School of Computer Science

Carleton University

Name: **A.M.** Problem Set: **4**
Course: **COMP 3804** Week: **1-7**

Problems

Consult [1, 2, 3]. The problem set consists of problems that have been asked in Test 2 and problems on SSSP and MST in graphs.

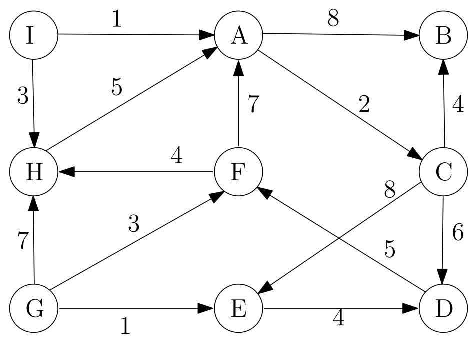
Problem 1. *Let $G = (V, E)$ be a directed graph consisting of 7 vertices with the following adjacency list representation:*

$A \to C, D, F$	$B \to A, F, G$
$C \to D, G$	$D \to E, F, G$
$E \to F, G$	$F \to G$
$G \to$	

Answer the following questions.

1. *Perform a depth first search traversal of G and determine if G has a backedge.*
2. *Is G a directed acyclic graph? Justify your answer.*

Problem 2. *Using Dijkstra's single source shortest path algorithm, compute the shortest path distances from vertex A to all the vertices in the following graph.*



Problem 3. *Let G be a connected weighted undirected graph $G = (V, E)$? Execute the following algorithm and answer whether the graph T returned by the algorithm is a minimum spanning tree of G. Justify your answer.*

Input: *A connected undirected graph* $G = (V, E)$*, where each edge* $e \in E$ *has a positive weight* $w(e) > 0$*. (You may assume that no two edges have the same weight.)*
Output: *A subgraph* T *of* G*.*

Step 1: *Sort the edges of* G *with respect to decreasing weight.*

Step 2: $T := E$.

Step 3: *For each edge* e *taken in the order of decreasing weight do:*
if $T - \{e\}$ *is connected, discard* e *from* T*.*

Step 4: *Return* T*.*

Note: For an edge $e = (uv) \in E$*, the operation* $T - \{e\}$ *discards only the edge* e*, but not the vertices* u *and* v*, from* T*.*

Problem 4. *Let* $G = (V, E)$ *be a directed acyclic graph, where the weight of each edge* $e \in E$ *is a real number. Note that some edges of* G *may have negative weight. Assume that* G *is represented in the adjacency list format, and there is a vertex* $s \in V$ *such that every vertex* $v \in V - \{s\}$ *can be reached from* s *by a directed path in* G*. Design an algorithm that computes the shortest path distances (even when some of the edges may have negative weights) from* s *to all the vertices in* G *in* $O(|V| + |E|)$ *time. Present the pseudocde of your algorithm. Show that the algorithm terminates, it is correct, and analyze its running time.*
(Hint: Think of processing the vertices of G *in the topological sorted order.)*

Problem 5. *Consider the same graph as in Problem 2. Ignore the directions on edges, and consider an undirected version of that graph. Execute Kruskal's and Prim's MST algorithm on this graph.*

Problem 6. *Given an undirected connected graph* $G = (V, E)$*, in adjacency list representation, can it be decided within* $O(|V| + |E|)$ *time whether there is a path between two specific vertices* x *and* y *consisting of at most 50 edges, where* $x, y \in V$*?*

Problem 7. *Can you devise a faster algorithm for computing single source shortest path distances from the source vertex* s *in an undirected graph* $G = (V, E)$ *when all the edge weights are 1? (Think of an algorithm that runs in* $O(|V| + |E|)$ *time on a graph* $G = (V, E)$*.)*

Problem 8. *Let* $G = (V, E)$ *be a weighted directed graph, where the weight of each edge is a positive integer and is bounded by a number* X*. Show how shortest paths from a given source vertex* s *to all vertices of* G *can be computed in* $O(X|V| + |E|)$ *time.*

Problem 9. *Let* T *and* S *be two minimum cost spanning tree of a graph* $G = (V, E)$ *on* n *vertices. Let* $c_1 \leq c_2 \leq ... \leq c_{n-1}$ *be the costs of the edges in* T*, and let* $d_1 \leq d_2 \leq ... \leq d_{n-1}$ *be the costs of the edges in* S*. Show that that* $c_i = d_i$*, for* $1 \leq i \leq n - 1$*.*

Problem 10. *Let* A *and* B *be two connected undirected weighted graphs. Let the minimum spanning trees of* A *and* B *be* T_A *and* T_B*, respectively. Show that the weight of MST of* $A \cup B$ *is same as the weight of MST of* $T_A \cup T_B$*. If required, we can assume that edges have distinct weights.*

References

[1] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms.* 3rd. ed., MIT Press, 2009.

[2] S. DasGupta, C. Papadimitriou, V. Vazirani. *Introduction to Algorithms.* McGraw Hill.

[3] A. Maheshwari. *Notes on Algorithm Design*, Chapter 1, `https://people.scs.carleton.ca/~maheshwa/Notes/DAA/notes.pdf`