Homework Assignment #8

Due Time/Date

2:10PM Tuesday, December 8, 2020. Late submission will be penalized by 20% for each working day overdue.

Note

Please write or type your answers on A4 (or similar size) paper. Drop your homework by the due time in Yih-Kuen Tsay's mail box on the first floor of Management College Building 2. You may discuss the problems with others, but copying answers is strictly forbidden.

Problems

There are five problems in this assignment, each accounting for 20 points. (Note: problems marked with "(X.XX)" are taken from [Manber 1989] with probable adaptation.)

1. (7.38) Given a directed acyclic graph $G = (V, E)$, find a simple (directed) path in G that has the maximum number of edges among all simple paths in G. The algorithm should run in linear time.

2. Dijkstra's algorithm for single-source shortest paths assumes that every edge of the input graph has a nonnegative weight. Suppose we are given a graph with negative weights on some edges, where the minimum weight of the edges is $-c$ for some $c > 0$. If we add c to the weight of every edge, then we obtain a new graph with nonnegative edge weights. We could then apply Dijkstra's algorithm to find the shortest paths for the new graph and thereafter subtract c from each edge of a path. Does this give us the shortest paths for the original graph? Please explain your answer.

3. (7.9) Prove that if the costs of all edges in a given connected graph are distinct, then the graph has exactly one unique minimum-cost spanning tree.

4. What is wrong with the following algorithm for computing the minimum-cost spanning tree of a given weighted undirected graph (assumed to be connected)?

   > If the input is just a single-node graph, return the single node. Otherwise, divide the graph into two subgraphs, recursively compute their minimum-cost spanning trees, and then connect the two spanning trees with an edge between the two subgraphs that has the minimum weight.

5. (7.61) 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 changed (*increased* or *decreased*); $\{u, v\}$ may or may not belong to T. Design an algorithm to either find a new MCST or to determine that T is still an MCST. The more efficient your algorithm is, the more points you will be credited for this problem. Explain why your algorithm is correct and analyze its time complexity.