CSE421: Introduction to Algorithms

Homework 4

Due: October 24, 2018

1. You are given an undirected graph G with n vertices and m edges, and a minimum spanning tree T of the graph. Suppose one of the edge weights $w(e)$ of the graph is updated.

   Give an algorithm that runs in time $O(m)$ to test if T still remains the minimum spanning tree of the graph. You may assume that all edge weights are distinct both before and after the update.

   HINT: If $e \in T$, consider the cut obtained by deleting e from T. If $e \notin T$, consider the cycle formed by adding e to T.

2. A small business faces the following scheduling problem: Each morning they get a set of jobs from customers. They want to do the jobs on their single machine in an order that keeps their customers happiest. Customer i's job will take t_i time to complete. Given a schedule (i.e., an ordering of jobs), let C_i denote the finishing time of job i. For example, if job j is the first to be done, we would have $C_j = t_j$; and if job j is done right after job i, we would have $C_j = C_i + t_j$. Each customer i also has a given weight w_i that represents his or her importance to the business. So, the company wants to order the jobs to minimize weighted sum of completion times, $\sum_{i=1}^{n} w_i C_i$.

   Design a polynomial time algorithm that orders the jobs so as to minimize the weighted sum of the completion times.

3. You are given a undirected graph $G = (V, E)$ whose edges have weights ℓ_{uv} for $(u, v) \in E$, and a tree T defined using a subset of the edges of G. Fix a node $s \in V$. The tree T is promised to have exactly one path from a vertex s to every other vertex in the graph. We call T is a *shortest path tree* for s if for any node $v \in V$, the path from s to v in T is a shortest s-v path in graph G. Give an $O(m + n)$ time algorithm (where $n = |V|$ and $m = |E|$) to decide whether or not T is a valid shortest path tree for G. Prove that your algorithm indeed gives the correct answer.

   Note that Dijkstra algorithm cannot be used here because it takes $O(m + n \log n)$ time.

   HINT: Let $d_T(s, u)$ be the distance between s and u on the tree T. Prove that T is a shortest path tree if and only if $d_T(s, u) \leq d_T(s, v) + \ell_{uv}$ for all edge (u, v) on the graph.

4. **Extra Credit:** In class we discussed an algorithm to color the vertices of an n vertex graph with 2 colors so that every edge gets exactly 2 colors (assuming such a coloring exists). We know of no such algorithm for finding 3-colorings in polynomial time. Here we'll figure out how to color a 3-colorable graph with $O(\sqrt{n})$ colors.

   (a) Give a polynomial time algorithm to color the vertices with at most $\Delta + 1$ colors, where Δ is the maximum degree of vertices in the graph.

   (b) Give a polynomial time algorithm that colors the graph with (at most) $O(\sqrt{n})$ colors, if the input graph is promised to have a 3-coloring.