Final

Note

This is a closed-book exam. Each problem accounts for 10 points, unless otherwise marked.

Problems

1. Please present the union-find algorithm with balancing and path compression in a suitable pseudocode.

2. Compute the *next* table as in the KMP algorithm for the string $B[1..10] = babbababbb$. Please show the details of calculation for $next[9]$ and $next[10]$.

3. Consider a chain A_1, A_2, A_3, A_4, A_5 of five matrices with dimensions 40×20, 20×30, 30×50, 50×40, and 40×30, respectively. Compute (by immitating an algorithm based on dynamic programming) the minimum number of scalar multiplications needed to evaluate the product $A_1A_2A_3A_4A_5$.

4. Given as input a connected undirected graph G, a spanning tree T of G, and a vertex v, design an algorithm to determine whether T is a valid DFS tree of G rooted at v. In other words, determine whether T can be the output of DFS under some order of the edges starting with v. The more efficient your algorithm is, the more points you get for this problem. Explain why the algorithm is correct and give an analysis of its time complexity.

5. Design an algorithm for determining whether a given *acyclic* directed graph $G = (V, E)$ contains a directed Hamiltonian path. (Note: A directed *Hamiltonian path* of a directed graph is a simple directed path that includes all vertices of the graph. An acyclic directed graph is one without directed cycles.) The more efficient your algorithm is, the more points you will be credited for this problem. Explain why your algorithm is correct and give an analysis of its time complexity.

   What does the existence of a directed Hamiltonian path imply (when the directed graph is meant to model the dependency among tasks)?

6. Let $G = (V, E)$ be a connected weighted undirected graph and T be a minimum-cost spanning tree (MCST) of G. Suppose that the cost of one edge $\{u, v\}$ in G is *decreased*; $\{u, v\}$ may or may not belong to T. Design an algorithm either to find a new MCST or to determine that T is still an MCST. The time complexity of your algorithm should be $O(|V| + |E|)$. Explain why your algorithm is correct and analyze its time complexity.

7. A connected undirected graph is called *edge-biconnected* if the graph remains connected after the removal of any edge. (Recall that a graph is *biconnected* if the graph remains connected after the removal of an arbitrary vertex and all edges incident to the vertex.)

   (a) Are biconnected graphs always edge-biconnected? Why? (3 points)

   (b) Draw a graph that is edge-biconnected but not biconnected. (3 points)

   (c) Describe the characteristics of edge-biconnected graphs that are useful for designing an algorithm that determines whether a graph is edge-biconnected. Is your characterization complete? (6 points)

   (d) Design an efficient algorithm, based on the characterization above, to determine whether a graph is edge-biconnected. It suffices to describe the main idea of the algorithm. (3 points)

8. Describe how the maximum matching problem for bipartite graphs can be reduced to the network flow problem.

9. Solve one of the following two problems. (Note: If you try to solve both problems, I will randomly pick one of them to grade.) (15 points)

   (a) The traveling salesman problem is as follows.

      The input includes a set of cities, the distances between all pairs of cities, and a number D. The problem is to determine whether there exists a (traveling-salesman) tour of all the cities having total length $\leq D$.

   Prove that the traveling salesman problem is NP-complete.

   (b) The knapsack problem is as follows.

      Given a set X, where each element $x \in X$ has an associated size $s(x)$ and value $v(x)$, and two other numbers S and V, is there a subset $B \subseteq X$ whose total size is $\leq S$ and whose total value is $\geq V$?

   Prove that the knapsack problem is NP-complete.

Appendix

- The partition problem: given a set X where each element $x \in X$ has an associated size $s(x)$, is it possible to partition the set into two subsets with exactly the same total size?
  The problem is NP-complete.
- The Hamiltonian cycle problem: given a graph G, does G contain a Hamiltonian cycle? (A Hamiltonian cycle in a graph is a cycle that contains each vertex, except the starting vertex of the cycle, exactly once.)
  The problem is NP-complete.