# CS5050 Advanced Algorithms

Spring Semester, 2018

## Assignment 7: Graph Algorithms II

**Due Date: 3:00 p.m.**, Tuesday, Apr. 24, 2018 (**at the beginning of CS5050 class**)

**Note:** In this assignment, we assume all input graphs are represented by adjacency lists.

1. **(20 points)** Given a directed graph $G$ of $n$ vertices and $m$ edges, each edge $(u, v)$ has a weight $w(u, v)$, which can be positive, zero, or negative. The *bottleneck-weight* of any path in $G$ is defined to be the **largest** weight of all edges in the path. Let $s$ and $t$ be two vertices of $G$. A *minimum bottleneck-weight path* from $s$ to $t$ is a path with the smallest bottleneck-weight among all paths from $s$ to $t$ in $G$. Refer to Fig. 1 for an example.

   Modify Dijkstra's algorithm to compute a minimum bottleneck-weight path from $s$ to $t$. Your algorithm should have the same time complexity as Dijkstra's algorithm.

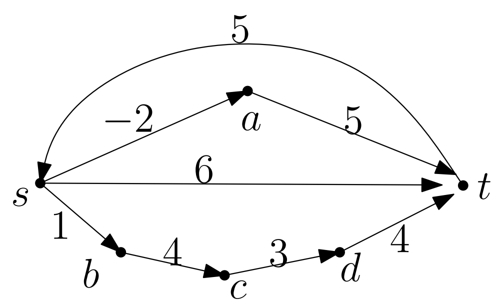


Figure 1: The following path is a minimum bottleneck-weight path from $s$ to $t$: $s, b, c, d, t$, whose bottleneck weight is 4.

2. Let $G = (V, E)$ be an undirected connected graph, and each edge $(u, v)$ has a positive weight $w(u, v) > 0$. Let $s$ and $t$ be two vertices of $G$. Let $\pi(s, t)$ denote a shortest path from $s$ to $t$ in $G$. Let $T$ be a minimum spanning tree of $G$. Please answer the following questions and explain why you obtain your answers.

   (a) Suppose we increase the weight of every edge of $G$ by a positive value $\delta > 0$. Then, is $\pi(s, t)$ still a shortest path from $s$ to $t$? **(10 points)**

   (b) Suppose we increase the weight of every edge of $G$ by a positive value $\delta > 0$. Then, is $T$ still a minimum spanning tree of $G$? **(10 points)**

   **Note:** For each of the two questions, your answer should be either "Yes" or "Not necessary". Again, please explain why you obtain your answers.

3. **(20 points)** Let $G = (V, E)$ be an undirected connected graph of $n$ vertices and $m$ edges. Suppose each edge of $G$ has a color of either *blue* or *red*. Design an algorithm to find a spanning tree $T$ of $G$ such that $T$ has as few red edges as possible. Your algorithm should run in $O((n + m)\log n)$ time.

**Total Points: 60**