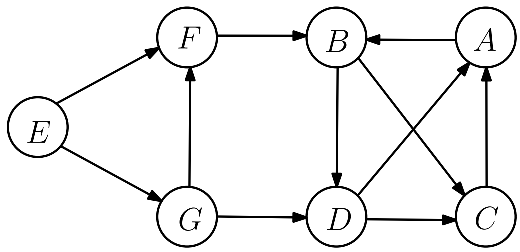
COMP 3804 — Tutorial February 14

Algorithm DFS(G):
for each vertex v
do $visited(v) = false$
endfor;
$clock = 1$;
for each vertex v
do if $visited(v) = false$
    then Explore(v)
    endif
endfor

Algorithm Explore(v):
$visited(v) = true$;
$pre(v) = clock$;
$clock = clock + 1$;
for each edge (v, u)
do if $visited(u) = false$
    then Explore(u)
    endif
endfor;
$post(v) = clock$;
$clock = clock + 1$

Problem 1: Consider the following directed graph:



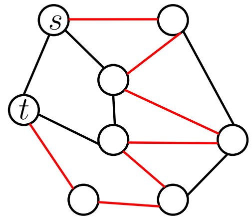
(1.1) Draw the *DFS*-forest obtained by running algorithm DFS. Classify each edge as a tree edge, forward edge, back edge, or cross edge. In the *DFS*-forest, give the *pre*- and *post*-number of each vertex. Whenever there is a choice of vertices, pick the one that is alphabetically first.

(1.2) Draw the *DFS*-forest obtained by running algorithm DFS. Classify each edge as a tree edge, forward edge, back edge, or cross edge. In the *DFS*-forest, give the *pre*- and *post*-number of each vertex. Whenever there is a choice of vertices, pick the one that is alphabetically last.

Problem 2: Let $G = (V, E)$ be a directed acyclic graph, and let s and t be two vertices of V.

Describe an algorithm that computes, in $O(|V|+|E|)$ time, the number of directed paths from s to t in G. As always, justify your answer and the running time of your algorithm.

Problem 3: A *Hamilton path* in an undirected graph is a path that contains every vertex exactly once. In the figure below, you see a Hamilton path in red. A *Hamilton cycle* is a cycle that contains every vertex exactly once. In the figure below, if you add the black edge $\{s, t\}$ to the red Hamilton path, then you obtain a Hamilton cycle.



If $G = (V, E)$ is an undirected graph, then the graph G^3 is defined as follows:

1. The vertex set of G^3 is equal to V.
2. For any two distinct vertices u and v in V, $\{u, v\}$ is an edge in G^3 if and only if there is a path in G between u and v consisting of at most three edges.

Question 3.1: Describe a *recursive* algorithm HAMILTONPATH that has the following specification:

Algorithm HamiltonPath(T, u, v)**:**
Input: A tree T with at least two vertices; two distinct vertices u and v in T such that $\{u, v\}$ is an edge in T.
Output: A Hamilton path in T^3 that starts at vertex u and ends at vertex v.

Hint: You do not have to analyze the running time. The base case is easy. Now assume that T has at least three vertices. If you remove the edge $\{u, v\}$ from T, then you obtain two trees T_u (containing u) and T_v (containing v).

1. One of these two trees, say, T_u, may consist of the single vertex u. How does your recursive algorithm proceed?
2. If each of T_u and T_v has at least two vertices, how does your recursive algorithm proceed?

Question 3.2: Prove the following lemma:

Lemma: For every tree T that has at least three vertices, the graph T^3 contains a Hamilton cycle.

Question 3.3: Prove the following theorem:

Theorem: For every connected undirected graph G that has at least three vertices, the graph G^3 contains a Hamilton cycle.