Due in class: Oct 11.

Graduate students should do problems 2–6, others do 1-5.

(1) Suppose we increase the weight of every edge in an undirected graph by a fixed constant c. Can this change the shortest path from s to t? Can this change the Minimum Spanning Tree T? Assume for simplicity that all edge weights are distinct (no two edges have the same weight).

(2) Describe an $O(E+V)$ time algorithm to find the length of the longest path from s to each vertex v in a Directed Acyclic Graph. You may assume that s can reach every vertex in the graph.

(3) Consider the single source shortest path problem in a directed graph. There may be situations when there are multiple paths of the same weight from s to v that are all shortest length paths. In such a situation, we may want a shortest length path with the least number of edges (hops). How can we design an algorithm for this problem? Assume that all edge weights are non-negative.

In other words, while computing shortest paths we normally ignore the number of edges. We may have two shortest paths of length 10, one with 3 edges and one with 4 edges. In Dijkstra's algorithm either one may be found. Suppose in this case we would like the path with fewer edges. Design an algorithm for this problem with running time the same as Dijkstra's algorithm.

(4) Problem 25-2 (page 546) OLD EDITION.

Problem 24-2 (page 615) NEW EDITION.

(5) Problem 25-3 (page 546) OLD EDITION.

Problem 24-3 (page 615) NEW EDITION.

(6) Consider the Bellman-Ford Algorithm. Suppose the algorithm halts and declares that the graph has a negative length cycle. Can you output the edges on this cycle? (In other words, design an algorithm to find a negative cycle in a graph if one exists.) The running time should be no worse than Bellman-Ford.