CSE 332 Data Abstractions, Winter 2015

Homework 5

Due: **Friday, Feb 20, 2015** at 23:00 (11:00pm) via the catalyst drop box. You should refer to the written homework guidelines on the course website for a reminder about what is acceptable pseudocode. You may enjoy the fact that homework Five has THREE (wait really, only 3?) thrilling questions!!

Submission instructions

Submit an electronic copy to the catalyst dropbox as a PDF file. You can either do the assignment on an electronic word processor (and convert to PDF) or do it on physical paper and scan it (or take a high res photo) and upload a single PDF of the file. It will be much easier to grade if every question starts on a separate page. Don't forget to put your name on the top of the first page.

Problem 1: Graph Representations (1 page)

Suppose a **directed** graph has a million nodes, most nodes have only a few edges, but a few nodes have hundreds of thousands of edges:

a) In what way(s) would an adjacency-matrix representation of this graph lead to inefficiencies?

b) In what way(s) would an adjacency-list representation of this graph lead to inefficiencies?

c) Design a representation for this sort of graph that avoids all the inefficiencies in your answers to parts (a) and (b).

Problem 2: Topological Sort (1 page)

Weiss, problem 9.1 (the problem is the same in the 2nd and 3rd editions of the textbook): "Find a topological ordering for the graph in figure 9.79 (2nd ed.)/9.81 (3rd ed.)." **For each step**, show the in-degree array and the queue in the table format shown below: Don't forget to state your final answer - the topological ordering. You should break ties using alphabetical ordering. Please use the template file posted on the homework page.

In-degree Array:

	s	A	B	C	D	E	F	G	H	I	t	Current Contents of Queue (Front, … , Back)
Step 1												
Step 2												
Etc.												

Please turn over for problem 3!!!

Problem 3: Dijkstra's Algorithm

a) Weiss, problem 9.5(a) (the problem is the same in the 2nd and 3rd editions of the textbook). Use Dijkstra's algorithm and show the results of the algorithm in the form used in lecture — a table showing for each vertex its best-known distance from the starting vertex and its predecessor vertex on the path. Also show the **order** in which the vertices are added to the "cloud" of known vertices as the algorithm progresses.

   Although the final table is all that is required, for potential partial credit I would recommend showing how your table changes over time by leaving lots of space and crossing things out as they change. Also be sure that you show the value of the path variable.

b) If there is more than one minimum cost path from v to w, will Dijkstra's algorithm always find the path with the fewest edges? If not, explain in a few sentences how to modify Dijkstra's algorithm so that if there is more than one minimum path from v to w, a path with the fewest edges is chosen. Assume no negative weight edges or negative weight cycles.

c) Give an example where Dijkstra's algorithm gives the wrong answer in the presence of a negative-cost edge but no negative-cost cycles. Explain briefly why Dijkstra's algorithm fails on your example. The example need not be complex; it is possible to demonstrate the point using as few as 3 vertices.

d) Suppose you are given a graph that has negative-cost edges but no negative-cost cycles. Consider the following strategy to find shortest paths in this graph: uniformly add a constant k to the cost of every edge, so that all costs become non-negative, then run Dijkstra's algorithm and return that result with the edge costs reverted back to their original values (i.e., with k subtracted).

   - Give an example where this technique fails (Dijkstra's would not find what is actually the shortest path) and explain why it fails.

   - Also, give a general explanation as to **why** this technique does not work. Think about your example and why the original least cost path is no longer the least cost path after adding k.