

Homework 2

Due: Tuesday, Feb. 19 in class.

General remark: whenever you are asked to provide an α -approximation algorithm, you need to prove that your algorithm outputs a feasible α -approximate solution.

1. **Metric k -cluster** In the metric k -cluster problem, we are given a complete undirected graph with non-negative weights w_e on edges satisfying the triangle inequality and an integer $k > 0$. The goal is to partition all vertices into k clusters V_1, \dots, V_k , minimizing the maximum distance of any pair of points in a cluster. So we seek to minimize the maximum, over all clusters V_i , of $\max_{u,v \in V_i} \{w_{(u,v)}\}$.
 - (a) Show a factor-2 approximation algorithm for the problem.
 - (b) Prove that the problem is hard to approximate up to any factor less than 2.
Hint: Reduction from 3-coloring.
2. **k -dispersion** The input to the k -dispersion problem is a complete undirected graph $G = (V, E)$ with weights $w_e \geq 0$ on edges $e \in E$, that obey the triangle inequality. Additionally, we are given an integer $k > 0$. The goal is to choose a subset $S \subseteq V$ of k vertices that are as far apart from each other as possible. Namely, we are trying to maximize $\min_{s,s' \in S} \{w_{(s,s')}\}$.
 - Consider the following greedy algorithm: start with $S = \{v\}$ for an arbitrary node $v \in V$, and repeatedly add a node furthest from the nodes currently in S , until $|S| = k$. Show that this is a 2-approximation algorithm.
 - Prove that the problem is hard to approximate up to any factor less than 2.
3. **Weighted k -center** Prove that weighted symmetric k -center problem is hard to approximate up to any factor better than 3.
Hint: Use the hardness construction for asymmetric k -center with $h = 3$ layers.
4. **Euclidean TSP** Show that the basic deterministic construction of the quad-tree (the one without the random shift) cannot be used for the PTAS for Euclidean TSP. (Show an example in which any well-behaved tour has cost at least $(1 + \alpha)\text{OPT}$ for some fixed constant α).
5. **Multidimensional knapsack and bin packing**
 - (a) In the multidimensional knapsack problem, the input is a set V of n non-negative d -dimensional vectors. Each vector $v_i \in V$ has a profit $w_i \geq 0$. Additionally, we are given a d -dimensional non-negative vector B (knapsack capacity). The goal is to find a maximum-profit subset $V' \subseteq V$ of vectors, whose sum is bounded by B coordinate-wise. Show a pseudo-polynomial time algorithm for multidimensional knapsack when d is a constant independent of n . What is the running time of your algorithm?
 - (b) In the multidimensional bin-packing problem, we are given a set V of n non-negative d -dimensional vectors. The goal is to partition the vectors into minimum number of bins, such that the sum of vectors in every bin is bounded by 1 in every coordinate. Show an $O(d)$ -approximation algorithm for multidimensional bin-packing.