \neq C' \in \mathcal{C}$ with $e \in \delta_H(C)$ and $e \in \delta_H(C')$, then we set the budget of the edge $e$ to be $B(e) = B_C(e) + B_{C'}(e)$. Lastly, if no cluster $C \in \mathcal{C}$ with $e \in \delta_H(C)$ exists, then we set $B(e) = 0$.

**Vertex budgets.** Vertex budgets are defined as follows. For every cluster $C \in \mathcal{C}^A$, for every vertex $v \in V(C)$, we set the budget $B(v) = \frac{c \deg_C(v) \log_{3/2} m \cdot \log_2(|E(C)|)}{8\tau'}$, where $c$ is the constant used in the definition of $\tau'$. The budgets of all other vertices are set to 0.

**Cluster budgets and total budget.** For a cluster $C \in \mathcal{C}$, we define its edge-budget $B^E(C) = \sum_{e\in\delta_H(C)} B_C(e)$, and its vertex-budget $B^V(C) = \sum_{v\in V(C)} B(v)$. The total budget of a cluster $C \in \mathcal{C}$ is $B(C) = B^E(C) + B^V(C)$, and the total budget in the system is $B^* = \sum_{C\in\mathcal{C}} B(C) = 2\cdot\sum_{e\in E(G)} B(e) + \sum_{v\in V(G)} B(v)$.

**Initial budget.** At the beginning of the algorithm, the budget of every vertex $v \in V(H) \setminus T$ is at most:

$$\frac{c \deg_{H\setminus T}(v) \log_{3/2} m \log_2 |E(H)|}{8\tau'} \leq \frac{\deg_H(v)}{8\tau},$$

the budget of every edge incident to a vertex in $T$ is at most $\log_{3/2}(2|T|) \leq 16 \log m$, while the budget of every other edge is 0. Therefore, the total budget $B^*$ in the system at the beginning of the algorithm is at most:

$$\frac{m}{4\tau} + 2 \cdot 16k \log m \leq \frac{m}{\tau},$$

since $k \leq m/(64\tau \log m)$.

We will ensure that, throughout the algorithm, the total budget $B^*$ never increases. Since, from the definition, $B^* \geq \sum_{C \in \mathcal{C}} |\delta_H(C)|$, this ensures that, when the algorithm terminates, $\sum_{C \in \mathcal{C}} |\delta_H(C)| \leq m/\tau$, so the set $\mathcal{C}^I$ of clusters is a valid output of the algorithm.

**Algorithm execution.** As mentioned above, the algorithm starts with $\mathcal{C}^I = \emptyset$, and $\mathcal{C}^A$ containing all connected components of $H \setminus T$, with $\mathcal{C} = \mathcal{C}^A \cup \mathcal{C}^I$. As long as $\mathcal{C}^A \neq \emptyset$, we perform iterations, where in each iteration we select an arbitrary cluster $C \in \mathcal{C}^A$ to process. We now describe the execution of an iteration in which a cluster $C \in \mathcal{C}^A$ is processed. The algorithm for processing cluster $C$ consists of two parts, that we describe next.

**Part 1: bandwidth property.** In this step we either establish that $C$ has the $\alpha'$-bandwidth property, or we partition it into smaller clusters that will replace $C$ in set $\mathcal{C}^A$. Recall that an augmentation $C^+$ of cluster $C$ is a graph that is obtained from graph $H$, by subdividing every edge $e \in \delta_H(C)$ with a vertex $t_e$, letting $T(C) = \{t_e \mid e \in \delta_H(C)\}$ be this new set of vertices, and then letting $C^+$ be the subgraph of the resulting graph induced by $V(C) \cup T(C)$. Recall that cluster $C$ has the $\alpha'$-bandwidth property iff the set $T(C)$ of vertices is $\alpha'$-well-linked in $C^+$. We apply the algorithm $\mathcal{A}_{\text{ARV}}$ to graph $C^+$ and terminal set $T(C)$, to obtain an $\beta_{\text{ARV}}(m)$-approximate sparsest cut $(X, Y)$ in graph $C^+$ with respect to the set $T(C)$ of terminals. We assume w.l.o.g. that $|X \cap T(C)| \leq |Y \cap T(C)|$.

We consider two cases. First, if $|E(X,Y)| \geq \alpha' \cdot \beta_{\text{ARV}}(m) \cdot |X \cap T(C)|$, then we are guaranteed that the set $T(C)$ of vertices is $\alpha'$-well-linked in $C^+$, and therefore cluster $C$ has the $\alpha'$-bandwidth property. In this case, we continue to Part 2 of the algorithm. Assume now that $|E(X,Y)| < \alpha' \cdot \beta_{\text{ARV}}(m) \cdot |X \cap T(C)|$.

We can assume without loss of generality that, for every vertex $t_e \in T(C)$, if $t_e \in X$, and $e' = (t_e, v)$ is the unique edge that is incident to $t_e$ in $C^+$, then $v \in X$ as well (since otherwise we can move vertex $t_e$ to $Y$, only making the cut sparser). Similarly, if $t_e \in Y$, then $v \in Y$ as well.

Let $X' = X \setminus T(C)$ and $Y' = Y \setminus T(C)$, so $(X', Y')$ is a partition of $C$. Note that $|T(C) \cap X| = |\delta_H(C) \cap \delta_H(X')|$ and similarly $|T(C) \cap Y| = |\delta_H(C) \cap \delta_H(Y')|$.

We remove cluster $C$ from $\mathcal{C}^A$ and from $\mathcal{C}$, and we all connected components of $C[X']$ and $C[Y']$ to $\mathcal{C}^A$ and to $\mathcal{C}$ instead. Observe that we are still guaranteed that $\bigcup_{C' \in \mathcal{C}} V(C') = V(H) \setminus T$. We now show that the total budget in the system does not increase as the result of this step.

Since every cluster that was newly added to $\mathcal{C}^A$ is contained in $C$, it is immediate to verify that, for every vertex $v$ of $C$, its budget may only decrease. The only edges whose budget may increase are the edges of $E_C(X', Y')$. The number of such edges is bounded by $\alpha' \cdot \beta_{\text{ARV}}(m) \cdot |X \cap T(C)| = \alpha' \cdot \beta_{\text{ARV}}(m) \cdot |\delta_H(C) \cap \delta_H(X')|$, and the budget of each such edge increases by at most $\log_{3/2}(2m) \leq 4 \log m$, so the total increase in the budget of all edges due to this step is bounded by:

$$4\alpha' \cdot \beta_{\text{ARV}}(m) \cdot |\delta_H(C) \cap \delta_H(X')| \cdot \log m \leq |\delta_H(C) \cap \delta_H(X')|,$$

since $\alpha' = \frac{1}{16\beta_{\text{ARV}}(m) \cdot \log m}$.

Consider now some edge $e \in \delta_H(C) \cap \delta_H(X')$. Since we have assumed that $|X \cap T(C)| \leq |Y \cap T(C)|$, it is easy to verify that $|\delta_H(X')| \leq 2|\delta_H(C)|/3$. Therefore, for every edge $e \in \delta_H(X') \cap \delta_H(C)$, if $C' \subseteq H[X']$ is the new cluster with $e \in \delta(C')$, then:

$$B_{C'}(e) = \log_{3/2}(2|\delta_H(C')|) \leq \log_{3/2}(2|\delta_H(X')|) \leq \log_{3/2}(2|\delta_H(C)|) - 1.$$

Therefore, the total decrease in the global budget due to the edges of $\delta_H(X') \cap \delta_H(C)$ is at least $|\delta_H(C) \cap \delta_H(X')|$. We conclude that overall the budget $B^*$ does not increase.

We assume from now on that algorithm $\mathcal{A}_{\text{ARV}}$ returned a cut $(X, Y)$ of $C^+$ with $|E(X, Y)| \geq \alpha' \cdot \beta_{\text{ARV}}(m) \cdot |X \cap T(C)|$, and so cluster $C$ has the $\alpha'$-bandwidth property.

If $|E(C)| \leq m/\tau$, then we remove cluster $C$ from $\mathcal{C}^A$, and add it to the set $\mathcal{C}^I$ of inactive clusters. It is easy to verify that total budget $B^*$ may not increase as the result of this step. Therefore, we assume from now on that $|E(C)| > m/\tau$.

**Part 2: sparse balanced cut.** In this step, we apply the algorithm from Theorem 4.11 to graph $C$ with parameter $\hat{c} = 3/4$, to obtain a $\hat{c}'$-edge-balanced cut $(Z, Z')$ of $C$ (where $1/2 < \hat{c}' < 1$), whose size is at most $O(\beta_{\text{ARV}}(m))$ times the size of a minimum 3/4-edge-balanced cut of $C$. We say that this step is *successful* if $|E_H(Z, Z')| < |E(C)|/\tau'$. Assume first that the step was successful. Then we remove cluster $C$ from $\mathcal{C}^A$ and from $\mathcal{C}$, and add all connected components of $C[Z], C[Z']$ to $\mathcal{C}^A$ and to $\mathcal{C}$ instead. We now show that the total budget in the system does not increase as the result of this step. Observe that the budget of every vertex may only decrease, and the same is true for the budget of every edge, except for the edges in set $\delta_H(Z) \cup \delta_H(Z')$.

We first bound the increase in the budgets of the edges of $\delta_H(Z) \cup \delta_H(Z')$. We consider two cases. The first case happens if $|\delta_H(C)| \leq \frac{8|E(C)|}{\tau'}$. Since the budget of every edge is always bounded by $\log_{3/2}(2m)$, and since $|E_H(Z, Z')| \leq \frac{|E(C)|}{\tau'}$, so the total increase in the budgets of all edges is bounded by $\frac{10|E(C)| \cdot \log_{3/2}(2m)}{\tau'}$.

Consider now the second case, where $|\delta_H(C)| > \frac{8|E(C)|}{\tau'}$, and assume without loss of generality that $|\delta_H(Z)| \leq |\delta_H(Z')|$. In this case, since $|E_H(Z, Z')| \leq \frac{|E(C)|}{\tau'}$, $|\delta_H(Z)| \leq |\delta_H(C)|$, so the budgets of the edges in $\delta_H(Z) \cap \delta_H(C)$ may not grow. As before, the budgets of the edges of $E_H(Z, Z')$ may grow by at most $|E_H(Z, Z')| \cdot \log_{3/2}(2m) \leq \frac{|E(C)| \cdot \log_{3/2}(2m)}{\tau'}$. Lastly, for every edge $e \in \delta_H(Z') \cap \delta_H(C)$, the original budget $B_C(e)$ is $\log_{3/2}(2|\delta_H(C)|)$, and, if $C' \subseteq H[Z']$ is the new cluster with $e \in \delta_H(C')$, then the new budget $B_{C'}(e) = \log_{3/2}(2|\delta_H(C')|)$. Since $|\delta_H(C')| \leq |\delta_H(C)| + |E_H(Z, Z')|$, we get that the increase in the budget of $e$ is bounded by $\log_{3/2}\left(\frac{|\delta_H(C)|+|E_H(Z,Z')|}{|\delta_H(C)|}\right) = \log_{3/2}\left(1 + \frac{|E_H(Z,Z')|}{|\delta_H(C)|}\right)$.

Since we have assumed that $|\delta_H(C)| > \frac{8|E(C)|}{\tau'}$, while $|E_H(Z, Z')| < \frac{|E(C)|}{\tau'}$, we get that $\frac{|E_H(Z,Z')|}{|\delta_H(C)|} < 1/2$. Since for all $\epsilon \in (0, 1/2)$, $\ln(1 + \epsilon) \leq \epsilon$, we get that the increase in the budget of $e$ is bounded by $\frac{|E_H(Z,Z')|}{|\delta_H(C)| \cdot \ln(3/2)} \leq \frac{4|E_H(Z,Z')|}{|\delta_H(C)|}$. Overall, we get that the budget of the edges of $\delta_H(Z') \cap \delta_H(C)$ increases by at most:

$$|\delta_H(Z') \cap \delta_H(C)| \cdot \frac{4|E_H(Z, Z')|}{|\delta_H(C)|} \leq 4|E_H(Z, Z')| \leq \frac{4|E(C)|}{\tau'}.$$

To summarize, regardless of which of the above two cases happened, the total increase in the budgets of all edges is bounded by $\frac{10|E(C)| \cdot \log_{3/2}(2m)}{\tau'}$. Next, we show that the total decrease in the budgets of the vertices is high enough to compensate for this increase.

Assume without loss of generality that $|E(Z)| \leq |E(Z')|$. From the definition of edge-balanced cut, $|E(Z')| \leq \hat{c}'|E(C)|$, for some universal constant $\hat{c}'$. In particular:

$$\sum_{v \in Z} \deg_Z(v) \geq 2(|E(C)| - |E(Z')| - |E(Z, Z')|) \geq 2(1 - \hat{c}' - 1/\tau') \cdot |E(C)|. \tag{32}$$

On the other hand, from our assumption that $|E(Z)| \leq |E(Z')|$, $\log_2(|E(Z)|) \leq \log_2(|E(C)|) - 1$. Recall that for every vertex $v \in Z$, its original vertex budget is: $B(v) = \frac{c \deg_C(v) \log_{3/2} m \cdot \log_2(|E(C)|)}{8\tau'}$,

and its new budget is:

$$B'(v) = \frac{c \deg_Z(v) \log_{3/2} m \cdot \log_2(|E(Z)|)}{8\tau'} \leq \frac{c \deg_Z(v) \log_{3/2} m \cdot (\log_2(|E(C)|) - 1)}{8\tau'}.$$

Therefore, for every vertex $v \in Z$, its budget decreases by at least $\frac{c \deg_Z(v) \log_{3/2} m}{8\tau'}$. Overall, the budget of the vertices in $Z$ decreases by at least:

$$\frac{c \log_{3/2} m}{8\tau'} \cdot \sum_{v \in Z} \deg_Z(v) \geq \frac{c \log_{3/2} m}{4\tau'} \cdot (1 - \hat{c}' - 1/\tau') \cdot |E(C)|$$

(from Equation 32.) Since $\tau' = c\tau \log_{3/2} m \log_2 m$, and since we can set $c$ to be a large enough constant, we can ensure that this is at least $\frac{16|E(C)| \cdot \log_{3/2}(2m)}{\tau'}$, so the overall budget in the system does not increase.

We assume from now on that the current step was not successful. In other words, the algorithm from Theorem 4.11 returned a cut $(Z, Z')$ with $|E_H(Z, Z')| \geq |E(C)|/\tau'$. Since the size of this cut is within factor $O(\beta_{\mathrm{ARV}}(m))$ from the minimum 3/4-edge-balanced cut, we conclude that the value of the minimum 3/4-edge-balanced cut in $C$ is at least $\rho = \Omega\left(\frac{|E(C)|}{\tau' \cdot \beta_{\mathrm{ARV}}(m)}\right)$.

Recall that, from Lemma 4.12, if the maximum vertex degree $\Delta$ in graph $C$ is at most $|E(C)|/2^{40}$, and $\mathsf{OPT}_{\mathsf{cr}}(C) \leq |E(C)|^2/2^{40}$, then graph $C$ must contain a (3/4)-edge-balanced cut of value at most $\tilde{c} \cdot \sqrt{\mathsf{OPT}_{\mathsf{cr}}(C) + \Delta \cdot |E(C)|}$ where $\tilde{c}$ is some universal constant. As the size of the minimum 3/4-balanced cut in $C$ is at least $\rho$, we conclude that either $\Delta \geq |E(C)|/2^{40}$, or $\mathsf{OPT}_{\mathsf{cr}}(C) > |E(C)|^2/2^{40}$, or $\sqrt{\mathsf{OPT}_{\mathsf{cr}}(C) + \Delta \cdot |E(C)|} \geq \rho/\tilde{c}$ must hold. The latter can only happen if either $\mathsf{OPT}_{\mathsf{cr}}(C) \geq \frac{\rho^2}{2\tilde{c}^2}$, or $\Delta \geq \frac{\rho^2}{2\tilde{c}^2 \cdot |E(C)|}$. Substituting the value of $\rho = \Omega\left(\frac{|E(C)|}{\tau' \cdot \beta_{\mathrm{ARV}}(m)}\right)$, and recalling that $|E(C)| > m/\tau$, while $\tau' = c\tau \log_{3/2} m \log_2 m$, we conclude that either (i) $\mathsf{OPT}_{\mathsf{cr}}(C) \geq \Omega\left(\frac{|E(C)|^2}{2(\tilde{c}\tau' \beta_{\mathrm{ARV}}(m))^2}\right) \geq \Omega\left(\frac{|E(C)|^2}{\tau^2 \log^5 m}\right) \geq \Omega\left(\frac{m^2}{\tau^4 \log^5 m}\right)$; or (ii) $\Delta \geq \Omega\left(\frac{|E(C)|}{2(\tilde{c}\tau' \beta_{\mathrm{ARV}}(m))^2}\right) \geq \Omega\left(\frac{m}{\tau^3 \log^5 m}\right)$; or (iii) $\Delta \geq \frac{|E(C)|}{2^{40}} \geq \frac{m}{2^{40}\tau}$. However, since we are guaranteed that $\Delta \leq \frac{m}{c^* \tau^3 \log^5 m}$ for a large enough constant $c^*$, we can rule out the latter two options, and conclude that $\mathsf{OPT}_{\mathsf{cr}}(H) \geq \mathsf{OPT}_{\mathsf{cr}}(C) \geq \Omega\left(\frac{m^2}{\tau^4 \log^5 m}\right)$.

If Phase 2 is unsuccessful, then we terminate the algorithm and declare that $\mathsf{OPT}_{\mathsf{cr}}(H) \geq \Omega\left(\frac{|E(H)|^2}{\tau^4 \log^5 m}\right)$.

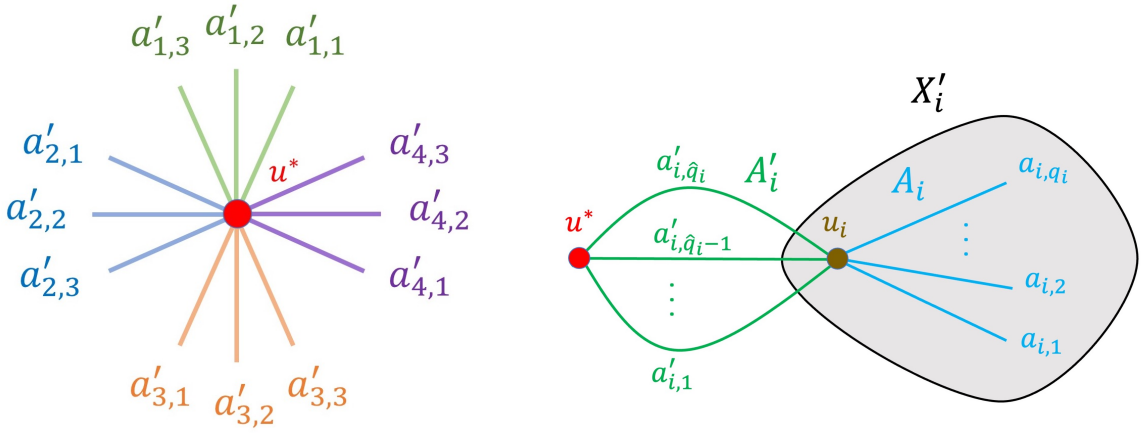
## I.2 Proof of Lemma 10.14

Let $\varphi'$ be a solution to instance $I'$. Throughout the proof, we denote by $D = D_{\varphi'}(u^*)$ the tiny $u^*$-disc in drawing $\varphi'$. Recall that $\delta_{G'}(u^*) = \left\{a'_{1,1}, \ldots, a'_{1,\hat{q}_1}, a'_{2,1}, \ldots, a'_{2,\hat{q}_2}, \ldots, a'_{k,1}, \ldots, a'_{k,\hat{q}_k}\right\}$. Moreover, the edges of $\delta_{G'}(u^*)$ appear in this circular order in the rotation $\mathcal{O}'_{u^*} \in \Sigma'$. We assume w.l.o.g. that the orientation of this ordering in $\varphi'$ is positive. In other words, the edges are encountered in this order as we traverse the boundary of $D$ so that the interior of $D$ always lies to our left (see, e.g. Figure 72(a), in which the orientation of the ordering $\mathcal{O}'_{u^*}$ is positive).

Consider now vertex $u_i$, for some $1 \leq i \leq k$. Recall that the set $\delta_{G'}(u_i)$ of edges is the union of two subsets: set $A'_i = \left\{a'_{i,1}, \ldots, a'_{i,\hat{q}_i}\right\}$ of parallel edges connecting $u_i$ to $u^*$, and set $A_i = \{a_{i,1}, \ldots, a_{i,q_i}\}$ of edges corresponding to the edge set $E_i \subseteq E(G)$. Recall that the ordering of the edges in $\delta_{G'}(u_i)$ in the rotation system $\Sigma'$ is: $(a'_{i,1}, a'_{i,2}, \ldots, a'_{i,\hat{q}_i}, a_{i,q_i}, a_{i,q_i-1}, \ldots, a_{i,1})$. We say that vertex $u_i$ is *synchronized* with $u^*$, if the orientation of the above ordering in $\varphi'$ is negative (see Figure 72(b)). In other words, if we traverse the boundary of the tiny $u_i$-disc $D_{\varphi'}(u_i)$ so that its interior always lies to our right, then we will encounter the edges of $\delta_{G'}(u_i)$ in the order $(a'_{i,1}, a'_{i,2}, \ldots, a'_{i,\hat{q}_i}, a_{i,q_i}, a_{i,q_i-1}, \ldots, a_{i,1})$. We need the following simple observation.

**Observation I.1** *If, for some index $1 \leq i \leq k$, vertex $u_i$ is not synchronized with $u^*$, then there are at least $\hat{q}_i^2/8$ crossings $(e, e')$ in $\varphi'$ with $e, e' \in A'_i$.*

**Proof:** We delete from drawing $\varphi'$ all vertices and edges except for vertices $u^*, u_i$ and edges of $A'_i$. For all $1 \leq j \leq \hat{q}_i$, let $s_j$ be the point on the boundary of $D$ that lies on the image of edge $e_{i,j}$, let $t_j$ be the point on the boundary of $D_{\varphi'}(u_i)$ that lies on the image of $e_{i,j}$, and let $\gamma_j$ be the segment of the image of edge $e_{i,j}$ between $s_j$ and $t_j$. We assume without loss of generality that $\gamma_j$ does not cross itself; if it does, then we remove self loops until $\gamma_j$ does not cross itself. Denote $\Gamma = \{\gamma_1, \ldots, \gamma_{\hat{q}_j}\}$ the resulting set of curves.

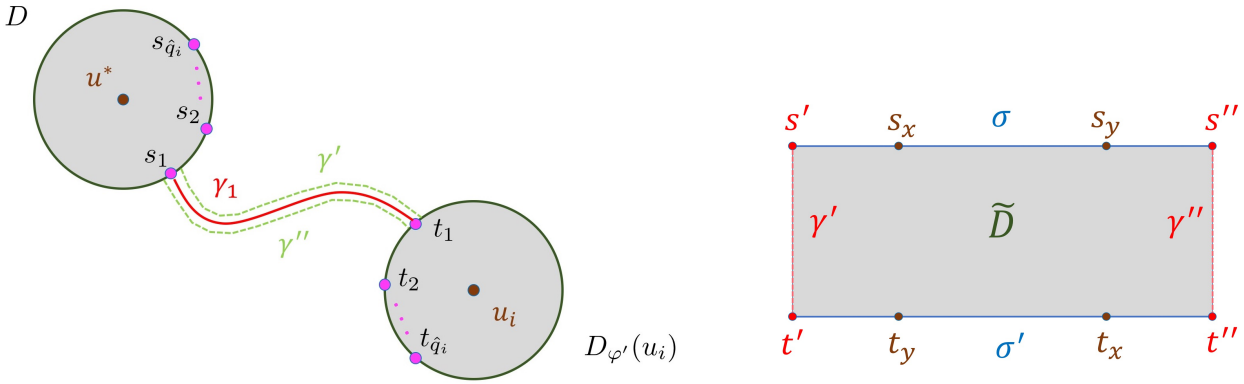



(a) The ordering $\mathcal{O}'_{u^*}$ with positive orientation.

(b) Vertex $u_i$ is synchronized with $u^*$. Note that edges of $A'_i$ may cross each other but they intersect the boundaries of the discs $D_{\varphi'}(u^*)$ and $D_{\varphi'}(u_i)$ in the order indicated above.

Figure 72: An illustration of the positive orientation of $\mathcal{O}'_{u^*}$ and the oriented rotation of the synchronized vertex $u_i$.

From our assumptions, points $s_1, \ldots, s_{\hat{q}_i}$ appear in this order on the boundary of $D$, when we traverse it so that the interior of $D$ lies to our left, while points $t_1, \ldots, t_{\hat{q}_i}$ appear in this order on the boundary of $\eta_i$, when we traverse it so that the interior of the disc is to our left (see Figure 73(a)).




(a) Points $\{s_k, t_k\}_{1 \leq k \leq \hat{q}_i}$ and curves $\gamma_1, \gamma', \gamma''$.

(b) Curves $\sigma, \gamma'', \sigma', \gamma'$ and disc $\tilde{D}$.

Figure 73: Illustrations for the proof of Observation I.1.

Assume for contradiction that there are fewer than $\hat{q}_i^2/8$ crossings $(e, e')$ in $\varphi'$ with $e, e' \in A'_i$. Then

there is some curve $\gamma_j \in \Gamma$, whose image crosses fewer than $\hat{q}_i/8$ curves in $\Gamma$. Assume w.l.o.g. that this curve is $\gamma_1$.

Let $\Gamma' \subseteq \Gamma \setminus \{\gamma_1\}$ be the set of curves that do not cross $\gamma_1$, so $|\Gamma'| \geq |\Gamma|/2$. We next show that every pair distinct of curves in $\Gamma'$ must cross, leading to a conradiction. Indeed, consider any pair $\gamma_x, \gamma_y \in \Gamma'$ of distinct curves in $\Gamma'$, and assume without loss of generality that $x < y$. Let $\gamma'$ and $\gamma''$ be two curves that follow curve $\gamma_1$ immediately to the left and immediately to the right, respectively. Let $s', s''$ be the endpoints of curves $\gamma'$ and $\gamma''$ lying on the boundary of $D$, respectively, and let $t', t''$ be be the endpoints of curves $\gamma'$ and $\gamma''$ lying on the boundary of $D_{\varphi'}(u_i)$, respectively. Notice that points $s', s''$ partition the boundary of $D$ into two segments, whose endpoints are $s'$ and $s''$; we let $\sigma$ be the segment that does not contain $s_1$. Similarly, points $t', t''$ partition the boundary of $D_{\varphi'}(u_i)$ into two segments, whose endpoints are $t'$ and $t''$; we let $\sigma'$ be the segment that does not contain $t_1$. Let $\lambda$ be the closed curve obtained by concatenating curves $\sigma, \gamma'', \sigma'$, and $\gamma'$ (see Figure 73(b)), and let $\tilde{D}$ be the disc whose boundary is $\lambda$, that contains the images of the curves $\gamma_x$ and $\gamma_y$. Then points $s_x, s_y, t_x, t_y$ appear on the boundary of $\eta^*$ in this order. Therefore, curves $\gamma_x, \gamma_y$ must cross. We conclude that every pair of curves in $\Gamma'$ must cross, a contradiction. □

In order to transform the drawing $\varphi'$ of $G'$ into a drawing $\varphi$ of $G$, we start by considering the tiny $u^*$-disc $D$ in the drawing $\varphi'$. For each edge $a'_{i,j} \in \delta_{G'}(u^*)$, the image of the edge in $\varphi'$ intersects the boundary of $D$ at exactly one point, that we denote by $p_{i,j}$. Recall that, from our assumptions, points $p_{1,1}, \ldots, p_{1,\hat{q}_1}, p_{2,1}, \ldots, p_{2,\hat{q}_2}, \ldots, p_{k,1}, \ldots, p_{k,\hat{q}_k}$ appear in this order on the boundary of $D$, if we traverse it so that the interior of $D$ lies to our left.

For all $1 \leq i \leq k$, we define a segment $\sigma_i$ of the boundary of $D$, that contains all points $p_{i,1}, \ldots, p_{i,\hat{q}_i}$. Observe that these segments can be defined so that they are mutually disjoint, and they appear on the boundary of $D$ in their natural order $\sigma_1, \ldots, \sigma_k$, as we traverse the boundary of $D$ so that its interior lies to our left. Next, for each $1 \leq i \leq k$, we define a disc $D_i$, that is contained in $D$, such that the intersection of the boundary of $D$ and the boundary of $D_i$ is precisely $\sigma_i$, the image of $u^*$ lies outside $D_i$, and all discs $D_1, \ldots, D_k$ are mutually disjoint. From the above discussion, for all $1 \leq i \leq k$, the points $p_{i,1}, \ldots, p_{i,\hat{q}_i}$ appear in this order on segment $\sigma_i$ of the boundary of $D_i$, as we traverse this boundary so that the interior of the disc $D_i$ lies to our left (see Figure 74).

Consider now some index $1 \leq i \leq k$. Let $\sigma'_i$ be any segment of non-zero length on the boundary of disc $D_i$, that is disjoint from segment $\sigma_i$. Let $p'_{i,1}, \ldots, p'_{i,q_i}$ be an arbitrary collection of distinct points on $\sigma'_i$, that appear on $\sigma'_i$ in this order, as we traverse the boundary of $D_i$ so that its interior lies to our right (see Figure 74). We can then define, for each $1 \leq i \leq k$ and $1 \leq j \leq q_i$, a curve $\zeta_{i,j}$, that originates at the image of $u^*$ and terminates at point $p'_{i,j}$, such that all curves in set $\{\zeta_{i,j} \mid 1 \leq i \leq k, 1 \leq j \leq q_i\}$ are mutually internally disjoint. From our definitions so far, the circular order in which these curves enter the image of $u^*$ is: $(\zeta_{1,1}, \ldots, \zeta_{1,q_1}, \zeta_{2,1}, \ldots, \zeta_{2,q_2}, \ldots, \zeta_{k,1}, \ldots, \zeta_{k,q_k})$ (see Figure 74). For all $1 \leq i \leq k$ and $1 \leq j \leq q_i$, we will use the curve $\zeta_{i,j}$ in order to draw the edge $e_{i,j} \in E_i$; in fact we will refer to $\zeta_{i,j}$ as the *first segment of the drawing of edge* $e_{i,j}$. We will later define a second segment of the drawing of this edge, and then eventually stitch the two segments together to complete the drawing of the edge.

Notice that so far, for all $1 \leq i \leq k$, we have defined a collection $\left\{p'_{i,1}, p'_{i,2}, \ldots, p'_{i,q_i}, p_{i,\hat{q}_i}, p_{i,\hat{q}_i-1}, \ldots, p_{i,1}\right\}$ of points on the boundary of $D_i$, that appear on it in this order, as we traverse the boundary so that the interior of $D_i$ lies to our right (see Figure 75). Lastly, for all $1 \leq i \leq k$, we define another disc $D'_i \subseteq D_i$, whose boundary is disjoint from the boundary of $D_i$ (see Figure 75).

In the remainder of the algorithm, we process each index $1 \leq i \leq k$ one by one. We let $G_0 = G'$, and for all $1 \leq i \leq k$, we let $G_i$ be the graph obtained from $G'$ by contracting vertices $u_1, \ldots, u_i$ into the vertex $u^*$; we delete self-loops but keep parallel edges. Note that graph $G_k$ is identical to graph $G$, except that some of the edges of $G$ are subdivided in $G_k$. Therefore, a drawing of graph

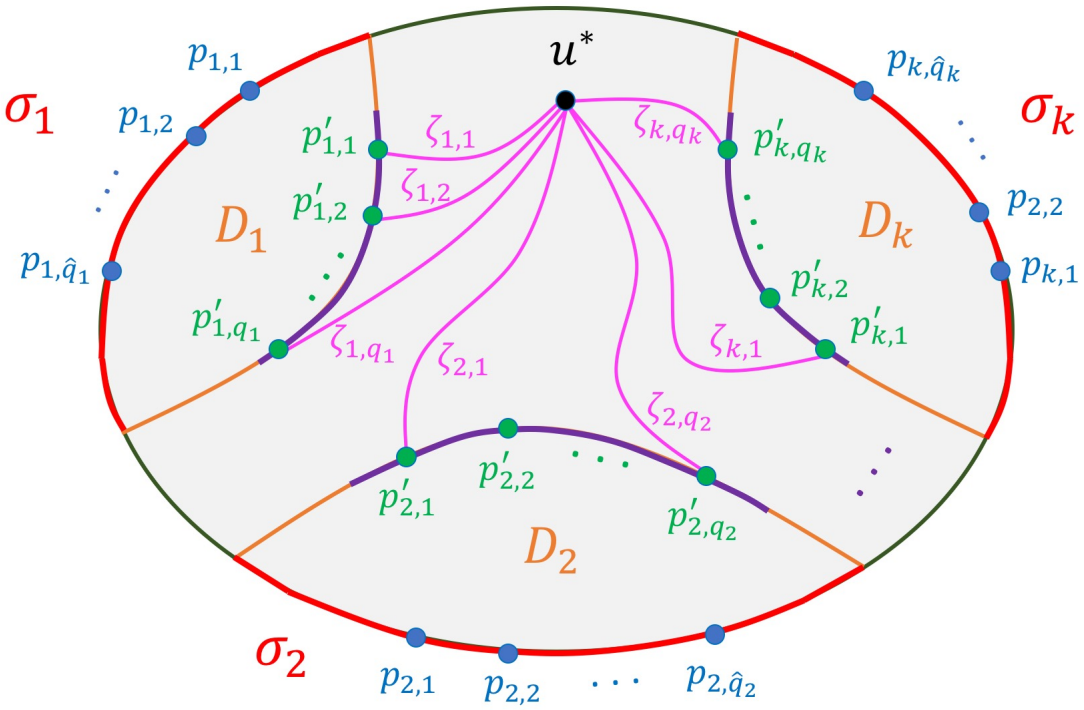



Figure 74: The interior of the disc $D$. Segments $\sigma'_1, \ldots, \sigma'_k$ are shown in purple.

$G_k$ immediately gives a drawing of graph $G$. For each $1 \leq i \leq k$, the input to the $i$th iteration is a drawing $\varphi_{i-1}$ of graph $G_{i-1}$, in which, for all $1 \leq i' \leq i-1$, all vertices and edges of $X_{i'}$ are drawn inside the disc $D'_{i'}$. The goal of the $i$th iteration is to produce a drawing $\varphi_i$ of graph $G_i$, in which, for all $1 \leq i' \leq i$, all vertices and edges of $X_{i'}$ are drawn inside the disc $D'_i$. The final drawing $\varphi_k$ of graph $G_k$, obtained at the end of the last iteration immediately provides a drawing of graph $G$. Let $\varphi_0 = \varphi'$ be the given drawing of graph $G_0$, that is a solution to instance $I'$ of MCNwRS. For all $1 \leq i \leq k$, we denote by $\mathsf{cr}_i$ the total number of crossings in $\varphi_0$, in which edges of $E(X'_i) \cup A'_i \cup \hat{A}_i$ participate. Clearly, $\sum_{i=1}^{k} \mathsf{cr}_i \leq 2\mathsf{cr}(\varphi)$. We will ensure that the following invariants hold, for all $1 \leq i \leq k$:

Inv1. over the course of iteration $i$, we may only change the images of the vertices and edges of $X'_i$, and the images of the edges of $\delta_G(X_i) \cup \delta_{G'}(X'_i)$; the images of the remaining edges and vertices of the graph remain unchanged;

Inv2. for every edge $e \in E(G_{i-1}) \setminus (E(X'_i) \cup A'_i \cup \hat{A}_i)$, the number of crossings in which edge $e$ participates in $\varphi_i$ is bounded by the number of crossings in which edge $e$ participated in $\varphi_{i-1}$; and

Inv3. $\mathsf{cr}(\varphi_i) \leq \mathsf{cr}(\varphi) + O(\mathsf{cr}_i)$.

From the above invariants, it is immediate to see that the final drawing $\varphi_k$ of graph $G_k$ has at most $O(\mathsf{cr}(\varphi))$ crossings.

In order to execute the $i$th iterations, we use the two sets $\mathcal{Q}_i, \mathcal{Q}'_i$ of paths that we have defined, in order to define two sets $\Gamma_i, \Gamma'_i$ of curves, that will serve as "guiding curves" for the transformation of the drawing $\varphi_{i-1}$. We also use the current drawing $\varphi_{i-1}$ in order to compute a "nice" drawing $\psi_i$ of graph $X_i$, together with a partial drawing of edges incident to vertices of $X_i$ in $G$. We then "plant" this drawing inside the disc $D'_i$, and then complete the drawings of the edges of $\delta_G(X_i)$.

The input to the first iteration is the initial drawing $\varphi_0 = \varphi'$ of graph $G_0 = G'$ on the sphere. We now fix a single index $1 \leq i \leq k$, and describe the iteration in which index $i$ is processed. Our starting

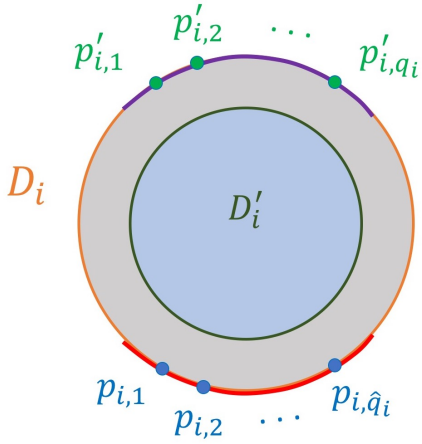



Figure 75: Discs $D_i$ and $D'_i$. Segments $\sigma_i$ and $\sigma'_i$ on the boundary of $D_i$ are shown in red and purple, respectively.

point is a drawing $\varphi_{i-1}$ of graph $G_{i-1}$. Note that, from Invariant Inv2, the total number of crossings in which the edges of $E(X'_i) \cup A'_i \cup \hat{A}_i$ participate in drawing $\varphi_{i-1}$ is at most $\mathsf{cr}_i$. We will use this fact later.

The algorithm for processing index $i$ consists of three stages. In the first stage, we use the set $\mathcal{Q}_i$ of paths in order to define the first set $\Gamma_i$, of "guiding" curves. In the second stage, we use the set $\mathcal{Q}'_i$ of paths in order to define the second set $\Gamma'_i$ of "guiding" curves, and we compute a drawing $\psi_i$ of $X_i$, together with a partial drawing of edges of $\delta_G(X_i)$. In the third and the final stage stage, we "plant" this drawing inside the disc $D'_i$, and complete the drawing $\varphi_i$. We now describe each of the three stages in turn.

### I.2.1 Stage 1: First Set of Guiding Curves, and Partial Drawing of Edges of $\hat{E}_i$

Recall that we have defined a collection $\mathcal{Q}_i = \left\{Q(e) \mid e \in \hat{E}_i\right\}$ of edge-disjoint paths in graph $G$, where for each edge $e \in \hat{E}_i$, path $Q(e)$ has $e$ as its first edge and $u^*$ as its last vertex, and all its inner vertices are contained in $X_i$. From the definition of graph $X'_i$, and from the fact that there are $|\hat{E}_i| = \hat{q}_i$ edges connecting $u_i$ to $u^*$ in $G'$ (the edges of $A'_i$), it is immediate to see that there must be a set $\hat{\mathcal{Q}}_i = \left\{\hat{Q}(\hat{a}) \mid \hat{a} \in \hat{A}_i\right\}$ of edge-disjoint paths in graph $G'_{i-1}$, where for each edge $\hat{a} \in \hat{A}_i$, path $\hat{Q}(\hat{a})$ contains $\hat{a}$ as its first edge and terminates at vertex $u^*$, such that all inner vertices of $\hat{Q}(\hat{a})$ are contained in $X'_i$.

We apply the algorithm from Theorem 4.37 to perform a type-2 uncrossing of the paths in $\hat{\mathcal{Q}}_i$. The input to this algorithm is graph $G_{i-1}$ and its drawing $\varphi_{i-1}$ on the sphere, and the set $\hat{\mathcal{Q}}_i$ of paths, which we view as being directed away from $u^*$. Let $\Gamma_i = \left\{\gamma(\hat{a}) \mid \hat{a} \in \hat{A}_i\right\}$ denote the set of curves that the algorithm outputs, that are aligned with the graph $\bigcup_{Q \in \hat{\mathcal{Q}}_i} Q$. For each edge $\hat{a} \in \hat{A}_i$, if $y(\hat{a})$ is the endpoint of $\hat{a}$ that does not lie in $X'_i$, then curve $\gamma(\hat{a})$ originates at the image of $u^*$ and terminates at the image of $y(\hat{a})$. Moreover, the curves in set $\Gamma_i$ do not cross each other.

Recall that, for every path $\hat{Q} \in \hat{\mathcal{Q}}_i$, the first edge of $\hat{Q}$ (the edge incident to $u^*$) must be an edge of $A'_i$. From the definition of aligned curves, the theorem guarantees that, for every edge $a'_i \in A'_i$, there is a unique curve $\gamma(\hat{a}) \in \Gamma_i$ that contains the segment of the image of $a'_i$ that lies inside disc $D$. In particular, for all $1 \leq j \leq \hat{q}_i$, there is a unique curve $\gamma(\hat{a}) \in \Gamma_i$ containing the point $p_{i,j}$ on the boundary of $D_i$. We denote $\hat{A}_i = \{\hat{a}_{i,1}, \ldots, \hat{a}_{i,\hat{q}_i}\}$, where for all $1 \leq j \leq \hat{q}_i$, edge $\hat{a}_{i,j}$ is the unique edge

whose corresponding curve $\gamma(\hat{a}_{i,j})$ contains the point $p_{i,j}$. From the definition of aligned curves, each such curve $\gamma(\hat{e}_{i,j})$ intersects the boundary of $D$ at a unique point - point $p_{i,j}$. For all $1 \leq j \leq \hat{q}_i$, we denote $\hat{a}_{i,j} = (\hat{x}_{i,j}, \hat{y}_{i,j})$, where $x_{i,j} \in X_i'$. For all $1 \leq j \leq \hat{q}_j$, we let $\hat{\zeta}_{i,j}$ be the segment of curve $\gamma(\hat{a}_{i,j})$ from the image of $y_{i,j}$ to point $p_{i,j}$ on the boundary of disc $D$. We denote the resulting set of curves by $\hat{Z}_i = \left\{\hat{\zeta}_{i,j} \mid 1 \leq j \leq \hat{q}_i\right\}$.

Consider now the drawing $\varphi_{i-1}$ of graph $G_{i-1}$. We slightly modify this drawing, as follows. First, we delete from $\varphi_{i-1}$ the images of all vertices of $X_i'$ and all edges of $E(X_i') \cup \hat{A}_i \cup A_i'$. Next, we add to this drawing the set $Z_i = \{\zeta_{i,j} \mid 1 \leq j \leq q\}$ of curves; recall that for all $1 \leq j \leq q$, curve $\zeta_{i,j}$ is contained in disc $D$, is internally disjoint from disc $D_i$, and connects the image of $u^*$ to point $p'_{i,j}$ on the boundary of disc $D_i$ (see Figure 74). Recall that we called curve $\zeta_{i,j}$ the first segment in the drawing of edge $e_{i,j}$. Additionally, we add to the current drawing the set $\hat{Z}_i = \left\{\hat{\zeta}_{i,j} \mid 1 \leq j \leq \hat{q}_i\right\}$ of curves. Recall that, for all $1 \leq j \leq \hat{q}_i$, curve $\hat{\zeta}_{i,j}$ is internally disjoint from disc $D$, and it connects the image of vertex $y_{i,j}$ (the endpoint of edge $\hat{a}_{i,j} \in \hat{A}_i$ lying outside $X_i'$) to point $p_{i,j}$ on the boundary of disc $D_i$ (see also Figure 75). We refer to curve $\hat{\zeta}_{i,j}$ as *the first segment in the drawing of edge* $\hat{a}_{i,j}$. We denote the resulting drawing by $\varphi'_{i-1}$. Note that, since the curves in $\Gamma_i$ are aligned with the graph $\bigcup_{Q \in \hat{\mathcal{Q}}_i} Q$, and, since the paths in set $\hat{\mathcal{Q}}_i$ are edge-disjoint, the total number of crossings in drawing $\varphi'_{i-1}$ (including crossings between curves representing edges that were not erased and curves in sets $Z_i$ and $\hat{Z}_i$) is bounded by $\mathsf{cr}(\varphi_{i-1})$. Moreover, for every edge $e \in E(G_{i-1}) \setminus (E(X_i') \cup A_i' \cup \hat{A}_i)$, the number of crossings in which $e$ participates in $\varphi'_{i-1}$ is bounded by the number of crossings in which $e$ participates in $\varphi_{i-1}$, and the image of $e$ is disjoint from disc $D_i$. This completes the first stage of the algorithm.

### I.2.2 Stage 2: Second Set of Guiding Curves and Drawing of $X_i$

In this stage we consider again drawing $\varphi_{i-1}$ of graph $G_{i-1}$. We will start by defining another set $\Gamma_i'$ of guiding curves in this graph (which we will eventually use in order to draw a second segment of each edge in $\hat{A}_i$). We then exploit the drawing $\varphi_{i-1}$ and the curves in $\Gamma_i'$ in order to compute a drawing $\psi_i$ of graph $X_i$, and, for each edge $e \in \delta_G(X_i)$, a drawing of a segment of $e$ that is incident to its endpoint that lies in $X_i$. We will also define a new collection $\Gamma_i^*$ of curves, that will be useful for us in Stage 3.

**Set $\Gamma_i'$ of guiding curves.** We start by defining a set $\Gamma_i'$ of guiding curves. Consider the drawing $\varphi_{i-1}$ of $G_{i-1}$.

Recall that petal $X_i$ is routable in graph $G$. Therefore, there is a set $\mathcal{Q}_i'$ of paths routing the edges of $\hat{E}_i$ to vertex $u^*$ in graph $G$, such that the paths in $\mathcal{Q}_i'$ are internally disjoint from $X_i$, and cause congestion at most 3000. Consider any path $Q = Q(\hat{e}) \in \mathcal{Q}_i'$, whose first edge is $\hat{e} \in \hat{E}_i$. Recall that we have subdivided each such edge $\hat{e} \in \hat{E}_i$ with a vertex in graph $G'$. Let $e', e''$ denote the two edges that we obtained from subdividing edge $\hat{e}$, and assume that $e' \in \hat{A}_i$. We then replace edge $\hat{e}$ with $\hat{e}''$ on path $Q$. Assume now that the last edge of $Q$ is $e_{i',j} \in E_{i'}$. Since path $Q$ is internally disjoint from $X_i$, $i' \neq i$ must hold. If $i' > i$, then, by replacing edge $e_{i',j}$ with the corresponding edge $a_{i',j}$, we obtain a new path $Q'$ in graph $G_{i-1}$, whose first vertex is an endpoint of an edge of $\hat{A}_i$, and last vertex is $u_{i'}$. If $i' < i$, then we set $Q' = Q$. Let $\mathcal{Q}_i'' = \{Q' \mid Q \in \mathcal{Q}_i'\}$ be the resulting set of paths in graph $G_{i-1}$.

Consider now any vertex $u_{i'}$, for $i < i' \leq k$. Every path $Q' \in \mathcal{Q}_i''$ that terminates at $u_{i'}$ must contain an edge of $\hat{A}_{i'}$. Therefore, the number of paths in $\mathcal{Q}_i''$ terminating at $u_{i'}$ is bounded by $3000\hat{q}_{i'}$. Since, for all $1 \leq i'' \leq k$, $|A'_{i''}| = \hat{q}_{i''}$, we can then extend all such paths, using the edges of $A'_{i'}$, to ensure that they terminate at vertex $u^*$, such that all such paths cause congestion at most 3000 in $G_{i-1}$. Therefore, we have established that there is a set $\mathcal{Q}_i''' = \left\{Q'''_{i,j} \mid 1 \leq j \leq \hat{q}_i\right\}$ of paths in graph $G_{i-1}$

that cause congestion at most 3000, such that, for all $1 \leq j \leq \hat{q}_i$, path $Q'''_{i,j}$ originates at vertex $\hat{y}_{i,j}$ (the endpoint of the edge $\hat{a}_{i,j} \in \hat{A}_i$ that does not lie in $X'_i$), terminates at vertex $u^*$, and is internally disjoint from $X'_i$.

In order to construct the set $\Gamma'_i$ of curves, we consider a graph $H$, that is obtained as follows. We start with $H = G_{i-1}$. We delete from this graph all edges $e \in E(G_{i-1}) \setminus (E(X'_i) \cup \hat{A}_i \cup A'_i)$ that do not participate in the paths of $\mathcal{Q}'''_i$. For all remaining edges $e \in E(G_{i-1}) \setminus (E(X'_i) \cup \hat{A}_i \cup A'_i)$, we replace $e$ with $\mathrm{cong}_{G_{i-1}}(\mathcal{Q}'''_i, e)$ parallel copies. Lastly, we delete all isolated vertices from the resulting graph. Note that drawing $\varphi_{i-1}$ of graph $G_{i-1}$ naturally defines a drawing $\varphi'$ of graph $H$: after deleting all edges of $E(G_{i-1}) \setminus E(H)$ and all vertices of $V(G_{i-1}) \setminus V(H)$ from the drawing, for every remaining edge $e \in E(H) \setminus (E(X'_i) \cup \hat{A}_i \cup A'_i)$, we draw the parallel copies of $e$ along the original image of edge $e$ in $\varphi_{i-1}$. We can now use the set $\mathcal{Q}'''_i$ paths in graph $G_{i-1}$ in order to define a set $\hat{\mathcal{Q}}'_i = \left\{\hat{Q}'_{i,j} \mid 1 \leq j \leq \hat{q}_i\right\}$ of edge-disjoint paths in graph $H$, where for all $1 \leq j \leq \hat{q}_i$, path $\hat{Q}_{i,j}$ originates at vertex $\hat{y}_{i,j}$, terminates at vertex $u^*$, and is disjoint from $X'_i$.

We use the algorithm from Theorem 4.37 in order to compute a type-2 uncrossing of the paths in $\hat{\mathcal{Q}}'_i$. Specifically, the algorithm is applied to graph $H$, its drawing $\varphi'$, and the set $\hat{\mathcal{Q}}'_i$ of edge-disjoint paths. The algorithm returns a set $\hat{\Gamma} = \{\hat{\gamma}_{i,j} \mid 1 \leq j \leq \hat{q}_i\}$ of internally disjoint curves, where, for all $1 \leq j \leq \hat{q}_i$, curve $\hat{\gamma}_{i,j}$ connects the image of $\hat{y}_j$ to the image of vertex $u^*$ in drawing $\varphi'$, and all curves in $\hat{\Gamma}$ are aligned with the graph $\bigcup_{\hat{Q} \in \hat{\mathcal{Q}}'_i} Q$. We will also consider the curves in set $\hat{\Gamma}$ in the drawing $\varphi_{i-1}$ of $G_{i-1}$. As before, each curve $\hat{\gamma}_{i,j}$ connects the image of $\hat{y}_{i,j}$ to the image of vertex $u^*$ in drawing $\varphi_{i-1}$. Since the paths in the original set $\mathcal{Q}'$ caused congestion at most 3000, it is immediate to verify that the number of crossings between the images of the edges of $E(X'_i) \cup A'_i \cup \hat{A}_i$ in $\varphi_{i-1}$ and the curves of $\hat{\Gamma}$ is at most $3000 \cdot \mathsf{cr}_i$.

Since the curves in $\hat{\Gamma}$ are aligned with the graph $\bigcup_{\hat{Q} \in \hat{\mathcal{Q}}'_i} Q$, each curve $\hat{\gamma}_{i,j} \in \hat{\Gamma}$ intersects the boundary of the tiny $u^*$-disc $D$ in a single point, that we denote by $z_j$. Sine the paths in $\hat{\mathcal{Q}}'$ may not use the edges of $A'_i$, we are guaranteed that each such point $z_j$ may not lie on the segment $\sigma_i$ (the segment containing the points $p_{i,1}, \ldots, p_{i,\hat{q}_i}$; point $p_{i,j'}$ is the intersection point of the image of edge $a'_{i,j'}$ and the boundary of $D$, see Figure 74). For convenience, we re-index the points in set $\{z_j\}_{1 \leq j \leq \hat{q}_i}$, so that points $z_1, z_2, \ldots, z_{\hat{q}_i}, p_{i,\hat{q}_i}, \ldots, p_{i,1}$ appear on the boundary of $D$ in this order. For each $1 \leq j \leq \hat{q}_i$, we denote by $\ell(j)$ the unique index such that curve $\hat{\gamma}_{i,j}$ contains the point $z_{\ell(j)}$. For all $1 \leq j \leq \hat{q}_i$, we let $\gamma'_{i,j}$ be the segment of curve $\hat{\gamma}_{i,j}$ from the image of vertex $\hat{y}_{i,j}$ to point $z_{\ell(j)}$. We then set $\Gamma'_i = \left\{\gamma'_{i,j} \mid 1 \leq j \leq \hat{q}_i\right\}$. From the above discussion, the total number of crossings between the images of the edges of $E(X'_i) \cup A'_i \cup \hat{A}_i$ in $\varphi_{i-1}$ and the curves of $\Gamma'_i$ is at most $3000 \cdot \mathsf{cr}_i$.

**Set $\Gamma^*_i$ of Auxiliary curves.** We need to define another set of curves, that we will use in Stage 3. Recall that we have defined a set of points $z_1, z_2, \ldots, z_{\hat{q}_i}, p_{i,\hat{q}_i}, \ldots, p_{i,1}$ that appear on the boundary of disc $D$ in this order. We can consider two orderings of elements of $\{1, \ldots, \hat{q}_i\}$: the first ordering is their natural ordering, while the second ordering is $\ell(1), \ell(2), \ldots, \ell(\hat{q}_i)$ – ordering that is defined by the curves in $\Gamma'_i$. In Stage 3 of our algorithm, we will need to show that the distance between these two orderings is small, in order to combine different segments of the drawings of the edges of $A_i$ to complete their drawing. The set $\Gamma^*_i$ of curves, that we define in the next observation, will be used in order to do so.

**Observation I.2** *There is an efficient algorithm to construct a collection $\Gamma^*_i = \left\{\gamma^*_{i,j} \mid 1 \leq j \leq \hat{q}_i\right\}$ of curves, such that, for all $1 \leq j \leq \hat{q}_i$, curve $\gamma^*_{i,j}$ connects point $p_{i,j}$ on the boundary of disc $D$ to point $z_{\ell(j)}$, and it is internally disjoint from disc $D$. Moreover, the total number of crossings between the curves of $\Gamma^*_i$ is $O(\mathsf{cr}_i)$.*

**Proof:** Consider an index $1 \leq j \leq \hat{q}_i$. Recall that we have defined a curve $\hat{\zeta}_{i,j}$, which is a sub-curve of some curve of $\Gamma_i$, connecting point $p_{i,j}$ to the image of vertex $\hat{y}_{i,j}$ in $\varphi_{i-1}$. We concatenate this curve with curve $\gamma'_{i,j} \in \Gamma'_i$, connecting the image of vertex $\hat{y}_{i,j}$ to point $z_{\ell(j)}$, obtaining the curve $\gamma^*_{i,j}$, that connects $p_{i,j}$ to $z_{\ell(j)}$. From the construction of curves in $\Gamma_i$ and $\Gamma'_i$, and the alignemnt properties of each such curve, we are guaranteed that each resulting curve is internally disjoint from disc $D$.

In order to bound the number of crossings between the curves of $\Gamma^*$, recall that the total number of crossings between the images of the edges of $E(X'_i) \cup A'_i \cup \hat{A}_i$ in $\varphi_{i-1}$ and the curves of $\Gamma'_i$ is at most $3000\mathsf{cr}_i$. Since the curves of $\Gamma_i$ are aligned with graph $\bigcup_{Q \in \hat{\mathcal{Q}}_i} Q$, and the paths of $\mathcal{Q}_i$ are edge-disjoint and contained in $X'_i \cup A'_i \cup \hat{A}_i$, we get that the total number of crossings between the curves of $\Gamma^*_i$ is $O(\mathsf{cr}_i)$. □

**Partial drawing of edges of $\hat{A}_i$.** Next, we define a collection $\hat{Z}'_i = \left\{\hat{\zeta}'_{i,j} \mid 1 \leq j \leq \hat{q}_i\right\}$, of curves, that we will use in order to obtain partial drawing of the edges of $\hat{A}_i$. For all $1 \leq j \leq \hat{q}_i$, curve $\hat{\zeta}'_{i,j}$ will connect the image of vertex $\hat{x}_{i,j}$ (the endpoint of edge $\hat{a}_{i,j}$ lying in $X'_i$) to a point on the boundary of the tiny $u_i$-disc in $\varphi_{i-1}$. We will then view curve $\hat{\zeta}'_{i,j}$ as the *second segment in the drawing of edge* $\hat{a}_{i,j}$, and we will add all such curves to the drawing $\psi_i$ that we compute in this stage.

In order to compute the set $\hat{Z}'_i$ of curves, we will utilize the cuves of $\Gamma'_i$, the images of the edges of $\hat{A}_i \cup A'_i$ in drawing $\varphi_{i-1}$, and another set of curves that we define next.

Recall that we have defined a set $z_1, z_2, \ldots, z_{\hat{q}_i}, p_{i,\hat{q}_i}, \ldots, p_{i,1}$ of points on the boundary of disc $D$, that appear on the boundary of $D$ in this order. We define, for all $1 \leq j \leq \hat{q}_i$ a curve $\rho_j$, connecting $z_j$ to $p_{i,j}$, such that all curves in $\{\rho_1, \ldots, \rho_{\hat{q}_i}\}$ are contained in disc $D$ and are disjoint from each other (see Figure 76).

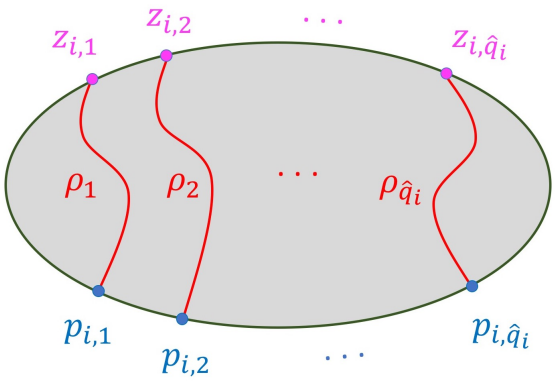



Figure 76: An illustration of curves $\rho_1, \ldots, \rho_{\hat{q}_i}$ in disc $D$.

We denote by $\tilde{D}_i = D_{\varphi_{i-1}}(u_i)$ the tiny $u_i$-disc in $\varphi_{i-1}$. Consider the images of the edges in set $A'_i = \left\{a'_{i,j} \mid 1 \leq j \leq \hat{q}_i\right\}$ in drawing $\varphi_{i-1}$ (these are the parallel edges connecting $u^*$ to $u_i$). From our definition, for all $1 \leq j \leq \hat{q}_i$, point $p_{i,j}$ is the unique point on the boundary of disc $D$ that lies on the image of edge $a'_{i,j}$. We denote the unique point of the image of $a'_{i,j}$ lying on the boundary of disc $\tilde{D}_i$ by $z'_j$. Recall that, from our assumptions, points $p_{i,1}, \ldots, p_{i,\hat{q}_i}, z_{\hat{q}_i}, \ldots, z_1$ appear in this order on the boundary of $D$, as we traverse it so that the interior of the disc lies to our left (see Figure 77). If $u_i$ is synchronized with $u^*$, then, from the definition of the rotation $\mathcal{O}'_{u_i} \in \Sigma'$, points $z'_1, \ldots, z'_{\hat{q}_i}$ appear in this order on the boundary of disc $\tilde{D}_i$, as we traverse it so that the interior of the disc lies to our right; if $u_i$ is not synchronized with $u^*$, then this order is reversed (see Figure 77).

We are now ready to define the curves of $\hat{Z}'_i$. Consider some index $1 \leq j \leq \hat{q}_i$. Curve $\hat{\zeta}'_{i,j}$ is the

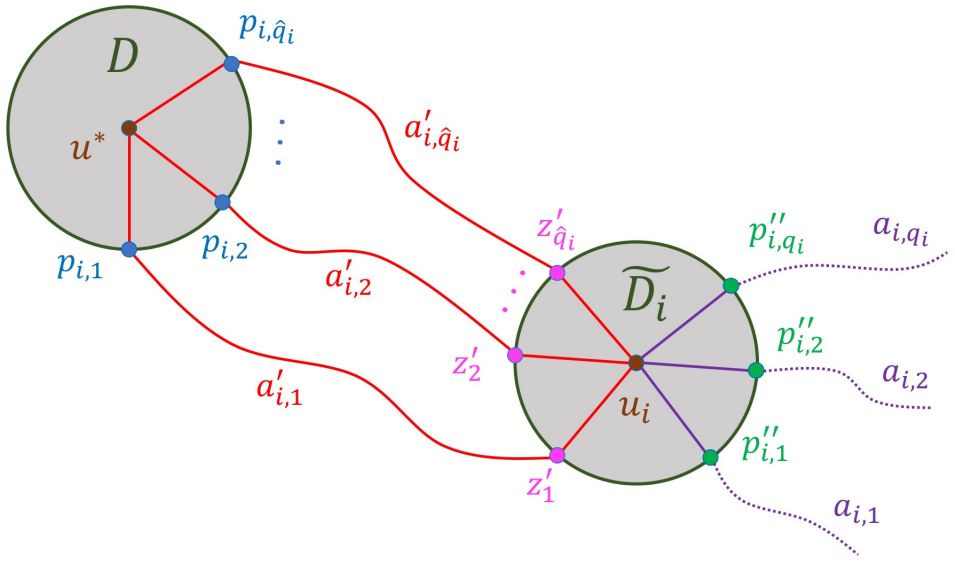



Figure 77: Schematic view of images of edges of $A'_i$ and $A_i$ and relevant points on the boundary of $D$ and $\tilde{D}_i$. Images of edges of $A_i$ are shown in red. Image of each edge $a_{i,j} \in A_i$ is shown in purple, with segments $\zeta_{i,j}$ dotted.

concatenation of four curves: (i) the image of the edge $\hat{a}_i = (\hat{x}_{i,j}, \hat{y}_{i,j})$ in $\varphi_{i-1}$; (ii) curve $\gamma'_{i,j} \in \Gamma'_i$ (that connects the image of $\hat{y}_{i,j}$ to point $z_\ell(j)$ on the boundary of $D$); (iii) curve $\rho_j$ (that connects $z_\ell(j)$ to $p_{i,j}$ and is contained in $D$); and (iv) a segment of the image of the edge $a'_{i,j}$ in $\varphi_{i-1}$, from point $p_{i,j}$ on the boundary of disc $D$, to point $z'_{\ell'(j)}$ on the boundary of disc $\tilde{D}_i$, where $1 \le \ell'(j) \le \hat{q}_i$. Note that, if $u_i$ is synchronized with $u^*$, then $\ell'(j) = \ell(j)$ must hold, while otherwise $\ell'(j) = \hat{q}_i - \ell(j) + 1$. We refer to the resulting curve $\hat{\zeta}'_{i,j}$ as the *second segment in the drawing of edge* $\hat{a}_{i,j}$, and we denote $\hat{Z}'_i = \left\{\hat{\zeta}'_{i,j} \mid 1 \le j \le \hat{q}_i\right\}$.

**Computing the drawing $\psi_i$.** Consider the current drawing $\varphi_{i-1}$ of graph $G_{i-1}$. We slightly modify this drawing in order to obtain a drawing $\psi_i$ of graph $X_i$, and, for each edge $e \in \delta_{G_i}(X_i) \setminus \delta_{G_i}(u^*)$, a drawing of a segment of $e$ that is incident to its endpoint that lies in $X_i$.

In order to do so, we start with the drawing $\varphi_{i-1}$ of graph $G_{i-1}$, and we delete from it the images of all edges and vertices, except for those lying in $X'_i$. We will also make use of the disc $\tilde{D}_{i-1}$ that we have defined. Next, for all $1 \le j \le q_i$, we consider the image of edge $a_{i,j}$ in the current drawing. Recall that this image intersects the boundary of $\tilde{D}_i$ at a single point, that we denote by $p''_{i,j}$. From the definition of the rotation $\mathcal{O}'_{u_i} \in \Sigma'$, points $z'_1, \ldots, z'_{\hat{q}_i}, p''_{i,q_i}, \ldots, p''_{i,2}, p''_{i,1}$ appear on the boundary of disc $\tilde{D}_i$ in this order, and, if $u_i$ is synchronized with $u^*$, then they are encountered in this order as we traverse the boundary of $\tilde{D}_i$ so its interior lies on our right; see Figure 77.

Consider now some edge $a_{i,j} \in A_i$, for $1 \le j \le q_i$, and assume that $a_{i,j} = (u_i, x_{i,j})$, where $x_{i,j} \in V(X_i)$. We denote by $\zeta'_{i,j}$ the segment of the image of edge $a_{i,j}$ from the image of $x_{i,j}$ to point $p''_{i,j}$ (see Figure 77), and we refer to $\zeta'_{i,j}$ *the second segment in the drawing of edge* $e_{i,j}$. We delete, from the current drawing, the portion of the image of $a_{i,j}$ lying in the interior of the disc $\tilde{D}_i$; in other words, we replace the image of $a_{i,j}$ with the curve $\zeta'_{i,j}$.

Lastly, we add the curves in set $\hat{Z}'_i = \left\{\hat{\zeta}'_{i,j} \mid 1 \le j \le \hat{q}_i\right\}$ to the current drawing. Recall that, for each $1 \le j \le \hat{q}$, curve $\hat{\zeta}'_{i,j}$ connects the image of vertex $\hat{x}_{i,j}$ (the endpoint of edge $\hat{a}_{i,j} \in \hat{E}_i$ that lies in $X_i$)

to point $z'_{\ell'(j)}$ on the boundary of $\tilde{D}_i$, where $1 \leq \ell'(j) \leq \hat{q}_i$.

This completes the drawing $\psi_i$. We now bound the number of crossings in this drawing. In order to bound the number of crossings, recall that every curve $\zeta'_{i,j} \in Z'_{i,j}$ is a segment of the image of edge $a_{i,j} \in A_{i,j}$, and every curve $\hat{\zeta}'_{i,j} \in \hat{Z}'_i$ is a concatenation of four curves: the image of the edge $\hat{a}_{i,j} \in \hat{A}_i$; the curve $\gamma'_{i,j} \in \Gamma'_i$; the curve $\rho_j$, and the image of the edge $a'_{i,\ell(j)} \in A'_i$. Since the curves in $\Gamma'_i$ are disjoint from each other, and the curves $\rho_1, \ldots, \rho_{\hat{q}_i}$ are disjoint from each other and are contained in disc $D$, the total number of crossings in $\psi_i$ is bounded by (i) the number of crossings between pairs of edges in $E(X'_i) \cup \hat{A}_i \cup A'_i$ (which is bounded by $\mathsf{cr}_i$ by definition); and (ii) the number of crossings between edges of $E(X'_i) \cup \hat{A}_i \cup A'_i$ and curves of $\Gamma'_i$ (which is bounded by $O(\mathsf{cr}_i)$ from the discussion above. We conclude that drawing $\psi_i$ has $O(\mathsf{cr}_i)$ crossings.

We will view the interior of the disc $\tilde{D}_i$ as the "outer face" of the drawing $\psi_i$. If we denote by $\tilde{D}'_i$ the disc in the sphere whose boundary is the same as the boundary of $\tilde{D}_i$, but its interior is disjoint from the interior of $\tilde{D}_i$, then the current drawing $\psi_i$ is contained in $\tilde{D}'_i$. To summarize, drawing $\psi_i$ consists of: (i) the drawing of all edges and vertices of $X_i \setminus \{u^*\}$; (ii) for every edge $a_{i,j} = (x_{i,j}, u^*) \in E_i$, a curve $\zeta'_{i,j}$, connecting the image of $x_{i,j}$ to point $p''_{i,j}$ on the boundary of $\tilde{D}'_i$; and (iii) for every edge $\hat{a}_{i,j} = (\hat{x}_{i,j}, \hat{y}_{i,j}) \in \hat{A}_i$ with $\hat{x}_{i,j} \in X_i$, a curve $\hat{\zeta}'_{i,j}$, connecting the image of $\hat{x}_{i,j}$ to point $z'_{\ell'(j)}$ on the boundary of $\tilde{D}'_i$, where $1 \leq \ell'(j) \leq \hat{q}_i$. Note that the points $p''_{i,1}, \ldots, p''_{i,q_i}, z'_{\hat{q}_i}, \ldots, z'_2, z'_1$ appear on the bounary of disc $\tilde{D}'_i$ in this order, and, if $u_i$ is synchronized with $u^*$, then they are encountered in this order as we traverse the boundary of $\tilde{D}'_i$ so its interior lies to our right.

### I.2.3 Stage 3: Computing the Drawing $\varphi_i$ of $G_i$

We start with the drawing $\varphi'_{i-1}$ that we computed in Stage 1. We consider two cases. The first case is when vertex $u_i$ is synchronized with vertex $u^*$. In this case, we place the drawing $\psi_i$ that we computed in Stage 2 of the algorithm inside the disc $D'_i$, so that the discs $D'_i$ and $\tilde{D}'_i$ coincide. In the second case, vertex $u_i$ is not synchronized with vertex $u^*$. In this case, we place a mirror image of the drawing $\varphi_i$ inside the disc $D'_i$, so that the boundaries of the discs $D'_i$ and (the mirror image of) disc $\tilde{D}'_i$ coincide (see Figure 78). In either case, we obtain two disjoint segments on the boundary of $D'_i$: segment $\tilde{\sigma}_i$, containing the points $p''_{i,1}, \ldots, p''_{i,q_i}$, and segment $\tilde{\sigma}'_i$, containing the points $z'_1, \ldots, z'_{\hat{q}_i}$. Moreover, we are now guaranteed that points $p''_{i,1}, \ldots, p''_{i,q_i}, z'_{\hat{q}_i}, \ldots, z'_2, z'_1$ are encountered in this order as we traverse the boundary of $D'_i$, so that the interior of $D'_i$ lies to our right (see Figure 78). Recall that we have defined a collection of points $\left\{p'_{i,1}, p'_{i,2}, \ldots, p'_{i,q_i}, p_{i,\hat{q}_i}, p_{i,\hat{q}_i-1}, \ldots, p_{i,1}\right\}$ on the boundary of $D_i$, that appear in this order on the boundary of $D_i$, as we traverse it so that the interior of $D_i$ lies to our right (see Figures 74 and 77). Consider now some edge $e_{i,j} = (u^*, x_{i,j}) \in E_i$, for $1 \leq i \leq q_i$. Recall that we have already defined a curve $\zeta_{i,j}$, that serves as the first segment of the drawing of $e_{i,j}$, and connects the image of $u^*$ to point $p'_{i,j}$ on the boundary of $D_i$, such that curve $\zeta_{i,j}$ is internally disjoint from $D_i$. We have also defined a curve $\zeta'_{i,j}$ inside the disc $D'_i$, that connects the image of vertex $x_{i,j}$ to point $p''_{i,j}$ on the boundary of $D'_{i,j}$.

Recall that we have also defined another segment $\sigma'_i$ on the boundary of $D_i$, that contains points $p'_{i,1}, \ldots, p'_{i,q_i}$, and a segment $\hat{\sigma}'_i$ on the boundary of $D'_i$, containing the points in $z'_1, \ldots, z'_{\hat{q}_i}$. We can then define two disjoint discs that are both contained in $D_i \setminus D'_i$: one disc, $D^1_i$, with segments $\sigma_i$ and $\tilde{\sigma}_i$ on its boundary, and another disc, $D^2_i$, with segments $\tilde{\sigma}'_i, \sigma'_i$ on its boundary (see Figure 79).

Observe that points $\left\{p'_{i,1}, p'_{i,2}, \ldots, p'_{i,q_i}, p''_{i,q_i}, \ldots, p''_{i,1}\right\}$ appear in this circular order on the boundary of disc $D^1_i$. Therefore, we can define, for all $1 \leq j \leq q_i$, a curve $\zeta''_{i,j}$, connecting point $p'_{i,j}$ to point $p''_{i,j}$, such that the interior of the curve is contained in disc $D^1_i$, and all curves in $\left\{\zeta''_{i,j} \mid 1 \leq j \leq q_i\right\}$ are mutually disjoint (see Figure 80(a)). For all $1 \leq j \leq q_i$, we then let the image of the edge $e_{i,j}$ be

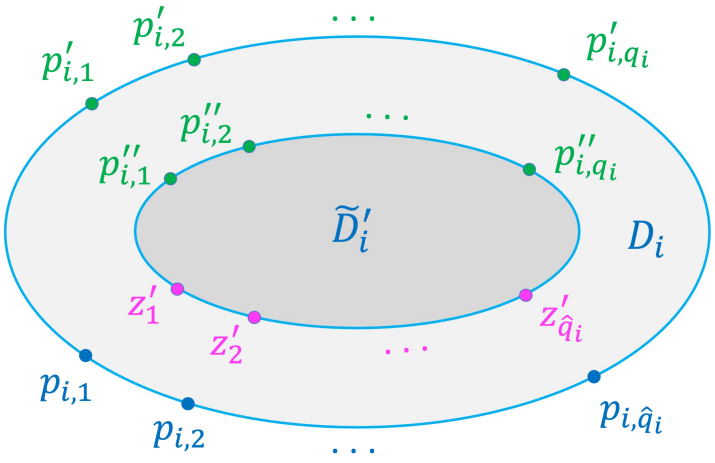



Figure 78: Planting disc $\tilde{D}'_i$ inside $D'_i$.

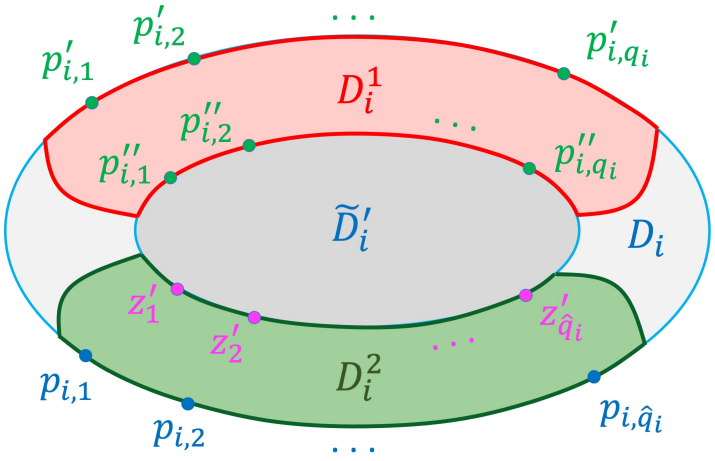



Figure 79: An illustration of discs $D_i^1$ and $D_i^2$.

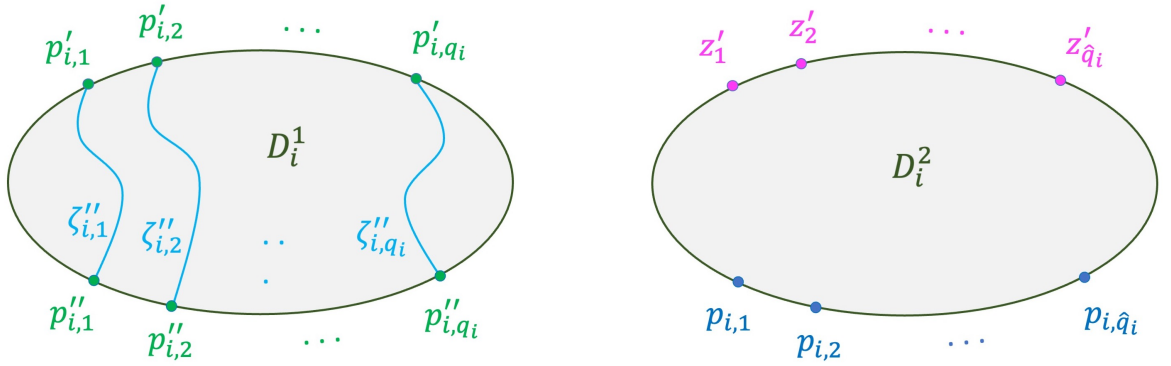
the concatenation of the curves $\zeta_{i,j}$, $\zeta''_{i,j}$, and $\zeta'_{i,j}$.

Lastly, it remains to complete the drawing of the edges of $\hat{A}_i$. Recall that for every edge $\hat{a}_{i,j} = (\hat{x}_{i,j}, \hat{y}_{i,j})$ (where $\hat{x}_{i,j} \in X_i$), we have already defined two segments of the drawing of $\hat{a}_{i,j}$. The first segment, $\hat{\zeta}_{i,j}$, is internally disjoint from disc $D$, and connects the image of $\hat{y}_{i,j}$ to point $p_{i,j}$ on the boundary of $D_i$. The second segment, $\hat{\zeta}'_{i,j}$, is contained in disc $D'_i$, and connect the image of $\hat{x}_{i,j}$ to point $z'_{\ell'(j)}$ on the boundary of disc $D'_i$. In order to complete the drawing of edge $\hat{a}_{i,j}$, we will define a third curve, $\hat{\zeta}''_{i,j}$, that is contained in disc $D_i^2$, and connects points $p_{i,j}$ and $z'_{\ell'(j)}$ to each other. See Figure 80(b) for an illustration.

We will use the following observation in order to complete the drawing.

**Observation I.3** *There is an efficient algorithm to compute a collection* $\left\{\hat{\zeta}''_{i,j} \mid 1 \le j \le \hat{q}_i\right\}$ *of curves that are contained in disc* $D_i^2$*, such that, for all* $1 \le j \le \hat{q}_i$*, curve* $\zeta''_{i,j}$ *connects points* $p_{i,j}$ *and* $z'_j$*, and the number of crossings between the curves in* $\left\{\hat{\zeta}''_{i,j} \mid 1 \le j \le \hat{q}_i\right\}$ *is at most* $O(\mathsf{cr}_i)$.

**Proof:** We consider two cases. The first case is when $u^*$ and $u_i$ are not synchonized. In this case, we




(a) An illustration of curves $\zeta_{i,1}, \ldots, \zeta_{i,q_i}$.

(b) An illustration of disc $D_i^2$ and points on its boundary.

Figure 80: Stitching the images of the edgs of $E_i$ and $\hat{E}_i$.

let $\left\{\hat{\zeta}''_{i,j} \mid 1 \leq j \leq \hat{q}_i\right\}$ of curves that are contained in disc $D_i^2$, such that, for all $1 \leq j \leq \hat{q}_i$, curve $\zeta''_{i,j}$ connects points $p_{i,j}$ and $z'_j$, and every pair of curves cross at most once. In this case, the number of crossings between the curves in $\left\{\hat{\zeta}''_{i,j} \mid 1 \leq j \leq \hat{q}_i\right\}$ is at most $\hat{q}_i^2$. Since, from Observation I.1, there were at least $\hat{q}_i^2/8$ crossings $(e, e')$ in $\varphi'$ with $e, e' \in A'_i$, we get that $\mathsf{cr}_i \geq \Omega(\hat{q}_i^2)$, and so the the number of crossings between the curves in $\left\{\hat{\zeta}''_{i,j} \mid 1 \leq j \leq \hat{q}_i\right\}$ is at most $O(\mathsf{cr}_i)$ as required.

In the second case, $u^*$ and $u_i$ are synchronized. Recall that in Stage 2 of the algorithm, in Observation I.2, we have constructed a collection $\Gamma_i^* = \left\{\gamma^*_{i,j} \mid 1 \leq j \leq \hat{q}_i\right\}$ of curves, such that, for all $1 \leq j \leq \hat{q}_i$, curve $\gamma^*_{i,j}$ connects point $p_{i,j}$ on the boundary of disc $D$ to point $z_{\ell(j)}$, and it is internally disjoint from disc $D$; the total number of crossings between the curves of $\Gamma_i^*$ is $O(\mathsf{cr}_i)$. Recall that points $z_1, z_2, \ldots, z_{\hat{q}_i}, p_{i,\hat{q}_i}, \ldots, p_{i,1}$ that appear on the boundary of disc $D$ in this order, while points $z'_1, z'_2, \ldots, z'_{\hat{q}_i}, p_{i,\hat{q}_i}, \ldots, p_{i,1}$ appear on the boundary of disc $D_i^2$ in this order. The key point is that, as observed in Stage 2 of the algorithm, if vertex $u_i$ is synchronized with vertex $u^*$, then for all $1 \leq j \leq \hat{q}_i$, $\ell(j) = \ell(j')$. Therefore, we can copy the collection $\Gamma_i^*$ of curves to the interior of the disc $D_i^2$, such that, for all $1 \leq j \leq \hat{q}_i$, one of the resulting curves, that we denote by $\hat{\zeta}''_{i,j}$ connects $p_{i,j}$ to $z'_{\ell'(j)} = z'_{\ell(j)}$.

□

### I.2.4 Analysis

We now show that, assuming that Invariants Inv1–Inv3 hold at the beginning of the $i$th iteration, they continue to hold at the end of the iteration. Indeed, it is immediate to see that we only change the images of vertices and edges of $X'_i$, and $A'_i \cup \hat{A}_i$, which establishes Invariant Inv1. Consider now some edge $e \in E(G_{i-1}) \setminus (E(X'_i) \cup A'_i \cup \hat{A}_i)$, and some other edge $e'$ that crosses $e$ in $\varphi_i$. If $e'$ is not an edge of $E(X_i) \cup \hat{A}_i$, then the image of $e'$ was not changed in the current iteration, and the crossing lies in $\varphi_{i-1}$ as well. Notice that $e$ may not cross edges of $E(X_i)$, as for each such edge $e'$, either $e'$ is drawn inside disc $D'_i$, or $e' \in E_i$, so the first segemnt of $e'$ is some curve $\zeta_{i,j}$ (that is contained in $D$), and the remainder of the image of $e'$ is contained in $D'_i$. Assume now that $e' \in \hat{A}_i$. In this case, only the first segment of the drawing $e'$, which is a segment of some curve in $\Gamma_i$, may cross edge $e$, as the remainder of the image of $e'$ lies in disc $D$. Overall, the number of crossings in which edge $e$ participates in drawing $\varphi_i$ is bounded by the number of crossings in which edge $e$ participates in drawing $\varphi'_{i-1}$ (that was defined in Stage 1), which is in turn bounded by the number of crossings in which $e$ participates in $\varphi_{i-1}$. We conclude that Invariant Inv2 continues to hold. Lastly, it remains to

establish Invariant Inv3. From the above discussion, for each edge $e \in E(G_{i-1}) \setminus (E(X_i') \cup A_i' \cup \hat{A}_i)$, the number of crossings in which $e$ participates in drawing $\varphi_i$ is bounded by the number of crossings in which $e$ participates in drawing $\varphi_{i-1}$. From our analysis of Stage 2, the number of crossings in $\psi_i$ is bounded by $O(\mathsf{cr}_i)$. Recall that the curves of $\Gamma_i$ cannot cross each other, and they are internally disjoint from disc $D$. The only additional crossings that we introduced are the crossings between the curves of $\left\{\hat{\zeta}''_{i,j} \mid 1 \leq j \leq \hat{q}_i\right\}$ that were computed in Observation I.3; from the observation, the number of such crossings is at most $O(\mathsf{cr}_i)$. This establishes Invariant Inv3.

Overall, after $k$ iterations, we obtain a drawing $\varphi_k$ of graph $G_k$ with $O(\mathsf{cr}(\varphi') + \sum_{i=1}^{k} \mathsf{cr}_i) \leq O(\mathsf{cr}(\varphi'))$ crossings. Since graph $G_k$ can be obtained from graph $G$ by subdividing some of its edges, this immediately provides a drawing of graph $G$ with $O(\mathsf{cr}(\varphi'))$ crossings. From our construction it is immediate to verify that the resulting drawing obeys the rotation system $\Sigma$, so we obtain a feasible solution to instance $I$ of MCNwRS.

# J Proofs Omitted from Section 11

## J.1 Proof of Lemma 11.1

Denote $z = \left\lceil \frac{|T|}{|T'|} \right\rceil$. We arbitrarily partition the vertices of $T \setminus T'$ into $(z-1)$ subsets $T_1, \ldots, T_{z-1}$ of cardinality at most $|T'|$ each. Consider some index $1 \leq i \leq z-1$. Since vertices of $T$ are $\alpha$-well-linked in $G$, using the algorithm from Theorem 4.17, we can compute a collection $\mathcal{Q}_i'$ of paths in graph $G$, routing vertices of $T_i$ to vertices of $T'$, such that $\mathrm{cong}_G(\mathcal{Q}_i') \leq \lceil 1/\alpha \rceil$, and each vertex of $T' \cup T_i$ is the endpoint of at most one path in $\mathcal{Q}_i'$. By concatenating the paths in $\mathcal{Q}_i'$ with paths in $\mathcal{P}$, we obtain a collection $\mathcal{Q}_i$ of paths in graph $G$ routing vertices of $T_i$ to vertex $x$. For every edge $e \in E(G)$, $\mathrm{cong}_G(\mathcal{Q}_i, e) \leq \lceil 1/\alpha \rceil + \mathrm{cong}_G(\mathcal{P}, e)$. Lastly, we set $\mathcal{P}' = \bigcup_{i=1}^{z-1} \mathcal{Q}_i$. It is clear that the paths in $\mathcal{P}$ route the vertices of $T$ to $x$. Moreover, for every edge $e \in E(G)$,

$$\mathrm{cong}_G(\mathcal{P}', e) \leq \sum_{i=1}^{z} \mathrm{cong}_G(\mathcal{Q}_i, e) \leq \left\lceil \frac{|T|}{|T'|} \right\rceil \left( \mathrm{cong}_G(\mathcal{P}, e) + \lceil 1/\alpha \rceil \right).$$

## J.2 Proof of Lemma 11.5

We denote $\tilde{\alpha}' = \frac{\tilde{\alpha}}{c \log^2 m}$, where $c$ is a large enough constant whose value we set later. Throughout the algorithm, we will maintain a collection $\mathcal{W}$ of disjoint clusters of $G \setminus T$. We will ensure that this collection $\mathcal{W}$ of clusters has some useful properties, that are summarized in the following definition.

**Definition J.1** *A collection $\mathcal{W}$ of disjoint clusters of $G \setminus T$ is a* legal clustering of $G$ *if the following hold:*

- $\bigcup_{W \in \mathcal{W}} V(W) = V(G) \setminus T$;
- *every cluster $W \in \mathcal{W}$ has the $\tilde{\alpha}'$-bandwidth property; and*
- *for every cluster $W \in \mathcal{W}$, $|\delta_G(W)| \leq \tilde{\alpha} k/64$.*

Given a legal clustering $\mathcal{W}$ of $G$, we associate with it a contracted graph $\hat{G} = G_{|\mathcal{W}}$; recall that $\hat{G}$ is obtained from graph $G$ by contracting every cluster $W \in \mathcal{W}$ into a supernode $v_W$; we keep parallel edges but delete self loops. Observe that from the definition of a legal clustering, the only regular (non-supernode) vertices of $\hat{G}$ are the vertices of $T$. Given a legal clustering $\mathcal{W}$ of $G$, we denote by

$\hat{E}^{\mathsf{out}}(\mathcal{W})$ the set of all edges $(u,v)$ of $G$, where $u$ and $v$ belong to different clusters of $\mathcal{W}$; equivalently, $\hat{E}^{\mathsf{out}}(\mathcal{W}) = E(\hat{G} \setminus T)$. We will not distinguish between the edges of $\hat{E}^{\mathsf{out}}(\mathcal{W})$ and the edges of $\hat{G} \setminus T$.

We need the following simple claim, whose analogues were proved in [Chu12, CL12, CC16, Chu16].

**Claim J.2** *Let $\mathcal{W}$ be a legal clustering of $G$. If the set $T$ of terminals is $\tilde{\alpha}$-well-linked in $G$, then $|\hat{E}^{\mathsf{out}}(\mathcal{W})| \geq \tilde{\alpha}k/4$.*

**Proof:** For every cluster $W \in \mathcal{W}$, let $T_W \subseteq T$ be the set containing every terminal $t \in T$, such that, if $e = (t,v)$ is the unique edge incident to $t$ in $G$, then $v \in W$. Denote $n_W = |T_W|$. Since we are guaranteed that for every cluster $W \in \mathcal{W}$, $|\delta_G(W)| \leq \tilde{\alpha}k/64$, $n_W \leq \tilde{\alpha}k/64$ must hold. Note that there is a partition $\mathcal{W}_1, \mathcal{W}_2$ of $\mathcal{W}$, such that $\sum_{W \in \mathcal{W}_1} n_W, \sum_{W \in \mathcal{W}_2} n_W \geq k/4$. Indeed, we can compute such a partition using a simple greedy algorithm: start with $\mathcal{W}_1, \mathcal{W}_2 = \emptyset$, and process the clusters $W \in \mathcal{W}$ one by one. When cluster $W \in \mathcal{W}$ is processed, we add it to $\mathcal{W}_1$ if $\sum_{W \in \mathcal{W}_1} n_W < \sum_{W \in \mathcal{W}_2} n_W$, and we add it to $\mathcal{W}_2$ otherwise. We are guaranteed that at the end of this procedure, $\left|\sum_{W \in \mathcal{W}_1} n_W - \sum_{W \in \mathcal{W}_2} n_W\right| \leq \max_{W \in \mathcal{W}} \{n_W\} \leq \tilde{\alpha}k/64$ holds. It is then immediate to verify that $\sum_{W \in \mathcal{W}_1} n_W, \sum_{W \in \mathcal{W}_2} n_W \geq k/4$.

We construct a partition $(X,Y)$ of $V(G)$, where $X = \bigcup_{W \in \mathcal{W}_1} (V(W) \cup T_W)$, and $Y = \bigcup_{W \in \mathcal{W}_2} (V(W) \cup T_W)$. Then $|X \cap T|, |Y \cap T| \geq k/4$, and, since we have assumed that the set $T$ of terminals is $\tilde{\alpha}$-well-linked in $G$, $|E_G(X,Y)| \geq \tilde{\alpha}k/4$. Notice that every edge in $E_G(X,Y)$ connects a pair of vertices lying in different clusters of $\mathcal{W}$, so $|\hat{E}^{\mathsf{out}}(\mathcal{W})| \geq |E_G(X,Y)| \geq \tilde{\alpha}k/4$. □

The following lemma is key to the proof of Lemma 11.5.

**Lemma J.3** *There is an efficient algorithm, that, given a legal clustering $\mathcal{W}$ of $G$ with $|\hat{E}^{\mathsf{out}}(\mathcal{W})| \geq \tilde{\alpha}k/4$, either produces another legal clustering $\mathcal{W}'$ of $G$ with $|\hat{E}^{\mathsf{out}}(\mathcal{W}')| < |\hat{E}^{\mathsf{out}}(\mathcal{W})|$, or it computes two disjoint subgraphs $C_1, C_2$ of $G$, each of which has the $\tilde{\alpha}'$-bandwidth property, such that, for all $i \in \{1,2\}$, there is a set $\mathcal{R}_i$ of at least $\Omega(\tilde{\alpha}^2 k/\log^2 m)$ edge-disjoint paths in $G$, routing a subset $T_i \subseteq T$ of terminals to the edges of $\delta_G(C_i)$.*

We prove Lemma J.3 in the following subsection, after we complete the proof of Lemma 11.5 using it. Throughout the algorithm, we maintain a legal clustering $\mathcal{W}$ of $G$. If, at any point in the algorithm's execution, we obtain a legal clustering $\mathcal{W}$ with $|\hat{E}^{\mathsf{out}}(\mathcal{W})| < \tilde{\alpha}k/4$, then, from Claim J.2, the set $T$ of terminals is not $\tilde{\alpha}$-well-linked in $G$. We then terminate the algorithm and return FAIL. Therefore, from now on we assume that every legal clustering $\mathcal{W}$ that the algorithm obtains has $|\hat{E}^{\mathsf{out}}(\mathcal{W})| \geq \tilde{\alpha}k/4$.

We start with an initial collection $\mathcal{W}$ of clusters of $V(G) \setminus T$, where for every vertex $v \in V(G) \setminus T$, we add a cluster $\{v\}$ to $\mathcal{W}$. It is easy to verify that $\mathcal{W}$ is a legal clustering of $G$, since the degree of every vertex in $G$ is guaranteed to be at most $\tilde{\alpha}k/64$. We then perform a number of iterations. In every iteration, we apply the algorithm from Lemma J.3 to the current legal clustering $\mathcal{W}$. If the outcome of the algorithm is another legal clustering $\mathcal{W}'$ of $G$ with $|\hat{E}^{\mathsf{out}}(\mathcal{W}')| < |\hat{E}^{\mathsf{out}}(\mathcal{W})|$, then we replace $\mathcal{W}$ with $\mathcal{W}'$ and continue to the next iteration. Assume now that the outcome of the algorithm from Lemma J.3 is a pair $C_1, C_2$ of disjoint subgraphs of $G$, each of which has the $\tilde{\alpha}'$-bandwidth property, such that for all $i \in \{1,2\}$, there is a set $\mathcal{R}_i$ of at least $\Omega(\tilde{\alpha}^2 k/\log^2 m)$ edge-disjoint paths in $G$, routing a subset $T_i \subseteq T$ of terminals to the edges of $\delta_G(C_i)$. In this case, we let $(X,Y)$ be a partition of $V(G)$ with $V(C_1) \subseteq X$ and $V(C_2) \subseteq Y$, that minimizes $|E_G(X,Y)|$ among all such partitions. Notice that such a partition $(X,Y)$ can be computed via a standard minimum $s$-$t$ cut computation. Moreover, from the Maximum Flow / Minimum Cut theorem, we are guaranteed that there is a collection $\mathcal{Q}$ of $|E_G(X,Y)|$ edge-disjoint paths, each of which connects a vertex of $V(C_1)$ to a vertex of $V(C_2)$; we can assume w.l.o.g. that each path in $\mathcal{Q}$ is simple and it does not contain vertices of $V(C_1) \cup V(C_2)$ as inner vertices. Notice that every path in $\mathcal{Q}$ must contain exactly one edge of the set $E' = E_G(X,Y)$, and every edge of $E'$ must lie on exactly one such path. For every path $Q \in \mathcal{Q}$, we denote by $Q_1$ the

subpath of $Q$ from its endpoint that lies in $C_1$ to an edge of $E'$, and we denote by $Q_2$ the subpath of $Q$ from an edge of $E'$ to a vertex of $C_2$. Let $\mathcal{Q}_1 = \{Q_1 \mid Q \in \mathcal{Q}\}$ and $\mathcal{Q}_2 = \{Q_2 \mid Q \in \mathcal{Q}\}$. Then $\mathcal{Q}_1$ is a set of edge-disjoint paths routing the edges of $E'$ to edges of $\delta_G(C_1)$, in graph $G[X] \cup E'$, and similarly, $\mathcal{Q}_2$ is a set of edge-disjoint paths routing the edges of $E'$ to edges of $\delta_G(C_2)$ in graph $G[Y] \cup E'$. We now show that the partition $(X, Y)$ of $V(G)$ has all required properties.

Assume w.l.o.g. that $|X \cap T| \geq |Y \cap T|$. Recall that there is a set $\mathcal{R}_2$ of at least $\Omega(\tilde{\alpha}^2 k/\log^2 m)$ edge-disjoint paths in $G$, routing a subset $T_2 \subseteq T$ of terminals to the edges of $\delta_G(C_2)$. Assume first that $|T_2 \cap X| \geq |T_2|/2$. Let $\mathcal{R}'_2 \subseteq \mathcal{R}_2$ be the set of paths whose endpoint lies in $T_2 \cap X$, so $|\mathcal{R}'_2| \geq |\mathcal{R}_2|/2 \geq \Omega(\tilde{\alpha}^2 k/\log^2 m)$. Then each path $R \in \mathcal{R}'_2$ connects a vertex of $T_2 \cap X$ to a vertex of $Y$, so it must contain an edge of $E_G(X, Y)$. By suitably truncating each such path $R \in \mathcal{R}'_2$, we obtain a collection $\mathcal{R}$ of $\Omega(\tilde{\alpha}^2 k/\log^2 m)$ edge-disjoint paths, routing the terminals of $T_2 \cap X$ to the edges of $E_G(X, Y)$.

Assume now that $|T_2 \cap X| < |T_2|/2$. Let $h = \lceil |T_2|/2 \rceil$. Then $|X \cap T|, |Y \cap T| \geq h$ must hold. We apply the algorithm from Theorem 4.17 to graph $G$ and two arbitrary subsets $T'_1 \subseteq X \cap T, T'_2 \subseteq Y \cap T$ of terminals, of cardinality $h$ each. If the set $T$ of terminals is $\tilde{\alpha}$-well-linked in $G$, the algorithm must return a collection $\mathcal{R}'$ of paths in graph $G$, such that $\mathcal{R}'$ is an one-to-one routing of vertices of $T'_1$ to vertices of $T'_2$, and $\mathrm{cong}_G(\mathcal{R}') \leq \lceil 1/\tilde{\alpha} \rceil$. If the algorithm fails to return such a collection of paths, then we are guaranteed that the set $T$ of terminals is not $\tilde{\alpha}$-well-linked in $G$. We then terminate the algorithm and return FAIL. Therefore, we assume from now on that the algorithm from Theorem 4.17 returned a collection $\mathcal{R}'$ of paths with $\mathrm{cong}_G(\mathcal{R}') \leq \lceil 1/\tilde{\alpha} \rceil$, such that $\mathcal{R}'$ is an one-to-one routing of $T'_1$ to $T'_2$. From Claim 4.2, there is a collection $\mathcal{R}''$ of at least $\Omega(h\tilde{\alpha}) = \Omega(\tilde{\alpha}^3 k/\log^2 m)$ edge-disjont paths in graph $G$, routing a subset $T''_1 \subseteq T'_1$ of terminals to a subset $T''_2 \subseteq T'_2$ of terminals. Each path $R \in \mathcal{R}''$ must then contain an edge of $E_G(X, Y)$. By suitably truncating each such path, we obtain a collection $\mathcal{R}$ of $\Omega(\tilde{\alpha}^3 k/\log^2 m)$ edge-disjoint paths, routing the terminals of $T''_1$ to the edges of $E_G(X, Y)$.

It is now enough to prove that each of the clusters $G[X], G[Y]$ has the $\tilde{\alpha}'/2$-bandwidth property, which we do in the following claim.

**Claim J.4** *Each of the clusters $G[X], G[Y]$ has the $\tilde{\alpha}'/2$-bandwidth property.*

**Proof:** We show this for $G[X]$; the proof for $G[Y]$ is symmetric.

Assume for contradiction that $G[X]$ does not have the $\tilde{\alpha}'/2$-bandwidth property. Then there must be a partition $(A, B)$ of $X$, such that $|E_G(A, B)| < \tilde{\alpha}' \cdot \min\left\{|\delta_G(X) \cap \delta_G(A)|, |\delta_G(X) \cap \delta_G(B)|\right\}/2$. We assume w.l.o.g. that $|\delta_G(X) \cap \delta_G(A)| \leq |\delta_G(X) \cap \delta_G(B)|$, and we denote $|\delta_G(X) \cap \delta_G(A)|$ by $r$.

Note that partition $(A, B)$ of $X$ naturally defines a partition $(A', B')$ of $V(C_1)$, with $A' = A \cap V(C_1)$ and $B' = B \cap V(C_1)$. Since cluster $C_1$ has the $\tilde{\alpha}'$-bandwidth property, while $|E_{C_1}(A', B')| \leq |E_G(A, B)| < \tilde{\alpha}' r/2$, either $|\delta_G(A') \cap \delta_G(C)| < r/2$, or $|\delta_G(B') \cap \delta_G(C_1)| < r/2$ must hold. Assume w.l.o.g. that it is the former. Recall that $|\delta_G(X) \cap \delta_G(A)| = r$, and there is a set $\mathcal{Q}_1$ is a set of edge-disjoint paths routing the edges of $E' = \delta_G(X)$ to edges of $\delta_G(C_1)$. Let $\mathcal{Q}' \subseteq \mathcal{Q}_1$ be the paths that originate at edges of $|\delta_G(X) \cap \delta_G(A)|$, so $|\mathcal{Q}'| \geq r$. Recall that each path in $\mathcal{Q}'$ terminates at an edge of $\delta_G(C_1)$. However, since $|\delta_G(A') \cap \delta_G(C_1)| < r/2$, at least $r/2$ of the paths in $\mathcal{Q}'$ must contain a vertex of $B$. Therefore, at least $r/2$ of the paths in $\mathcal{Q}'$ contain an edge of $E_G(A, B)$, and so $|E_G(A, B)| \geq r/2$, a contradiction. □

## Proof of Lemma J.3

We need the following claim, which is a constructive version of Lemma 5.8 from [Chu16]; the proof is almost identical to that in [Chu16] and is included here for completeness.

**Claim J.5** *There is an efficient algorithm that, given any graph $G'$ with maximum vertex degree at most $\Delta$, computes a partition $(A,B)$ of $V(G')$, with $|E(A)|, |E(B)| \geq \frac{|E(G')|}{4} - \Delta$.*

**Proof:** For every vertex $v \in V(G')$, let $d(v)$ denote its degree in $G'$. For a subset $S \subseteq V(G')$ of vertices, let $\mathrm{vol}(S) = \sum_{v \in S} d(v)$. We start by computing an initial partition $(A,B)$ of $V(G')$, with $|\mathrm{vol}(A) - \mathrm{vol}(B)| \leq \Delta$, using a simple greedy algorithm: start with $A = B = \emptyset$, and process the vertices of $G'$ one-by-one. When $v$ is processed, add it to $A$ if $\mathrm{vol}(A) < \mathrm{vol}(B)$ currently holds, and add it to $B$ otherwise. It is easy to see that at the end of this procedure, we obtain a partition $(A,B)$ of $V(G')$ with $|\mathrm{vol}(A) - \mathrm{vol}(B)| \leq \Delta$.

We then iterate. The input to iteration $i$ is a partition $(A_i, B_i)$ of $V(G')$ with $|\mathrm{vol}(A_i) - \mathrm{vol}(B_i)| \leq 2\Delta$, where the input to the first iteration is the partition $(A_1, B_1) = (A, B)$ that we have just computed. We assume w.l.o.g. that $\mathrm{vol}(A_i) \geq \mathrm{vol}(B_i)$ holds, so $|E(A_i)| \geq |E(B_i)|$. If $|E(B_i)| < \frac{|E(G')|}{4} - \Delta$, then the outcome of the $i$th iteration is a partition $(A_{i+1}, B_{i+1})$ of $V(G')$ with $|\mathrm{vol}(A_{i+1}) - \mathrm{vol}(B_{i+1})| \leq 2\Delta$, and $|E(A_{i+1}, B_{i+1})| < |E(A_i, B_i)|$; otherwise, the algorithm terminates. In the latter case, we get that $|E(A_i)| \geq |E(B_i)| \geq \frac{|E(G')|}{4} - \Delta$, as required.

We now describe the execution of the $i$th iteration, whose input is is a partition $(A_i, B_i)$ of $V(G')$ with $|\mathrm{vol}(A_i) - \mathrm{vol}(B_i)| \leq 2\Delta$, such that $\mathrm{vol}(A_i) \geq \mathrm{vol}(B_i)$ and $|E(B_i)| < \frac{|E(G')|}{4} - \Delta$ hold.

For every vertex $v \in A_i$, let $d_1(v)$ be the number of edges incident to $v$ whose other endpoint belongs to $A_i$, and let $d_2(v)$ be the number of edges incident to $v$ whose other endpoint belongs to $B_i$. As we show later, there must exist a vertex $v \in A_i$ with $d_1(v) < d_2(v)$. Let $v$ be any such vertex. We then define a new partition $(A_{i+1}, B_{i+1})$ of $V(G')$ as follows: $A_{i+1} = A_i \setminus \{v\}$ and $B_{i+1} = B_i \cup \{v\}$. It is easy to verify that $|E(A_{i+1}, B_{i+1})| < |E(A_i, B_i)|$, while:

$$|\mathrm{vol}(A_{i+1}) - \mathrm{vol}(B_{i+1})| \leq \max\{|\mathrm{vol}(A_i) - \mathrm{vol}(B_i)|, 2d(v)\} \leq 2\Delta.$$

We then output the partition $(A_{i+1}, B_{i+1})$ of $V(G')$ and terminate the iteration.

It now remains to show that, if $|E(B_i)| < \frac{|E(G')|}{4} - \Delta$, there must exist a vertex $v \in A_i$ with $d_1(v) < d_2(v)$. Indeed, assume for contradiction that for every vertex $v \in A_i$, $d_1(v) \geq d_2(v)$.

Then $|E(A_i)| = \frac{1}{2}\sum_{v \in A_i} d_1(v) \geq \frac{1}{2}\sum_{v \in A_i} d_2(v) = \frac{1}{2}|E(A_i, B_i)|$. Altogether, $|E(G')| = |E(A_i)| + |E(B_i)| + |E(A_i, B_i)| \leq 4|E(A_i)|$, and so $|E(A_i)| \geq |E(G')|/4$.

On the other hand:

$$|E(B_i)| = \frac{\mathrm{vol}(B_i) - |E(A_i, B_i)|}{2} \geq \frac{\mathrm{vol}(A_i) - 2\Delta - |E(A_i, B_i)|}{2} \geq \frac{2|E(A_i)| - 2\Delta}{2} \geq \frac{|E(G')|}{4} - \Delta,$$

a contradiction to our assumption that $|E(B_i)| < \frac{|E(G')|}{4} - \Delta$.

The algorithm terminates once we obtain a partition $(A_i, B_i)$ of $V(G')$ with $|E(A_i)|, |E(B_i)| \geq \frac{|E(G')|}{4} - \Delta$. From the above discussion, this is guaranteed to happen after at most $|E(G')|$ iterations. Since each iteration can be executed efficiently, the claim follows. □

We apply the algorithm from Claim J.5 to graph $\hat{G}' = \hat{G} \setminus T$. Recall that, since for every cluster $W \in \mathcal{W}$, $|\delta_G(W)| \leq \tilde{\alpha}k/64$, every vertex in graph $\hat{G}'$ has degree at most $\tilde{\alpha}k/64$. Moreover, $|E(\hat{G}')| = |\hat{E}^{\mathsf{out}}(\mathcal{W})| \geq \tilde{\alpha}k/4$. Therefore, we obtain a partition $(A,B)$ of $V(\hat{G}')$ with $|E_{\hat{G}'}(A)|, |E_{\hat{G}'}(B)| \geq |E(\hat{G}')|/4 - \tilde{\alpha}k/64 \geq |E(\hat{G}')|/8 \geq \tilde{\alpha}k/32$.

Let $\mathcal{W}_1$ be the set of all clusters $W \in \mathcal{W}$ with $v_W \in A$, and let $\mathcal{W}_2$ be the set of all clusters $W \in \mathcal{W}$ with $v_W \in B$. Clearly, $(\mathcal{W}_1, \mathcal{W}_2)$ is a partition of $\mathcal{W}$. Our next step is summarized in the following claim.

**Claim J.6** *There is an efficient algorithm, that, given any subset $\mathcal{C} \subseteq \mathcal{W}$ of clusters, such that the total number of edges $e = (u, v) \in E(G)$ where $u$ and $v$ lie in distinct clusters of $\mathcal{C}$ is at least $|E(\hat{G}')|/8$, outputs one of the following:*

- *either a legal clustering $\mathcal{W}'$ of $G$ with $|\hat{E}^{\mathsf{out}}(\mathcal{W}')| < |\hat{E}^{\mathsf{out}}(\mathcal{W})|$; or*
- *a cluster $C$ with $V(C) \subseteq \bigcup_{C' \in \mathcal{C}} V(C')$, such that $C$ has the $\tilde{\alpha}'$-bandwidth property, and there exists a collection $\mathcal{R}$ of at least $\tilde{\alpha} \cdot \tilde{\alpha}' k/256$ edge-disjoint paths in $G$ routing a subset of terminals to the edges of $\delta_G(C)$.*

Observe that Claim J.6 finishes the proof of Lemma J.3, as follows. Let $A' = \bigcup_{W \in \mathcal{W}_1} V(W)$, and let $B' = \bigcup_{W \in \mathcal{W}_2} V(W)$; clearly, $A' \cap B' = \emptyset$. We aply the algorithm from Claim J.6 to the set $\mathcal{W}_1$ of clusters, and then separately to the set $\mathcal{W}_2$ of clusters. If the outcome of any of the two algorithms is a legal clustering $\mathcal{W}'$ of $G$ with $|\hat{E}^{\mathsf{out}}(\mathcal{W}')| < |\hat{E}^{\mathsf{out}}(\mathcal{W})|$, then we return this legal clustering $\mathcal{W}'$ and terminate the algorithm. Therefore, we assume from now on that the outcome of the algorithm from Claim J.6 when applied to cluster set $\mathcal{W}_1$ is a cluster $C_1$ with $V(C_1) \subseteq A'$, such that $C_1$ has the $\tilde{\alpha}'$-bandwidth property, and there exists a collection $\mathcal{R}_1$ of at least $\tilde{\alpha} \cdot \tilde{\alpha}' k/256 = \Omega(\tilde{\alpha}^2 k/\log^2 m)$ edge-disjoint paths in $G$ routing some subset $T_1 \subseteq T$ of terminals to the edges of $\delta_G(C_1)$. Similarly, the outcome of the algorithm from Claim J.6 when applied to cluster set $\mathcal{W}_2$ is a cluster $C_2$ with $V(C_2) \subseteq B'$, such that $C_2$ has the $\tilde{\alpha}'$-bandwidth property, and there exists a collection $\mathcal{R}_2$ of at least $\tilde{\alpha} \cdot \tilde{\alpha}' k/256 = \Omega(\tilde{\alpha}^2 k/\log^2 m)$ edge-disjoint paths routing some subset $T_2 \subseteq T$ of terminals to the edges of $\delta_G(C_2)$. We then return $C_1$ and $C_2$. From the above discussion, $C_1$ and $C_2$ are both disjoint and have the required properties. In order to complete the proof of Lemma J.3, it is now enough to prove Claim J.6, which we do next.

**Proof of Claim J.6.** Let $S \subseteq V(G)$ be a set of vertices that contains, for every cluster $W \in \mathcal{C}$, the vertices of $W$. Since every edge that is incident to a vertex of $S$ either has a terminal of $T$ as its other endpoint, or belongs to $\hat{E}^{\mathsf{out}}(\mathcal{W})$, from Claim J.2, we get that $|\delta_G(S)| \leq k + |E(\hat{G}')| \leq 8|E(\hat{G}')|/\tilde{\alpha}$, since, from Claim J.2, $|E(\hat{G}')| = |\hat{E}^{\mathsf{out}}(\mathcal{W})| \geq \tilde{\alpha} k/4$.

We apply the algorithm from Theorem 4.19 to graph $G$ and its cluster $G[S]$, with parameter $\alpha = \tilde{\alpha}'$ (recall that $\tilde{\alpha}' = \frac{\tilde{\alpha}}{c \log^2 m}$ for a large enough constant $c$, so the requirement that $\tilde{\alpha}' < \min\left\{\frac{1}{64\beta_{\mathrm{ARV}}(m)\log m}, \frac{1}{48\log^2 m}\right\}$ is satisfied), to obtain a collection $\mathcal{C}'$ of disjoint clusters of $G[S]$ (if grah $G[S]$ is not connected, then we apply the algorithm from Theorem 4.19 to each connected component of $G[S]$ separately; this does not affect the remainder of the analysis). Recall that $\{V(C') \mid C' \in \mathcal{C}'\}$ partitions $S$; every cluster $C' \in \mathcal{C}'$ has the $\tilde{\alpha}'$-bandwidth property, and:

$$\begin{aligned}
\sum_{C' \in \mathcal{C}'} |\delta_G(C')| &\leq |\delta_G(S)| \cdot \left(1 + O(\tilde{\alpha}' \cdot \log^{3/2} m)\right) \\
&\leq |\delta_G(S)| + O\left(\frac{8|E(\hat{G}')|\tilde{\alpha}' \log^{3/2} m}{\tilde{\alpha}}\right) \\
&\leq |\delta_G(S)| + O\left(\frac{|E(\hat{G}')|}{c \log^{1/2} m}\right) \\
&\leq |\delta_G(S)| + \frac{|E(\hat{G}')|}{64},
\end{aligned}$$

since $\tilde{\alpha}' = \tilde{\alpha}/c\log^2 m$, and $c$ is a large enough constant.

We consider now a new clustering $\mathcal{W}'$ of $G$, that is obtained as follows: start from $\mathcal{W}' = \mathcal{W} \setminus \mathcal{C}$, and then add the clusters of $\mathcal{C}'$ to $\mathcal{W}'$. It is easy to verify that the clusters in $\mathcal{W}'$ are all mutually disjoint,

and that $\bigcup_{W' \in \mathcal{W}} V(W) = V(G) \setminus T$. Moreover, every clsuter $W \in \mathcal{W}$ has the $\tilde{\alpha}'$-bandwidth property. Next, we show that $|\hat{E}^{\mathsf{out}}(\mathcal{W}')| < |\hat{E}^{\mathsf{out}}(\mathcal{W})|$.

Indeed, we can partition the edge set $\hat{E}^{\mathsf{out}}(\mathcal{W})$ into three subsets: set $E_1$ contains all edges $e = (u, v)$, where $u$ and $v$ lie in different clusters of $\mathcal{W} \setminus \mathcal{C}$; set $E_2$ contains all edges $e = (u, v)$, where $u$ lies in a cluster of $\mathcal{W} \setminus \mathcal{C}$, and $v$ lies in a cluster of $\mathcal{C}$; lastly, $E_3$ contains all edges $e = (u, v)$, where $u$ and $v$ lie in different clusters of $\mathcal{C}$.

We can partition $\hat{E}^{\mathsf{out}}(\mathcal{W}')$ into three subsets $E'_1, E'_2, E'_3$ similarly, using cluster set $\mathcal{C}'$ instead of $\mathcal{C}$. Clearly, $E_1 = E'_1$, and $E_2 = E'_2 = \delta_G(S)$. From the statement of Claim J.6, $|E_3| \geq |E(\hat{G}')|/8$. On the other hand, since we have established that $\sum_{C \in \mathcal{C}'} |\delta_G(C)| \leq |\delta_G(S)| + \frac{|E(\hat{G}')|}{64}$, and since $E_2 = \delta_G(S) \subseteq \bigcup_{C \in \mathcal{C}'} \delta_G(C)$, we get that $|E'_3| \leq \frac{|E(\hat{G}')|}{64}$. We conclude that $|E'_3| < |E_3|$, and $|\hat{E}^{\mathsf{out}}(\mathcal{W}')| < |\hat{E}^{\mathsf{out}}(W)|$. Note however that $\mathcal{W}'$ may not be a legal clustering of $G$, since we do not guarantee that for every cluster $C \in \mathcal{C}'$, $\delta_G(C) \leq \tilde{\alpha}k/64$ (this property is guaranteed to hold for every cluster of $\mathcal{W} \setminus \mathcal{C}'$ though, since $\mathcal{W}$ is a legal clustering). In the remainer of the algorithm, we will attempt to "fix" the clustering $\mathcal{W}'$ so it becomes a legal clustering of $G$, and, if we fail to do so, we will produce the desired cluster $C$.

In the remainder of the algorithm, we will maintain a set $\mathcal{W}^*$ of clusters, starting with $\mathcal{W}^* = \mathcal{W}'$, and we will ensure that the following invariants hold for $\mathcal{W}^*$ at all times:

I1. all clusters in $\mathcal{W}^*$ are disjoint from each other, and $\bigcup_{W \in \mathcal{W}^*} V(W) = V(G) \setminus T$;

I2. every cluster $W \in \mathcal{W}^*$ has the $\tilde{\alpha}'$-bandwidth property;

I3. $|\hat{E}^{\mathsf{out}}(\mathcal{W}^*)| < |\hat{E}^{\mathsf{out}}(\mathcal{W})|$; and

I4. if $|\delta_G(W)| > \tilde{\alpha}k/64$ for some cluster $W \in \mathcal{W}^*$, then $V(W) \subseteq S$.

Note that all these invariants hold for the initial setting $\mathcal{W}^* = \mathcal{W}'$. The algorithm performs a number of iterations, as long as there is some cluster $W \in \mathcal{W}^*$ with $|\delta_G(W)| > \tilde{\alpha}k/64$. We now describe the execution of a single iteration.

Let $W \in \mathcal{W}^*$ be any cluster with $|\delta_G(W)| > \tilde{\alpha}k/64$. Using the standard max-flow computation, we compute a maximum-cardinality set $\mathcal{R}$ of edge-disjoint paths in graph $G$, where each path in $\mathcal{R}$ connects a distinct terminal of $T$ to a distinct edge of $\delta_G(W)$.

We now consider two cases. The first case happens if $|\mathcal{R}| \geq \tilde{\alpha}\tilde{\alpha}'k/256$. In this case, we terminate the algorithm and return the cluster $W$. Notice that, from Invariant I4, we are guaranteed that $V(W) \subseteq S = \bigcup_{C' \in \mathcal{C}} V(C)$, and from Invariant I2, $W$ has the $\tilde{\alpha}'$-bandwidth property.

Assume now that the second case happens, that is, $|\mathcal{R}| < \tilde{\alpha}\tilde{\alpha}'k/256$. From the maximum flow / minimum cut theorem, there is a partition $(A, B)$ of $V(G)$, with $V(W) \subseteq A$, $T \subseteq B$, and $|E_G(A, B)| \leq |\mathcal{R}| < \tilde{\alpha}\tilde{\alpha}'k/256$.

We slightly modify the cut $(A, B)$ in graph $G$, to compute a new cut $(A', B')$ with $W \subseteq A'$, $T \subseteq V(B')$, such that for every cluster $C \in \mathcal{W}^*$, either $V(C) \subseteq A'$, or $V(C) \subseteq B'$ holds. In order to do so, we process every cluster $C \in \mathcal{W}^* \setminus \{W\}$ one by one. Consider an iteration when cluster $C$ is processed. If $V(C) \subseteq A$, or $V(C) \subseteq B$, then no further updates for cluster $C$ are necessary. Otherwise, we denote by $E'(C) = E_G(A, B) \cap E(C)$ – the set of edges that cluster $C$ contributes to $E_G(A, B)$. We partition the set $\delta_G(C)$ of edges into two subsets: set $\delta^A(C)$, $\delta^B(C)$, as follows. Let $e = (u, v) \in \delta_G(C)$ be any such edge, and assume that $v \in V(C)$. If $v \in A$, then we add $e$ to $\delta^A(C)$, and otherwise we add $e$ to $\delta^B(C)$. If $|\delta^A(C)| < |\delta^B(C)|$, then we move all vertices of $V(C) \cap A$ to $B$. Notice that in this case, since $C$ has the $\tilde{\alpha}'$-bandwidth property, $|\delta^A(C)| \leq |E'(C)|/\tilde{\alpha}'$. Once the vertices of $V(C) \cap A$ are moved to $B$, the edges of $E'(C)$ no longer lie in the cut $E_G(A, B)$, and the only new edges that

may have been added to the cut $E_G(A,B)$ are the edges of $\delta^A(C)$. Since $|\delta^A(C)| \leq |E'(C)|/\tilde{\alpha}'$, we charge every edge of $E'(C)$ $1/\tilde{\alpha}'$ units for the edges of $\delta^A(C)$. Note that the edges of $E'(C)$ will never be charged again by the algorithm. Otherwise, $|\delta^B(C)| \leq |\delta^A(C)|$ holds, and we move the vertices of $B \cap V(C)$ to $A$. As before, $|\delta^B(C)| \leq |E'(C)|/\tilde{\alpha}'$ must hold, and the edges of $E'(C)$ no longer belong to the cut $E_G(A,B)$. The only new edges that may have been added to the cut are the edges of $\delta^B(C)$. As before, we charge every edge of $E'(C)$ $1/\tilde{\alpha}'$ units for the edges of $\delta^B(C)$.

Once every cluster in $\mathcal{W}^* \setminus \{W\}$ is processed, we obtain a new partition $(A', B')$ of $V(G)$, with $W \subseteq A'$, and $T \subseteq V(B')$, such that for every cluster $C \in \mathcal{W}^*$, either $V(C) \subseteq A'$, or $V(C) \subseteq B'$ holds. Moreover, since every edge in $E_G(A',B') \setminus E_G(A,B)$ is charged to some edge of $E_G(A,B) \setminus E_G(A',B')$, and the charge to each edge of $E_G(A,B)$ is at most $1/\tilde{\alpha}'$, we get that $|E_G(A',B')| \leq |E_G(A,B)|/\tilde{\alpha}' \leq \tilde{\alpha}k/256$.

We modify the clustering $\mathcal{W}^*$ in two steps. In the first step, we remove from $\mathcal{W}^*$ all clusters $W'$ with $V(W') \subseteq A'$, and we add to it cluster $G[A']$ (notice that we can assume w.l.o.g. that graph $G[A']$ is connected, since otherwise, we can move the sets of vertices corresponding to all connected components of $G[A']$ that are disjoint from $W$ to $B'$). Let $\mathcal{W}^{**}$ denote this new clustering. It is immediate to verify that Invariants I1 and I4 continue to hold in $\mathcal{W}^{**}$, since $|\delta_G(A')| < \tilde{\alpha}k/64$. Note also that the edges of $\delta_G(W) \setminus \delta_G(A')$ no longer belong to $\hat{E}^{\mathsf{out}}(\mathcal{W}^{**})$, while no new edges were added to $\hat{E}^{\mathsf{out}}(\mathcal{W}^{**})$. Since $|\delta_G(W)| > \tilde{\alpha}k/64$, while $|\delta_G(A')| = |E_G(A',B')| \leq \tilde{\alpha}k/256$, we get that $|\hat{E}^{\mathsf{out}}(\mathcal{W}^{**})| \leq |\hat{E}^{\mathsf{out}}(\mathcal{W}^{*})| - \tilde{\alpha}k/64$.

Note that Invariant I2 is guaranteed to hold for every cluster of $\mathcal{W}^{**}$ except for possibly $G[A']$. In our last step, we apply the algorithm from Theorem 4.19 to compute a well-linked decomposition of the cluster $G[A']$ with parameter $\tilde{\alpha}'$. Recall that the algorithm computes a collection $\mathcal{C}^*$ of clusters, such that vertex sets $\{V(C') \mid C' \in \mathcal{C}^*\}$ partition $A'$, each cluster $C' \in \mathcal{C}^*$ has the $\tilde{\alpha}'$-bandwidth property, and $\delta_G(C') \leq \delta_G(A') < \tilde{\alpha}k/64$ for all $C' \in \mathcal{C}^*$. Moreover, we are guaranteed that:

$$\sum_{C' \in \mathcal{C}^*} |\delta_G(C')| \leq |\delta_G(A')| \cdot \Big(1 + O(\tilde{\alpha}' \cdot \log^{3/2} m)\Big) \leq 2|\delta_G(A')| \leq \tilde{\alpha}k/128.$$

We let $\mathcal{W}^{***}$ be obtained from $\mathcal{W}^{**}$ by replacing $G[A']$ with the collection $\mathcal{C}^*$ of clusters. From the above discussion, it is immediate to verify that Invariants I1, I2 and I4 hold for $\mathcal{W}^{***}$. Moreover, $\hat{E}^{\mathsf{out}}(\mathcal{W}^{***}) \subseteq \hat{E}^{\mathsf{out}}(\mathcal{W}^{**}) \cup \big(\bigcup_{C' \in \mathcal{C}^*} \delta_G(C')\big)$. Therefore, $|\hat{E}^{\mathsf{out}}(\mathcal{W}^{***})| \leq |\hat{E}^{\mathsf{out}}(\mathcal{W}^{**})| + \sum_{C' \in \mathcal{C}^*} |\delta_G(C')| \leq |\hat{E}^{\mathsf{out}}(\mathcal{W}^{*})|$. This establishes Invariant I3 for $\mathcal{W}^{***}$. We then set $\mathcal{W}^* = \mathcal{W}^{***}$, and continue to the next iteration.

If the algorithm never terminates with a cluster $W$ and a collection $\mathcal{R}$ of at least $\tilde{\alpha}\tilde{\alpha}'k/256$ edge-disjoint paths routing a subset of terminals to the edges of $\delta_G(W)$, then the algorithm terminates once every cluster $W' \in \mathcal{W}^*$ has $|\delta_G(W')| \leq \tilde{\alpha}k/64$. We are then guaranteed that $\mathcal{W}^*$ is a legal clustering of $G$, and moreover, $|\hat{E}^{\mathsf{out}}(\mathcal{W}^*)| < |\hat{E}^{\mathsf{out}}(\mathcal{W})|$. We return the clustering $\mathcal{W}^*$ as the output of the algorithm.

□

### J.3 Proof of Lemma 11.9

We assume for contradiction that for every regular vertex $x \in V(\hat{H}_1)$, $|\mathcal{J}(x)| < \tilde{k}'$, and yet $\mathsf{OPT}_{\mathsf{cnwrs}}(I) < \frac{(\tilde{k}\tilde{\alpha}\alpha')^2}{c'\eta' \log^{20} m}$, where $c'$ is the constant from the definition of $\tilde{k}'$. The high-level idea of the proof is the following. We use the algorithm from Claim 4.23 in order to embed an expander $W$ over the set $\tilde{T}$ of terminals into $\hat{H}_1$, which, from Observation 4.24 has a high crossing number. On the other hand, from Claim 4.42, there is a drawing of the contracted graph $\hat{H}_1$ with relatively few crossings. We exploit this latter drawing of $\hat{H}_1$, the embedding of the expander $W$ into $\hat{H}_1$, and the fact that any set $\mathcal{J}'(x)$ of edge-disjoint paths routing a subset of terminals to a vertex $x$ of $\hat{H}_1 \cap V(H_1)$ must have a small cardinality, in order to obtain a drawing of the expander $W$ with relatively few crosings, reaching a

contradiction. We now proceed with a formal proof.

By applying Claim 4.23 to graph $\hat{H}_1$ and the set $\tilde{T}$ of terminals (that is $\tilde{\alpha}$-well-linked in $\hat{H}_1$ from Property P4), we conclude that there exist a graph $W$ with $V(W)=\tilde{T}$ and maximum vertex degree at most $c_{\mathrm{CMG}}\log^2\tilde{k}$, and an embedding $\hat{\mathcal{P}}$ of $W$ into $\hat{H}_1$ with congestion at most $\frac{c_{\mathrm{CMG}}\log^2\tilde{k}}{\tilde{\alpha}}$, such that $W$ is a 1/4-expander. Moreover, from Observation 4.24, the crossing number of $W$ is at least $\tilde{k}^2/(\tilde{c}\log^8\tilde{k})\geq\tilde{k}^2/(\tilde{c}\log^8 m)$, for some constant $\tilde{c}$. Recall that we have assumed for contradiction that $\mathsf{OPT}_{\mathsf{cnwrs}}(I)<\left(\frac{(\tilde{k}\tilde{\alpha}\alpha')^2}{c'\eta'\log^{20}m}\right)$ for some large enough constant $c'$. We will exploit this fact in order to show that there exists a drawing of $W$ with fewer than $\tilde{k}^2/(\tilde{c}\log^8 m)$ crossings, reaching a contradiction.

Recall that, from Claim 4.42, there is a drawing $\varphi$ of the contracted graph $\hat{H}_1$, whose number of crossings is bounded by:

$$O\left(\frac{\mathsf{OPT}_{\mathsf{cnwrs}}(I)\cdot\log^8 m}{(\alpha')^2}\right)\leq O\left(\frac{\tilde{k}^2\tilde{\alpha}^2}{c'\eta'\log^{12}m}\right),$$

from our assumption that $\mathsf{OPT}_{\mathsf{cnwrs}}(I)<\left(\frac{(\tilde{k}\tilde{\alpha}\alpha')^2}{c'\eta'\log^{20}m}\right)$.

In the remainder of the proof, we gradually modify the drawing $\varphi$ of $\hat{H}_1$, to transform it into a drawing of the expander $W$ with fewer than $\tilde{k}^2/(\tilde{c}\log^8 m)$ crossings, leading to a contradiction. The drawing of $W$ is obtained from the drawing $\varphi$ of $\hat{H}_1$ by exploiting the embedding $\hat{\mathcal{P}}$ of $W$ into $\hat{H}_1$. Intuitively, we would like to use the images of the paths in $\hat{\mathcal{P}}$ in $\varphi$ in order to draw the edges of the graph $W$. Unfortunately, the curves representing the images of these paths are not in general position. This is since that paths in $\hat{\mathcal{P}}$ may share edges, and they may also share vertices other than their endpoints. We modify the drawing $\varphi$ in three stages. In the first stage, we create a number of copies of every edge in $\hat{H}_1$, so that the paths in $\hat{\mathcal{P}}$ no longer share edges, and in the second stage, we modify the drawing of the curves corresponding to the images of the paths in $\hat{\mathcal{P}}$ in the vicinity of the vertices they share, using a nudging operation. Then in the third and the last stage we define the final drawing of the expander $W$.

**Stage 1: shared edges.** For every edge $e\in E(\hat{H}_1)$, let $N_e$ be the total number of paths in $\hat{\mathcal{P}}$ containing $e$; recall that $N_e\leq O((\log^2\tilde{k})/\tilde{\alpha})\leq O((\log^2 m)/\tilde{\alpha})$ must hold. Let $\hat{H}'$ be the graph obtained from graph $\hat{H}_1$ by deleting from it all edges that do not participate in paths in $\hat{\mathcal{P}}$, and, for each remaining edge $e$, creating $N_e$ parallel copies of edge $e$. Drawing $\varphi$ of graph $\hat{H}_1$ can be easily extended to a drawing $\varphi'$ of graph $\hat{H}'$, by deleting the images of all edges $e$ with $N_e=0$, and, for every edge $e$ with $N_e>1$, drawing the parallel copies of $e$ in parallel to the original image of $e$, at a very small distance from it, so that the images of the copies of $e$ do not cross. Since, for every edge $e$, $N_e\leq O((\log^2 m)/\tilde{\alpha})$, every crossing in the drawing $\varphi$ may give rise to at most $O((\log^4 m)/\tilde{\alpha}^2)$ crossings in the drawing $\varphi'$, and so the number of crossings in $\varphi'$ is bounded by:

$$\mathsf{cr}(\varphi)\cdot O\left(\frac{\log^4 m}{\tilde{\alpha}^2}\right)\leq O\left(\frac{\tilde{k}^2\tilde{\alpha}^2}{c'\eta'\log^{12}m}\right)\cdot O\left(\frac{\log^4 m}{\tilde{\alpha}^2}\right)\leq O\left(\frac{\tilde{k}^2}{c'\eta'\log^8 m}\right).$$

Notice that the set $\hat{\mathcal{P}}$ of paths in graph $\hat{H}_1$ naturally defines a set $\hat{\mathcal{P}}'$ of edge-disjoint paths in graph $\hat{H}'$, embedding the expander $W$ into $\hat{H}'$ (where for every edge $e\in E(\hat{H})$, every path in $\hat{\mathcal{P}}$ containing $e$ now uses a different copy of $e$).

**Stage 2: shared vertices.** We process every vertex $x\in V(\hat{H}')\setminus\tilde{T}$ one by one. Let $\hat{\mathcal{P}}'(x)=\{P_1,\ldots,P_r\}$ be the set of all paths in $\hat{\mathcal{P}}'$ containing $x$. For each such path $P_i(x)$, let $e(P_i,x)$ and

$e'(P_i,x)$ be the edges immediately preceding and immediately following $x$ on path $P_i$. Let $D(x)$ be a very small disc containing the image $x$ in the drawing $\varphi'$ in its interior. Consider now some path $P_i \in \hat{\mathcal{P}}'(x)$, and let $q_i, q_i'$ be the points where the images of $e(P_i,x)$ and $e'(P_i,x)$ intersect the boundary of $D(x)$, respectively. We define a curve $\gamma(P_i,x)$ inside disc $D(x)$, connecting $q_i$ and $q_i'$, such that, for every pair $P_i, P_j \in \hat{\mathcal{P}}'(x)$ of distinct paths, the two corresponding curves $\gamma(P_i,x)$, $\gamma(P_j,x)$ cross at most once, and every point of $D(X)$ lies on at most two such curves.

**Stage 3: final drawing of $W$.** We are now ready to define the final drawing $\varphi''$ of the expander $W$. Recall that $V(W)=\tilde{T}$. For every terminal $t \in \tilde{T}$, the image of $t$ in $\varphi''$ remains the same as the image of $t$ in $\varphi'$.

Consider now some edge $\hat{e}=(t,t') \in E(W)$, and let $P(\hat{e}) \in \hat{\mathcal{P}}'$ be its embedding path. Denote $P=(e_1,e_2,\ldots,e_\ell)$, and denote the vertices of $P$ by $t=x_0,x_1,x_2,\ldots,x_{\ell-1},x_\ell=t'$ in the order of their appearance on $P$. For each edge $e_i$, let $\gamma(e_i)$ be its image in the drawing $\varphi'$. If $i>1$, then we delete the portion of $\gamma(e_i)$ that lies inside the disc $D(x_{i-1})$, and similarly, if $i<\ell$, then we delete the portion of $\gamma(e_i)$ that lies inside the disc $D(x_i)$. The image of the edge $\hat{e}$ is obtained by concatenating the curves $\gamma(e_1),\gamma(P,x_1),\gamma(e_2),\ldots,\gamma(P,x_{\ell-1}),\gamma(e_\ell)$.

This completes the definition of the drawing $\varphi''$ of the expander $W$. Our last step is to show that this drawing contains few crossings, leading to a contradiction. The following claim will then complete the proof of Lemma 11.9.

**Claim J.7** $\mathsf{cr}(\varphi'') < \tilde{k}^2/(\tilde{c}\log^8 m)$.

**Proof:** Recall that drawing $\varphi'$ of $\hat{H}'$ contained at most $O\left(\frac{\tilde{k}^2}{c'\eta'\log^8 m}\right)$ crossings. For every vertex $u \in V(\hat{H}_1)$, let $N_u$ de note the number of paths in $\hat{\mathcal{P}}$ containing $u$. It is then easy to verify that the total number of crossings in $\varphi''$ is at most:

$$\mathsf{cr}(\varphi') + \sum_{u\in V(\hat{H}_1)} N_u^2 \le O\left(\frac{\tilde{k}^2}{c'\eta'\log^8 m}\right) + \sum_{u\in V(\hat{H}_1)} N_u^2.$$

Assuming that $c'$ is a large enough constant, the above expression is bounded by $\tilde{k}^2/(2\tilde{c}\log^8 m) + \sum_{u\in V(\hat{H}_1)} N_u^2$. Therefore, it is now enough to prove that $\sum_{u\in V(\hat{H}_1)} N_u^2 \le \tilde{k}^2/(2\tilde{c}\log^8 m)$.

We partition the vertices of $\hat{H}_1$ into two subsets: the set $U=\{v_C \mid C\in\mathcal{C}_X\}$ of supernodes, and the set $U'=V(\hat{H}_1)\setminus U$ of regular vertices, that lie in $H_1$.

Consider first a supernode $v=v_C$. Since the paths in $\hat{\mathcal{P}}$ cause edge-congestion at most $O(\log^2 m/\tilde{\alpha})$ in graph $\hat{H}_1$, $N_v \le O(\deg_{\hat{H}_1}(v)\log^2 m/\tilde{\alpha})$, and so $N_v^2 \le O(|\delta_{H_1}(C)|^2\log^4 m/\tilde{\alpha}^2)$. Recall that from Property P6:

$$\sum_{C\in\mathcal{C}} |\delta_H(C)|^2 \le \frac{(\tilde{k}\tilde{\alpha}\alpha')^2}{c_1\log^{20} m}.$$

Therefore, we conclude that:

$$\begin{aligned}\sum_{v\in U} N_v^2 &\leq \sum_{C\in\mathcal{C}} |\delta_H(C)|^2 \cdot O\left(\frac{\log^4 m}{\tilde{\alpha}^2}\right)\\ &\leq \frac{(\tilde{k}\tilde{\alpha}\alpha')^2}{c_1\log^{20} m}\cdot O\left(\frac{\log^4 m}{\tilde{\alpha}^2}\right)\\ &\leq \frac{\tilde{k}^2}{4\tilde{c}\log^8 m},\end{aligned}$$

since we have assumed that $c_1$ is a large enough constant. Lastly, it remains to show that $\sum_{v\in U'} N_v^2 \leq \frac{\tilde{k}^2}{4\tilde{c}\log^8 m}$. We do so using the following claim.

**Claim J.8** *For every vertex $v \in U'$, $N_v < \frac{8c_{\text{CMG}}^2\tilde{k}'\log^4 m}{\tilde{\alpha}}$.*

**Proof:** Assume for contradiction that there is some vertex $v \in U'$, with $N_v \geq \frac{8c_{\text{CMG}}^2\tilde{k}'\log^4 m}{\tilde{\alpha}}$. Then there is a set $\mathcal{P} \subseteq \hat{\mathcal{P}}$ of at least $\frac{8c_{\text{CMG}}^2\tilde{k}'\log^4 m}{\tilde{\alpha}}$ paths in the embedding $\hat{\mathcal{P}}$ of $W$ into $\hat{H}_1$, containing $v$. Recall that for every path $P \in \hat{\mathcal{P}}$, both endpoints of $P$ are terminals in $\tilde{T}$, and that, since the maximum vertex degree in $W$ is at most $c_{\text{CMG}}\log^2\tilde{k} \leq c_{\text{CMG}}\log^2 m$, every terminal may serve as an endpoint of at most $c_{\text{CMG}}\log^2 m$ paths in $\mathcal{P}$.

Consider any path $P \in \mathcal{P}$, and let $t, t'$ be its two endpoints. Let $P'$ be the subpath of $P$ between $t$ and $v$. We then set $\mathcal{P}_1 = \{P' \mid P \in \mathcal{P}\}$. Furthermore, since every terminal may serve as an endpoint of at most $c_{\text{CMG}}\log^2 m$ paths in $\mathcal{P}_1$, there is a subset $\mathcal{P}_2 \subseteq \mathcal{P}_1$ of at least $\frac{|\mathcal{P}|}{c_{\text{CMG}}\log^2 m}$ paths, each of which originates at a distinct terminal of $\tilde{T}$, and terminates at $v$. Recall that the paths in $\hat{\mathcal{P}}$ cause edge-congestion at most $\frac{c_{\text{CMG}}\log^2 m}{\tilde{\alpha}}$ in $\hat{H}_1$. Lastly, from Claim 4.2, there is a collection $\mathcal{P}_3$ of edge-disjoint paths in $\hat{H}_1$, routing a subset of terminals to $v$, such that:

$$|\mathcal{P}_3| \geq \frac{\tilde{\alpha}\cdot|\mathcal{P}_2|}{2c_{\text{CMG}}\log^2 m} \geq \frac{\tilde{\alpha}|\mathcal{P}|}{2c_{\text{CMG}}^2\log^4 m} > \tilde{k}',$$

contradicting the fact that the largest number of edge-disjoint paths routing terminals of $\tilde{T}$ to $v$ is bounded by $\tilde{k}'$. □

We group the vertices of $U'$ into groups $S_1, \ldots, S_q$, where $q = \left\lceil \log\left(\frac{8c_{\text{CMG}}^2\tilde{k}'\log^4 m}{\tilde{\alpha}}\right)\right\rceil + 1$. Set $S_i$ contains all vertices $v \in U'$, with $2^{i-1} \leq N_v < 2^i$.

Since paths in $\hat{\mathcal{P}}$ cause edge-congestion $O((\log^2 m)/\tilde{\alpha})$, for all $1 \leq i \leq q$, for every vertex $v \in S_i$, $\deg_{\hat{H}}(v) \geq \Omega(\tilde{\alpha}N_v/\log^2 m) \geq \Omega(\tilde{\alpha}\cdot 2^i/\log^2 m)$. Since the total number of edges in graph $\hat{H}_1$ is $|E(\hat{H}_1)| \leq O(\tilde{k}\cdot\eta\log^8 m/\alpha^3)$ from Property P3, while $\tilde{\alpha} = \Omega(\alpha/\log^4 m)$ from Property P4, we get that, for all $1 \leq i \leq q$,

$$\sum_{v\in S_i} N_v \leq \sum_{v\in S_i} O\left(\frac{\deg_{\hat{H}}(v)\log^2 m}{\tilde{\alpha}}\right) \leq O\left(\frac{|E(\hat{H}_1)|\log^6 m}{\alpha}\right) \leq O\left(\frac{\tilde{k}\eta\log^{14} m}{\alpha^4}\right).$$

Therefore:

$$\sum_{v\in S_i} N_v^2 \leq 2^{i+1}\cdot\sum_{v\in S_i} N_v \leq O\left(\frac{2^i\cdot\tilde{k}\eta\log^{14} m}{\alpha^4}\right).$$

Summing up over all sets $S_1, \ldots, S_q$, we get that:

$$\begin{aligned}\sum_{v \in U'} N_v^2 &\leq O\left(\frac{2^q \tilde{k}\eta \log^{14} m}{\alpha^4}\right)\\ &\leq O\left(\frac{c_{\mathrm{CMG}}^2 \tilde{k}'\tilde{k}\eta \log^{18} m}{\alpha^4 \tilde{\alpha}}\right)\\ &\leq O\left(\frac{c_{\mathrm{CMG}}^2 \tilde{k}^2}{c' \log^8 m}\right).\end{aligned}$$

since $q = \left\lceil \log\left(\frac{8c_{\mathrm{CMG}}^2 \tilde{k}' \log^4 m}{\tilde{\alpha}}\right)\right\rceil + 1$, $\tilde{\alpha} = \Omega(\alpha/\log^4 m)$, and $\tilde{k}' = \tilde{k}\alpha^5/(c'\eta \log^{36} m)$.

Lastly, since we can fix $c'$ to be a large enough constant, we get that $\sum_{v \in U'} N_v^2 < \frac{k^2}{4\tilde{c}\log^8 m}$.

To conclude, we have shown that $\mathsf{cr}(\varphi'') \leq \frac{\tilde{k}^2}{\tilde{c}\log^8 m}$, a contradiction. □

## J.4 Proof of Lemma 11.10

We start by defining a new expanded graph $H'$, whose construction is similar to that of $H^*$, except that now we expand every vertex $v \in V(H) \setminus (\tilde{T} \cup \{x\})$ into a grid. Specifically, we start with $H' = \emptyset$, and then process every vertex $u \in V(H) \setminus (\tilde{T} \cup \{x\})$ one by one. We denote by $d(u)$ the degree of the vertex $u$ in graph $H$. We now describe an iteration when a vertex $u \in V(H) \setminus (\tilde{T} \cup \{x\})$ is processed. Let $e_1(u), \ldots, e_{d(u)}(u)$ be the edges that are incident to $u$ in $H$, indexed according to their ordering in $\mathcal{O}_u \in \Sigma$. We let $\Pi(u)$ be the $(d(u) \times d(u))$ grid, and we denote the vertices on the first row of this grid by $s_1(u), \ldots, s_{d(u)}(u)$ indexed in their natural left-to-right order. We add the vertices and the edges of the grid $\Pi(u)$ to graph $H'$. As before, we refer to the edges in the resulting grids $\Pi(u)$ as inner edges. Once every vertex $u \in V(H) \setminus (\tilde{T} \cup \{x\})$ is processed, we add the vertices of $\tilde{T}$ to the graph $H'$. Recall that every terminal $t \in \tilde{T}$ has degree 2 in $H$. We denote $s_1(t) = s_2(t) = t$, and we arbitrarily denote the two edges incident to $t$ by $e_1(t)$ and $e_2(t)$. We also add vertex $x$ to $H'$. We denote $s_1(x) = \cdots = s_{d(x)}(x) = x$, and we denote the edges incident to $x$ by $e_1(x), \ldots, e_{d(x)}(x)$, indexed consistently with the circular ordering $\mathcal{O}_x \in \Sigma$, where $\Sigma$ is the rotation system for graph $H$. Next, we add a collection of outer to graph $H'$, exactly as before. Consider any edge $e = (u, v) \in E(H)$. Assume that $e$ is the $i$th edge of $u$ and the $j$th edge of $v$, that is, $e = e_i(u) = e_j(v)$. Then we add an edge $e' = (s_i(u), s_j(v))$ to graph $H'$, and we view this edge as the copy of the edge $e \in E(H)$. This completes the definition of graph $H'$.

The partition $(X, Y)$ of the vertices of $V(H) \setminus \tilde{T}$ naturally defines a partition $(X', Y')$ of the vertices of $V(H') \setminus \tilde{T}$, as follows: $X' = \left(\bigcup_{u \in X \setminus \{x\}} V(\Pi(u))\right) \cup \{x\}$ and $Y' = \bigcup_{u \in Y} V(\Pi(u))$. We denote $H_1' = H'[X' \cup \tilde{T}]$ and $H_2' = H'[Y \cup \tilde{T}]$. Let $\mathcal{W}_X = \{\Pi(u) \mid u \in X \setminus \{x\}\}$. Then $\mathcal{W}_X$ is a collection of disjoint clusters in graph $H_1'$. Moreover, since, for every vertex $u \in X \setminus \{x\}$, the set of vertices on the first row of a grid $\Pi(u)$ is 1-well-linked in $\Pi(u)$ (from Observation 4.14), the corresponding cluster $\Pi(u)$ has the 1-bandwidth property.

**Observation J.9** *There is a collection $\mathcal{J}'$ of paths in graph $H_1'$, routing all terminals of $\tilde{T}$ to $x$, with $\mathrm{cong}_{H_1'}(\mathcal{J}') \leq O\left(\frac{\eta \log^{36} m}{\alpha^6 (\alpha')^2}\right)$.*

**Proof:** Recall that there exists a set $\mathcal{J}$ of edge-disjoint paths in graph $\hat{H}_1$, routing a subset $\tilde{T}_0 \subseteq \tilde{T}$ of terminals to $x$, with $|\mathcal{J}| \geq \tilde{k}'$. Since every cluster in $\mathcal{C}$ has the $\alpha'$-bandwidth property, from

Claim 4.41, there is a collection $\mathcal{J}_0$ of edge-disjoint paths in graph $H_1$, routing a subset $\tilde{T}'_0 \subseteq \tilde{T}_0$ to $x$, where $|\tilde{T}'_0| \geq \alpha'\tilde{k}'/2$. Using the 1-bandwidth property of the clusters $\Pi(u)$ in graph $H'_1$, it is easy to verify that there is a collection $\mathcal{J}'_0$ of edge-disjoint paths in graph $H'_1$, routing the terminals of $\tilde{T}'_0$ to vertex $x$. Recall that $\tilde{k}' = \tilde{k}\alpha^5/(c'\eta\log^{36} m)$, where $c'$ is a constant whose value was fixed in the proof of Lemma 11.9.

From Property P4, the set $\tilde{T}$ of terminals is $\tilde{\alpha}$-well-linked in $\hat{H}_1$, and so, from Claim 4.39, it is $(\tilde{\alpha}\alpha')$-well-linked in $H_1$. Moreover, since $H_1 = (H'_1)_{|\mathcal{W}_X}$, and since each cluster in $\mathcal{W}_X$ has the 1-bandwidth property, from Claim 4.39, the set $\tilde{T}$ of terminals is $(\tilde{\alpha}\alpha')$-well-linked in $H'_1$.

From Lemma 11.1, there is a set $\mathcal{J}'$ of paths in graph $H'_1$, routing the terminals in $\tilde{T}$ to vertex $x$, with:

$$\mathrm{cong}_{H'_1}(\mathcal{J}') \leq O\left(\frac{|\tilde{T}|}{|\tilde{T}'_0|}\cdot\frac{1}{\alpha\alpha'}\right) \leq O\left(\frac{\tilde{k}}{\tilde{k}'\alpha(\alpha')^2}\right) \leq O\left(\frac{\eta\log^{36} m}{\alpha^6(\alpha')^2}\right).$$

□

For convenience, we denote by $\rho = O\left(\frac{\eta\log^{36} m}{\alpha^6(\alpha')^2}\right)$ the bound on $\mathrm{cong}_{H'_1}(\mathcal{J}')$. Note that we can compute such a collection $\mathcal{J}'$ of paths efficiently using standard maximum $s$-$t$ flow computation. In order to compute the ordering $\tilde{\mathcal{O}}$ of the terminals in $\tilde{T}$, we need one additional property from the paths in $\mathcal{J}'$: we need them to be *confluent*. In order to define confluent paths, we need to assign each path a direction, so that one of its endpoint becomes the first vertex on the path. If $P$ is a simple directed path, whose first endpoint is $s$ and last endpoint is $t$, a *suffix* of $P$ is any subpath $P' \subseteq P$ that contains the vertex $t$.

**Definition J.10 (Confluent Paths)** *Let $\mathcal{P}$ be a collection of directed paths. We say that the paths in $\mathcal{P}$ are* confluent *iff for every pair $P_1, P_2 \in \mathcal{P}$ of paths, either $P_1 \cap P_2 = \emptyset$, or $P_1 \cap P_2$ is a suffix of both $P_1$ and $P_2$.*

The following claim, that easily follows from the work of [CKL+07], allows us to transform the set $\mathcal{J}'$ of paths into a confluent one.

**Claim J.11** *There is an efficient algorithm that computes a set $\mathcal{J}''$ of confluent paths in graph $H'_1$, routing the vertices of $\tilde{T}$ to $x$, (where every path is directed towards $x$), with $\mathrm{cong}_{H'_1}(\mathcal{J}'') \leq O(\rho\log m)$.*

**Proof:** We use the notion of confluent flows from [CKL+07]. The following definitions are from [CKL+07]. Let $G = (V, E)$ be a directed graph, and let $\mathrm{Dem} : V \to \mathbb{R}^+$ be a demand function for vertices $v \in V$. Let $S \subseteq V$ denote a collection of sink vertices. We assume that every edge incident to a sink vertex $s \in S$ is directed towards $s$. A flow $f : E \to \mathbb{R}^+$ is a *valid flow* if it satisfies, for every vertex $v \in V \setminus S$:

$$\sum_{e=(v,w)\in E} f(e) - \sum_{e=(u,v)\in E} f(e) = \mathrm{Dem}(v).$$

In other words, the total amount of flow leaving $v$ is equal to the demand on $v$ plus the total amount of flow entering $v$. The *congestion* on vertex $v$ is defined to be as:

$$\sum_{e=(u,v)\in E} f(e) + \mathrm{Dem}(v),$$

the total amount of flow entering $v$ plus the demand at $v$. The congestion of $f$ is the maximum, over all vertices $v \in V$, of the congestion of $f$ at $v$.

We say that a flow $f$ is *confluent* if for every vertex $v \in V$, there is at most one edge $(v,u)$ with $f(v,u) > 0$. A confluent flow therefore defines a subgraph of $G$ (induced by edges carrying non-zero flow), consisting of disjoint components $\{T_1, \ldots, T_k\}$, where each $T_i$ is an arborescence directed towards the root $s_i \in S$. In each such arborescence $T_i$, the maximum vertex congestion occurs at the sink $s_i$, and is equal to the total demand of all vertices of $T_i$. The following result was proved in [CKL$^+$07].

**Theorem J.12 (Theorem 3 in [CKL$^+$07])** *There is an efficient algorithm, that, given a directed $n$-vertex graph $G$ with a collection $S \subseteq V(G)$ of sinks, and demands $\mathrm{Dem}(v) \geq 0$ for vertices $v \in V(G)$, such that there exists a (regular, splittable) flow $f$ with node congestion 1 satisfying the demands in graph $G$, computes a confluent flow satisfying all demands, with congestion $O(\log n)$.*

We construct a flow network from graph $H_1'$, as follows. First, we subdivide every edge $e$ that is incident to $x$ with a new sink vertex $s_e$, setting $S = \left\{ s_e \mid e \in \delta_{H_1'}(x) \right\}$, and delete the vertex $x$ from the graph. Next, we bi-direct every edge of the resulting graph (by replacing it with two anti-parallel edges), except that for every sink vertex $s_e \in S$, the unique edge incident to $s_e$ is directed towards $s_e$. For every terminal $t \in \tilde{T}$, we set its demand $\mathrm{Dem}(t) = 1/(4\rho)$, and we set the demands of all other vertices to 0. Let $G$ be the resulting flow network. Denote $n = |V(G)|$, and observe that $|V(G)| = O\left(\sum_{u \in V(H_1)} (\deg_{H_1}(u))^2\right) \leq O(m^2)$. Note that the collection $\mathcal{J}'$ of paths in graph $H_1'$ immediately defines a collection $\tilde{\mathcal{J}}$ of paths, routing the set $\tilde{T}$ of terminals to the vertices of $S$, in graph $G$, with edge-congestion at most $\rho$. Since the degree of every vertex in $G$ is at most 4, by sending $1/(4\rho)$ flow units on every path in $\mathcal{J}''$, we obtain a valid flow from vertices of $\tilde{T}$ to vertices of $S$, satisfying all demands, with vertex-congestion at most 1. From Theorem J.12, there is a confluent flow $f$ in graph $G$ with vertex-congestion $O(\log n) \leq O(\log m)$, satisfying all demands. We can use standard flow-decomposition of $f$ to obtain a collection $\tilde{\mathcal{J}}'$ of flow-paths, routing the terminals in $\tilde{T}$ to vertices of $S$, such that the paths in $\mathcal{J}''$ are confluent. Each such flow-paths carries $1/(4\rho)$ flow units in the flow $f$, so the total edge-congestion caused by paths in $\tilde{\mathcal{J}}'$ is at most $O(\rho \log m)$. Moreover, every sink vertex $s_e \in S$ is an endpoint of at most $O(\rho \log m)$ paths. The set $\tilde{\mathcal{J}}'$ of paths naturally define a set $\mathcal{J}''$ of confluent paths in graph $H_1'$, routing the set $\tilde{T}$ of terminals to vertex $x$, with edge-congestion $O(\rho \log m)$. □

We will use the set $\mathcal{J}''$ of confluent paths to both compute the desired ordering $\tilde{\mathcal{O}}$ of the terminals, and to show that there exists the desired drawing $\varphi$ of graph $H^*$.

For convenence, let $d$ denote the degree of vertex $x$ in $H$ and the edges incident to $x$ by $e_1(x), \ldots, e_d(x)$, indexed consistently with the circular ordering $\mathcal{O}_x \in \Sigma$, where $\Sigma$ is the rotation system for graph $H$. For all $1 \leq i \leq d$ let $\mathcal{P}_i \subseteq \mathcal{J}''$ be the subset of paths whose last edge is $e_i$, so $(\mathcal{P}_1, \ldots, \mathcal{P}_d)$ is a partition of $\mathcal{J}''$. We define a circular ordering of the paths in $\mathcal{J}''$ as follows: for each $1 \leq i \leq d$, the paths in $\mathcal{P}_i$ appear consecutively in this ordering, in an arbitrary order, and paths belonging to different subsets appear in the natural ordering $\mathcal{P}_1, \ldots, \mathcal{P}_d$ of these subsets. Denote $\mathcal{J}'' = \left\{P_1, \ldots, P_{\tilde{k}}\right\}$, where the paths are indexed by the ordering that we have just defined. For all $1 \leq j \leq |\tilde{T}|$, let $t_j$ be the terminal of $\tilde{T}$ that serves as an endpoint of path $P_j$. We have then defined an ordering $t_1, \ldots, t_{\tilde{k}}$ of the terminals in $\tilde{T}$, that we denote by $\tilde{\mathcal{O}}$. For all $1 \leq i \leq \tilde{k}$, let $e_i$ denote the edge $(x, t_i)$ in graph $H^*$.

It is now enough to show that there is a drawing $\varphi$ of graph $H^*$, in which the inner edges do not participate in crossings, and the images of edges $e_1, \ldots, e_{\tilde{k}}$ enter the image of $x$ in this order, such that $\mathsf{cr}(\varphi) \leq O\left(\mathsf{OPT}_{\mathsf{cnwrs}}(I) \cdot \frac{\eta^2 \log^{74} m}{\alpha^{12}(\alpha')^4}\right) + \left(\frac{\tilde{k}\eta \log^{37} m}{\alpha^6(\alpha')^2}\right)$. The following observation will then finish the proof of Lemma 11.10.

**Observation J.13** *There is a drawing $\varphi$ of graph $H^*$ with at most $O\left(\mathsf{OPT}_{\mathsf{cnwrs}}(I) \cdot \frac{\eta^2 \log^{74} m}{\alpha^{12}(\alpha')^4}\right) + O\left(\frac{\tilde{k}\eta \log^{37} m}{\alpha^6(\alpha')^2}\right)$ crossings, in which all crossings are between pairs of outer edges, and the images of edges $e_1, \ldots, e_{\tilde{k}}$ enter the image of $x$ in this circular order.*

**Proof:** Let $\varphi_1$ be the optimal solution to instance $I$ of $\mathsf{MCNwRS}$, and denote by $\chi_1 = \mathsf{OPT}_{\mathsf{cnwrs}}(I)$ its number of crossings. We can easily transform drawing $\varphi_1$ to a drawing $\varphi_2$ of graph $H'$, with the same number of crossings, such that every crossing in $\varphi_2$ is between a pair of outer edges. In order to do so, we consider, for every vertex $v \in V(H) \setminus (\tilde{T} \cup \{x\})$, the tiny $v$-disc $D_\varphi(v)$. We place a drawing of the grid $\Pi(v)$ inside the disc, using the natural layout of the grid (depending on the orientation of vertex $v$ in $\varphi$, we may need to flip the image of $\Pi(v)$).

Next, we slightly modify the graph $H'$, as follows. First, for every edge $e_i$ that is incident to $x$, we subdivide $e_i$ with a new vertex $s_i$, and then delete $x$ from the graph. Let $S = \{s_1, \ldots, s_d\}$ be the resulting set of new vertices. We modify the paths in $\mathcal{J}''$ so that each path now connects a distinct vertex of $\tilde{T}$ to some vertex of $S$, and the paths remain confluent. As before, we denote by $\mathcal{P}_i \subseteq \mathcal{J}''$ the set of paths that terminate at vertex $s_i$.

For every edge $e$ of the resulting graph, we let $N_e$ the number of paths in $\mathcal{J}''$ in which edge $e$ participates; recall that $N_e \leq O(\rho \log m)$ must hold. For every edge $e \in E(H'_1)$, if $N_e = 0$, then we delete edge $e$, and otherwise we replace $e$ with $N_e$ parallel copies. We denote the resulting graph by $H''$, and we denote by $H''_1$ the subgraph of $H''$ corresponding to $H'_1$, that is, if we let $X'' = V(H'_1) \cap V(H'')$, then $H''_1$ is the subgraph of $H''$ that is induced by vertices of $X'' \cup S$. Observe that graph $H''_1$ can be viewed as consisting of disjoint trees $\tau_1, \ldots, \tau_d$ (though some of the trees may consist of a single vertex $s_i$), where for all $1 \leq i \leq d$, the root of $\tau_i$ is the vertex $s_i$, but these trees may have parallel edges. If $v \in V(\tau_i) \setminus \{s_i\}$, and the subtree rooted at $v$ contains $n_v$ terminal vertices, then the edge connecting $v$ to its parent has exactly $n_v$ parallel copies, and $n_v \leq O(\rho \log m)$. We further modify the paths set $\mathcal{J}''$, so that the paths become edge-disjoint in graph $H''_1$, that is, we ensure that for every edge $e \in E(H'_1)$, each path in $\mathcal{J}''$ containing $e$ uses a different copy of the edge $e$.

We can modify the drawing $\varphi_2$ of graph $H'$ to obtain a drawing $\varphi_3$ of graph $H''$, as follows. First, consider the tiny $x$-disc $D = D_{\varphi_2}(x)$. Recall that, that for every edge $e_i$, the intersection of the image of $e_i$ in $\varphi_2$ and the boundary of the disc $D$ is a single point, that we denote by $p_i$. We place the image of the new vertex $s_i$ on point $p_i$, and we erase the portion of the image of $e_i$ that lies in disc $D$. We also delete the image of $x$. We delete images of edges and vertices as needed, and then, for every edge $e \in E(H''_1)$ with $N_e > 0$, we create $N_e$ copies of $e$, all of which are drawn in parallel to the original image of $e$, very close to it, so that the images of these copies of $e$ do not cross each other. Let $\varphi_3$ denote this resulting drawing of the graph $H''$. Since, for every edge $e$, $N_e \leq O(\rho \log m)$, it is easy to verify that $\mathsf{cr}(\varphi_3) \leq \mathsf{cr}(\varphi_2) \cdot O(\rho^2 \log^2 m) \leq O(\mathsf{OPT}_{\mathsf{cnwrs}}(I) \cdot \rho^2 \log^2 m)$.

For every vertex $v \in V(H'')$, this drawing $\varphi_3$ of $H''$ naturally defines a circular ordering of the edges of $\delta_{H''}(v)$, that we denote by $\mathcal{O}^3_v$ – the order in which the images of the edges of $\delta_{H''}(v)$ enter the image of $v$ in this drawing. (Note that the drawing $\varphi_3$ depends on the optimal solution to instance $I$ of $\mathsf{MCNwRS}$, so we cannot compute the drawing or the resulting orderings $\mathcal{O}^3(v)$ for $v \in V(H'')$ efficiently; we only use their existence here). Let $\Sigma^3 = \left\{\mathcal{O}^3_v\right\}_{v \in V(H'')}$ be the resulting rotation system for graph $H''$.

Recall that for all $1 \leq i \leq d$, $\mathcal{P}_i \subseteq \mathcal{J}''$ is a set of paths routing a subset of the terminals of $\tilde{T}$ to $s_i$, and for $1 \leq i \neq j \leq d$, no vertex may belong to a path in $\mathcal{P}_i$ and to a path in $\mathcal{P}_j$. For all $1 \leq i \leq d$, let $\tilde{T}_i \subseteq \tilde{T}$ be the set of terminals that serve as endpoints of paths in $\mathcal{P}_i$.

We apply Lemma 4.7 to each such path set $\mathcal{P}_i$ separately, to obtain a path set $\mathcal{P}'_i$, that is non-transversal with respect to the rotation system $\Sigma^3$. The lemma ensures that the paths in $\mathcal{P}'_i$ route terminals of $\tilde{T}_i$ to vertex $s_i$ in $H''$; the paths are edge-disjoint, and moreover, an edge $e \in E(H'')$ may only belong to a path of $\mathcal{P}'_i$ if it belonged to a path of $\mathcal{P}_i$. Let $\mathcal{J}''' = \bigcup_{i=1}^{d} \mathcal{P}'_i$.

We are now ready to define a drawing $\varphi$ of the graph $H^*$. Notice that graph $H''$ can be viewed as the union of graph $H^* \setminus \{x\}$, and the paths in $\mathcal{J}'''$. As discussed above, there is a drawing $\varphi_3$ of $H''$ with at most $O(\mathsf{OPT}_{\mathsf{cnwrs}}(I) \cdot \rho^2 \log^2 m)$ crossings. We have defined a rotation system $\Sigma^3$ for graph

$H''$, such that the drawing $\varphi_3$ is consistent with this rotation system. Moreover, from the definition of the drawing $\varphi_3$, there is a disc $D$ (that originally contained the image of the vertex $x$), whose interior is disjoint from the drawing $\varphi_3$, with vertices $s_1,\ldots,s_d$ appearing on the boundary of $D$ in this circular order. Lastly, the paths in sets $\mathcal{P}'_1,\ldots,\mathcal{P}'_d$ are all non-transversal with respect to $\Sigma_3$, and for all $1\leq i\neq j\leq d$, paths in $\mathcal{P}_i$ and paths in $\mathcal{P}_j$ cannot share vertices with each other.

In order to obtain the drawing $\varphi$ of $H^*$, we slightly modify the drawing $\varphi_3$, as follows. First, we place an image of vertex $x$ in the interior of the disc $D$. For every edge $e_j=(t_j,x)$ of $H^*$, let $P_j\in\mathcal{J}'''$ be the path whose endpoint is $t_j$, and let $s_{i_j}$ be its other endpoint. Let $\gamma_j$ be the image of the path $P_j$ in the drawing $\varphi_3$. Since all paths in set $\mathcal{J}'''$ are non-transversal with respect to $\Sigma_3$ we can apply the nudging algorithm from Claim 4.34 to the image of every vertex of $H''$ that lies on some path in $\mathcal{J}'''$, in order to compute, for each $1\leq j\leq\tilde{k}$, a curve $\gamma'_j$ connecting the image of $t_j$ to the image of $s_{i_j}$, so that, if we delete from $\varphi_3$ the images of all inner vertices and of all edges participating in the paths in $\mathcal{J}'''$, and add instead the curves $\gamma'_1,\ldots,\gamma'_{\tilde{k}}$, then the number of crossings does not increase. For all $1\leq j\leq d$, curve $\gamma'_j$ is obtained from $\gamma_j$ by "nudging" it in the vicinity of every vertex $v\in V(P_j)$ using the algorithm from Claim 4.34.

Note that for all $1\leq i\leq d$, for every terminal $t_j\in\tilde{T}_i$, the curve $\gamma'_j$ terminates at the image of the vertex $s_i$, and moreover, the images of the vertices $s_1,\ldots,s_d$ appear in this circular order on the boundary of the disc $D$. However, the order in which the curves in $\Gamma'_i=\left\{\gamma'_j\mid t_j\in\tilde{T}_i\right\}$ enter the image of $s_i$ may be different from the ordering of the corresponding terminals in $\tilde{T}$. We need to reorder the curves in $\Gamma'_i$ in the vicinity of $s_i$, so that they enter the image of $s_i$ in the order consistent with $\tilde{\mathcal{O}}$. Since for all $1\leq j\leq d$, $|\Gamma_j|\leq O(\rho\log m)$, we can perform these reorderings while introducing crossings whose number is bounded by:

$$\sum_{i=1}^{d}|\Gamma'_i|^2\leq\sum_{i=1}^{d}|\Gamma'_i|\cdot O(\rho\log m)\leq O(\tilde{k}\rho\log m).$$

For all $1\leq j\leq\tilde{k}$, let $\gamma''_j$ be the curve obtained form $\gamma'_j$ after the reordering, so that $\gamma''_j$ connects the image of $t_j$ to the image of $s_{i_j}$. By slighty extending this curve inside the disc $D$, we can ensure that it terminates at the image of $x$. This can be done for all $1\leq j\leq\tilde{k}$ without introducing any new crossings, while ensuring that the resulting curves $\gamma''_1,\ldots,\gamma''_{\tilde{k}}$ enter the image of $x$ in this order. We then let, for all $1\leq j\leq\tilde{k}$, $\gamma''_j$ be the image of the edge $(t_j,x)$, obtaining a drawing $\varphi$ of $H^*$. It is immediate to verify that every crossing in $\varphi$ is between a pair of outer edges, and from our construction, the edges $(t_1,x),\ldots,(t_{\tilde{k}},x)$ enter the image of $x$ in this circular order. From the above discussion, the total number of crossings in $\varphi$ is at most:

$$O(\mathsf{OPT}_{\mathsf{cnwrs}}(I)\cdot\rho^2\log^2 m)+O(\tilde{k}\rho\log m)\leq O\left(\mathsf{OPT}_{\mathsf{cnwrs}}(I)\cdot\frac{\eta^2\log^{74}m}{\alpha^{12}(\alpha')^4}\right)+\left(\frac{\tilde{k}\eta\log^{37}m}{\alpha^6(\alpha')^2}\right).$$

□

### J.5 Proof of Observation 11.11

Assume that $\mathsf{OPT}_{\mathsf{cnwrs}}(I)<\frac{(\tilde{k}\tilde{\alpha}\alpha')^2}{c_1\eta'\log^{20}m}$. Then, from Lemma 11.10:

$$\begin{aligned}\mathsf{cr}(\varphi) \leq & O\left(\frac{(\tilde{k}\tilde{\alpha}\alpha')^2\eta^2\log^{54} m}{c_1\eta'\alpha^{12}(\alpha')^4}\right)+O\left(\frac{\tilde{k}\eta\log^{37} m}{\alpha^6(\alpha')^2}\right)\\ \leq & O\left(\frac{\tilde{k}^2\log^{46} m}{c_1\eta^7\alpha^{10}(\alpha')^2}\right)+O\left(\frac{\tilde{k}\eta\log^{37} m}{\alpha^6(\alpha')^2}\right),\end{aligned} \tag{33}$$

since $\tilde{\alpha}=\Theta(\alpha/\log^4 m)$ and $\eta'\geq\eta^{13}$ (from the statement of Theorem 6.4).

Recall that, since we have assumed that Special Case 4 did not happen, $k\geq\eta^6$, and from Property P1, $\tilde{k}\geq\Omega(\alpha^3 k/\log^8 m)$. Therefore, $\tilde{k}\geq\Omega(\eta^6\alpha^3/\log^8 m)$, and $\eta\leq O\left(\frac{\tilde{k}\log^8 m}{\eta^5\alpha^3}\right)$. We can now bound the second term in Equation 33 as follows:

$$O\left(\frac{\tilde{k}\eta\log^{37} m}{\alpha^6(\alpha')^2}\right)\leq O\left(\frac{\tilde{k}^2\log^{45} m}{\eta^5\alpha^9(\alpha')^2}\right)$$

Recall that we have assumed that $\log m>c_0'$, for some large enough constant $c_0'$, whose value we can set to be greater than $c_1$. Therefore, $\mathsf{cr}(\varphi)\leq O\left(\frac{\tilde{k}^2\log^{46} m}{c_1\eta^6\alpha^{10}(\alpha')^2}\right)$ must hold.

Lastly, from the conditions of Theorem 6.4, $\eta\geq c^*\log^{46} m/(\alpha^{10}(\alpha')^2)$. Since we can assume that $c_1$ is a sufficiently large constant, we conclude that, if $\mathsf{OPT}_{\mathsf{cnwrs}}(I)<\frac{(\tilde{k}\tilde{\alpha}\alpha')^2}{c_1\eta'\log^{20} m}$, then $\mathsf{cr}(\varphi)\leq\frac{\tilde{k}^2}{c_2\eta^5}$ holds, where $c_2$ is an arbitrarily large constant whose value we can set later.

### J.6 Proof of Observation 11.12

In order to obtain the distribution $\mathcal{D}'$, for every router $\mathcal{Q}\in\Lambda'$, whose probability value in $\mathcal{D}$ is $p(\mathcal{Q})>0$, we construct a router $\mathcal{Q}'\in\Lambda(H,\tilde{T})$, as follows. Let $y'$ be the center vertex of $\mathcal{Q}$, and assume that $y'\in\Pi(y)$ for some vertex $y\in V(H)$. For every terminal $t\in\tilde{T}$, let $Q_t\in\mathcal{Q}$ be the unique path connecting $t$ to $y'$ in $H''$. By suppressing all inner edges on path $Q_t$, we obtain a path $Q_t'$ in graph $H$, connecting $t$ to $y$. We then set $\mathcal{Q}'=\left\{Q_t'\mid t\in\tilde{T}\right\}$. It is easy to verify that paths in $\mathcal{Q}'$ route $\tilde{T}$ to $y$ in graph $H$, so $\mathcal{Q}'\in\Lambda(H,\tilde{T})$. Moreover, for every edge $e\in E(H)$, $\mathrm{cong}_H(\mathcal{Q}',e)\leq\mathrm{cong}_{H''}(\mathcal{Q},e)$. We assign to the router $\mathcal{Q}'\in\Lambda(H,\tilde{T})$ the same probability value $p(\mathcal{Q})$. Let $\mathcal{D}'$ be the resulting distribution over the routers of $\Lambda(H,\tilde{T})$. Since every edge $e\in E(H)$ is an outer edge of $H''$, it is immediate to verify that $\mathbf{E}_{\mathcal{Q}'\sim\mathcal{D}'}\left[(\mathrm{cong}_H(\mathcal{Q}',e))^2\right]\leq\beta$.

## References


[ACNS82] M. Ajtai, V. Chvátal, M. Newborn, and E. Szemerédi. Crossing-free subgraphs. *Theory and Practice of Combinatorics*, pages 9–12, 1982.

[AMS07] Christoph Ambuhl, Monaldo Mastrolilli, and Ola Svensson. Inapproximability results for sparsest cut, optimal linear arrangement, and precedence constrained scheduling. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pages 329–337. IEEE, 2007.

[And10] Matthew Andrews. Approximation algorithms for the edge-disjoint paths problem via raecke decompositions. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 277–286. IEEE, 2010.

[ARV09] Sanjeev Arora, Satish Rao, and Umesh V. Vazirani. Expander flows, geometric embeddings and graph partitioning. *J. ACM*, 56(2), 2009.

[Cab13] Sergio Cabello. Hardness of approximation for crossing number. *Discrete & Computational Geometry*, 49(2):348–358, 2013.

[CC16] Chandra Chekuri and Julia Chuzhoy. Polynomial bounds for the grid-minor theorem. *Journal of the ACM (JACM)*, 63(5):1–65, 2016.

[CH11] Markus Chimani and Petr Hliněnỳ. A tighter insertion-based approximation of the crossing number. In *International Colloquium on Automata, Languages, and Programming*, pages 122–134. Springer, 2011.

[Chu11] Julia Chuzhoy. An algorithm for the graph crossing number problem. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 303–312. ACM, 2011.

[Chu12] Julia Chuzhoy. Routing in undirected graphs with constant congestion. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 855–874, 2012.

[Chu16] Julia Chuzhoy. Improved bounds for the excluded grid theorem. *arXiv preprint arXiv:1602.02629*, 2016.

[CKL+07] Jiangzhuo Chen, Robert D Kleinberg, László Lovász, Rajmohan Rajaraman, Ravi Sundaram, and Adrian Vetta. (almost) tight bounds and existence theorems for single-commodity confluent flows. *Journal of the ACM (JACM)*, 54(4):16–es, 2007.

[CKS04] Chandra Chekuri, Sanjeev Khanna, and F Bruce Shepherd. Edge-disjoint paths in planar graphs. In *45th Annual IEEE Symposium on Foundations of Computer Science*, pages 71–80. IEEE, 2004.

[CL12] Julia Chuzhoy and Shi Li. A polylogarithmic approximation algorithm for edge-disjoint paths with congestion 2. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 233–242. IEEE, 2012.

[CMM16] Julia Chuzhoy, Vivek Madan, and Sepideh Mahabadi. In *Private Communication*, 2016.

[CMS11] Julia Chuzhoy, Yury Makarychev, and Anastasios Sidiropoulos. On graph crossing number and edge planarization. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete algorithms*, pages 1050–1069. SIAM, 2011.

[CMT20] Julia Chuzhoy, Sepideh Mahabadi, and Zihan Tan. Towards better approximation of graph crossing number. To appear at FOCS'20.

[CS13] Chandra Chekuri and Anastasios Sidiropoulos. Approximation algorithms for euler genus and related problems. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 167–176. IEEE, 2013.

[CT19] Julia Chuzhoy and Zihan Tan. Towards tight (er) bounds for the excluded grid theorem. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1445–1464. SIAM, 2019.

[EGS02] Guy Even, Sudipto Guha, and Baruch Schieber. Improved approximations of crossings in graph drawings and vlsi layout areas. *SIAM Journal on Computing*, 32(1):231–252, 2002.

[GH61] Ralph E Gomory and Tien Chung Hu. Multi-terminal network flows. *Journal of the Society for Industrial and Applied Mathematics*, 9(4):551–570, 1961.

[GJ83] M. R. Garey and D. S. Johnson. Crossing number is NP-complete. *SIAM J. Algebraic Discrete Methods*, 4(3):312–316, 1983.

[Hli06] P. Hlinený. Crossing number is hard for cubic graphs. *J. Comb. Theory, Ser. B*, 96(4):455–471, 2006.

[HT74] John Hopcroft and Robert Tarjan. Efficient planarity testing. *Journal of the ACM (JACM)*, 21(4):549–568, 1974.

[KRV09] Rohit Khandekar, Satish Rao, and Umesh Vazirani. Graph partitioning using single commodity flows. *Journal of the ACM (JACM)*, 56(4):1–15, 2009.

[KS17] Ken-ichi Kawarabayashi and Anastasios Sidiropoulos. Polylogarithmic approximation for minimum planarization (almost). In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 779–788, 2017.

[KS19] Ken-ichi Kawarabayashi and Anastasios Sidiropoulos. Polylogarithmic approximation for euler genus on bounded degree graphs. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 164–175. ACM, 2019.

[Lei83] F. T. Leighton. *Complexity issues in VLSI: optimal layouts for the shuffle-exchange graph and other networks*. MIT Press, 1983.

[LR99] Tom Leighton and Satish Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *Journal of the ACM (JACM)*, 46(6):787–832, 1999.

[LT79] Richard J Lipton and Robert Endre Tarjan. A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*, 36(2):177–189, 1979.

[Mat02] J. Matoušek. *Lectures on discrete geometry*. Springer-Verlag, 2002.

[PSS96] János Pach, Farhad Shahrokhi, and Mario Szegedy. Applications of the crossing number. *Algorithmica*, 16(1):111–117, 1996.

[PSŠ09] Michael J Pelsmajer, Marcus Schaefer, and Daniel Štefankovic. Odd crossing number and crossing number are not the same. In *Twentieth Anniversary Volume:*, pages 1–13. Springer, 2009.

[PSŠ11] Michael J Pelsmajer, Marcus Schaefer, and Daniel Štefankovič. Crossing numbers of graphs with rotation systems. *Algorithmica*, 60(3):679–702, 2011.

[PT00] J. Pach and G. Tóth. Thirteen problems on crossing numbers. *Geombinatorics*, 9(4):194–207, 2000.

[Rac02] Harald Racke. Minimizing congestion in general networks. In *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.*, pages 43–52. IEEE, 2002.

[RS86] Neil Robertson and Paul D Seymour. Graph minors. v. excluding a planar graph. *Journal of Combinatorial Theory, Series B*, 41(1):92–114, 1986.

[RS09] R. B. Richter and G. Salazar. Crossing numbers. In L. W. Beineke and R. J. Wilson, editors, *Topics in Topological Graph Theory*, chapter 7, pages 133–150. Cambridge University Press, 2009.

[RST94] Neil Robertson, Paul Seymour, and Robin Thomas. Quickly excluding a planar graph. *Journal of Combinatorial Theory, Series B*, 62(2):323–348, 1994.

[Sch12] Marcus Schaefer. The graph crossing number and its variants: A survey. *The electronic journal of combinatorics*, pages DS21–Sep, 2012.

[Sid10] Anastasios Sidiropoulos. Personal communication, 2010.

[Tur77] P. Turán. A note of welcome. *J. Graph Theory*, 1:1–5, 1977.

[Vrt] Imrich Vrto. Crossing numbers of graphs: A bibliography. `http://www.ifi.savba.sk/~imrich`.

[Whi92] Hassler Whitney. Congruent graphs and the connectivity of graphs. In *Hassler Whitney Collected Papers*, pages 61–79. Springer, 1992.