Algorithms Lab 9

In lab

1. Finish "Applications of BFS and DFS" handout.
2. 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? What does this mean as far as a topological order of G is concerned?

Homework

1. (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. Check SSSP chapter on shortest paths on DAGs.

2. Assume you are given a DAG, and you want to compute "longest" paths; the edges do not have weights, and we use the sttandard 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.

3. (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.

4. (4.2.31 Sedgewick Wayne) Describe a linear time algorithm for computing the strong component containing a given vertex v. On the basis of that algorithm, describe a simple quadratic time algorithm for computing the strong components of a digraph.