CS 344: Design and Analysis of Computer Algorithms **Rutgers: Fall 2019**

Practice Midterm Exam #2

Name: _____ *NetID:* _____

Instructions

1. Do not forget to write your name and NetID above.
2. The exam contains 4 problems worth 100 points in total *plus* one extra credit problem worth 10 points. You have 75 minutes to finish the exam. The exam is closed-book and closed notes.
3. **Note that problems appear on both odd and even numbered pages.** There should be more than enough space to write down your solution for each problem below the problem itself. But if you ran out of space, you can also use the extra sheet at the end of the exam; if you do so, be clear about which problem you are solving.
4. Remember that you can leave a problem (or parts of it) entirely blank and receive 25% of the grade for that problem (or part). However, this should not discourage you from attempting a problem if you think you know how to approach it as you will receive partial credit more than 25% if you are on the right track. But keep in mind that if you simply do not know the answer, writing a very wrong answer may lead to 0% credit.

   The only **exception** to this rule is the extra credit problem: you do not get any credit for leaving the extra credit problem blank, and it is harder to get partial credit on that problem.
5. **You should always prove the correctness of your algorithm and analyze its runtime.** Also, as a general rule, avoid using complicated pseudo-code and instead explain your algorithm in English.
6. You may use any algorithm presented in the class as a building block for your solutions.

Suggestion: Leave the extra credit problem for last as it is harder than the rest and worths fewer points.

Problem. #	Points	Score
1	25	
2	25	
3	25	
4	25	
5	+10	
Total	100 + 10	

Problem 1. Prove or disprove the following assertions.

(a) Suppose $G(V, E)$ is a directed acyclic graph (DAG) with *two* sources s_1, s_2 and *two* sinks t_1, t_2. If we add a directed edge from any arbitrary sink to any arbitrary source, then the new graph will always contain a cycle. **(5 points)**

(b) Suppose $G(V, E)$ is an undirected graph and $(S, V - S)$ is a cut with no cut edges in G. Suppose we add an edge $e = \{u, v\}$ for $u \in S$ and $v \in V - S$ to G; then edge e *cannot* belong to any cycle in G.

(10 points)

(c) Suppose $G(V, E)$ is an undirected connected graph with *distinct* weight w_e for each edge $e \in E$, i.e., $w_e \neq w_{e'}$ for any two edges $e \neq e' \in E$. Then G has a *unique* minimum spanning tree (MST).

(10 points)

Problem 2. You are given a set of n (closed) intervals on a line:

$$[a_1, b_1], [a_2, b_2], ..., [a_n, b_n].$$

Design an $O(n \log n)$ time *greedy* algorithm to select the *minimum* number of intervals whose union is the same as the union of all intervals. You may assume that the union of intervals is equal to $[\min_i a_i, \max_j b_j]$.

Example: If the following 4 intervals are given to you:

$$[2, 5], [3, 9], [2.5, 9.5], [4, 8],$$

then a correct answer is: $[2, 5], [2.5, 9.5]$. **(25 points)**

Problem 3. Suppose you are given a directed acyclic graph (DAG) $G(V, E)$, a source vertex s, and a sink vertex t (these two are *not* the only sources or sinks in G). We say that a vertex $v \in V - \{s, t\}$ is an (s, t)-*independent vertex* in G if there is *no* path in G from s to t that passes through v. Design an $O(n+m)$ time algorithm for finding all (s, t)-independent vertices in G. **(25 points)**

Problem 4. Consider the following different (and less efficient) algorithm for computing an MST of a given undirected and connected graph $G(V, E)$ with edge weight w_e on each $e \in E$:

1. Sort the edges in decreasing (non-increasing) order of their weights.
2. Let $H = G$ be a copy of the graph G.
3. For $i = 1$ to m (in the sorted ordering of edges):
   (a) If removing e_i from H does not make H disconnected, remove e_i from H.
4. Return H as a minimum spanning tree of G.

Our goal in this question is to prove the correctness of this algorithm, i.e., that it outputs an MST of any given graph G (we ignore the runtime of this algorithm in this problem).

(a) Prove that in any graph $G(V, E)$, if an edge $e \in E$ has the largest weight among edges of some cycle in G, then there exists an MST of G that does *not* contain the edge e. **(12.5 points)**

(b) Use Part (a) to argue the correctness of the above algorithm. **(12.5 points)**

Problem 5. [**Extra credit**] You are given a set of n cities with R roads and H highways between different cities, a starting city s, and a destination city t. Your goal is to go from city s to city t by using the minimum number of roads. You are also allowed to take any highway but that requires paying a toll and you only have enough money to pay at most one toll. Design an $O(n + R + H)$ time algorithm that finds the set of roads and the single highway (if any) you should take. You may assume that there is always a way to get from s to t, all roads and highways are one-way, and the best solution always involve taking a highway.

(+10 points)

Extra Workspace