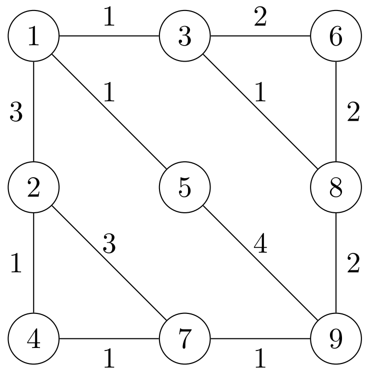
# COMPSCI330 Design and Analysis of Algorithms
# Assignment 5

Due Date: Thursday March 7, 2019

## Guidelines

- **Describing Algorithms** If you are asked to provide an algorithm, you should clearly define each step of the procedure, establish its correctness, 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 the problems to GradeScope. You will be asked to **label your solution for individual problems**. Failing to label your solution can cost you 5% of the total points (3 points out of 60 for this homework).
- 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 need more time for your homework please use this form and submit a STINF.
- **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** (Dijkstra)**.** (20 points) We are going to perform Dijkstra's algorithm on the following graph



This is an undirected graph, the numbers close to the edges are the edge weights. The numbers on the vertices are the indices of the vertices. We will perform Dijkstra using vertex 1 as the starting vertex.

(a) (5 points) Give the ordering in which the vertices are visited in Dijkstra's algorithm. (When there are multiple options, the algorithm chooses the vertex with smaller index.)

(b) (5 points) After vertex 7 is visited by Dijkstra's algorithm, what are the distance values for all the vertices (if a vertex cannot be reached yet its distance value is $+\infty$)?

(c) (10 points) If the edge weights are allowed to be negative, but the graph does not have any negative cycles, give an example where Dijkstra's algorithm fails to find the shortest path. You should specify the number of vertices, starting vertex, ending vertex and the set of edges (for each edge specify its starting vertex, ending vertex and weight). Your example should not contain more than 5 vertices.

**Problem 2** (Public Transit)**.** (20 points) Alice wants to go to Durham downtown by the public transit system. She has a transit map of the area, which in this problem is abstracted as a graph. Alice is at a vertex $s$ and she is going to vertex $t$. The edges on the graph represents possible bus schedules. Each directed edge $e = (u, v)$ means there is a bus that goes from $u$ to $v$, and the weight $w(u, v) > 0$ is the time that it takes to go from $u$ to $v$ using this bus. Taking each bus (going through each edge) costs \$1. Alice's goal is to find a path from $s$ to $t$ that (1) minimizes the amount of money spent; (2) among all paths that spend the minimum amount of money, choose the one that takes the minimum amount of time. Time to wait for buses is ignored in this problem.

(a) (10 points) Design an algorithm for Alice that finds the optimal path she wants. Your algorithm should run in $O(m + n \log n)$ time.

(Hint: One way to solve the problem is to modify the weight of the edges, so a path with $k$ edges is always shorter than a path with $k + 1$ edges, how do you do that?)

(b) (10 points) Prove the correctness of the algorithm that you designed in (a). You can use the correctness of Dijkstra's algorithm directly if you used Dijkstra's algorithm as a subroutine in your algorithm. However, if you modified Dijkstra's algorithm you will need to prove the correctness of the modified algorithm.

**Problem 3** (Minimum Average Length Cycle)**.** (20 points) Given a weighted directed graph $G = (V, E)$ where every edge $(u, v) \in E$ has a positive weight $w(u, v) > 0$. The goal of this problem is to find a cycle $(u_1, u_2, ..., u_k)$ in the graph such that the average edge length of the cycle, defined as

$$average((u_1, ..., u_k)) = \frac{w(u_k, u_1) + \sum_{i=1}^{k-1} w(u_i, u_{i+1})}{k},$$

is as small as possible. The average edge length is just the total length of edges in the cycle, divided by the number of edges. Note that the length of the cycle $k$ is not given, you can choose a cycle of arbitrary length as long as it has the minimum average edge length.

You can also assume that starting from a vertex $s$ in the graph, it is possible to reach all other vertices in the graph.

(a) (12 points) Given a number $c$, design an algorithm that decides whether the minimum average length cycle has an average length at least $c$.

(Hint: If you change the weight of edges to $w(u, v) - c$, what would happen to the weight of a cycle?)

(b) (8 points) Assume for simplicity that the average edge length of the minimum average length cycle is an integer between 0 and $W$. Design an algorithm that finds the minimum average edge length. Your algorithm should run in time $O(nm \log W)$.