CS264: Homework #10

Due by midnight on Wednesday, December 10, 2014

Instructions:

(1) Students taking the course pass-fail should complete 5 of the exercises. Students taking the course for a letter grade should complete 8 of the exercises. Students taking the course for a letter grade should also complete some of the problems — we'll grade your problem solutions out of a total of 40 points (with any additional points counting as extra credit).

(2) All other instructions are the same as in previous problem sets.

Lecture 19 Exercises

Exercise 58

Recall the "diamond graphs" from lecture. Each such graph has a "left" and "right" vertex. G_0 is a single edge. For $i > 0$, G_i is obtained from four copies of G_{i-1} by arranging them in a diamond, with the left (right) endpoints of the first (last) two copies identified together, and the right endpoints of the first and third copies identified with the left endpoints of the second and fourth copies, respectively.

Assign all edges of G_k a cost of 1 and designate the rightmost vertex as the root vertex r. Prove that for every deterministic online Steiner tree algorithm A, there exists a sequence of terminals $s_1, \ldots, s_t$ in G_k such that the minimum-cost Steiner tree spanning r and $s_1, \ldots, s_t$ has cost 2^k, while the tree produced by A has cost $\Omega(k \cdot 2^k)$.

[Hint: make sure that all of the terminals lie on a shortest path between the left- and rightmost vertices of G_k. Define the terminals in "rounds" that track the inductive definition of G_k.]

Exercise 59

In this exercise you'll prove that the "MST heuristic" is a 2-approximation algorithm for the (offline) Steiner tree problem. Consider an undirected graph $G = (V, E)$ with a cost $c_e \geq 0$ for each edge, a root vertex $r \in V$, and terminals $S = \{s_1, \ldots, s_t\}$. Consider the algorithm:

1. Initialize $R = \{r\}$ and $T = \emptyset$.
2. While S is non-empty:
   (a) Let s_i denote a terminal of S with minimum distance (in G) to a vertex of R.
   (b) Let P denote a shortest path from s_i to T.
   (c) Add P to T.
   (d) Remove s_i from S and add it to R.

Now consider a complete graph H on the vertex set $\{r, s_1, s_2, \ldots, s_k\}$, where the cost of edge (u, v) in H is defined as the shortest-path distance between u and v in G. Prove that the cost of the tree (in G) produced by the algorithm above is at most the cost of a minimum spanning tree of H.

[Hint: consider relating the iterations of Prim's MST algorithm in H to the iterations of the above Steiner tree algorithm in G.]

Exercise 60

To complete the previous exercise, prove that the cost of an MST of H is at most twice the cost of an optimal Steiner tree in G.

[Hint: consider the minimum-cost Steiner tree of G, make a second copy of each of its edges, and consider an Eulerian tour of the resulting multi-graph. Relate the cost of this tour to that of a subgraph of H.]

Exercise 61

Exhibit a family of graphs $G = (V, E)$ (with edge costs and a root vertex) and distributions D over vertices such that, when t terminals $s_1, \ldots, s_t$ are drawn i.i.d. from D, the expected cost of the greedy online Steiner tree algorithm is more than a constant factor times that of the minimum-cost Steiner tree (as $|V|$ and t tend to infinity).

[Hint: consider line graphs, augmented with "shortcuts."]

Exercise 62

In lecture we gave an online Steiner tree algorithm that, given t i.i.d. samples $q_1, q_2, \ldots, q_t$ from a distribution D over terminals, incurs expected cost at most 4 times that of a minimum-cost Steiner tree on random instances with t terminals $s_1, \ldots, s_t$ drawn i.i.d. from D. Prove that if the number of samples s from D is less than the number t of terminals in a random instance, then the same algorithm produces a Steiner tree with expected cost at most $2\frac{t}{s}$ times that of a minimum-cost Steiner tree.

Exercise 63

Recall the secretary problem: an adversary chooses a set S of n arbitrary distinct numbers, and nature presents them as input one by one, with the order chosen uniformly at random. An online algorithm for the problem decides, at each time step, whether to stop or to continue to the next number. The goal is to maximize the probability of stopping on the largest of the n numbers.

Consider the algorithm that never picks any of the first $\frac{n}{e}$ numbers, and then picks the first subsequent number that is larger than any of the first $\frac{n}{e}$ numbers. (Assume $\frac{n}{e}$ is an integer, for simplicity of notation.) Explain why the probability that this algorithm correctly guesses the maximum number is

$$\frac{1}{n} \sum_{i=(n/e)+1}^{n} \frac{n}{e} \frac{1}{i-1} = \frac{1}{e} \sum_{i=(n/e)+1}^{n} \frac{1}{i-1}.$$

Approximate the sum with an integral to argue that this is roughly $\frac{1}{e}$ for large n.

Exercise 64

In this exercise you will prove that the acyclic subgraphs of an undirected graph form a matroid. Formally, let (V, E) be a graph and let $\mathcal{C} \subseteq 2^E$ denote the subsets $F \subseteq E$ such that (V, F) is acyclic. It is clear that if $S \in \mathcal{C}$ and $T \subseteq S$ then $T \in \mathcal{C}$. Prove the exchange property: if $S, T \in \mathcal{C}$ with $|T| < |S|$, then there exists an edge $e \in S \setminus T$ such that $T \cup \{e\} \in \mathcal{C}$.

Lecture 20 Exercises

Exercise 65

Recall the Follow the Leader (FTL) algorithm for online decision-making: on each day $t = 1, 2, \ldots, T$, pick the action with minimum cumulative cost $\sum_{s<t} c^s(a)$ so far. Assume that the algorithm breaks ties among actions lexicographically. Prove that FTL is *not* a no-regret algorithm — that there are arbitrarily long cost vector sequences (of length $T \to \infty$) for which the cost incurred by FTL is $\Omega(T)$ more than that by the best fixed action.

[Hint: two actions are enough.]

Exercise 66

Recall that we proved an upper bound of $O(\sqrt{T \log n})$ on the expected regret of the Perturbed Follow the Leader (PFTL) algorithm, where T is the time horizon and $n = |A|$ is the number of actions. It turns out that no online algorithm has better regret (modulo a constant factor). Prove a lower bound of $\Omega(\sqrt{T})$ for the case where $n = 2$.

[Hint: suppose an adversary randomizes uniformly between the cost vectors $(1, 0)$ and $(0, 1)$. What is the expected cost incurred by every online algorithm? What is the expected cost of the best fixed action in hindsight?]

Exercise 67

Recall the third claim in our analysis of the PFTL algorithm: if $X_1, \ldots, X_n$ are i.i.d. random variables, each a geometric random variable with parameter ϵ,[1] then $\mathbf{E}[\max_{i=1}^{n} X_i] = O(\frac{\log n}{\epsilon})$.

[Hint: this is a Union Bound and straightforward computations.]

Problems

Problem 32

The point of this problem is to prove that the greedy algorithm for the online Steiner tree problem achieves an approximation ratio of $O(\log t)$, where t is the number of terminals. Recall that the initial input is an undirected graph $G = (V, E)$ with a nonnegative cost $c_e \geq 0$ per edge, and a root vertex $r \in V$. Terminals $s_1, \ldots, s_t$ arrive online, and the algorithm is responsible for maintaining a tree that spans r and the terminals-so-far. The greedy algorithm, when presented with a new terminal s_i, augments the current solution T with a shortest path between s_i and a vertex already spanned by T.

(a) (7 points) Let $a_1, a_2, \ldots, a_t$ denote the costs incurred by the greedy algorithm over the t iterations, sorted so that $a_1 \geq a_2 \geq \cdots \geq a_t$. Define $S = \{r, s_1, \ldots, s_t\}$. Prove that for every $i = 1, 2, \ldots, t$, there is a set $S_i \subseteq \{r, s_1, \ldots, s_t\}$ such that every pair $u, v \in S_i$ of distinct vertices of S_i are at distance at least a_i apart in G.

(b) (5 points) Let OPT denote the cost of the minimum-cost Steiner tree spanning S. Prove that for every $i = 1, 2, \ldots, t$, $OPT \geq i a_i / 2$.

[Hint: compare to Exercise 60.]

(c) (3 points) Prove that OPT is at least an $\Omega(1/\log t)$ fraction of the cost incurred by the greedy algorithm.

Problem 33

(10 points) In the *k-secretary problem*, the online algorithm is allowed to pick up to k numbers. Give an online algorithm with the property that, for each of the k highest numbers x in the input, the algorithm chooses x with probability at least $\frac{1}{e}$.

[Hint: don't do any new calculations. Instead, give an algorithm for which the 1-secretary guarantee applies to each of the k highest elements in the input.]

Problem 34

(15 points) In the *graphical secretary problem*, a vertex set V is known up front and the edges E of a graph arrive online. Each arrives with a value v_e. An online algorithm either accepts or rejects an edge when it arrives, and must obey the constraint that the edges F accepted so far form an acyclic subgraph (V, F).

Give an online algorithm A with the following property: for every set E of edges and nonnegative edge values, if the edges are presented in an order chosen uniformly at random, then the expected (over orderings)

[1] I.e., the number of coin flips necessary to get "heads," where the probability of "heads" is ϵ.

total value of the acyclic subgraph output by A is at least a constant fraction of the maximum total value of an acyclic subgraph of (V, E).

Problem 35

(20 points) Recall from Exercise 65 that the Follow the Leader (FTL) algorithm for online decision-making has large regret in the worst case. The point of this problem is to prove that it has small expected regret in smoothed instances. While many perturbation models can be used to prove this, we use the following one to keep the proofs relatively simple.

1. An adversary picks a sequence of cost vectors $c^1, c^2, \ldots, c^T$, each a function from the action set A to $[0, 1]$.

2. Independently for each day t and each action a, nature subtracts 1 from $c^t(a)$ with probability δ, a positive constant (at most $\frac{1}{2}$, say).

Prove that the expected regret (over the perturbations) of the FTL algorithm grows sublinearly in T (i.e., $o(T)$ as $T \to \infty$).

[Hint: Adapt the analysis from Lecture #20. Your regret bound can be larger than $O(\sqrt{T \log n})$, as long as the dependence on T is sublinear.]