CSOR 4231 Final Exam
December 17, 2003, 9AM

Rules; Answer each question completely and concisely. When you give an algorithm, be sure to give the most efficient one you can, to prove that it is correct, and to analyze its running time. Any NP-completeness proof must be a reduction from one of the problems we studied in class (SAT, 3-SAT, Clique, Independent Set, Vertex Cover, Hamiltonian Path, Set Cover, Subset Sum). When you give an algorithm, be sure to give the most efficient one you can, to prove that it is correct, and to analyze its running time.

The first two problems are loosely related, but do not depend on each other.

Answer 4 of the following 5 questions.

Exam

Problem 1. In a fit of patriotism, the president decides that we need to emphasize the the theme of independence. He therefore signs the *Extremely Independent Graph Act* which decrees that all $|V|$-node graphs that are studied with federal money must have an independent set of size at least $|V|/2$.

You have been hired to enforce this law. Show that enforcement is difficult, i.e. prove that the problem of deciding whether a graph has an independent set of size at least $|V|/2$ is NP-complete.

Problem 2. In an effort to save face, and to placate the lumber industry, the president declares that all graphs studied with federal money must be trees (but not necessarily binary trees). Suppose that each node v in a tree has a weight $w(v)$. Give a polynomial time algorithm to find an independent set I that maximizes the sum of the weights of the nodes in the independent set, $\sum_{v \in I} w(v)$.

Prove your algorithm is correct and analyze its running time.

Problem 3. You are given a connected graph which has n nodes and m edges. The edges are colored either red or blue. Give the most efficient algorithm you can to find the spanning tree that has the maximum possible number of blue edges.

Problem 4. You need to maintain a graph dynamically, that is you want to allow vertices and edges to be inserted and deleted over time. You need to support the following operations:

- InsertVertex(v) - add a vertex v to the graph.
- InsertEdge(u, v, w) - add an edge (u, v) with weight w to the graph.
- DeleteMaxEdge(v) - Let F be the set of edges incident to vertex v. Delete the maximum weight edge in F from the graph.

You can assume that each edge or vertex inserted is unique, that the vertices are labeled with integers between 1 and n, and that a when DeleteMaxEdge(v) is called, v has at least one incident edge. Let n be the total number of operations performed.

a) Explain how to implement this data structure so that each operation has a worst-case cost of $O(\log n)$ time.

b) Explain how to implement this data structure so that each InsertVertex and InsertEdge each has a worst case cost of $O(1)$ time and each operation has an amortized cost of $O(\log n)$.

Problem 5. Consider the following scenario. You are given a number k, n numbers to sort, code for insertion sort and code for heapsort. You must flip k coins, and if $k-1$ come up heads, you run insertion sort, otherwise you run heapsort. It takes $O(1)$ time to flip a coin.

a) What is the expected running time of this algorithm when $k = 2$.

b) What is the expected running time of this algorithm for arbitrary k and n, in terms of k and n.

c) Is there a value of k so that the expected running time is $O(n \log n)$? If so, please give such a value for k.