Eric Yang

304263623

CS 180

Dis 1B

Homework 4

1. Given an array of people N

Copy N into another array M

For each person X in M

For each person Y in M

If X knows Y

X is not the celebrity, remove from array M

Else

Y is not the celebrity, remove from array M

If M is not empty

For each person X in N

If X does not know person Y in M

There is no celebrity

Else there is no celebrity

1. A. We can test the base case of a DAC with two nodes X and Y. The DFS would order Y (1) first then X (2). There is a directed edge (X,Y) and number assigned to X (2) is indeed higher than number assigned to Y (1).

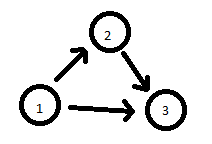
Next we have a DAC with N nodes. We assume that the condition that all nodes are in reverse topological ordering when DFS is run is true for this graph. If we add a node to this DAC, it now has N+1 nodes. If we do a DFS on this graph, the new node N+1 would get a smaller number than its parent node X. There is an edge (X,N+1) and the number given to N+1 is smaller than number of X. Thus the condition that DFS on a DAC gives reverse topological ordering must be true.

B. If A is a strongly connected component, we can isolate A and imagine if it were a directed graph itself. Now if we run DFS on graph A, the first vertex visited will be the vertex at the end of the reverse topological ordering. Thus it must be true that the number assigned to the first vertex of A has a higher number than any other vertex in A.

C. We can find the maximum of B using the idea of the previous proof so the maximum is the first vertex in B. Also, we use the idea we proved in the first part that if there is an edge (X,Y) the number of X must be higher than the number of Y. Thus if there is an edge from a node in A to a node in B, there must be an edge from node in A to maximum node in B. number of node in A is higher than maximum node in B so maximum of A must also be higher.

1. A. Let’s assume that there is an odd cycle in the graph and that the graph is colored in red and blue such that no edge connecting blue to blue or red to red exists. Let’s label the nodes of this odd cycle N1, N2, N3 all the way to Nx. We color N1 blue and N2 red and so on. Nx must be colored blue since x is odd due to the odd cycle. However, N1 and Nx have an edge between them and they are both blue. Thus there is a contradiction that tells us there is an odd cycle if and only if there is an edge connecting red to red or blue to blue.

B. In a strongly connected graph, each vertex is reachable from every other vertex. If the graph is not strongly connected, we can’t assume that a cycle even exists. Thus if a color conflict does happen, we can’t assume there even is a cycle at all. An example of such a graph would be:



If we color 1 blue, then we must color 2 and 3 red. There is an edge from 2 to 3 so there is color conflict; however, there is no cycle here at all in this graph.

C. Use DFS to assign post numbers to each vertex

Start another DFS at first vertex and assign that vertex blue

For each vertex, assign color opposite of previous

If there is a an edge (X,Y) between two vertices of same color and post value of X < value of Y

There is an odd cycle in graph

1. A. Let S be the set of explored cities and V be the set of all cities

For each u in S, store a distance d(u)

Initially, S = {Los Angeles}, d(Los Angeles) = 0, miles = 200

While S is not equal to V

Select a city v not in S with at least one edge from S for which *d’(v)* =

is as small as possible and (miles – miles from S to v >= 0)

Add v to S, define d(v) = *d’(v),*

If v has charging station

miles = 200

else

miles = (miles – miles from S to v)

B. Let S be the set of explored cities and V be the set of all cities

Initially, S = {Los Angeles}, miles = 200

While S is not equal to V

Select a city v not in S with at least one edge from S and (miles – miles from S to v >= 0) and there is no charging station

If such a city doesn’t exist

Select a city v not in S with at least one edge from S and (miles – miles from S to v >= 0)

Add v to S, define d(v) = *d’(v),*

If v has charging station

miles = 200

else

miles = (miles – miles from S to v)