MAT-72006 Advanced Algorithms and Data Structures
November 24, 2016
HW 10: 24 Single-Source Shortest Paths, 25 All-Pairs Shortest Paths

1.

The *diameter* of a graph G is the length of a longest shortest path in G. Give a polynomial-time algorithm for computing the diameter of a given graph G. For concreteness, you can assume G is directed and weighted with no negative edge weights.

2.

Let G be a weighted graph with no negative edge weights. Let $P = \{v_1, v_2, \ldots, v_k\}$ be a shortest path between two vertices v_1 and v_k. Is it true that all subpaths of P are also shortest paths?

3.

Let G be a weighted undirected graph with no negative edge weights. Is it true that for any two distinct vertices x and y, a shortest x-y path is contained in some minimum spanning tree of G?

4.

Let G be a weighted directed graph with no negative edge weights. In addition, let $\mathcal{S} = \{\{s_1, t_1\}, \ldots, \{s_\ell, t_\ell\}\}$ be a collection of ℓ terminal pairs. Given G and $\mathcal{S}$, the goal is to determine a closest terminal pair, i.e., the index i such that the distance $d(s_i, t_i)$ is minimized (note that there are possibly several choices of i giving the smallest distance).

A rather obvious solution is to run Dijkstra's algorithm for each terminal pair and record the distance. Suggest a faster algorithm for the problem.

5.

The lecture slides consider several algorithms for finding an s-t shortest path (i.e., a shortest path from vertex s to vertex t). All these algorithms make the assumption that edge weights are non-negative (or that there are no negative-weight cycles). Perhaps interestingly, there is no apparent reason for these assumptions.

Why do you think such assumptions are needed? In other words, why couldn't we have a fast algorithm for finding a shortest s-t path even in the presence of negative edge weights (and negative-weight cycles)? (Hint: a "correct" answer is outside the scope of the course; however, your answer can give thoughts, observations and/or intuitive ideas).