CS264: Homework #8

Due by midnight on Thursday, March 9, 2017

Instructions:

(1) Form a group of 1-3 students. You should turn in only one write-up for your entire group. See the course site for submission instructions.

(2) Please type your solutions if possible and feel free to use the LaTeX template provided on the course home page.

(3) All students should complete all of the exercises. Students taking the course for a letter grade should also complete Problems 29 and 30.

(4) Write convincingly but not excessively. Exercise solutions rarely need to be more than 1-2 paragraphs. Problem solutions rarely need to be more than a half-page (per part), and can often be shorter.

(5) You may refer to your course notes, and to the textbooks and research papers listed on the course Web page *only*. You cannot refer to textbooks, handouts, or research papers that are not listed on the course home page. (Exception: feel free to use your undergraduate algorithms textbook.) Cite any sources that you use, and make sure that all your words are your own.

(6) If you discuss solution approaches with anyone outside of your team, you must list their names on the front page of your write-up.

(7) Exercises are worth 5 points each. Problem parts are labeled with point values.

(8) No late assignments will be accepted.

Lecture 15 Exercises

Exercise 41

The *nonnegative rank* of a nonnegative $m \times n$ matrix $\mathbf{M}$ is the smallest value of k such that $\mathbf{M}$ can be written as the product of a $m \times k$ nonnegative matrix $\mathbf{A}$ and a $k \times n$ nonnegative matrix $\mathbf{B}$. (The usual rank of a matrix is defined similarly, but without the nonnegativity constraint.)

Give an example of a nonnegative matrix whose nonnegative rank is strictly larger than its rank.

Exercise 42

Prove that the following problem reduces to linear programming: given as input $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$ and $\mathbf{y} \in \mathbb{R}^d$, can $\mathbf{y}$ be written as a convex combination of the $\mathbf{x}_i$'s?

Exercise 43

This exercise fills in one of the missing details of the correctness argument for the greedy algorithm for computing anchor words. Consider distinct points $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$. The convex hull of the $\mathbf{x}_i$'s is defined as the set of convex combinations of the $\mathbf{x}_i$'s:

$$CH = \left\{ \sum_{i=1}^{n} \lambda_i \mathbf{x}_i \,:\, \lambda_1, \ldots, \lambda_n \geq 0, \sum_{i=1}^{n} \lambda_i = 1. \right\}.$$

A point $\mathbf{y}$ of this convex hull is a *vertex* if the only way to express $\mathbf{y}$ as a convex combination of the $\mathbf{x}_i$'s is by taking $\lambda_i = 1$ for some i (and hence $\lambda_j = 0$ for $j \neq i$). By definition, every vertex of the convex hull is one of the original points, but not conversely (why?).

Fix some point $\mathbf{x}_i$ arbitrarily. Prove that a point $\mathbf{x}_j$ that maximizes (over $\mathbf{x}_1, \ldots, \mathbf{x}_n$) the Euclidean distance $\|\mathbf{x}_i - \mathbf{x}_j\|_2$ must be a vertex of CH.

[Hint: use the strict convexity of the ℓ_2 norm.]

Lecture 16 Exercises

Exercise 44

This exercise shows that the random-order assumption is necessary to obtain a constant competitive ratio for the online facility location problem.

Consider the following input sequence: when the ith input point arrives, the distance between the ith point and jth point (for $j < i$) is $2^{-j} - 2^{-i}$. Thus, the distances are the same as those between the points $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \ldots$ on the real line.[1] Prove that for every constant $\alpha \geq 1$, every deterministic online algorithm has competitive ratio at least α as $n \to \infty$.[2]

[Hints: Prove that the optimal cost is at most 2. For an algorithm to be α-competitive, how frequently must it open a new center?]

Problems

Problem 29

(10 points) In the *k-secretary problem*, the online algorithm is allowed to pick up to k numbers (out of n total, which arrive in random order). Give an online algorithm with the following property: for some constant $c > 0$ (independent of k and n), for each of the k highest numbers x in the input, the algorithm chooses x with probability at least c.

Problem 30

Let's reconsider the online facility location algorithm from Lecture #16, and see how well it does in the worst case. (By Exercise 44, we know that at best it is $O(\log n)$-competitive in the worst case.) Fix an arbitrary input, with an arbitrary arrival order.

(a) (6 points) Continuing with the notation from lecture, consider a center c^* of the optimal solution, and let A be the points assigned to c^*. Suppose that every point $x \in A$ is roughly the same distance from c^*, between α and 2α for some α. Prove that there is a constant $a > 0$ such that the expected cost incurred by our online algorithm on the points of A is at most a times the cost $1 + \sum_{x \in A} d(x, c^*)$

[1] But remember the model: the online algorithm is not given the metric space up front, and cannot open a center at a point that it has not yet seen.

[2] It is not hard to extend the lower bound to randomized algorithms also, but you don't need to do this.

incurred by the optimal solution on these points. (Expectations are now solely with respect to the random coin flips of the algorithm.)

(b) (9 points) Prove that the online algorithm from lecture has expected cost at most $a \log n$ times that of the optimal solution (where again $a > 0$ is some constant independent of n).

[Hint: bucket the points of an optimal cluster according to their distances from the cluster center.]

Problem 31 (Extra Credit)

(10 extra credit points) The lower bound in Exercise 44 is a bit unsatisfying, as it exploits the constraint that an online algorithm cannot create a center at a point that it has not yet seen. Consider the following variant of the online facility location problem: the algorithm is given in advance an entire metric space (X, d); points $x_1, \ldots, x_n \in X$ arrive online; when a point x_i arrives, the algorithm can open any number of centers at whatever points of X that it wants (at a cost of 1 per center), and then assigns x_i to the closest center. (The algorithm can never delete a center, however.) Is it still true that there is no $O(1)$-competitive online algorithm (deterministic or randomized) for online facility location when the order of point arrival is determined by an adversary?