CS 211 Data Structures and Algorithms Lab
July -- December, 2018
Assignment 8
Total Marks: 10
Due on 24th October

The objective of this assignment is to implement Breadth First Search (BFS) to solve single-source shortest path problem on directed graphs and to implement Depth First Search (DFS) to do topological sorting of a Directed Acyclic Graph (DAG).

Inputs

Your program should accept an input file and an integer as command-line arguments. A typical execution of your program will be ./a.out sample.graph 25

The input file represents a directed graph. Every node (vertex) in the graph is uniquely labelled with a non-negative integer. Every line in the input file is of the form x y, which represents a directed edge from node x to node y. No directed edge is repeated in the input file. Since topological sorting makes sense only for DAGs, it is guaranteed that the input graph is a DAG.

The second command-line argument is an integer, which represents the label of a vertex in the given graph, which is the source from which the shortest distance of every vertex has to be calculated using BFS. Note that this argument is irrelevant for doing topological sorting and can be ignored for the same.

Your program should create an adjacency list of the input graph which can be used for both Task 1 and Task 2.

Task 1 (5 Marks)

Implement BFS and use it to find the shortest distance of every vertex from the source vertex (second command-line argument). The output file should be named as ‘shortest_distance.txt’. Every line in the output file should be of the form <vertex-label> <shortest-distance-from-source>. For example if the shortest distance of vertex with label 35 from the source vertex is 10, then there is a line in the output file which has ‘35 10’. The vertex-distance pair in the output file can be ordered in any fashion. If there is no directed path from the source to a vertex, then the corresponding distance should be denoted by -1.

Task 2 (5 Marks)

Implement DFS and use it to do a topological sorting of the input DAG. The output file should be named as ‘topological_sort.txt’. The output file must contain the vertices - one vertex per line - which represents a topological sorting of the input DAG.

Submission and Evaluation

- The program you submit should output two files named 'shortest_distance.txt' and 'topological_sort.txt' when run.
- There should be only one main file and it should be named as main.<extension>, where the extension depends on the language you choose (You must use either C or C++).
- Test well before submission. We have some hidden inputs with us to test your program. The mark you obtain is purely based on whether your program correctly gives outputs for the hidden inputs.
- Submit your code as a zip file (even if there is only one file) where the name of the zip file is your roll number**.** It is important that you follow the input/output conventions exactly (including the naming scheme) as we may be doing an automated evaluation.
- This assignment is due on 24th October. Penalty for late submission is 5% per week; i.e., if you submit on 26th October, you will get only 95% of the mark you deserve otherwise.
- Follow some coding style uniformly. Provide proper comments in your code.
- Submit only through Moodle. Submit well in advance. Any hiccups in the Moodle/internet at the last minute is never acceptable as an excuse for late submission.
- Acknowledge the people (other than the instructor and TAs) who helped you to solve this assignment. The details of the help you received and the names of the people who helped you (including internet sources, if applicable) should come in the main file or in a separate file (acknowledge.txt). Copying others' programs is a serious offence.
- Honesty policy of the institute will be strictly followed. Note that we have access to a very good software to check plagiarism.