CS 161 Final Exam – Autumn 2016-17

(Do not turn this page until you are instructed to do so!)

Instructions: Please answer the following questions to the best of your ability. Provide full and rigorous proofs and include all relevant calculations unless the question says otherwise. When writing proofs, please strive for clarity and brevity (in that order). You have **180 minutes** to complete the exam. The exam is closed-book, closed-note, closed-internet, etc. However, you may use two letter-sized sheets (front and back) of notes as reference.

All of the intended answers can be written well within the space provided. Do not use the back side of printed pages for your answers. We have provided one blank page at the end for your rough work. You may also use the back side of printed pages for rough work. Good luck!

The following is a statement of the Stanford University Honor Code:

1. *The Honor Code is an undertaking of the students, individually and collectively:*

   (1) that they will not give or receive aid in examinations; that they will not give or receive unpermitted aid in class work, in the preparation of reports, or in any other work that is to be used by the instructor as the basis of grading;

   (2) that they will do their share and take an active part in seeing to it that others as well as themselves uphold the spirit and letter of the Honor Code.

2. *The faculty on its part manifests its confidence in the honor of its students by refraining from proctoring examinations and from taking unusual and unreasonable precautions to prevent the forms of dishonesty mentioned above. The faculty will also avoid, as far as practicable, academic procedures that create temptations to violate the Honor Code.*

3. *While the faculty alone has the right and obligation to set academic requirements, the students and faculty will work together to establish optimal conditions for honorable academic work.*

By signing your name below, you acknowledge that you have abided by the Stanford Honor Code while taking this exam.

Signature: _____

Name: _____

SUNetID: _____

Problem	#1	#2	#3	#4	#5	#6	#7	Total
Maximum	20	15	15	20	30	30	30	160

Question 1: Recurrences (20 points)

Solve the recurrences below giving tight upper bounds of the form $T(n) = O(f(n))$ for an appropriate function f. You do *not* need to prove that the upper bounds are tight or give lower bounds ($T(n) = \Omega(f(n))$). You can use any method from class. Show your work. If you wish, you may assume that n initially has the form $n = a^i$, for an appropriate constant a. Each recurrence is worth 10 points.
Note: log refers to log base 2.

(a) $T(n) = 2T(n/2) + n^k$ where $k > 0$ is a constant.

Hint: Consider cases depending on the value of k.

(b) $T(n) = 2T(n/2) + \frac{n}{\log n}$.

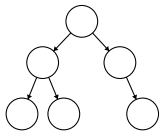
Hint: Draw the recursion tree.

Question 2: Binary Search Trees (15 points)

You are given a binary tree with n nodes and a set of n distinct keys (numbers).

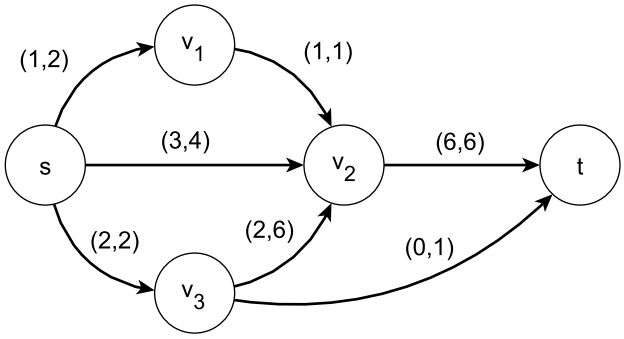
Prove or disprove: There is exactly one way to assign keys to nodes such that the resulting tree is a valid binary search tree.

Example: You are given the binary tree drawn on the right and the set of keys $\{1, 2, 3, 4, 5, 6\}$. The question asks whether there is exactly one way to assign the keys to nodes such that the tree will be a binary search tree. (If you prove the statement, it should be for any input and not just this example.)



Question 3: Maximum Flow (15 points)

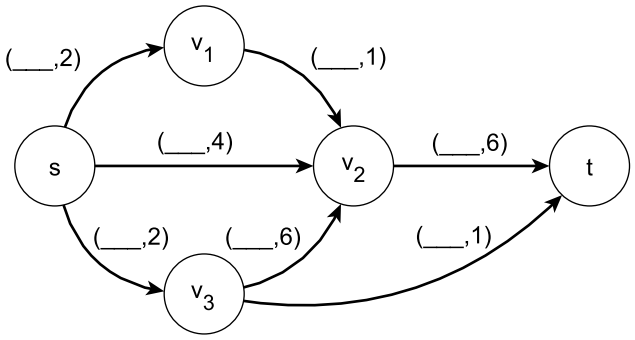
Consider the following flow network.



The figure describes a flow f and the capacity of the edges: if (x, y) appears next to an edge e, then the capacity of the edge c_e is y, and the flow f_e that goes through e in f is x. For example, if $e = (s, v_1)$, then $c_e = 2$ and $f_e = 1$.

(a) (5 points) Find the residual network G_f with respect to f (draw a graph containing all the nodes, edges, and the residual capacities). No explanation necessary.

(b) (5 points) Find an augmenting path that will increase the flow by 1. You only need to list the vertices in the path and indicate the resulting flow in the following figure (using the same notation as the figure on the previous page). No explanation necessary.



(c) (5 points) Find a minimum s-t cut in the graph (where the weight of an edge is its capacity). Briefly justify why the cut you found is a minimum cut.

Question 4: Minimum Spanning Trees (20 points)

Suppose we have weighted graphs $G_1 = (V, E)$ and $G_2 = (V, E)$ over the same set of nodes and edges, but with different weight functions on the edges: $w_1(e)$ is the weight of edge e in G_1 and $w_2(e)$ is the weight of edge e in G_2. Let $G_3 = (V, E)$ be another graph on the same set of nodes and edges, with weight $w_3(e) = w_1(e) + w_2(e)$ for every $e \in E$. In other words, the weight of an edge in G_3 is the sum of the weights of the corresponding edges in G_1 and G_2.

Let T_1, T_2, and T_3 be the minimum spanning trees of G_1, G_2, and G_3, respectively. We define the cost of T_i to be $cost(T_i) = \sum_{e \in E(T_i)} w_i(e)$ where $E(T_i)$ is the set of edges in T_i.

(a) (10 points) Prove that $cost(T_1) + cost(T_2) \leq cost(T_3)$.

(b) (10 points) Show an example where the inequality of part (a) is strict. In particular, draw an example of a connected graph on 3 nodes and two weight functions w_1 and w_2, and show that $cost(T_1) + cost(T_2) < cost(T_3)$.

Question 5: Matching Points on a Line (30 points)

Input: Two arrays of n points on the number line: red points $r_1, r_2, \ldots, r_n \in \mathbb{R}$ and blue points $b_1, b_2, \ldots, b_n \in \mathbb{R}$. You may assume that all red points are distinct and all blue points are distinct.
Task: Pair up red and blue points, i.e., find a bijection $\pi : \{1, \ldots n\} \to \{1, \ldots n\}$ so that r_i is matched with $b_{\pi(i)}$. Note that each red point is matched with a unique blue point and vice versa.
Objective: Minimize $\sum_{i=1}^{n} |r_i - b_{\pi(i)}|$

Design an $O(n \log n)$ time algorithm that finds the matching (bijection) that minimizes the sum of distances between the matched points. Prove the correctness of the algorithm and analyze its running time. *Suggestion*: Experiment with examples for small values of n. Drawing figures is helpful for understanding the problem.

Example: Consider the input where the red points are $r_1 = 8, r_2 = 1$ and the blue points are $b_1 = 3, b_2 = 9$. There are two possible matchings:

1. Matching r_1 to b_1 and r_2 to b_2. The cost is $|8 - 3| + |1 - 9| = 13$.
2. Matching r_1 to b_2 and r_2 to b_1. The cost is $|8 - 9| + |1 - 3| = 3$.

The algorithm will return the second matching.

(additional space for solution to question 5)

Question 6: Clustering (30 points)

In this problem, we consider a problem about clustering n points on the number line into k clusters.
Input: n distinct points $x_1, \ldots, x_n \in \mathbb{R}$ in sorted order, a parameter $k \leq n$
Task: Divide the n points into k disjoint non-empty clusters $S_1, \ldots, S_k$ such that $\bigcup_{i=1}^{k} S_i = \{x_1, \ldots, x_n\}$ and all points in S_i are to the left of all points in S_{i+1} for $1 \leq i < k$, i.e., for any $y \in S_i, z \in S_{i+1}$, $y < z$.
Objective: Minimize $\sum_{i=1}^{k} \text{cost}(S_i)$
where $\text{cost}(S_i) = (\max(S_i) - \min(S_i))^2$.

Note that $\min(S_i)$ is the minimum element of S_i, i.e., the leftmost point of S_i. Similarly, $\max(S_i)$ is the maximum element of S_i, i.e., the rightmost point of S_i.

For example, if $S_i = \{x_j\}$, then $\text{cost}(S_i) = 0$. If $S_i = \{x_j, x_{j+1}, \ldots, x_{j+t}\}$, $x_j < x_{j+1} < \ldots x_{j+t}$, then $\text{cost}(S_i) = (x_{j+t} - x_j)^2$.

Design an $O(n^2 k)$ time algorithm to find the optimal clustering using dynamic programming. Briefly argue correctness and analyze running time.

Example: Consider clustering 4 points $\{1, 5, 8, 10\}$ into 2 clusters. There are three options:
1. $S_1 = \{1\}, S_2 = \{5, 8, 10\}$, with total cost $0^2 + 5^2 = 25$.
2. $S_1 = \{1, 5\}, S_2 = \{8, 10\}$, with total cost $4^2 + 2^2 = 20$.
3. $S_1 = \{1, 5, 8\}, S_2 = \{10\}$, with total cost $7^2 + 0^2 = 49$.

The output of the algorithm is the optimal solution $S_1 = \{1, 5\}, S_2 = \{8, 10\}$.

(additional space for solution to question 6)

Question 7: Strenuous Trail (30 points)

You are trying to find the most strenuous hiking trail between two points: the path with the most elevation gain or loss per mile. We model this problem as follows: You are given a connected undirected graph $G = (V, E)$ with two designated nodes $a, b \in V$, where each node v is associated with a height $h(v) \in \mathbb{R}$ and each edge e is associated with a positive length $w(e)$. In addition, $h(a) = h(b) = 0$.

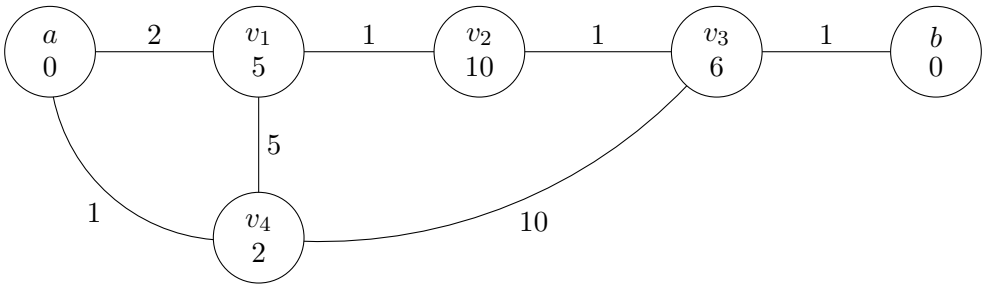
We say that a path from a to b is an *up-and-down* path if there is some node v in the path such that all the nodes from a to v have increasing heights, and all the nodes from v to b have decreasing heights. (Figuratively, think about climbing a hill where v is the top of the hill.)

Formally, let $p = (v_1, v_2, \ldots, v_{k-1}, v_k)$ be a path. The path p is up-and-down if there is some vertex v_i $(1 < i < k)$, such that for all $1 \leq j < i$, $h(v_j) < h(v_{j+1})$ and for all $i \leq j < k$, $h(v_j) > h(v_{j+1})$. The path may not be a simple path and can visit vertices multiple times (technically this is called a walk). We introduce the following definitions:

- The length of p is the sum of the lengths of the edges of the path: $\text{length}(p) = \sum_{i=1}^{k-1} w((v_i, v_{i+1}))$.
- The elevation of p is defined as the maximum height of a node in the path: $\text{elevation}(p) = \max_{1 \leq i \leq k} h(v_i)$.
- The difficulty of p is $\text{difficulty}(p) = \frac{\text{elevation}(p)}{\text{length}(p)}$.

Design an $O(|E| + |V| \log |V|)$ time algorithm that finds an up-and-down path from a to b that has the maximum difficulty or reports that there is no up-and-down path from a to b. Briefly argue correctness and analyze running time.

Example: Consider the following graph.



The number inside each node is its height (for example, $h(v_1) = 5$), and the number next to each edge is its length (for example, $w((s, v_1)) = 2$).

The output of the algorithm is the most difficult up-and-down path from a to b, which is (a, v_1, v_2, v_3, b). The elevation of the path is 10, its length is 5, and its difficulty is 2. The path (a, v_4, v_3, b) is another up-and-down path from a to b that has difficulty $1/2$. The path (a, v_1, v_4, v_3, b) is not an up-and-down path.

Hint: If you are looking for an a-b path of maximum difficulty where the vertex with the highest height is v, how could you find the portion of the path from a to v and the portion of the path from v to b?

(space for solution to question 7)

(additional space for solution to question 7 if needed)

This page left blank for your rough work. **Anything written here will not be graded**.