CS 3330: Homework 4

Fall 2017

1. Consider the directed graph in the figure for this problem. We are given a non-negative length corresponding to each edge. Compute the shortest path lengths from S to each vertex in the graph. Also show the corresponding shortest path tree. This problem corresponds to Section 4.4 of the text. (2 points)

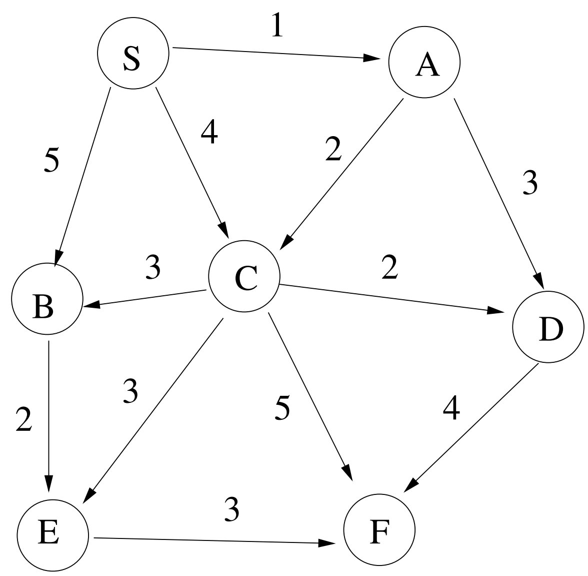

Figure 1: Figure for Problem 1

2. Consider the undirected graph in the figure for this problem. We are given a non-negative cost corresponding to each edge.

   (a) List the edges in a minimum spanning tree of the graph, in the order in which Kruskal's algorithm adds them.

   (b) List the edges in a minimum spanning tree of the graph, in the order in which Prim's algorithm, starting at vertex S, adds them.

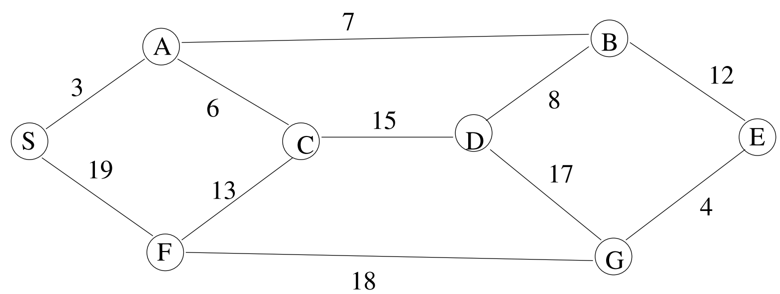

Figure 2: Figure for Problem 2

   This problem corresponds to Section 4.5 of the text. (3 points)

3. You just discovered your best friend from elementary school on Twitbook. You both want to meet as soon as possible, but you live in two different cities that are far apart. To minimize travel time, you agree to meet at an intermediate city, and then you simultaneously hop in your cars and start driving toward each other. But where exactly should you meet?

   You are given a weighted, directed, graph $G = (V, E)$, where the vertices V represent cities and the edges E represent roads that directly connect cities. Each edge e has a weight $w(e)$ equal to the time required to travel between the two cities. You are also given a vertex p, representing your starting location, and a vertex q, representing your friend's starting location.

   Describe an algorithm to find the target vertex t that allows you and your friend to meet as quickly as possible. Make the algorithm as efficient as you can, in terms of running time. (2 points)

4. Let $G = (V, E)$ be a directed graph with non-negative edge lengths, let s and t be vertices of G, and let H be a subgraph of G obtained by deleting some edges. Suppose we want to reinsert exactly one edge from G back into H, so that the shortest path from s to t in the resulting graph is as short as possible. Describe an algorithm that chooses the best edge to reinsert. For full credit, your algorithm should run in $O(|E| \log |E|)$ time. Assume $|E| \geq |V|$. (3 points)

The last two problems have been adapted from Lecture 21 of Jeff Erickson's notes at `http://www.cs.uiuc.edu/~jeffe/teaching/algorithms/`