1 Short Answer

For each of the following, either prove the statement is true or give a counterexample to show it is false.

(a) If \((u, v)\) is an edge in an undirected graph and during DFS, \(\text{post}(v) < \text{post}(u)\), then \(u\) is an ancestor of \(v\) in the DFS tree.

(b) In a directed graph, if there is a path from \(u\) to \(v\) and \(\text{pre}(u) < \text{pre}(v)\) then \(u\) is an ancestor of \(v\) in the DFS tree.

(c) In any connected undirected graph \(G\) there is a vertex whose removal leaves \(G\) connected.

2 Graph Traversal

(a) For the directed graph above, perform DFS starting from vertex A, breaking ties alphabetically. As you go, label each node with its pre- and post-number, and mark each edge as Tree, Back, Forward or Cross.

(b) What are the strongly connected components of the above graph?

(c) Draw the DAG of the strongly connected components of the graph.

3 Connectivity vs Strong Connectivity

(a) Prove that in any connected undirected graph \(G = (V, E)\) there is a vertex \(v \in V\) such that removing \(v\) from \(G\) gives another connected graph.
(b) Give an example of a strongly connected directed graph \( G = (V, E) \) such that, for every \( v \in V \), removing \( v \) from \( G \) gives a directed graph that is not strongly connected.

(c) Let \( G = (V, E) \) be a connected undirected graph such that \( G \) remains connected after removing any vertex. Show that for every pair of vertices \( u, v \) where \( (u, v) \not\in E \) there exist two different \( u-v \) paths.

4 Updating Labels

You are given a tree \( T = (V, E) \) with a designated root node \( r \), and a non-negative integer label \( l(v) \). If \( l(v) = k \), we wish to relabel \( v \), such that \( l_{\text{new}}(v) \) is equal to \( l(w) \), where \( w \) is the \( k \)th ancestor of \( v \) in the tree. We follow the convention that the root node, \( r \), is its own parent. Give a linear time algorithm to compute the new label, \( l_{\text{new}}(v) \) for each \( v \) in \( V \).

Slightly more formally, the parent of any \( v \neq r \), is defined to be the node adjacent to \( v \) in the path from \( r \) to \( v \). By convention, \( p(r) = r \). For \( k > 1 \), define \( p^k(v) = p^{k-1}(p(v)) \) and \( p^1(v) = p(v) \) (so \( p^k \) is the \( k \)th ancestor of \( v \)). Each vertex \( v \) of the tree has an associated non-negative integer label \( l(v) \). We want to find a linear-time algorithm to update the labels of all vertices in \( T \) according to the following rule: \( l_{\text{new}}(v) = l(p^{l(v)}(v)) \).