CS570

Analysis of Algorithms

Spring 2008

Final Exam (B)

Name: ______________________

Student ID: __________________

Section:__2:00-5:00 or __5:00-8:00

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	10	
Problem 6	20	
Total	100	

Note: The exam is closed book closed notes.

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

[TRUE/FALSE]
In a flow network whose edges have capacity 1, the maximum flow always corresponds to the maximum degree of a vertex in the network**.**

[TRUE/FALSE]
If all edge capacities of a flow network are unique, then the min cut is also unique.

[TRUE/FALSE]
A minimum weight edge in a graph G must be in one minimum spanning tree of G.

[TRUE/FALSE]
When the size of the input grows, any polynomial algorithm will eventually become more efficient than any exponential one.

[TRUE/FALSE/UNKNOWN]
NP is the class of problems that are not solvable in polynomial time.

[TRUE/FALSE/UNKNOWN]
If a problem is not solvable in polynomial time, it is in the NP-Complete class.

[TRUE/FALSE/UNKNOWN]
Linear programming can be solved in polynomial time.

[TRUE/FALSE]
$10^{2 \log 4n+3} + 9^{2 \log 3n+21}$ is O(n).

[TRUE/FALSE]
f(n) = O(g(n)) implies g(n) = O(f(n)).

[TRUE/FALSE]
If X can be reduced in polynomial time to Y and Z can be reduced in polynomial time to Y , then X can be reduced in polynomial time to Z.

2) 15 pts

Suppose you are given a number x and an array A with n entries, each being a distinct number. Also it is known that the sequence of values A[1],A[2], ...,A[n] is **unimodal**. In other words for some unknown index p between 1 and n, we have

$A[1] < \ldots < A[p-1] < A[p] > A[p+1] > \ldots > A[n]$. (Note that A[p] holds the peak value in the array).

Give a algorithm with running time O(log n) to determine if x belongs to A, if yes the algorithm should return the index j such that A[j] = x. You should justify both the correctness of your algorithm and the running time.

3) 15 pts

You are given two sequences a[1],...,a[m] and b[1],...,b[n]. You need to find their longest common subsequence; that is, find a subsequence $a[i_1]$, ..., $a[i_k]$ and $b[j_1]$,...,$b[j_k]$, such that $a[i_1] = b[j_1]$, ..., $a[i_k] = b[j_k]$ with k as large as possible. You need to show the running time of your algorithm.

4) 20 pts

   In Linear Programming, variables are allowed to be real numbers. Consider that you are restricting variables to be only integers, keeping everything else the same. This is called Integer Programming. Integer Programming is nothing but a Linear Programming with the added constraint that variables be integers. Prove that integer programming is NP-Hard

5) 10 pts

A carpenter makes tables and bookshelves, and he wants to determine how many tables and bookshelves he should make each week for a maximum profit. The carpenter knows that a net profit for each table is $25 and a net profit for each bookshelf is $30. The wooden material available each week is 690 units, its working hours are 120 hours per week. The estimated wood and working hours for making a table are 20 units and 5 hours respectively, while for making a bookshelf are 30 units and 4 hours. Formulate the problem using any technique we have covered in class. You do not need to solve it numerically.

6) 20 pts

A variation of the satisfiability problem is the MIN 2-SAT problem. The goal in the MIN 2-SAT problem is to find a truth assignment that minimizes the number of satisfied clauses. Give the best approximation algorithm that you can find for the problem.