CS570 Spring2022: Analysis of Algorithms Exam III

	Points		Points
Problem 1	20	Problem 5	9
Problem 2	9	Problem 6	15
Problem 3	18	Problem 7	17
Problem 4	12		
	Total	**100**	

Instructions:

1. This is a 2-hr exam. Open book and notes. No electronic devices or internet access.
2. If a description to an algorithm or a proof is required, please limit your description or proof to within 150 words, preferably not exceeding the space allotted for that question.
3. No space other than the pages in the exam booklet will be scanned for grading.
4. If you require an additional page for a question, you can use the extra page provided within this booklet. However please indicate clearly that you are continuing the solution on the additional page.
5. Do not detach any sheets from the booklet. Detached sheets will not be scanned.
6. If using a pencil to write the answers, make sure you apply enough pressure, so your answers are readable in the scanned copy of your exam.
7. Do not write your answers in cursive scripts.
8. This exam is printed double sided. Check and use the back of each page.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE** by circling the correct answer. No need to provide any justification.

[TRUE/FALSE]
If P = NP, then all problems in P are NP-complete.

[TRUE/FALSE]
Assuming that we have found a correct optimal solution to an LP problem, if someone claims to have found a different optimal solution, we know that that solution must be incorrect.

[TRUE/FALSE]
Let X and Y be decision problems for which there exist efficient certifiers, 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]
Minimum Spanning Tree (MST) $\leq_P$ Hamiltonian Path.

[TRUE/FALSE]
The optimization version of the Linear Programming is not in NP.

[TRUE/FALSE]
If P=NP, then some NP-hard problems can be solved in polynomial time.

[TRUE/FALSE]
Even if P≠NP, there is a polynomial-time algorithm for the 0/1 Knapsack problem when all items have the same weight-to-value ratio

[TRUE/FALSE]
If P = NP, then every decision problem whose solution can be verified in polynomial time can itself be solved in polynomial time.

[TRUE/FALSE]
Integer Max Flow (where flows and capacities are integer-valued) is polynomial-time reducible to Linear Programming.

[TRUE/FALSE]
Maximize $\Sigma\, x_i(1 - x_i)$ $\quad (1 \leq i \leq n)$
subject to $x_i + x_j < 1$ and $x_i \in \{0, 1\}$ $\quad (1 \leq i \leq n,\ 1 \leq j \leq m)$
where n and m are constants, is an integer linear program.

2) 9 pts

i) Which of the following problems have been proved to be not solvable in polynomial time? Circle all correct answers

A. Traveling Salesman
B. Integer Linear Programming
C. Subset sum
D. All of the above
E. None of the above

Ans : E

ii) Let *X, Y, Z,* and *W* be problems. Suppose *Z* is polynomial-time reducible to *X*, and *X* is NP-complete. Which of the following statements are true? Circle all correct answers.

A. If Vertex Cover is polynomial-time reducible to problem *Z*, then *Z* is NP-Complete.
B. If there exists an efficient certifier for problem *Z*, then *Z* is NP-Complete.
C. *X* is polynomial-time reducible to Independent Set.
D. If *X* is polynomial-time reducible to *Y*, then *Z* is polynomial-time reducible to *Y*.
E. If *Z* is polynomial-time reducible to *W*, then *X* is polynomial-time reducible to *W*.

Ans: C, D

iii) If a problem *X* is in NP, then it must be NP-complete if… (Circle all correct answers)

a. It is polynomial-time reducible to 3-SAT.
b. 3-SAT can be reduced to it in polynomial time.
c. It can be reduced to every other problem in NP in polynomial time.
d. Some problem in NP can be reduced to it in polynomial time.

Answer: (b)
Explanation: A problem in NP becomes NPC if all NP problems can be reduced to it in polynomial time. This is same as reducing any of the NPC problem to it. 3-SAT being an NPC problem, reducing it to a NP problem would mean that NP problem is NPC.

3) 18 pts

A paper company is building warehouses to supply its $m \geq 1$ customers. The company is considering whether to build a warehouse at each of $n \geq 1$ potential construction sites. The cost of building a warehouse at site i is denoted by $b_i \geq 0$. After the warehouses are built, the company assigns each customer to a warehouse that will supply it. Each warehouse that is built may supply paper to any number of customers, but each customer is supplied by exactly one warehouse. The cost of supplying customer j from a warehouse built at site i is denoted by $s_{ij} \geq 0$.

Consider the Warehouse Construction Optimization problem:
Given construction costs b_i $(1 \leq i \leq n)$ and warehouse-customer supply costs s_{ij} $(1 \leq i \leq n,\ 1 \leq j \leq m)$, make a selection of a subset of construction sites and an assignment of customers to warehouses such that the total cost of building a warehouse at each selected site and supplying each customer from their assigned warehouse is minimized.

a) Reduce the above Warehouse Construction problem to Integer Linear Programming. (16 pts total)

i) Describe what your ILP variables represent. (5 pts)

Let the variables $x_i \in \{0,1\}$ $(1 \leq i \leq n)$ indicate whether construction site i was selected, and let the variables $y_{ij} \in \{0,1\}$ $(1 \leq i \leq n,\ 1 \leq j \leq m)$ indicate whether a warehouse built at site i supplies customer j.

Rubric:
2.5 for each correctly defined variable x and y
-1 if more variables are provided
-2.5 if a variable is missing

ii) Show your objective function. (4 pts)

Min $C_B + C_S$

s.t. a) $C_B =$ $\sum_{1 \leq i \leq n} x_i\, b_i$ — C_B *is the total cost of building the warehouses.*

b) $C_S =$ $\sum_{1 \leq i \leq n} \sum_{1 \leq j \leq m} y_{ij}\, s_{ij}$ — C_S *is the total cost of supplying the customers.*

Rubric:
1.5 for each correct functions for Cb and Cs
1 mark for correct overall objective function
-0.5 for minor mistakes

iii) Show your constraints (7 pts)

c)	$\sum_{1 \le i \le n} y_{ij} = 1$	$1 \le j \le m$	*Each customer is supplied by exactly one warehouse.*
d)	$x_i - y_{ij} \ge 0$	$1 \le i \le n$ $1 \le j \le m$	*A warehouse built at site* i *may only supply customer* j *if site* i *was selected.*
	$x_i \in \{0,1\}$	$1 \le i \le n$	*Construction site* i *is either selected or not.*
	$y_{ij} \in \{0,1\}$	$1 \le i \le n$ $1 \le j \le m$	*A warehouse at site* i *supplies customer* j*, or does not.*

Rubric:
2 each for c) & d)
1.5 each for xi and you
-0.5 for any missing constraints

b) **[TRUE/FALSE]** (2 pts)
The above reduction shows that the Warehouse Construction optimization problem is NP-hard.

Rubric:
2 marks for False, 0 for True answer.

4) 12 pts
Suppose that we have developed an approximation algorithm that takes as input a graph $G=(V,E)$ and a real number $\rho > 1$ and returns a vertex cover whose size is less than ρ times the size of a minimum vertex cover.

Prove that the running time of this approximation algorithm is not polynomial in the size of the input unless P=NP.

Brief Solution
We can use this algorithm to solve the vertex cover problem exactly:
Set ρ to some value slightly above 1, e.g. $1< \rho < 1+1/n$ (where n is the number of nodes in G). Then the approximation algorithm must find the optimal vertex cover set, since even if the size of the set is 1 more than the size of the optimal set the approximation ratio will be violated.

Rubrics
1 Attempts that use the approximation Vertex Cover algorithm to solve any other known instance of NP complete problem will get 4 points.(Attempts that don't use reduction or have a wrong direction will get 0 points.)

2 Attempts that use the approximation Vertex Cover algorithm to solve the original Vertex Cover problem will get 6 points. (Attempts that simply paraphrase the algorithm or give some concrete examples e.g., V={1, 2 …}, get a min VC …, will get 0 points)

3 Attempts that explicitly solve VC by approximation VC algorithm with p slightly above 1 will receive 8 points.

4 Attempts that mention the reasonable formula (p < 1+1/n) of increasing the value of p slightly will receive 10 points(match the solution)

5 Attempts that point out the contradiction will receive full points(match the solution)

-2pts for every false statement

Alternative Solution 1:
We can use this algorithm to solve the IS problem exactly: IS<=p VC.
Lemma: I is a Maximal Independent Set of G(V,E) if and only if V\I is a Minimal Vertex Cover G(V,E).
instance of IS with max IS = k, then p satisfies: p(n-k)<(n-k)+1=>p<1+1/(n-k), we can select 1<p<1/n, Then the approximation algorithm must find the optimal IS, since even if the size of the set is 1 smaller than the size of the optimal set the approximation ratio will be violated.

-2pts for every false statement

Alternative Solution 2:
Idea: Contradict with the Theorem that Unless P=NP, there is no $1/n^{1-e}$ approximation for max IS.
Lemma: I is a Maximal Independent Set of G(V,E) if and only if V\I is a Minimal Vertex Cover G(V,E).

For each instance in IS, I* as max IS, thus V\I* is min VC. Let n=|V|, x*=|I*|.
Apply appropriation algorithm on G(V, E), we get a VC with size at most p(n-x*).
Thus, we have a IS with size at least n-p(n-x*).
Set p to some value slightly above 1, e.g. $1 < p < 1+1/n$
there exists a constant $0<e<1$, s.t. $[n-p(n-x^*)]/x^*=p-(p-1)n/x^*>=1/n^{1-e}$, which contradicts the theorem.
1. Attempts that IS<=p VC, and use the approximation VC algorithm. (4pts)

2. Mention theorem that Unless P=NP, there is no $1/n^{1-e}$ approximation for max IS. (1pts)

3. State the Lemma correctly "max IS ⇔ min VC". (1pts)
part I 1+2+3 = 6pts

4. Attempts that explicitly solve VC/IS by approximation VC algorithm with p slightly above 1. (8pts) if analysis in part I is incorrect (0pts)

5. Attempts that mention the reasonable formula ($p < 1+1/n$) of increasing the value of p slightly (match the solution) (10pts) if analysis in part I is incorrect (0pts)

6. Attempts that apply the theorem in a correct way, must mention "there exists a e". (12pts) if analysis in part I is incorrect (0pts)

-2pts for every false statement

Alternative Solution 3:
Idea: reduce VC to app VC with p that satisfies $1<p<1+1/n$.
2 points for using reduction without explicitly mentioning p selection.
4 points for just mentioning set p "slightly" greater than 1.
6 points for setting e.g., $1 < p < 1+1/n$
10 points for correct reduction.
12 points for correct application of reduction.

-2pts for every false statement

5) 9 pts
For each of the problems defined below, determine which class the problem belongs to from the options below. Assume P ≠ NP.

1. The problem is in P.
2. The problem is in NP but not in P.
3. The problem is neither in P nor NP.

a) Given a list of positive integers *A[1..n]* and a positive integer *k*, does there exist an increasing subsequence of *A* of length at least *k*?

b) Given an undirected graph with weights on its vertices, return an independent set with maximum total weight.

c) Given a computer program and an input to the program, will the program terminate for the given input?

- Brief Solution:
  - a) The problem is in P.
  - b) The problem is in neither P nor NP
  - c) The problem is in neither P nor NP

- Explanation

(a) Longest increasing subsequence with running time of $O(n^2)$

(b) No, It's an optimization problem and not a decision problem. We cannot verify in polynomial time if it is the maximal weight. To verify so, we'll have to re-compute maximal weight.

(c) Halting problem in unsolvable -> Not in NP

6) 15 pts

Several artificial lights are installed in a greenhouse to provide light for several plants. Each light may provide light to multiple plants, and each plant receives sufficient light so long as it is exposed to at least one light. The greenhouse operators are planning to uninstall some lights to reduce their costs.

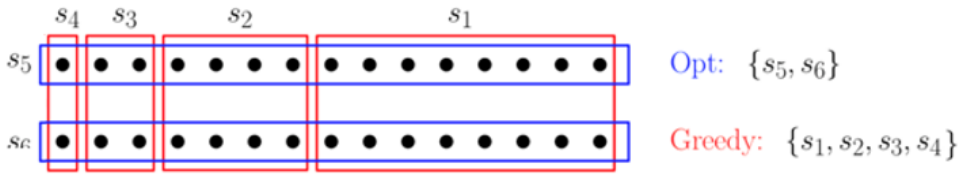
The greenhouse operators have designed a greedy algorithm for minimizing the number of lights in the greenhouse such that each plant is still exposed to at least one light:

> *Begin by turning off all the lights. Then, in each iteration, turn on the light that exposes the most plants to light that are not exposed yet. Repeat until all plants are exposed to at least one light. Then, uninstall the lights that are still turned off.*

a) Give a counterexample to prove that this algorithm does not always give the optimal solution. (5 pts)

b) Prove that if we used the above algorithm as an approximation algorithm, this approximation algorithm has no constant approximation ratio > 1. In other words, if the minimum number of lights needed is C^*, prove that there is no constant ρ ($\rho > 1$) such that we can guarantee the algorithm returns a solution with total number of lights $\leq \rho C^*$. (10 pts)

- Brief Solution:
  - a)



  - b) Imagine a case like the above where the light corresponding to red box S_i covers half the plants that S_{i-1}, and finally two lights corresponding to the two blue boxes each cover half of the total plants as shown. If there are n lights corresponding to the red boxes, then the greedy solution needs n lights whereas the optimal solution still takes 2. Hence, if we increase n, the number of lights that have been chosen becomes arbitrarily large. In particular, choosing n > 2p makes the ratio to the optimal solution more than p for any p > 1, making a p-approximation algorithm not possible.

- Rubrics:
  - For part a.
    - +5 pts for correct counter example. Counter example should contain more lights than the optimal.
    - -2 pts, if the counterexample has the right idea but is incomplete.
    - -5 pts, if the counterexample is wrong or doesn't make sense.
  - For part b.
    - -6 pts, if the explanation is not correct as to why the number of lights can go large or why there is no constant approximation possible.
    - -2 pts, for **each** incorrect proof statement or logical implication, if most of the rest is correct.
    - Note that the fact that this is analogous to Set cover and hence (or otherwise) is NP-hard, is not a correct explanation for there being no constant-factor approximation.

7) 17 pts

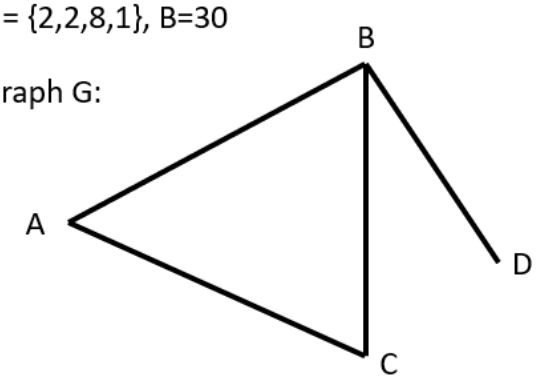
Consider the following problem:

> *Given an undirected graph* G(V,E) *with* n *vertices, a list* L[1..n] *of* n *non-negative integers, and a non-negative integer* B*, does there exist a function* f : L → V *that assigns each number in* L *to a vertex in* G *such that*
>
> $$\sum_{(u,v) \in E} f(u) \cdot f(v) \leq B?$$

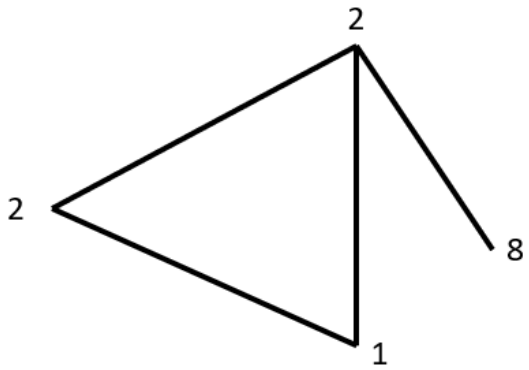
For example, the following is a YES instance of this problem (i.e. an instance for which such a function f exists):

L = {2,2,8,1}, B=30

Graph G:



Here is one possible assignment function:
$L_1 \rightarrow A$, $L_2 \rightarrow B$, $L_3 \rightarrow D$, $L_4 \rightarrow C$



$\sum_{(u,v) \in E} f(u) \cdot f(v) = (2*2) + (2*1) + (2*1) + (2*8) = 24 \leq 30$

Continued of next page

a) Prove that the problem is in NP. (5 pts)

b) Prove that the problem is NP-Complete. (Hint: use Vertex Cover) (12 pts)

SOLUTION

a) **[Polynomial length certificate. 2pt]** Polynomial-length certificate: the function f that maps each integer in L to a vertex in G.
[Polynomial-time certifier. 3pt] Polynomial-time certifier: Verify that each integer in L is assigned to exactly one vertex, compute the value $\sum_{(u,v) \in E} f(u) \cdot f(v)$ and compare to B, and return "Yes" if $\sum_{(u,v) \in E} f(u) \cdot f(v) \leq B$. This can be done in polynomial time (specifically, $O(|E| + |V|)$ time).

b) **[Describe Vertex Cover instance. 1pt]** Given an instance of Vertex Cover, described by a graph $G=(V,E)$ and vertex cover size bound k, construct an instance of the Q7 problem as follows: **[Construction of Q7 *L*. 1pt]** Let L be a list of n integers, where the first k integers are all zero and the remaining $n - k$ integers are all one. **[Construction of Q7 *G*. 1pt]** The graph G in the Q7 problem instance will be the same as in the Vertex Cover instance. **[Construction of Q7 *B*. 1pt]** Let the bound B in the Q7 problem instance be 0.

[Forward assumption. 1pt] Suppose there exists a vertex cover C of the graph G of size $|C| \leq k$. **[Construct *f*. 1pt]** Let f assign the first $|C|$ integers of L arbitrarily to the $|C|$ vertices in the cover C and assign the remaining $n - |C|$ integers of L arbitrarily to the $n - |C|$ vertices of G that are not in the cover. **[Prove $\sum_{(u,v) \in E} f(u) \cdot f(v) \leq B$. 2pt]** Since $|C| \leq k$ and the first k integers in L are all zero, f maps each vertex in the vertex cover C to zero. Since C is a vertex cover, for all edges $(u,v) \in E$, at least one of u and v is in the cover C. Therefore, for all edges edges $(u,v) \in E$, either $f(u) = 0$, or $f(v) = 0$, or both. In any case, $f(u) \cdot f(v) = 0$. This is true for all edges $(u,v) \in E$, so $\sum_{(u,v) \in E} f(u) \cdot f(v) \leq B = 0$. Therefore, if there exists a vertex cover of the graph G of size $\leq k$, then there exists a function f that maps

each integer in L to a vertex in G such that $\sum_{(u,v) \in E} f(u) \cdot f(v) \leq B$.

[Backward assumption. 1pt] Suppose there exists a function f that maps each integer in L to a vertex in G such that $\sum_{(u,v) \in E} f(u) \cdot f(v) \leq B$. **[Construct vertex cover. 1pt]** Let C be the set of vertices u for which $f(u) = 0$. **[Prove C is a vertex cover with $|C| \leq k$. 2pt]** Since all the integers in L are non-negative and $B = 0$, we must have $\sum_{(u,v) \in E} f(u) \cdot f(v) = 0$. This can only happen if, for every edge $(u,v) \in E$, either $f(u) = 0$, or $f(v) = 0$, or both. Then C is a vertex cover of G, since for every edge $(u,v) \in E$, either u is in C, or v is in C, or both. Furthermore, we must have $|C| = k$, since exactly k of the integers in L are zero. Therefore, if there exists a function f that maps each integer in L to a vertex in G such that $\sum_{(u,v) \in E} f(u) \cdot f(v) \leq B$, then there exists a vertex cover of the graph G of size $\leq k$.

Therefore, there exists a vertex cover of the graph G of size $\leq k$ if and only if there exists a function f that maps each integer in L to a vertex in G such that $\sum_{(u,v) \in E} f(u) \cdot f(v) \leq B$. Therefore, Vertex Cover is polynomial-time reducible to the Q7 problem. Since Vertex Cover is NP-Complete, and the Q7 problem is in NP by the previous part, we conclude that the Q7 problem is NP-Complete.

Rubric: [Every time we assign partial credit, list the student's response here, so that if we see similar responses in the future, we can assign partial credit consistently / in the same way.]

a)

[Q7 a1. Polynomial length certificate. 2pt]

Example 2 point answers:

- Describes a valid polynomial-length certificate.
  -"The function f that maps each integer in L to a vertex in G."
  - "The graph G with an assigned number from L to each vertex…"
  - "For a given solution…"
- Describes a polynomial-length certificate identifying a "yes" instance, even though answers of this type are incorrect.
  - "A function f satisfying $\sum_{(u,v) \in E} f(u) \cdot f(v) \leq B$."

Example 0 point answers:

- Does not not describe a valid polynomial-length certificate.
  - "The certificate will be $\sum_{(u,v) \in E} f(u) \cdot f(v)$"
  - "The certificate will be the sum of the function values on the