CS345A Algorithms-II

Assignment-3

Deepanshu Bansal (150219)
Mukul Chaturvedi (150430)

Q1 Searching for a special path

Algorithm

1. Make a topological ordering of G given source s. Let's say it is stored in array D.
2. Now if there exists a path $s \rightsquigarrow x_1 \rightsquigarrow x_2 \rightsquigarrow \dots \rightsquigarrow x_k \rightsquigarrow t$ then they must appear in the same order in topological ordering of vertices.
3. J ← 0 and let's say this order of $s \rightsquigarrow x_1 \rightsquigarrow x_2 \rightsquigarrow \dots \rightsquigarrow x_k \rightsquigarrow t$ is provided in array T.
4. For i = 0 to n-1 do
    a. If (T[j] == D[i]) then j++;
5. If (j == k+2) then return true; // all element are present in same order
6. Else return false;

Time Complexity - **O(m+n)** // O(m+n) for step 1. And O(n) for step 4.

Q2 Unique path graph

DFS(v)

{ Visited(v) ← true;

D[v] ← count++;

Back-edges[v] ← 0;

For each edge (v,w)

{ If(Visited[w] = false)

```
        DFS(w);
        Else{
                If(Finished[w]) {  UniquePathGraph ← false;  break;  }
                Else{ Back-edges[v]++;
                        If( Back-edges[v]>1){ UniquePathGraph ← false;  break;  }
                        }
        }
}
Finished[v] ← true;
F[v] ← count++;
}
```

Algorithm - If there are any forward or cross edges then finished value of w node of (v,w) edge will not be zero. Therefore, forward and cross edges are handled. One node can only have at-most one back-edge as it only makes a cycle and thus no two paths are made in graph. Thus, we do not have unique path graph only if there are more than one back edges towards a node. As we are applying the DFS on u so all the vertices on the graph are checked and it takes **O(m+n)** time.