CS 344: Design and Analysis of Computer Algorithms **Rutgers: Spring 2021**

Final Exam

Due: Monday, May 10, 1:00pm EST

Name: _____ *NetID:* _____

Instructions

1. Do not forget to write your name and NetID above, and to sign Rutgers honor pledge below.
2. The exam contains 4 problems worth 100 points in total *plus* one extra credit problem worth 10 points.
3. This is a take-home exam. You have exactly 48 hours to finish the exam.
4. The exam should be done **individually** and you are not allowed to discuss these questions with anyone else. This includes asking any questions or clarifications regarding the exam from other students or posting them publicly on Piazza (any inquiry should be sent directly to the Instructor or posted privately on Piazza). You may however consult all the materials used in this course (video lectures, notes, textbook, etc.) while writing your solution, but **no other resources are allowed**.
5. 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.
6. **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.
7. You may use any algorithm presented in the class or homeworks as a building block for your solutions.

Rutgers honor pledge:

On my honor, I have neither received nor given any unauthorized assistance on this examination.

Signature:_____

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

Problem 1.

(a) Mark each of the assertions below as True or False and provide a short justification for your answer.

(i) If $f(n) = 2^{\sqrt{\log n}}$ and $g(n) = n$, then $f(n) = \Omega(g(n))$. **(2.5 points)**

(ii) If $T(n) = T(n/3) + T(n/4) + O(n)$, then $T(n) = O(n)$. **(2.5 points)**

(iii) If P = NP, then all NP-complete problems can be solved in polynomial time. **(2.5 points)**

(iv) If P $\neq$ NP, then no problem in NP can be solved in polynomial time. **(2.5 points)**

(b) Prove the following statements.

(i) Suppose G is a directed acyclic graph (DAG) with a unique source s. Then, there is a path from s to v for any vertex v in G. **(7.5 points)**

(ii) Consider a flow network G and a flow f in G. Suppose there is a path from the source to sink such that $f(e) < c_e$ for all edges of the path, i.e., the flow on each edge is strictly less than its capacity. Then, f is *not* a maximum flow in G. **(7.5 points)**

Problem 2. We consider a different variant of the Knapsack problem in this question. You are given n items with integer weights $w_1, \ldots, w_n$ and integer values $v_1, \ldots, v_n$ and a target value V. Your goal is to determine the *smallest* knapsack size needed so that you can fit a set items in the knapsack with total value at least V. In other words, you want to *minimize* $\sum_{i \in S} w_i$ subject to $\sum_{i \in S} v_i \geq V$ (over the choice of S from n items).

Design an $O(n \cdot V)$ time dynamic programming algorithm for this problem.

(a) *Specification of recursive formula for the problem (in plain English)*: **(5 points)**

(b) *Recursive solution for the formula:* **(7.5 points)**

(c) *Proof of correctness of the recursive formula:* **(7.5 points)**

(d) *Runtime analysis:* **(5 points)**

Problem 3. You are given a directed graph $G = (V, E)$ such that every *edge* is colored red, yellow, or green, and two vertices s and t. We say that a path from s to t is a *good* path if (1) it has *at least one* edge of each color, and (2) all the red edges in the path appear before all the yellow edges, and all the yellow edges appear before the green edges. For instance, a $(red, red, yelow, green, green, green)$ path is a good path but neither a $(red, green, green)$ path nor a $(red, green, yellow)$ path are good.

Design and analyze an $O((m + n) \cdot n)$ time algorithm that outputs the size of the *largest* collection of *edge-disjoint good* paths from s to t in a given directed graph $G = (V, E)$ with n vertices and m edges.

(25 points)

Problem 4. Prove that the following problems are NP-hard. For each problem, you are only allowed to use a reduction from the problem specified.

(a) **4-Coloring Problem:** Given an undirected graph $G = (V, E)$, is there a 4-coloring of vertices of G? (A 4-coloring is an assignment of colors $\{1, 2, 3, 4\}$ to vertices so that no edge gets the same color on both its endpoints). **(12.5 points)**

For this problem, use a reduction from the 3-*Coloring problem*. Recall that in the 3-Coloring problem, you are given a graph $G = (V, E)$ and the goal is to find whether there is a 3-coloring of G or not. A 3-coloring is an assignment of colors $\{1, 2, 3\}$ to vertices so that no edge gets the same color on both its endpoints

(b) **Hamiltonian Path Problem:** Given an undirected graph $G = (V, E)$, does G contain a path that goes through all vertices, i.e., a Hamiltonian path? **(12.5 points)**

For this problem, use a reduction from the *s-t Hamiltonian Path problem*. Recall that in the *s-t* Hamiltonian Path problem, you are given a graph $G = (V, E)$ and two vertices s, t and the goal is to decide whether there is a *s-t* path in G that passes through all other vertices.

(Note that the difference between Hamiltonian Path problem and *s-t* Hamiltonian Path problem is that in the former problem, the path can start from any vertex and end in any vertex as long as it goes through all vertices, while in the latter it should start from s and ends at t.)

Problem 5. [**Extra credit**] Alice wants to throw a party and is deciding who to invite. She has n people to choose from and she has made up a list of which pairs of these people know each other. She wants to pick as many people as possible, subject to the constraint that at the party, each person should know at least five other people.

Give a polynomial time algorithm that takes as input a list of n people and the list of pairs who know each other and outputs the maximum number of guests that Alice can invite.

(+10 points)

Hint: Get creative and design an algorithm for this problem from scratch; this problem is *not* about using reductions to the problems you have already seen in this course.

Extra Workspace

Extra Workspace

Extra Workspace