Cpt S 350 Homework #7

1. Describe an algorithm that answers the following question: Given: a graph G = (V, E) Question: is G a DAG? What is the worst-case complexity of your algorithm?

Answer:

G is a DAG iff G does not have a loop, Run SCC to verify this. Complexity: linear – O(V + E).

2. Let G be a decorated graph. That is, each edge in G is labeled with a color drawn from a finite set C. Let v and v 0 be two given nodes in G. Design an algorithm that answers the following question: is it true that, for every path (a path could contain loops) from v to v 0 , the number of red edges is greater than the number of green edges?

Answer:

Construct a machine M from G as follows. M is armed with an int var ‘i’. Nodes in G are considered states in M. One run of M corresponds a walk over G. Initially, M starts with state v and with i equals 0. M walks on the graph while changing its state (for instance, if M’s current state is v1 and, then, when it walks on the edge (v1, v2), the next state of M will be v2). M increments/decrements ‘i’ by one when a red or green edge is passed. M halts when the node v ′ is the current state and i greater than or equal to zero. M halts iff false. Use PDA algorithm to check whether a PDA represented as M halts and gives us the solution to the problem.

3. Let G be a decorated graph. That is, each edge in G is labeled with a color drawn from a finite set C. Let v and v 0 be two given nodes in G. Design an algorithm that answers the following question: is it true that, for every path (a path could contain loops) from v to v 0 , the number of red edges and the number of green edges are both at least 4?

Answer:

Run DFS algorithm over G to determine if we have path from v to v’ where number of green edges is 0, 1, 2, or 3 or the same for red edges. If the algorithm returns false, then we have a path from v to v’.

4. Let G be an activity graph. That is, each edge in G is labeled with an activity drawn from a finite set A. Each node in G is also called a state. Let s0 be a given initial state. In the definition of G, each node is either marked with good or not-good. Recall that a liveness property is to argue that something good will eventually happen. That is, it is not true that, from s0, there is an infinite path on G on which every state is marked with not-good. Design an algorithm to decide whether G satisfies the liveness property or not.

Answer:

Fact: infinite path must contain loops.

We must determine: there is a finite path q on G such that q starts with s0 and q contains a loop and every state on q is marked with not-good.

Alg: Eliminate all ‘good’ states from graph G. Run SCC on G. For each state r in G do the following: DFS to check whether s0 can reach r

check whether there is an edge from r to r in G or r is contained in a SCC that has at least two nodes if both checking’s are yes, return yes.

Else return no.

5. Let G be a graph where each edge is colored and multiple edges can share the same color. We are given three distinct nodes v1, v2 and v3 (the graph could have many nodes). The graph is good if there is an infinite walk from v1 that passes v2 for only finitely many times and passes v3 for infinitely many times and, after certain point on the walk, the walk only contains either red degs or green edges. Design an algorithm that decides whether a graph is good.

Answer:

We can run SCC on the graph initially to determine if the nodes form loops. Since we are focusing on v1, v2, v3 we can check the paths using DFS and see if there is a loop where two nodes can reach each other infinitely whilst making the loop touching the intermediate node v2. Checking the edge to determine red or green will be easy to factor since we drop check the liveliness factor the same as in problem four. Run SCC after this check and the return is Yes then graph is good otherwise it is false.