Exercise Set 3

CSE 5311 Design and Analysis of Algorithms

SPRING 2007

1. Let G = (V,E) be a **directed** graph and let **v** and **w** be two vertices in G. Design a linear time algorithm to find **the number of different shortest paths** (not necessarily vertex disjoint) **between v and w**. **The path** can be in terms of **the number of edges.**

2. Let G = (V,E) be an (undirected) graph with costs $c_e \geq 0$ on the edges $e \in E$. Assume you are given a minimum-cost spanning tree T in G. Now assume that a new edge is added to G, connecting two nodes v, w $\in$V with cost c.
    a. Give an efficient algorithm to test if T remains the minimum-cost spanning tree with the new edge added to G (but not to the tree T). Make your algorithm run in time O ($|E|$). Can you do it in O($|V|$) time?
    b. Suppose T is no longer the minimum-cost spanning tree. Give a linear-time algorithm (time O ($|E|$)) to update the tree T to the new minimum-cost spanning tree.

3. Let *G* = (*V*,*E*) be a connected undirected weighted graph. Assume for simplicity that the weights are positive and distinct. Let *e* be an edge of *G*. Denote by *T*(*e*) the spanning tree of *G* that has minimum cost among all spanning trees of *G* that contains *e*. Design an algorithm to find *T*(*e*) for all edges $e \in E$. The algorithm should run in time O ($|V|^2$).

4. An Euler circuit of an undirected graph *G*(*V*,*E*) is a path that starts and ends at the same node and contains each edge of G exactly once.
    a. Show that a connected, undirected graph has an Euler circuit if and only if each node is of even degree.
    b. Let *G* (*V*,*E*) be an undirected graph with m edges in which every node is of even degree. Give an O($|V|$) algorithm to construct an Euler circuit for G.
5. Let *G* = (*V*, *E*) be a directed weighted graph such that all weights are positive. Let *v* and *w* be two vertices of *G* and $k \leq |V|$ be an integer. Design an algorithm to find the shortest path from *v* to *w* that contains exactly k edges. The path need not be simple.