Hard

CS570
Analysis of Algorithms
Spring 2015
Exam III

Name: _____

Student ID: _____

Email Address:_____

_____Check if DEN Student

	Maximum	Received
Problem 1	20	
Problem 2	16	
Problem 3	16	
Problem 4	16	
Problem 5	16	
Problem 6	16	
Total	100	

Instructions:

1. This is a 2-hr exam. Closed book and notes
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.

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

[TRUE/FALSE]
If SAT $\leq_P$ A, then A is NP-hard.

[TRUE/FALSE]
If a problem X can be reduced to a known NP-hard problem, then X must be NP-hard.

[TRUE/FALSE]
If P equals NP, then NP equals NP-complete.

if P=NP => P = NP = NPC
P里的问题可以相互归约

[TRUE/FALSE]
Let X be a decision problem. If we prove that X is in the class NP and give a poly-time reduction from X to Hamiltonian Cycle, we can conclude that X is NP-complete.

[TRUE/FALSE]
The recurrence $T(n) = 2T(n/2) + 3n$, has solution $T(n) = \theta(n \log(n^2))$.

[TRUE/FALSE]
On a connected, directed graph with only positive edge weights, Bellman-Ford runs asymptotically as fast as Dijkstra.

O(mn)
O(mlogn)

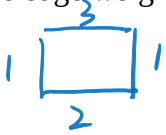
[TRUE/FALSE]
Linear programming is at least as hard as the Max Flow problem in a flow network.

[TRUE/FALSE]
If you are given a maximum s-t flow in a graph then you can find a minimum s-t cut in time O(m) where m is the number of the edges in the graph.

[TRUE/FALSE]
Fibonacci heaps can be used to make Dijkstra's algorithm run in O(|E| + |V| log|V|) time on a graph G=(V,E)

O(m+nlogn)

[TRUE/FALSE]
A graph with non-unique edge weights will have at least two minimum spanning trees



2) 16 pts

Given a graph G=(V, E) with an even number of vertices as the input, the HALF-IS problem is to decide if G has an independent set of size |V|/2. Prove that HALF-IS is in NP-Complete.

certificate: a set of nodes

certifier: check the size of the set is $|V|/2$, check each pair of the nodes in set do not have an edge between them.

The certifier can be done in polynomial time.

Claim: $IS \leq_p HALF\text{-}IS$

Claim: if G has IS of size k, iff G' has HALF-IS of size $|V'|/2$

if $k = |V|/2$ $G'(V', E') = G(V, E)$

if $k < |V|/2$ $V' = V + V_0$ $E' = E$

$|V_0| = |V| - 2k$ and has no edge between any node in $|V|$ and each other.

if $k > |V|/2$ $V' = V + V_0$ $E' = E + E_0$

$|V_0| = 2k - |V|$ E_0 = edges between V_0 nodes each other and each v_0 to every V

3) 16 pts

A variant on the decision version of the subset sum problem is as follows: Given a set of n integer numbers A = $\{a_1, a_2, \ldots, a_n\}$ and a target number *t*. Determine if there is a subset of numbers in A whose product is precisely *t*. That is, the output is *yes* or *no*. Describe an algorithm (and provide pseudo-code) to solve this problem, and analyze its complexity.

OPT (i, j) $\{a_1 \cdots a_i\}$ j

OPT (i, j) = OPT (i-1, j) || OPT (i-1, j/a_i) || 1 if $a_i = j$

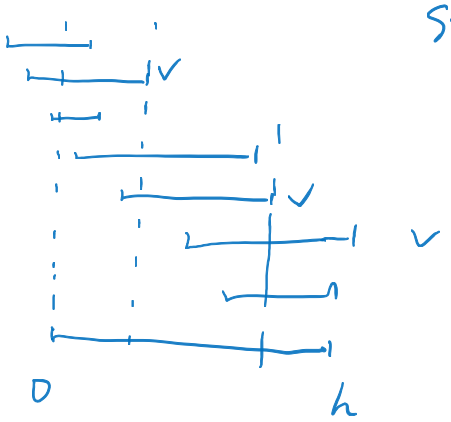
O(tn)

4) 16 pts

We've been put in charge of a phone hotline. We need to make sure that it's staffed by at least one volunteer at all times. Suppose we need to design a schedule that makes sure the hotline is staffed in the time interval $[0, h]$. Each volunteer i gives us an interval $[s_i, f_i]$ during which he or she is willing to work. We'd like to design an algorithm which determines the minimum number of volunteers needed to keep the hotline running. Design an efficient greedy algorithm for this problem that runs in time $O(n \log n)$ if there are n student volunteers. Prove that your algorithm is correct.

You may assume that any time instance has at least one student who is willing to work for that time.

sort $V_1 \cdots V_n$ in $s_i \uparrow$



Initially select a stu V_1, $s_{V_1} \leq 0$ and f_{V_1} is the latest

while the last stu V_i $f_{V_i} < h$,

select a stu V_{i+1}, $s_{V_{i+1}} \leq f_{V_i}$ and $f_{V_{i+1}}$ is the latest

end while

Proof. suppose an opt. sol. $q_1 \cdots q_n$

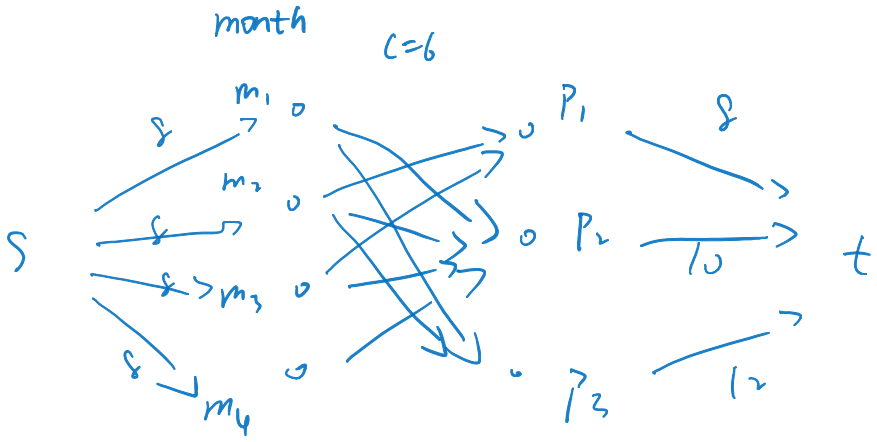
for our sol. $f_{p_n} \geq f_{q_n} \geq h$

our sol won't choose next stu.

$\therefore$ our sol is opt.

5) 16 pts

A software house has to handle 3 projects, P_1, P_2, P_3, over the next 4 months. P_1 can only begin after month 1, and must be completed within month 3. P_2 and P_3 can begin at month 1, and must be completed, respectively, within month 4 and 2. The projects require, respectively, 8, 10, and 12 man-months. For each month, 8 engineers are available. Due to the internal structure of the company, at most 6 engineers can be working, at the same time, on the same project. Determine whether it is possible to complete the projects within the time constraints. Describe how to reduce this problem to the problem of finding a maximum flow in a flow network and justify your reduction.



max flow == 8 + 10 + 12 = 30 !

6) 16 pts

There are n people and n jobs. You are given a cost matrix, C, where C[i][j] represents the cost of assigning person i to do job j. You want to assign all the jobs to people and also only one job to a person. You also need to minimize the total cost of your assignment. Can this problem be formulated as a linear program? If yes, give the linear programming formulation. If no, describe why it cannot be formulated as an LP and show how it can be reduced to an integer program.

x_{ij} denotes whether job i assign to person j

$x_{ij} = \{0, 1\}$ discrete variable, so integer programming.

Objective: $\min\left(\sum_i \sum_j x_{ij} C_{ij}\right)$

s.t.: $\sum_i x_{ij} = 1$ for $j = 1 \dots n$

$\sum_j x_{ij} = 1$ for $i = 1 \dots n$

Additional Space

Additional Space