“CS 374” Spring 2015 — Homework 5

Due Tuesday, March 10, 2015 at 10am

• • • Some important course policies • • •

- **You may work in groups of up to three people.** However, each problem should be submitted by exactly one person, and the beginning of the homework should clearly state the names and NetIDs of each person contributing.
- **You may use any source at your disposal**—paper, electronic, or human—but you ***must*** cite ***every*** source that you use. See the academic integrity policies on the course web site for more details.
- **Submit your pdf solutions in Moodle.** See instructions on the course website and submit a separate pdf for each problem. Ideally, your solutions should be typeset in LaTex. If you hand write your homework make sure that the pdf scan is easy to read. Illegible scans will receive no points.
- **Avoid the Three Deadly Sins!** There are a few dangerous writing (and thinking) habits that will trigger an automatic zero on any homework or exam problem. Yes, we are completely serious.
  - Give complete solutions, not just examples.
  - Declare all your variables.
  - Never use weak induction.
- Unlike previous editions of this and other theory courses we are not using the “I don’t know” policy.

See the course web site for more information.

If you have any questions about these policies,
please don’t hesitate to ask in class, in office hours, or on Piazza.

---

1. Suppose you are given a directed graph $G = (V, E)$ with non-negative edge lengths; $\ell(e)$ is the length of $e \in E$. You are interested in the shortest path distance between two given locations/nodes s and t. It has been noticed that the existing shortest path distance between s and t in G is not satisfactory and there is a proposal to add exactly one edge to the graph to improve the situation. The candidate edges from which one has to be chosen is given by $E' = \{e_1, e_2, \ldots, e_k\}$ where $e_i = (u_i, v_i)$ for $1 \leq i \leq k$. You can assume that $E \cap E' = \emptyset$. The length of the e_i is $\alpha_i \geq 0$. Your goal is figure out which of these k edges will result in the most reduction in the shortest path distance from s to t. Describe an algorithm for this problem that runs in time $O(m + n \log n + k)$ where $m = |E|$ and $n = |V|$. Note that one can easily solve this problem in $O(k(m + n \log n))$ by running Dijkstra's algorithm k times, one for each G_i where G_i is the graph obtained by adding e_i to G. In other words, if you run Dijkstra's algorithm, you can only do it only a constant number of times where the constant does not depend on k.

2. You are given an array A with n distinct numbers in it, and another array B of ranks $1 \leq i_1 < i_2 < \ldots < i_k \leq n$. An element x of A has rank u if there are exactly $u - 1$ numbers in A smaller than it. Design an algorithm that outputs the k elements in A that have ranks $i_1, i_2, \ldots, i_k$.

   (A) Describe a $O(n \log k)$ recursive algorithm for this problem. Prove the bound on the running time of the algorithm. (As a warm up first obtain an $O(nk)$ time algorithm). Note that there is an easy $O(n \log n)$ time algorithm based on sorting A. However this is not as efficient as $O(n \log k)$ when $k \ll n$, say $k = O(\log n)$.

   (B) Show, that if this problem can be solved in $T(n, k)$ time, then one can sort n numbers in $O(n + T(n, n))$ time (i.e., give a reduction). What is an informal implication of this reduction in terms of being able to solve the problem in time faster than $O(n \log k)$ time.

   No proof of correctness necessary.

3. Suppose $p = (x_1, y_1)$ and $q = (x_2, y_2)$ are two points in the Euclidean plane. We say that p dominates q if $x_1 \geq x_2$ *and* $y_1 \geq y_2$. Note that domination is a partial order on points in Euclidean plane. Given a set $P = \{p_1, p_2, \ldots, p_n\}$ of points in the Euclidean plane we say $p_j \in P$ is undominated in P if there is no $p_i \in P, i \neq j$ such that p_i dominates p_j.

   - Draw a figure with at least 8 points and point out the undominated points.
   - Describe an algorithm that given set P of n points in the Euclidean plane (each point is specified by its two coordinates) outputs all the undominated points in P. Your algorithm should ideally run in $O(n \log n)$ time; note that an $O(n^2)$ time algorithm is trivial. You can assume without loss of generality that all points are in the positive quadrant of the plane (that is $p = (x, y)$ and $x, y \geq 0$).