Recitation Problems - Com S 311

Week of Mar 25th - Mar 30th

1. There are $n-$boxers. There are $m-$pairs among these boxers who are bitter rivals of each other. Write an algorithm which checks whether the boxers can be arranged in 2 groups such that rival boxers are in separate groups. If the check is successful, then your algorithm should output the two groups.

   Solution Strategy: Model this as a graph and check if the graph is bipartite.

   - Arrange n boxers as vertices of a graph.
   - Connect vertices i and j if there is a rival relationship between i and j.
   - Conduct a BFS of the entire graph. Put vertices in alternate layers in different groups.
   - If there is an edge between vertices in the same layer in the BFS exploration, then a grouping of boxers in 2 groups is not possible.

Algorithm 1 $bfsExploration$ (given graph $G = (V, E)$)

```
for all $v \in V$ do
    $v.explored = false$
    $v.layer = -1$
for all $v \in V$ do
    if $v.explored == false$ then
        if ! $bfs(v)$ then
            return no grouping possible
Put vertices in even layers in one group
Put vertices in odd layers in another group
```

Algorithm 2 bfs (given a vertex v)

```
add $v$ to queue $Q$
$v.layer = 0$ // start with layer 0 for root
$v.explored = true$
while $Q$ is not empty do
    $u = dequeue(Q)$
    for all $w \in Neighbor(u)$ do
        if $w.explored == false$ then
            add $w$ to queue $Q$
            $w.explored = true$
            $w.layer = u.layer + 1$
        else
            if $w.layer == u.layer$ then // vertices in same layer are neighbors
                return $false$
return $true$
```

Runtime Analysis: $O(|V| + |E|)$ because $bfs(v)$ is called only for vertices which have not been explored.

2. In a well-planned city, a road network connects intersections. If there is a road between two intersections, then one can go from one intersection to the connected intersection following the road connecting the two intersections. Each road has exactly the same length.

   The city planning commission is trying to determine whether electric buses will be a viable option for public transportation in the city. Each bus is routed to travel along the *shortest* path from a starting intersection to a destination intersection. To reduce the range anxiety of the electric bus drivers, the commission wants to determine the longest such shortest path bus route in the city. Write an algorithm that finds the maximum such shortest path bus route.

   Each bus route is the shortest path between two intersections. Consider the intersections as vertices of a graph and roads as edges between vertices.

Do a BFS exploration from each vertex and find the layers for every vertex. This will produce the length of the shortest path from any vertex v_i to any vertex v_j. Capture this information in a 2-D array as $d[i][j]$. Runtime is $O(|V|(|V| + |E|))$ as we are doing the BFS $|V|$ times.

Then, loop over all vertex pairs to find the one with the maximum shortest path.

Algorithm 3 Find the maximum shortest path (given a 2-D array d)

```
1: $max = 0$
2: **for** all $v_i \in V$ **do**
3:     **for** all $v_j \in V$ **do**
4:         **if** $d[i][j] > max$ **then**
5:             $max = d[i][j]$
6: return $max$
```

The runtime of finding the maximum shortest path is $O(|V|^2)$. As a result, the overall runtime is $O(|V|(|V| + |E|))$.