CS 180: Introduction to Algorithms and Complexity
Midterm Exam

May 6, 2020

Name	
UID	
Section	

1	2	3	4	5	Total

★ **Print your name, UID and section number in the boxes above, and print your name at the top of every page**.

★ **Your Exams need to be uploaded in Gradescope. Use Dark pen or pencil. Handwriting should be clear and legible.**

- There are 5 problems.
- Do not write code using C or some programming language. Use English or clear and simple pseudo-code. Explain the idea of your algorithm and why it works.
- Your answers are supposed to be in a simple and understandable manner. Sloppy answers are expected to receive fewer points.
- Don't spend too much time on any single problem. If you get stuck, move on to something else and come back later.

1. For each of the following problems answer True or False and briefly justify you answer.

   (a) (5pt) For a connected and undirected graph G, if removing edge e disconnects the graph, then e is a tree edge in DFS of G.

   (b) (5pt) For a DAG G, if there is only one node with no incoming edge, then there exists only one topological ordering.

   (c) (5pt) For the stable matching problem, if there is a man m_1 and woman w_1 such that w_1 has the lowest ranking in m_1's preference list and m_1 has the lowest ranking in w_1's preference list, then any stable matching will not contain the pair (m_1, w_1).

   (d) (5pt) If we run DFS on a DAG and node u is the first leaf node in the DFS tree, then u has no outgoing edge.

2. (10pt) Take the following list of functions and arrange them in ascending order of growth rate. That is, if function $g(n)$ immediately follows function $f(n)$ in your list, then it should be the case that $f(n)$ is $O(g(n))$.

   - $f_1(n) = 3n^3$
   - $f_2(n) = n(\log n)^{100}$
   - $f_3(n) = 2^{n \log n}$
   - $f_4(n) = 2^{\sqrt{n}}$
   - $f_5(n) = 2^{0.8 \log n}$

3. (20pt) For a DAG with n nodes and m edges (and assume $m \geq n$), design an algorithm to test if there is a path that visits every node exactly once. The algorithm should run in $O(m)$ time.

4. (20pt) Given an array A of n distinct integers and assume they are sorted in increasing order. Design an algorithm to find whether there is an index i with $A[i] = i$. The algorithm should run in $O(\log n)$ time.

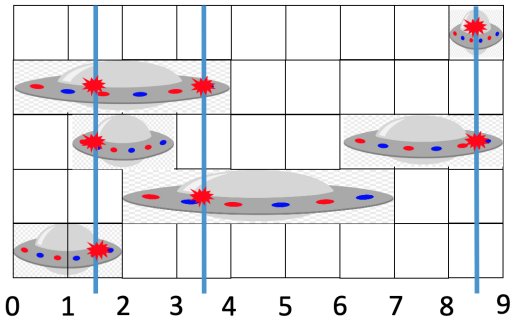

Figure 1: In this example, there are 6 flying saucers with $(L_1, R_1) = (0, 2)$, $(L_2, R_2) = (2, 7)$, $(L_3, R_3) = (1, 3)$, $(L_4, R_4) = (6, 9)$, $(L_5, R_5) = (0, 4)$, $(L_6, R_6) = (8, 9)$. We need at least 3 laser canons to destroy all of them, and $1.5, 3.5, 8.5$ is a set of valid positions of these laser canons.

5. (30pt) There are several flying saucers on the sky to attack the Earth. For simplicity, we assume Earth surface is 1-D and the flying saucers are on the sky, as shown in Figure 1. We know there are n flying saucers and each of them occupies the open interval (L_i, R_i) (assume L_i, R_i are integers). To destroy those flying saucers, we are going to fire the laser canon at some locations. If the laser canon is fired at position x to the sky, it will destroy all the saucers that intersects with this vertical line, i.e., all the flying saucers with $x \in (L_i, R_i)$ will be destroyed, as illustrated in Figure 1. However, firing the laser canon is expensive so we want to find a way to destroy all the flying saucers using as few laser canons as possible.

   Mathematically, given n intervals $\{(L_i, R_i) \mid i = 1, \ldots, n\}$, our goal is to find a minimum set of numbers $X = \{x_1, \ldots, x_k\}$ such that for every interval i, there is at least one x_j in X contained in the interval $(L_i < x_j < R_i)$. Give an $O(n \log n)$ time algorithm to solve this problem, and prove the correctness of your algorithm.