Algorithms Lab 9

csci 2200, Laura Toma. Bowdoin College

In lab: (COLLABORATION LEVEL 0: EVERYTHING ALLOWED)

1. Continue working on the "Applications of BFS and DFS" handout. It is important to go over all problems and discuss with your group your ideas and solutions.

2. Suppose you have an undirected graph given as an adjacency list, and you want to figure out if they form a *tree* (a tree is an undirected graph that's connected and has no cycles). Describe an algorithm to solve this problem, and analyze its running time.

3. The two-colorability problem from handout: Is it possible that the vertices of a given graph be assigned one of two colors, such that no edge connects vertices of the same color? (Note: this is equivalent to the question: is G bipartite?)) Describe an algorithm to solve this problem, and analyze its running time.

4. Draw a small DAG (< 10 vertices), perform a DFS on G and mark the finish times of all vertices. Now consider an arbitrary edge (u, v): what can you say about the finish time of u compared to the finish time of v? Can you prove it?

Homework problems (COLLABORATION[1] LEVEL:1)

1. (4.2.31 Sedgewick Wayne) In a directed graph, two vertices u and v are said to be in the same *strongly connected component (SCC)* if u can reach v and v can reach u.

   (a) Describe a linear time algorithm for computing the *strong component* containing a given vertex v.

   (b) On the basis of that algorithm, describe a simple quadratic time algorithm for computing the strong components of a directed graph G.

2. (CLRS 22.4-2) Give a linear-time algorithm that takes as input a directed acyclic graph $G = (V, E)$ and two vertices s and t, and returns the number of simple paths from s to t in G. For example, the DAG in Fig. 22.8 CLRS contains exactly four simple paths from p to v: pov, $poryv$, $posryv$ abd $psryv$. Your algorithms needs to only count the simple paths, not list then.

   Hint: dynamic programming on DAGs.

3. Assume you are given a DAG, and you want to compute "longest" paths; the edges do not have weights, and we use the standard convention that the length of a path is the number of edges on the path.

   (a) Describe how to compute the longest path in a DAG starting from a specified vertex.

   (b) Describe how to compute the longest path in a DAG.

   Hint: dynamic programming on DAGs.

4. (4.2.32 Sedgewick Wayne) (Hamiltonian paths in DAGs) Given a DAG, design a linear time algorithm to determine whether there is a directed path that visits each vertex exactly one.

[1] Collaboration level 1: verbal collaboration without solution sharing. You are allowed and encouraged to discuss ideas with other class members, but the communication should be verbal and additionally it can include diagrams on board. Noone is allowed to take notes during the discussion (being able to recreate the solution later from memory is proof that you actually understood it). Communication cannot include sharing pseudocode for the problem. Check complete guidelines at: https://turing.bowdoin.edu/dept/collab.php