COMP 3711H – Fall 2014
Tutorial 7

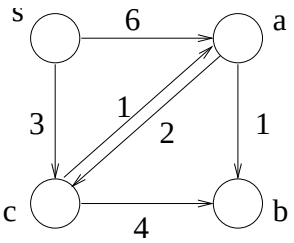
1. Given an undirected graph $G = (V, E)$, recall that the *complement*, $\overline{G}$, is a graph (V, E') such that for all $u \neq v$, $\{u, v\} \in E'$ if and only if $\{u, v\} \notin E$. Prove that either G or $\overline{G}$ is connected.

2. An (undirected) graph $G = (V, E)$ is *bipartate* if there exists some $S \subset V$ such that, for every edge $\{u, v\} \in E$, either (i) $u \in S$, $v \in V - S$ or (ii) $v \in S$, $u \in V - S$.

   Let $G = (V, E)$ be a connected graph. Design an $O(n + e)$ algorithm that checks whether G is bipartite. *Hint: Run BFS.*

   How can you modify your algorithm so that it also works for unconnected graphs?

3. Execute Dijkstra's algorithm on the following digraph, where s is the source vertex.



   You need to indicate only the following:

   (a) the order in which the vertices are removed from the priority queue?
   (b) the final distance values $d[]$ for each vertex?
   (c) the different distance values $d[]$ assigned to vertex b, as the algorithm executes.

4. Give a simple example of a directed graph with negative-weight edges for which Dijkstra's algorithm produces incorrect answers. Why does the correctness proof of Dijkstra's algorithm not go through when negative-weight edges are allowed?

5. Suppose that instead of using a heap to store the tentative vertex distances, Dijkstra's algorithm just kept an array in which it stored each vertex's tentative distance. It then finds the next vertex by running through the array and choosing the vertex with lowest tentative distance. What would the algorithm's running time be? Is this better than our implementation for some graphs?

6. Suppose that we are given a cable network of n sites connected by duplex communication channels. Unfortunately, the communication channels are not perfect. Each channel may fail with certain probability (also given in the input). The probabilities of failure may differ for different channels and they are mutually independent events. One of the n sites is the central station and your problem is to compute the most reliable paths from the central station to all other sites (i.e., the paths of lowest failure probabilities from the central station to all other sites). Design an algorithm for solving this problem, justify its correctness, and analyze its time and space complexities. Let $f(u, v)$ denote the failure probability for the channel between sites u and v.