COMPSCI 311: Introduction to Algorithms **Spring 2019**

Homework 3

Released 2/22/2019 Due 3/8/2019 11:59pm in Gradescope

Instructions. You may work in groups, but you must write solutions yourself. List collaborators on your submission.

If you are asked to design an algorithm, please provide: (a) the pseudocode or precise description in words of the algorithm, (b) an explanation of the intuition for the algorithm, (c) a proof of correctness, (d) the running time of your algorithm and (e) justification for your running time analysis.

Submissions. Please submit a PDF file. You may submit a scanned handwritten document, but a typed submission is preferred. Please assign pages to questions in Gradescope.

1. **(20 points) Recurrences.** Give asymptotic upper and lower bounds in the following recurrences. Assume $T(n)$ constant for $n \leq 2$. Make your bounds as tight as possible and justify your answers.

   (a) $T(n) = T(n/3) + T(n/2) + n$

   (b) $T(n) = T(n-1) + \log n$

   (c) $T(n) = 2T(n/2) + n/\log n$ Fact: $\sum_{k=1}^{n} 1/k = \Theta(\log n)$

2. **(20 points) Greedy Stays Ahead.** You fail to land a good internship for the summer so you end up working in the UMass mail room. The job is really boring. You stand at a conveyor belt and put mail items from the conveyor belt into boxes. It turns out that all of the mail is headed to the CS department! Each box has a fixed limit W on how much weight it can hold, and the items arrive on the conveyor belt one by one: the ith item that arrives has weight w_i. The rules of the job are really draconian: you must fill one box at a time and send it to the CS department before starting on the next box, and you must pack items into boxes in exactly the order they arrive on the conveyor belt. So, your only real decision is how many items to pack in each box before you send it off to the CS department.

   You decide to try a simple greedy algorithm: pack items into the current box in the order they arrive, and, whenever the next item does not fit, send the current box and start a new one.

   Is it possible that this will cause you to use more boxes than necessary? That is, could you decrease the overall number of boxes by packing one box less full, so that items somehow fit more efficiently into later boxes?

   Prove that, for a given set of items with specified weights, your greedy algorithm minimizes the number of boxes that are needed. Use a "greedy stays ahead" argument for your proof.

   Here is some notation and a few definitions to help formulate the problem precisely.

   - Assume the items are numbered $1, 2, \ldots, n$ and arrive in order, and that item i has weight w_i.
   - Let i_k be the number of items packed in the first k boxes by the greedy algorithm (equivalently, i_k is the number of the last item packed in box k).
   - Similarly, consider any optimal solution O and let j_k be the number of items packed in the first k boxes by O.

(a) (3 points) Create an example where you select a specific value for W, and make up weights for a sequence of n items that requires at least three boxes. Design your example so there are at least two different optimal solutions. Indicate the values $i_1, i_2, \ldots, i_p$ for the greedy solution, as well as the values $j_1, \ldots, j_q$ for a *different* optimal solution.

(b) (2 points) Write down an inequality that is always true for the quantities i_1 and j_1, and explain your reasoning.

(c) (2 points) Formulate a "claim": an inequality comparing i_k and j_k that is true for $k \geq 1$, which you will prove by induction.

(d) (10 points) Prove your claim by induction.

(e) (3 points) Now argue that your claim implies that the greedy algorithm is optimal, i.e., that $p = q$ where p is the number of boxes used by the greedy algorithm and q is the number of boxes used by the optimal solution.

3. **(20 points) Database Medians.** You are working as a programmer for UMass administration, and they ask you to determine the median GPA for all students. However, student GPAs are stored in two different databases, one for in-state students and one for out-of-state students. Assume there are n students of each type, so there are $2n$ students total. You'd like to determine the median of this set of $2n$ values, **which we will define here to be the nth smallest value.** (Note: this is *the definition* of the median for the purposes of this problem. Consider it carefully and ignore other definitions you may know.)

   However, security is very tight, so the only way you can access these values is through queries to the databases. In a single query, you can specify a value k to one of the two databases, and the chosen database will return the kth smallest value that it contains. Since queries are expensive, you would like to compute the median using as few queries as possible.

   Give an algorithm that finds the median value using at most $O(\log n)$ queries.

4. **(10 points) Spanning Trees.** Consider the problem of designing a spanning tree for which the *most expensive edge* (as opposed to the total edge cost) is as cheap as possible. Let $G = (V, E)$ be a connected graph with n vertices, m edges, and positive edge costs that are all distinct. Let $T = (V, E')$ be a spanning tree of G; we define the bottleneck edge of T to be the edge of T with the greatest cost.

   A spanning tree T of G is a minimum-bottleneck spanning tree if there is no spanning tree T' of G with a cheaper bottleneck edge.

   (a) Is every minimum-bottleneck tree a minimum spanning tree of G? Prove or give a counterexample.

   (b) Is every minimum spanning tree a minimum-bottleneck tree of G? Prove or give a counterexample.

5. **(10 points) More Spanning Trees.** Show that a graph has a unique minimum spanning tree if, for every cut of the graph, there is a unique cheapest edge crossing the cut. Show that the converse is not true by giving a counterexample.

6. **(20 points) Shortest Road Trip.** Mary and Tom are taking a road trip from New York City to San Francisco. Because it is a 44-hour drive, Mary and Tom decide to switch off driving at each rest stop they visit. However, because Mary has a better sense of direction than Tom, she should be driving both when they depart and when they arrive (to navigate the city streets). Given a route map represented as a weighted directed graph $G = (V, E)$ with positive edge lengths (the length of edge e is $\ell_e > 0$), where vertices represent rest stops and edges represent routes between rest stops, devise an efficient algorithm to find a route (if possible) of minimum distance between New York City and San Francisco such that Mary and Tom alternate edges and Mary drives the first and last edge.

   Hint: think how to modify either the shortest path computation *or* the problem input to account for alternating driving.

7. **(0 points)** Have you provided all 5 items (mentioned at the beginning of this assignment) for all your algorithm designs?

8. **(0 points)** How long did it take you to complete this assignment?