Practice Problems 2

Problem 1. In this problem you will simulate the strong-components algorithm given in class on the digraph shown in Fig. 1.

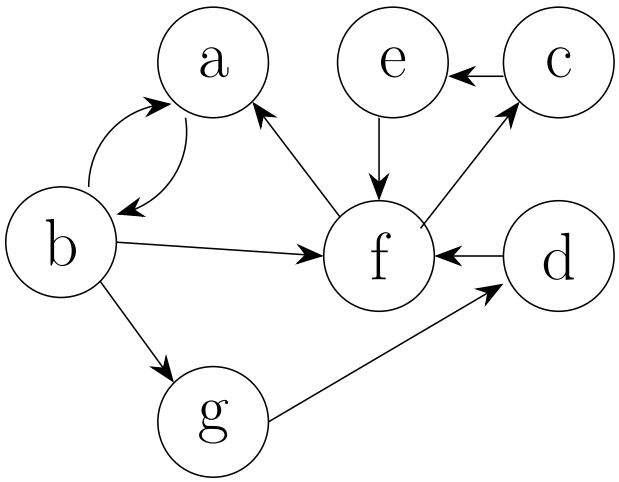

Figure 1: A directed graph

(a) Show the result of applying DFS to G. Label each vertex u with its discovery and finish times ($d[u]/f[u]$). You need only show the *final* DFS tree, not the intermediate results. Whenever you have a choice of which vertex to visit next, select the lowest vertex in alphabetic order.

(b) Draw the reverse graph G^R. (Be careful to note the directions of the edges.)

(c) Show the result of running DFS on G^R, where the main program selects the next vertex to visit based on decreasing order of finish times from part (a).

(d) Illustrate (e.g., by circling them or listing them out) the strong components of G.

Problem 2. A **source** of a directed graph is a node with no incoming edges. A **sink** of a directed graph is a node with no outgoing edges.

(a) Prove that if $G = (V, E)$ is a directed acyclic graph, G contains at least one source and at least one sink.

(b) Further prove that if $G = (V, E)$ is a directed acyclic graph, then for every vertex v there exists a source s and a sink t such that there exists a path from s to v and a path from v to t.

Problem 3. For both parts below, assume that graphs and digraphs are represented using an adjacency list. Given a graph or digraph $G = (V, E)$, let $n = |V|$ and $m = |E|$.

(a) Present an algorithm which, given an undirected graph $G = (V, E)$, determines whether G contains a cycle in $O(n)$ time. If the graph has a cycle, it should output the vertices of any cycle.

It is important that the running time of your algorithm is independent of the number of edges m, which may generally be as high as $\Omega(n^2)$. This means that your algorithm will need to terminate with the correct answer, possibly even before reading the entire adjacency list. You may assume that the graph is presented to your algorithm in constant time as reference to its existing adjacency list.

(b) Prove that there is no corresponding algorithm for digraphs. In particular, prove that any algorithm that correctly determines whether a digraph has a cycle may need to inspect $\Omega(n^2)$ edges.

Hint: Prove this by contradiction. Suppose that such an algorithm existed. Show that there exists a digraph having $\Omega(n^2)$ edges, such that any correct algorithm must inspect *every* edge of this graph.

Problem 4. In this problem, we seek to characterize graphs that have only one topological ordering. A **Hamiltonian path** is a path that contains every vertex in a graph. Finding a Hamiltonian Path in a general undirected graph is a hard problem, but we will show that it is easier.

(a) Show that if $G = (V, E)$ is a DAG with a Hamiltonian path, then it has only one valid topological ordering.

(b) Show the converse: that if $G = (V, E)$ is a DAG with only one valid topological ordering, then it has a Hamiltonian path.

(c) Provide an algorithm for determining whether a DAG $G = (V, E)$ has a Hamiltonian path. Your algorithm should run in $O(n + m)$ time, where $|V| = n, |E| = m$.