Assignment 2

COMP 3804, Fall 2022

Upload in Brighspace by 11:59 PM on November 10, 2022

1 Guidelines

General guidelines are as follows:

1. Since we are only accepting assignments via the Brighspace system, and we will discuss the solutions immediately after the due date, no late submissions will be entertained after the cut-off time & date.
2. Please write clearly and answer questions precisely. It is your responsibility to ensure that what is uploaded is readable. If we can't read, we can't mark!
3. Please cite all the references (including websites, names of friends, etc.) that you used/consulted as the source of information for each of the questions.
4. All questions carry equal marks.
5. When a question asks you to design an algorithm - it **requires** you to
   (a) Clearly spell out the **steps** of your algorithm as a pseudo code.
   (b) **Prove** that your algorithm is correct
   (c) **Analyze** the running time.

2 Problems

1. For the undirected graph in Figure 1, perform a depth-first search starting at the vertex marked A, and whenever there is a choice of vertices, pick the one that is lexicographic smallest. Classify each edge as a tree edge or a back edge.

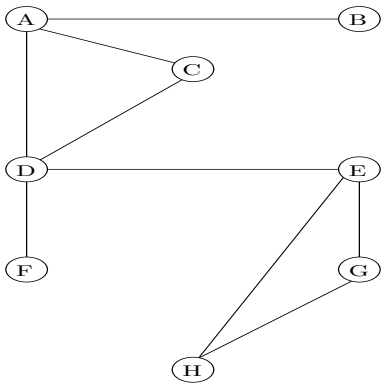

Figure 1: Graph for DFS traversal

2. Let $G = (V, E)$ be a simple undirected graph given in the adjacency list representation. Provide an algorithm running in $O(|V| + |E|)$ time that outputs whether G contains an even cycle. If G contains an even cycle, we need to output one such cycle.
   Note: A cycle of length $k > 2$ in a graph is a sequence of k distinct vertices, say without loss of generality $v_1, v_2, \ldots, v_k$, such that $\{v_1v_2, v_2v_3, \ldots, v_{k-1}v_k, v_kv_1\} \subseteq E$. We say a cycle is even if k is even.

3. For the directed graph G in Figure 2, classify each edge's type (tree, forward, backward, cross). First, represent this graph in the adjacency list format, and execute the DFS algorithm. Justify the classification of edges based on the DFS traversal that you performed on G.

4. Let $G = (V, E)$ be a directed graph. Suppose we have performed a DFS traversal of G, and for each vertex v, we know its *pre* and *post* numbers. Show the following:

   (a) If for a pair of vertices $u, v \in V$, $pre(u) < pre(v) < post(v) < post(u)$, then there is a directed path from u to v in G.

   (b) If for a pair of vertices $u, v \in V$, $pre(u) < post(u) < pre(v) < post(v)$, then there is no directed path from u to v in G.

5. Show that in a DAG (directed acyclic graph) G, if there is a directed path from u to v, then $post(u) > post(v)$. Does this statement hold for any directed graph that contains directed cycles?

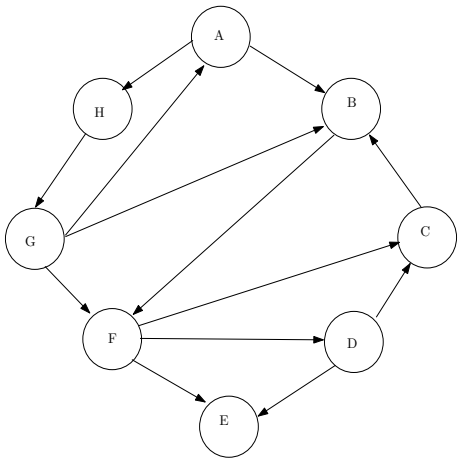

Figure 2: Graph for DFS traversal

6. Let G be a $n \times n$ integer grid graph, where n is a positive integer. Each vertex of G is specified by its coordinate (i, j), where $1 \leq i \leq n$ and $1 \leq j \leq n$. The neighbors of a vertex (i, j) are the vertices $(i-1, j)$, $(i+1, j)$, $(i, j-1)$, and $(i, j+1)$ (if they exist). Each vertex holds a unique number. We say a vertex (i, j) is dominant if the number stored at that vertex is larger than the numbers stored at all of its neighbors. For an illustration, see Figure 3. Answer the following:

   (a) Show that G always contains a dominant vertex.

   (b) Design an algorithm running in $O(n)$ time to report a dominant vertex in G. Note that G has $\Theta(n^2)$ vertices, and we are asking for an algorithm with running time $O(n)$.

7. Let $G = (V, E)$ be a DAG. We say that G is *semi-connected* if for every pair of distinct vertices $u, v \in V$, we have a directed path from u to v or there is a directed path from v to u in G. Given G in the adjacency list representation, design an algorithm running in $O(|V|+|E|)$ time to determine whether G is semi-connected. (Hint: First, construct examples of directed acyclic graphs on four vertices that are semi-connected and that aren't. Determine what property (with respect to their linear order) distinguishes them.)

8. Let $G = (V, E)$ be a directed graph. The input to the problem consists of G in the adjacency list format, a pair of vertices $s, t \in V$, and an integer $k > 0$. Design an algorithm that determines whether there is a directed path in G from s to t consisting of at most k edges. Your algorithm must run in $O(|V| + |E|)$ time.

9. In Internet routing, there are delays on edges and significant delays on switches (i.e. vertices). Suppose that in addition to having non-negative edge weights in the graph $G = (V, E)$, we have a positive cost associated with each vertex in G. Let the cost

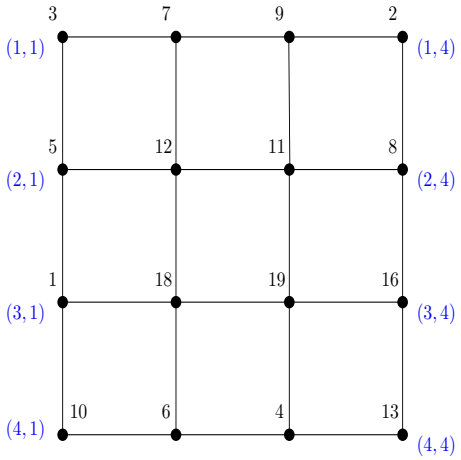

Figure 3: 4×4 grid graph G. Vertex coordinates are in blue color. Neighbors of vertex $(1, 1)$ are $(1, 2)$ and $(2, 1)$. Vertex at coordinate $(3, 3)$ with value 19 is a dominant vertex of G. Vertex $(4, 1)$ is also a dominant vertex.

associated to a vertex v be $c(v)$. The cost of a path is defined as the sum total of the weights of the edges in the path + the sum total of the costs of all the vertices in the path. Present an efficient algorithm to compute the shortest path costs from vertex s to all the vertices in the graph with this modified definition of the path's cost. Present a Pseudocode - Analyze the Complexity and - Justify why your algorithm is correct. You can assume G is a directed graph represented in the adjacency list format.

10. Given a directed acyclic graph $G = (V, E)$, where each vertex has a distinct integer label. For each vertex $v \in V$, define $R(v)$ to be the set of all the vertices $w \in V$ for which there is a directed path from v to w in G. Furthermore, for each vertex $v \in V$, define $\mathsf{MinLabel}(v)$ to be the vertex with the minimum label in the set $R(v)$. Provide an algorithm, running in $O(|V| + |E|)$ time, that computes $\mathsf{MinLabel}(v)$ for all vertices $v \in V$.