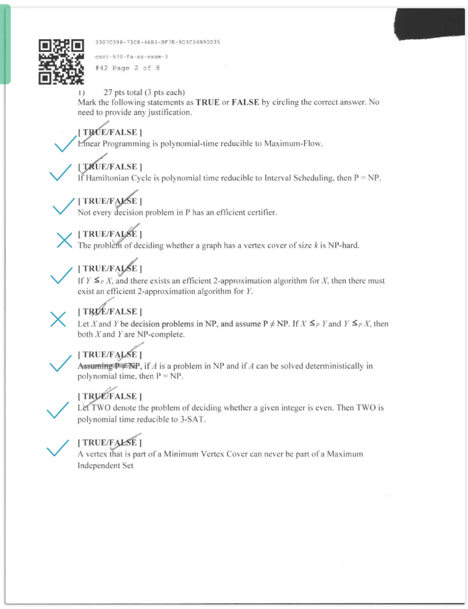
Q1 21



[redacted]

3307C398-73C8-46B3-BF7E-9D3C34B90D35

csci-570-fa-ss-exam-3

#42 Page 2 of 8

1) 27 pts total (3 pts each)
Mark the following statements as **TRUE** or **FALSE** by circling the correct answer. No need to provide any justification.

[TRUE/FALSE]
Linear Programming is polynomial-time reducible to Maximum-Flow.

[TRUE/FALSE]
If Hamiltonian Cycle is polynomial time reducible to Interval Scheduling, then P = NP.

[TRUE/FALSE]
Not every decision problem in P has an efficient certifier.

[TRUE/FALSE]
The problem of deciding whether a graph has a vertex cover of size k is NP-hard.

[TRUE/FALSE]
If $Y \leq_P X$, and there exists an efficient 2-approximation algorithm for X, then there must exist an efficient 2-approximation algorithm for Y.

[TRUE/FALSE]
Let X and Y be decision problems in NP, and assume P $\neq$ NP. If $X \leq_P Y$ and $Y \leq_P X$, then both X and Y are NP-complete.

[TRUE/FALSE]
~~Assuming P ≠ NP~~, if A is a problem in NP and if A can be solved deterministically in polynomial time, then P = NP.

[TRUE/FALSE]
Let TWO denote the problem of deciding whether a given integer is even. Then TWO is polynomial time reducible to 3-SAT.

[TRUE/FALSE]
A vertex that is part of a Minimum Vertex Cover can never be part of a Maximum Independent Set

Q2 8

[redacted]

30D0F505-867A-45B5-A47E-D14906F57F31

2) 16 pts

Circle ALL correct answers and only correct answers (no partial credit when missing some of the correct answers). No need to provide any justification.

I. If there exists a strongly polynomial-time algorithm for the decision version of the Traveling Salesman problem, which of the following statements is necessarily true? (4 pts)

a) There exists a strongly polynomial-time algorithm for the decision version of 0/1 Knapsack.
b) NP-hard problems are at least as hard as problems in P.
c) P $\neq$ NP.
d) All decision problems can be solved efficiently.

II. Consider an undirected graph G. Which of the following statements are true? (4 pts)

a) The problem of determining whether there exists a simple cycle in G is in NP.
b) The problem of determining whether there exists a simple cycle of length k in G is in NP-complete.
c) The problem of determining whether there exists a simple cycle that visits all nodes in G is in NP-complete.
d) The problem of determining whether there exists a simple cycle that visits all nodes in G is in NP-hard.

III. Which of the following statements are known to be true? (4 pts)

a) It has been proven that P $\neq$ NP
b) There are problems in NP that cannot be solved in polynomial time.
c) There are problems in NP that can be solved in polynomial time.
d) There are NP-complete problems that cannot be solved in polynomial time.

IV. Given a directed graph G, two distinct nodes S and T in G, and a number k, which of the following decision problems are NP-complete? (4 pts)

a) Determine whether there exists a simple path of length at most k from S to T.
b) Determine whether there exists a simple path of length at least k from S to T.
c) Determine whether there exist at least k simple edge-disjoint paths from S to T.
d) Determine whether there exist at least k simple node-disjoint paths from S to T.

Q3 0

[redacted]

71F71587-7FB9-4C73-AB61-6CAD2299AE28

csci-570-fa-ss-exam-3

#42 Page 4 of 8

3) 17 pts

A chef is cooking meals for a number of food critics. The chef keeps n different ingredients in the kitchen, where n is a positive integer. Let a_i be the amount of ingredient i the chef has stored in the kitchen measured in grams, where a_i is a positive integer and $1 \le i \le n$. The chef knows how to cook m different recipes. Each recipe requires a different amount of each ingredient. Let b_{ij} be the amount of ingredient i required to cook recipe j measured in grams, where b_{ij} is a non-negative integer and $1 \le i \le n$ and $1 \le j \le m$. In order to maximize the variety of meals presented to the critics, the chef will make at most one of each recipe.

Consider the following problem:

Given the number of available ingredients n, *the number of possible recipes* m, *the available ingredient amounts* a_i $(1 \le i \le n)$, *and the recipe requirements* b_{ij} $(1 \le i \le n, 1 \le j \le m)$, *what is the maximum number of meals the chef can make without making the same meal more than once?*

Follow the steps below to express this problem as an instance of Integer Linear Programming.

a) What are the variables used by your integer linear program? What are the domains of your variables, i.e., what are the possible values for your variables? In plain English, describe what your variables represent. (6 pts)

The decision variables would be x_{ij} (boolean)

$x_{ij} = 1$ if ingredient a_i is used in recipe j

Wrong variable. Need a variable that tells if the chef cooks recipe j or not.

$x_{ij} \le 1$ for all

$\sum_{i}^{n} x_{ij} \le a_i$ for al

mize the num of meals.

Q4 **4**

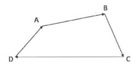
[redacted]

A0C3E4E2-1DC5-4885-821C-3752D23B4A36
csci-570-fa-ss-exam-3
#42 Page 5 of 8

4) 20 pts

Consider the following problem: Given a directed graph G, remove some edges to turn G into a Directed Acyclic Graph (DAG) of maximum size (i.e. with maximum number of edges).

For example, for the following graph, removing any one of the edges, e.g. AB will result in a maximum size DAG of size 3.



Our goal is to find a ½-approximation to this problem. In other words, we want to end up with a DAG that includes at least ½ the number of edges in the optimal DAG.

a) Describe an algorithm to achieve a ½ approximation for this problem (10 pts)
Hint 1: Remember topological ordering of nodes in a DAG
Hint 2: As a first try, see if a random ordering of the nodes can help yo… identify which edges to remove in order to achieve this approximation ratio.

① Given graph G, we randomly remove any … edges. We then check to see if there … node which has no incoming edge. If n… will remove more edges until we ge… has no incoming edges. ② We might al… edges and see if there exists a node w… ③ We might also order the nodes, as they… and remove first half of the edges, … with no incoming edge else rem…

… have only have the no. of edges … of the $DAG_{approx} \leq \frac{1}{2} DAG^{OPT}$.

Mentioned **3** deleting some edges …

Mentioned **1** random ordering of the nodes

Unclear what's "first half of edges"

Final inequality in the wrong direction

Q5 10

[redacted]

5E1C60F6-4879-433A-8307-6FE547C46F65

csci-570-fa-ss-exam-3

#42 Page 6 of 8

5) 20 pts

Consider a special case of the Traveling Salesman Problem called TSP379 where each edge in the graph has a weight of either 3, 7, or 9. Prove that TSP379 is NP-Complete.

a) Prove that TSP379 is in NP. (5 pts)

Certificate: a tour of cost which D

Certifier: all vertices are visited once only.

this can be done in poly time -1

We take an instance of Hamiltonian cycle on Graph G.

We give edge weights of 3 to all the edges in the

Graph G and construct a new gr

the remaining edges to make it fu

and give them edge weights which

no.s which divisible by 3,7 and 9).

is tour with a cost of atmost

is the number of vertices in G. Supp

cost at most 3n in G'. ∵ the missing

high cost compared to 3, the tour e

contains edges from G. Hence, the t

cycle in G. Suppose we have a Ham

∵ the missing edges h

will contain only the

G is the tour in G'.

for weights 7 an

then we

The incorrect construction of the graph invalidates the proof. -6

make the graph connected by adding edges with weight 7 & 9. -3

[redacted]

Additional Space

Over here we consider D to be LCM 3, 7 and 9, 3n, 7n and 9n. We can also have a case where we give weights combination of 3, 7 and 9. In that case if we get tour cost atmost LCM (3, 7 and 9) n then the Hamiltonian cycle exists and vice versa as proved earlier.

[redacted]

D765215E-76E6-45E2-8615-7D847F14CBAE

csci-570-fa-ss-exam-3

Additional Space