CS570
Analysis of Algorithms
Summer 2011
Final Exam

Name: _____

Student ID: _____

____Check if DEN student

	Maximum	Received
Problem 1	20	
Problem 2	13	
Problem 3	14	
Problem 4	13	
Problem 5	20	
Problem 6	20	
Total	100	

2 hr exam
Close book and notes

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification except for the question at the bottom of the page.

[TRUE/FALSE]
In a flow network, if all edge capacities are distinct, then the max flow of this network is unique.

[TRUE/FALSE]
To find the minimum element in a max heap of n elements, it takes O(n) time.

[TRUE/FALSE]
Let T be a spanning tree of graph G(V, E), let k be the number of edges in T, then k=O(V)

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

[TRUE/FALSE]
Consider problem A: given a flow network, find the maximum flow from a node s to a node t. problem A is in NP.

[TRUE/FALSE]
Given n numbers, it takes O(n) time to construct a binary min heap.

[TRUE/FALSE]
Kruskal's algorithm for finding the MST works with positive and negative edge weights.

[TRUE/FALSE]
Breadth first search is an example of a divide-and-conquer algorithm.

[TRUE/FALSE]
If a problem is not in P, then it must be in NP.

[TRUE/FALSE]
L1 can be reduced to L2 in Polynomial time and L1 is in NP, then L2 is in NP

2) 13 pts

Imagine that you constructed an approximation algorithm for the Traveling Salesman Problem that could always calculate a solution that is correct within a factor of $1/k$ of the optimal tour in $O(n^{2k})$ time. Would you be able to use this approximation algorithm to obtain a "good" solution to all other NP-Complete problems? Explain why or why not.

Yes. You can use it.

In the Traveling Salesperson Problem, we are given an undirected graph G = (V,E) and cost c(e) > 0 for each edge e 2 E. Our goal is to find a Hamiltonian cycle with minimum cost. A cycle is said to be Hamiltonian if it visits every vertex in V exactly once.
TSP is known to be NP-complete, and so we cannot expect to exactly solve TSP in polynomial time. What is worse, there is no good approximation algorithm for TSP unless P = NP. This is because if one can give a good approximation solution to TSP in polynomial time, then we can exactly solve the NP-Complete Hamiltonian cycle problem (HAM) in polynomial time, which is impossible unless P = NP. Recall that HAM is the decision problem of deciding whether the given graph G = (V,E) has a Hamiltonian cycle or not.
Theorem 1 ([2]) The Traveling Salesperson Problem cannot be approximated within any factor, unless P = NP.
Proof: For the sake of contradiction, suppose we have an approximation algorithm A on TSP with an approximation ratio C. We show a contradiction by showing that using A, we can exactly solve HAM in polynomial time. Let G = (V,E) be the given instance of HAM. We create a new graph H = (V,E0) with cost c(e) for each e 2 E0 such that c(e) = 1 if e 2 E, otherwise c(e) = B, where B = nC + 1 and n = |V |.
We observe that if G has a Hamiltonian cycle, OPT = n, otherwise OPT § n−1+B = n(C+1).
(Here, OPT denotes the cost of an optimal TSP solution in H.) Note that there is a "gap" between when G has a Hamiltonian cycle and when it does not. Thus, if A has an approximation ratio of C, we can tell whether G has a Hamiltonian cycle or not: Simply run A on the graph H; if G has a Hamiltonian cycle, A returns a TSP tour in H of cost at most COPT = Cn. Otherwise, H has no TSP tour of cost less than n(C + 1), and so A must return a tour of at least this cost.

3) 14 pts

Prove the following statement or give a counterexample: "For any weighted graph G there is node v in G such that a Shortest Path Tree (tree found by running Dijkstra's) with v as source is identical to a Minimum Spanning Tree of G."

False. Note that the MST will always avoid the largest weight edge in a cycle (assuming there is a unique largest), whereas the shortest path tree might use it. Consider the following graph where the diagonal edges

have weight 3 and the other edges have weight 2. A MST consists of three of the weight 2 edges, whereas a shortest path tree will use one of the diagonal edges.

4) 13 pts

Define the capacity of a node as the maximum incoming flow it can have. Briefly show how to reduce the problem of finding the maxflow in a network with capacity limits on the nodes to the usual maxflow problem. Answer in one or two sentences.

Replace the node v with two nodes v' and v". Add an edge from v' to v" with the desired node capacity. Replace all of the edges (u, v) that enter v to (u, v') so that they enter v'. Replace all of the edges (v, w) that leave v to (v", w) so that they leave v".

5) 20 pts

Let G = (V,E) be an undirected graph in which the vertices represent small towns and the edges represent roads between those towns. Each edge e has a positive integer weight d(e) associated with it, indicating the length of that road. The distance between two vertices (towns) in a graph is defined to be the length of the shortest weighted path between those two vertices.

Each vertex v also has a positive integer c(v) associated with it, indicating the cost to build a fire station in that town.

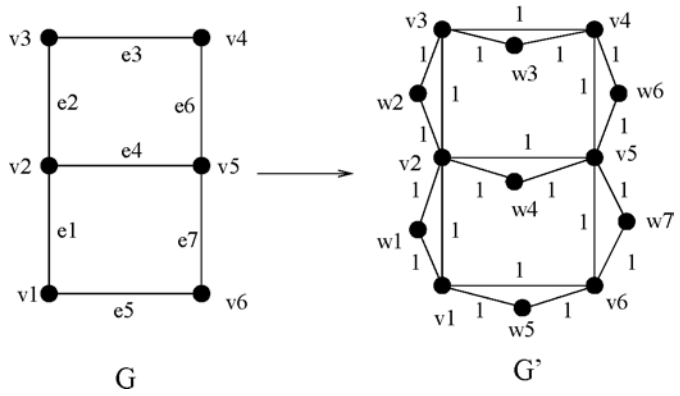
In addition, we are given two positive integer parameters D and C. Our objective is to determine whether there is a way to build fire stations such that the total cost of building the fire stations does not exceed C and the distance from any town to a fire station does not exceed D. This problem is known as the Rural Fire Station (RFS) Problem.

Your company has been hired by the American League of Rural Fire Departments to study this problem. After spending months trying unsuccessfully to find an efficient algorithm for the problem, your boss has a hunch that the problem is NP-complete. Prove that RFS is NP-complete.

We prove that FIRE-STATION-PLACEMENT is NP-complete. To see that it is NP we use the set of vertices for placing the fire stations as a certificate. Clearly, there is a polynomial time verification algorithm.

We now prove that VERTEX-COVER $\leqslant_p$ FIRE-STATION-PLACEMENT. Let (G, k) be the input for VERTEX-COVER. Without loss of generality, we assume that no vertices in G have degree 0. (If G had any vertices of degree 0, then remove them. Notice that this won't affect the size of a minimum vertex cover of G.)

The graph for the fire-station-placement problem $G' = (V',E')$ is obtained from $G = (V,E)$ as follows. Let $V = \{v_1, \ldots, v_n\}$ and $E = \{e_1, \ldots, e_m\}$. We will create a new vertex w_i for each edge in E. Let $W = \{w_1, \ldots, w_m\}$. Then $V' = V \cup W$ and for each edge $e = (u, v) \in E$ (with associated vertex w) we place the following 3 edges in E': (u, v), (u,w), (v,w). All edges have a weight of 1. So $|V'| = |V| + |E|$ and $|E'| = 3|E|$. Here is an example of this transformation:



Clearly G' can be constructed in polynomial time. The input for FIRE-STATION-PLACEMENT is (G', C = k, D = 1) and the cost of building each fire station is 1.
We now prove that G has a vertex cover of size k if and only if G' has a placement of k fire stations so that each vertex has distance of at most 1 to its nearest fire station. Suppose that G has a vertex cover C of size k. We show that by placing a fire station in G' at each vertex in C all vertices are directly connected to a fire station (or have a fire station). Suppose not. Let u be such a vertex greater than distance 1 from its nearest fire station. First suppose that $u \in V$ (i.e. an original vertex from G). Since all vertices in G have at least one edge, u must be connected in G' to some vertex $v \in V$ where v is also not in C. This contradicts that C is a vertex cover. Next suppose that $u \in W$ (it is one of the added vertices), then u is connected to two vertices v_1 and v_2 where $(v_1, v_2) \in E$ and there is not a fire station at v_1 or v_2 thus contradicting that C was a vertex cover.
We first prove that if F is the set of vertices to place fire stations so that $|C| = k$ and all vertices have distance at most 1 to some fire station, then there is a vertex cover of size k in G.

We first transform F into a fire station placement solution F_0 in which $|F'| = |F|$ and in which F' is a subset of V (i.e. F' only includes original vertices). We build F' from F by doing the following: for each vertex $w \in W \setminus F$ (in any order). Suppose w is the vertex added corresponding to edge (u_1, u_2). In this case, we can simply move the fire station at vertex w to u_1 (or anywhere else if u_1 already has a fire station). Notice that the fire station at w could only be a nearest fire station for w, u_1 and u_2 and hence if there is a fire station at u_1 then each of w, $u_1$and u_2 can use u_1 as a nearest fire station. Hence we maintain the invariant (started with F) that our solution at each step (and hence F') satisfies the requirements that each vertex in $V' = V \cup W$ is within distance 1 from a fire station.
We now prove that F' is a vertex cover in G. Suppose not. Then there is an edge $(u, v) \in G$ such that neither of u or v belong to F'. Consider the vertex w corresponding to the edge (u, v) (i.e the triangle u,v,w). Notice that w is connected only to u and v and thus w must have a distance of greater than 1 to its nearest fire station thus contradicting the requirements for a proper fire station placement in G'.

6) 20 pts
You are given a array of n positive numbers A[1]…A[n]. You are not allowed to re-order them. You need to find two indexes i and j, where 1<=i<j<=n, such that A[j]-A[i] is maximized. Solve this problem in O(n) time using a dynamic programming method.

Let OPT[i] be the max difference of the first i numbers.

OPT[i+1]=max{OPT[i], A[i+1]-min[i]}
Where min[i] is the minimum of the first i numbers.
min[i+1]=min{A[i+1], min[i]}.
To find the indices, we can record the choice OPT[] and min[] of each step.
Calculate this takes O(n) because each element of the array is visited once.