max planck institut informatik

SAARLAND UNIVERSITY
COMPUTER SCIENCE

Karl Bringmann, Erik Jan van Leeuwen **Winter 2016/2017**

Exercises for Algorithms and Data Structures

http://www.mpi-inf.mpg.de/departments/algorithms-complexity/teaching/winter16/algorithms-and-data-structures/

Exercise Sheet 6 Due: **5.12.2016**

The homework must be handed in on Monday before the lecture. You may collaborate with other students on finding the solutions for this problem set, but every student must hand in a writeup in their own words. We also expect you to state your collaborators and sources (books, papers, course notes, web pages, etc.) that you used to arrive at your solutions.

You need to collect at least 50% of all points on the first six exercise sheets, and at least 50% of all points on the remaining exercise sheets.

Whenever you are asked to design an algorithm in this exercise sheet you have to give a proof of its correctness as well as an asymptotic upper bound on its worst case running time.

Exercise 1 (*10 points*)

a) (5 points) Give an example of a directed graph with at least one arc of *negative* length in which Dijkstra's Algorithm will fail to compute the shortest paths from some vertex v to any other vertex.

b) (5 points) Consider the following algorithm that claims to compute the single-source shortest path lengths from a vertex s to all vertices in a graph under length function ℓ. Initially, let $C = \{s\}$, $d(s) = 0$, and $d(v) = \infty$ for all $v \neq s$. Then, while $C \neq V$, perform the following operations: find an arc (x, y) of minimum length among all arcs (u, v) with $u \in C$ and $v \notin C$, add y to C, and set $d(y)$ to $d(x) + \ell(x, y)$. The algorithm outputs the table d.

Give an example of a graph where this algorithm fails to compute the single-source shortest path lengths from a designated vertex s. Your graph should have non-negative arc lengths, and every vertex should be reachable from s.

Exercise 2 (*10 points*)

Give an efficient algorithm that given a directed graph with arbitrary arc lengths decides whether the graph has a directed cycle in which all arcs have negative length.

Exercise 3 (*10 points*)

It is a well known that the problem of finding a path of *maximum* length in general graphs is NP-hard. In this exercise, however, you are given a directed *acyclic* graph G with n vertices and m arcs.

A *topological ordering* of G is a linear ordering of the vertices in V such that for every arc (u, v) in A, from vertex u to vertex v, u comes before v in the ordering.

a) (2 points) Show that every directed acyclic graph has a topological ordering.

b) (4 points) Give an $O(n + m)$ time algorithm to compute a topological ordering in G.

c) (4 points) Given the algorithm of part b), show that the length of a longest directed path between two nodes in G can be found in linear time, i.e. time $O(m + n)$.

Exercise 4 (*10 points*)

Let $G = (V, E)$ be an undirected graph with n vertices and $m > n$ edges. All edges in this graph have length 1. Let v be a vertex in this graph and assume that for every other vertex $x \in V$, a new edge (v, x) is added to the graph and each such new edge is assigned a (possibly different) integer length from the range $\{1, \ldots, n\}$.

Now you want to find the single-source shortest path lengths on this amended graph with start vertex v. Show that one can use the breadth-first-search algorithm to solve this problem in time $O(m + n)$.