Data Structures and Algorithms

COSC 336 Assignment 9

Instructions.

1. Due Dec. 3.
2. This is a team assignment. Work in teams of 3-4 students. Submit one assignment per team, with the names of all students making the team.
3. Your programs must be written in Java or C/C++.
4. Write your programs neatly - imagine yourself grading your program and see if it is easy to read and understand. At the very beginning present your algorithm in plain English or in pseudo-code (or both). Comment your programs reasonably: there is no need to comment lines like "i++" but do include brief comments describing the main purpose of a specific block of lines.
5. You will submit in class a printed document with the solutions to Problems 1 and 2 and with descriptions in English or in pseudocode of the algorithms for the programming task you are required to do and the results that you are required to report. Append to your document an annex containing the Java or C++ sources of your program. Also append images/screenshots with the output you obtain for each testing data.

   Staple all pages together. You should have the electronic copy of your programs with you (for example on a memory stick) because you may be asked to make a demo.

   For editing the above document. I recommend that you use Latex, see the template posted on the course website.

   http://orion.towson.edu/~mzimand/adatastruct/assignment-template.tex and

   http://orion.towson.edu/~mzimand/adatastruct/assignment-template.pdf

Problem 1 Modify the pseudo-code for the FLOYD-WARSHALL algorithm given in the textbook at the top of page 695 so that for all pairs (i, j) of nodes in the graph, a shortest path from node i to node j can be computed. More precisely, the modified algorithm should compute numbers $pred_{i,j}^{(k)}$, where $pred_{i,j}^{(k)}$ is defined to be the predecessor of j on a shortest path from i to j with all intermediate nodes in the set $\{1, \ldots, k\}$. (Look at equations 25.6 and 25.7 in the book, where they use π instead of $pred$). Write the pseudo-code of the modified algorithm.

Problem 2

For a certain application, you need to compute the shortest path for every pair (i, j) of vertices in a weighted graph G in which $m = n^{1.5}$ (as usual m is the number of edges and n is the number of vertices in G). All the edges have positive weights. You have

to determine which of Bellman-Ford, Dijkstra, or Floyd-Warshall algorithms to use, based on the running time of these algorithms. Which algorithm would you choose? Present a complete motivation for your choice.

Programming Task. The input is a directed graph with (possibly negative) weighted edges, a source node s, a target node t, and a positive integer k. The task is to compute the length of a shortest path from s to t that has at most k edges, or to report that there is no path from s to t with k edges. For example, for the graph in Figure 25.2, the shortest path from 3 to 1 with $k = 2$ edges is 3-2-1 (note that there is a shorter path 3-2-4-1, but this one has 3 edges, and so it does not satisfy the constraint of having at most 2 edges.).

Your algorithm should run in time $O(k|E|)$.

Describe briefly your algorithm (which is a modification of one of the algorithms that we discussed in class, and so you only need to explain what modification you did.)

Run your program on the graph in Figure 25. 2, page 691 in the textbook, for

(1) $s = 6, t = 1, k = 3$

(2) $s = 3, t = 1, k = 2$

(3) $s = 3, t = 1, k = 3$

and show screen shots with the results for all three test cases.