HW 6 (due Tuesday, at noon, October 21, 2014)

CS 473: Fundamental Algorithms, Fall 2014 Version: **1.1**

Make sure that you write the solutions for the problems on separate sheets of paper. Write your name and netid on each sheet.

Collaboration Policy: The homework can be worked in groups of up to 3 students each.

---

1. (30 PTS.) The IGSS rules.

Every year the famous inter-galactic surfing school (IGSS) elects their president using a somewhat bizarre process. They all go and travel to Planet Earth and try to padde from Monterey Bay Harbor to Mavericks at Half Moon Bay. The rule of the game is that they all start from the Monterey Bay Harbor and paddle for 5 hours, and then they stop (it is a competition, so each surfer probably reached a different location in the water). After that, the base station starts calling the surfers one after the other, and ask them for their location, and more significantly how close are they to Mavericks (the surfers have each a unique ID between 1 and n which they can use to call them and also carry water-proof cell phone with them). The surfer that is the closest to Mavericks is the new president. (This rule has been questioned by the IGSS and possibly in a next homework, the president would be determined only if they also manage to ride one of the 50-foot waves to shore.)

To avoid a vacuum in leadership, the president is being updated as the results come in. Specifically, if after contacting the first i surfers the last one contacted is the closest, then the base station sends n messages to all the surfers telling them that this surfer is the new president, where n is the number of surfers participating.

(A) (10 PTS.) Given that the given surfers are $v_1, \ldots, v_n$, describe a strategy that minimizes the overall number of messages sent. How many messages does you scheme sends in the worst case? If you decide on a randomized strategy, how many messages does your scheme sends in expectation? The smaller the number of messages, the better. Prove your answer.

(B) (10 PTS.) How many times does the president changes in the worst case (and also in expectation if your scheme is randomized) under your scheme (the fewer the better). Prove your answer.

(C) (10 PTS.) A new scheme was suggested that minimizes the number of messages sent: Whenever a new president is discovered, you announce it only to the surfers already contacted. Describe a scheme that orders the surfers, such that the number of messages sent is minimized. What is the number of messages sent by your scheme (either in the worst case, and in expectation if your scheme is randomized). The fewer the better.

For this question, if you have two bounds x and y, then if $x < y$ then x is preferable to y, even if x holds only in expectation, and y is a worst case bound.

2. (40 PTS.) My Kingdom for a Tree.

let G be a graph with m edges and n vertices with weights on the edges.

(A) (10 PTS.) You are given a minimum spanning tree T of G. The weight of one edge e of the graph had changed (the edge might be an edge of T). Describe an $O(n + m)$ time algorithm that computes the MST of G with the updated weights.

(B) (10 PTS.) You are given a minimum spanning tree T of G. For social reasons that are

still not well understood, Vogon children just broke into your house and stole k edges of T. Describe an algorithm to compute an MST for the graph G without these k edges. The running time of your algorithm should be $O(k \log k + m)$.

(C) (10 PTS.) You are given a minimum spanning tree T of G. The weight of k edges of G (that are not in T) had been suddenly decreased. Describe an $O((n + k) \log n)$ time algorithm that computes the MST of the new graph.

(D) (10 PTS.) You are given a graph G and a minimum spanning tree T. The ***max-price*** of a path π is the price of the most expensive edge on π. Describe an algorithm, as efficient as possible, for computing the minimum max-price path between two given vertices x and y of G. (For full credit, your algorithm should work in $O(n)$ time.)
Prove the correctness of your algorithm.

(And no, you can not use hashing in the solution for this question.)

3. (30 PTS.) Selection revisited.

You are given two arrays of numbers $X[1 \ldots n]$ and $Y[1 \ldots m]$, where n is smaller than m.

(A) (20 PTS.) Given a number k, and assuming both X and Y are sorted (say in increasing order), describe an algorithm, as fast as possible, for finding the k smallest number in the set $X \cup Y$ (assume all the numbers in X and Y are distinct).

(B) (10 PTS.) Solve the same problem for the case that X is not sorted, but Y is sorted. You algorithm should be faster than, $O(n \log n)$(this will be worth only IDK credit).