College of Computer & Information Science
Northeastern University
Fall 2018

CS5800
Algorithms
October 26, 2018

Quiz 2 – Answer Key

1. **Problem 1: (4 points) Graph Traversal**
Let $G = (V, E)$ be a connected, undirected graph. Give an $O(V + E)$-time algorithm to compute a path in G that traverses each edge in E exactly once in each direction. It is adequate to provide a high-level outline for your algorithm, with enough detail to establish its worst-case running time.

Answer: The required path that traverses each edge of G exactly once in each direction can be achieved through adapting a Depth First search of G starting from any node s for composing such a path.

Outline of the algorithm:

```
Find_Path(G, s)
   for each vertex u in V
        do status[u] <- "unexplored"
   return DFS_Path(s)

 DFS_Path(u)
     status[u] <- "in process"
     R = () // initialize path R to be empty
     for each node v that is connected to u
          if status[v]  = "unexplored"  // Explore edge (u, v)
               // (u, v) is a tree edge
               // add (u,v) and (v,u) before and after exploring node v
               then R <- R + (u,v)
                     R <- R + DFS_Path(v)
                     R <- R + (v,u)
         else if  status[v] = "in process"
                 // v is an ancestor of u and (u, v) is a back edge
                 // add (u,v) and (v,u)
                 then R <- R + (u,v)
                       R <- R + (v,u)
     status[u] <- "explored"
     return R
```

As with DFS, the initialization takes O(V) time and the traversal of the graph for computing the path takes O(E) time. The overall worst-case complexity of the algorithm is O(V+E).

2. **Problem 2: (6 points) Wrestling Rivalry**
There are two types of professional wrestlers: "good guys" and "bad guys." Between any pair of professional wrestlers, there may or may not be a rivalry. Suppose we have n professional wrestlers and we have a list of r pairs of wrestlers for which there are rivalries. Give an $O(n + r)$-time algorithm that determines whether it is possible to designate some of the wrestlers as good guys and the remainder as bad guys such that each rivalry is between a good guy and a bad guy. If it is possible to perform such a designation, your algorithm should produce it.

Answer: Create a graph $G = (V, E)$ where each node corresponds to a professional wrestler. Create an edge $e = (u, v)$ between nodes u and v if there is a rivalry between the two professional wrestlers represented by u and v. Graph G is an undirected graph. The question of whether there is a designation of the set of wrestlers as good guys and bad guys such that each rivalry is between a good guy and a bad guy is equivalent to testing the bipartiteness of the graph G defined above.

Outline of the algorithm:

```
Label_wrestlers(G)
     for each vertex u in V  // G = (V,E)
        do Label[u] <- "undefined"
     RemainingWrestlers = V   // Keep track of wrestlers that are not yet labeled.
     while size of RemainingWrestlers > 0
         // care for multiple connected components in G
         s <-- arbitrary node in V
         // find the bread first component that includes s
         Q < - {}   // initilaize queue of nodes to be explored to be empty
        Label(s) <- "Good guy"
        RemainingWrestlers <- RemainingWrestlers - {s}
        Add s to Q
        while Q is not empty
             u <- node at the head of Q
             for each node v that is connected to u in E
                 if Label[v]  = "undefined"  // first visit to node v
                      // Label v to be the opposite of u
                      then if Label[u] = "Good guy"
                                 then Label[v]  <- "Bad guy"
                                 else Label[v]  <- "Good guy"
                                 Add v to Q
                                 RemainingWrestlers <- RemainingWrestlers - {v}
                else // node v already has a label
                      if Label[u] == Label[v]
                         then  Return(False)
                         // it is not feasible to designate the wrestlers as good guys and
                         // guys such that all rivalries are between the good guys and bad
                         else continue
```

```
// end of while loop
if RemainingWrestlers  is empty
   // all wrestlers have been labeled
   then
        Labels = []
        for u in V
           do Labels <- Labels + (u, Label[u])
       return(Labels)
```

The initialization step is O(V) for each node. The nested while loops process each edge in E by processing the neighbors of each node in V. The time taken in the nested while loops is, therefore, O(E). The result is composed in the end in O(V) time. The overall worst-case time complexity of the algorithm is O(V+E).