hw4 4102

jh4ctf

September 2021

1

In an adjacency matrix, it consists rows and columns that contains 1 and 0 values indicating if vertices are pointing towards other vertices. To deliver an algorithm that runs in O(V) run-time in a matrix with rows and columns, we have to assure to not loop within a loop. To prove that there exists a vertex with in-degree —V — 1 and out-degree 0, our algorithm will need simply find it in the graph. Because in the table, rows indicates its in degree and column indicates its out-degree. Thus, the first step of the algorithm is to search through all the value of a row i and see if the whole row i is all value 1 except i-th column of its row. If we find one value except i-th column in row i to be not 1, then search i+1 row. To loop through all rows takes θ (V). If it does finds such row i with this condition, then check if i column all contains 0. If it does, then there exists a vertex with in-degree —V — 1 and out-degree 0. But if it finds no row with all 1 except its column, then there doesn't exist a vertex with in-degree —V — 1 and out-degree 0. Checking a column to see if there is only 0 requires linear run-time. Thus its run-time is O(V).

2

```
def dfs(graph, start):
    indent visited = {}
    indent path=[]
    indent dfs_recurse(graph, start, visited)
def dfs_recurse(graph, curnode, visited,path):
    visited[curnode] = True
    try:
        alist = graph.get_adjlist(curnode)
    except:
        return path
    for v in alist:
        if v not in visited:
            path.append(v)
```

```
        print(" dfs traversing edge:", curnode, v)
        dfs_recurse(graph, v, visited)
```

return

3

Given that T_d is the depth-first Search Tree on G, we can conclude that the DFS reached the bottom height of nodes and is complete. Given that T_b is also a breath-first Search Tree, the BFS also reached to final nodes and is complete. Also by that two search tree is the same, we can infer that there is no back-edge or that there is no cycle on the tree. Thus, it indicates the graph G that already produces BFS and DFS is already a tree.

4

From the question, we can assume s to be the root in graph G. And for graph tree T_{BFS}, it has depth d, which means that there is a vertex t at distance d from s in this graph G. Also we have that t cannot be at a depth smaller than d in any T_{BFS} and T_{DFS} because there is no other route form s to t that spans length d-1. So all other spanning trees of G rooted with s, which is T_{DFS}, must have a depth taht cannot be smaller that of T_{BFS}.