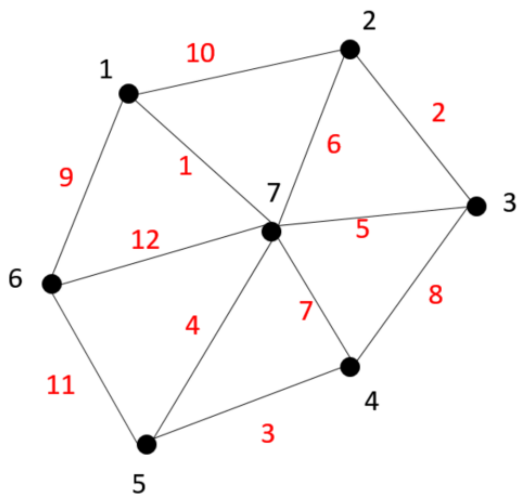
COMPSCI330 Design and Analysis of Algorithms
Assignment 7

Due Date: Wednesday, November 8, 2017

Guidelines

- **Describing Algorithms** If you are asked to provide an algorithm, you should clearly define each step of the procedure, and then analyze its overall running time. There is no need to write pseudo-code; an unambiguous description of your algorithm in plain text will suffice. If the running time of your algorithm is worse than the suggested running time, you might receive partial credits.
- **Typesetting and Submission** Please submit each problem as an *individual* pdf file for the correct problem on Sakai. LaTeX is preferred, but answers typed with other software and converted to pdf is also accepted. Please make sure you submit to the correct problem, and your file can be opened by standard pdf reader. **Handwritten answers or pdf files that cannot be opened will not be graded.**
- **Timing** Please start early. The problems are difficult and they can take hours to solve. The time you spend on finding the proof can be much longer than the time to write. If you submit within one week of the deadline you will get half credit. Any submission after that will not receive any credit.
- **Collaboration Policy** Please check this page for the collaboration policy. You are **not** allowed to discuss homework problems in groups of more than 3 students. **Failure to adhere to these guidelines will be promptly reported to the relevant authority without exception.**

Problem 1 (Prim and Kruskal)**.** (10 points) We will run Prim's algorithm and Kruskal's algorithm for the following graph:



In this graph, the black numbers are indices for vertices, and the red numbers are the lengths of edges.

(a) (5 points) Suppose we run Prim's algorithm with starting vertex 7, list the edges added by Prim's algorithm in the order that they are added by the algorithm.

(b) (5 points) Now we will run Kruskal's algorithm, list the edges added by Kruskal's algorithm in the order that they are added by the algorithm.

Problem 2 (Optimal Tree)**.** (25 points) In this problem we want to decide whether a tree T is a minimum spanning tree of a graph G. We want to prove: T is a minimum spanning tree of G if and only if for every cut $(S, \bar{S})$, T contains an edge e that is one of the minimum cost edges in the cut $(S, \bar{S})$.

(a) (10 points) Show that if there exists a cut $(S, \bar{S})$, such that T does not contain any of its minimum cost edges, then T cannot be a minimum spanning tree.

(Hint: use the swap operation.)

(b) (15 points) Show that if for every cut $(S, \bar{S})$, T contains edges e that is one of the minimum cost edges in the cut $(S, \bar{S})$, the tree T must be a minimum spanning tree.

(Hint: Show that the tree can be found by the Prim's algorithm (or use the Key Lemma).)

Problem 3 (Alternative Link)**.** (25 points) Suppose you have a graph G and a Minimum Spanning Tree T. These are already in memory, stored in whatever form (adjacency list/array) you want. Now there is a new edge $e = (u, v)$ added to G with weight $w(u, v)$.

(a) (10 points) Design an algorithm to update the tree T to be a Minimum Spanning Tree for $G \cup \{e\}$. Your algorithm should run in time $O(n)$.

(b) (15 points) Prove the correctness of your algorithm.

(Hint: You may want to use the conclusion of Problem 2. You don't have to use it but using it gives a clean proof.)