

Homework due 11/3. Please write the name of any collaborator you worked with.

1. Caching. Consider the least frequently used (LFU) algorithm. When a cache miss occurs, it evicts the page that has been used least in total, i.e., it keeps a counter for the number of accesses to each page and evicts a page with least count. Show that LFU has an unbounded competitive ratio. In other words, for every  $R$ , give a sequence such that the cost of LFU is at least  $R$  times more than the cost of the best caching policy in hindsight.
2. Good and bad exploration paths. We are given a fixed graph  $G$  with  $n$  nodes and  $m$  edges. Each path can be represented as a vector in  $\{0, 1\}^m$  with 1's for the edges that are in the path. As such, the paths span a  $d$ -dimensional subspace of  $\mathbb{R}^m$  for some  $d \leq m$ . (The exact value of  $d$  will depend on the graph.)<sup>1</sup> That is, one can choose  $d$  basis paths  $b_1, b_2, \dots, b_d$ , such that any path  $p \in \{0, 1\}^m$  can be written uniquely as a linear combination of these basis paths  $p = \sum_{i=1}^d w_i b_i$  for  $w_i \in \mathbb{R}$ . (This also implies that if we knew the times on the basis paths we could compute the time on any other path.)

A good basis is one in which every path can be written as  $p = \sum_{i=1}^d w_i b_i$  with relatively small  $|w_i|$ 's. A bad basis is one in which there is a path  $p$  whose representation has large  $|w_i|$ . Give an example of a graph with  $m$  edges, a bad basis, and a path whose representation using that basis requires at least  $w_i \geq c^{m-c'}$  for some  $i$ , some constants  $c > 1, c' \in \mathbb{R}$  and all  $m > M$  for some  $M$  sufficiently large. (In other words, you are really giving a family of graphs and bases for larger and larger  $m$ .) Bonus points to the people who get the largest  $c$ .

---

<sup>1</sup>Since linear algebra is not a prerequisite for this class, please email me if you have not studied linear algebra and would like a to substitute a different problem for this one.