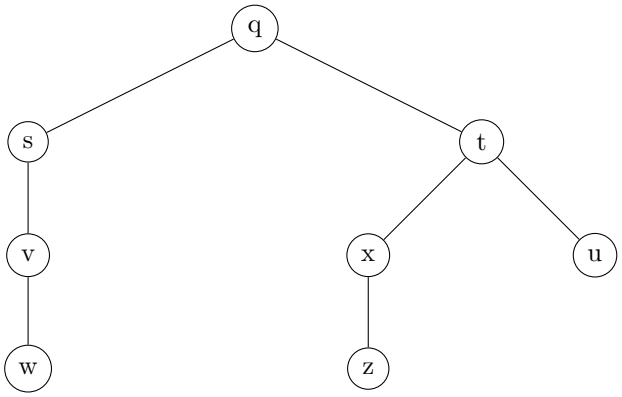
CSCI 6470
Homework 4
Jiakuan Li

1.



2.

Algorithm 1 DFS-CONSTRAINED(G,v, t)

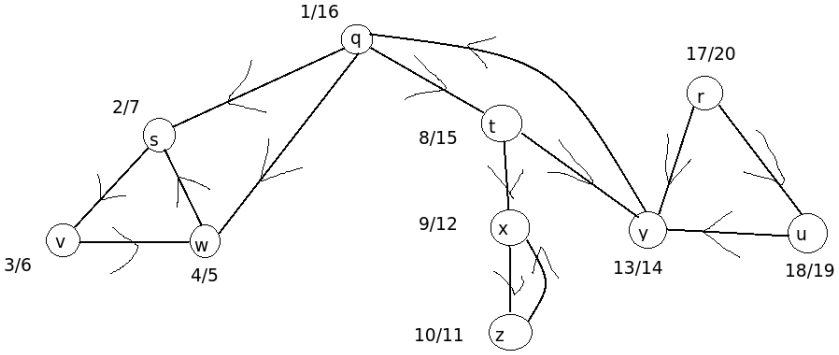
(1)
```
if v = t then
  return True
end if
for each vertex u adjacent to v do
  u.π = v
  if u.μ = true then
    DFS-CONSTRAINED(G, u, t)
  end if
end for
return False
```

(2) Just like a normal DFS algorithm, but this algorithm does not visit deeper once it meets a vertex in U. So the worst case for this algorithm is it visit all the edges once. The time complexity of visiting all edges are $O(|E|)$. Plus the setting "visit" and "parent" properly for every node, the total time complexity if $O(|V| + |E|)$.

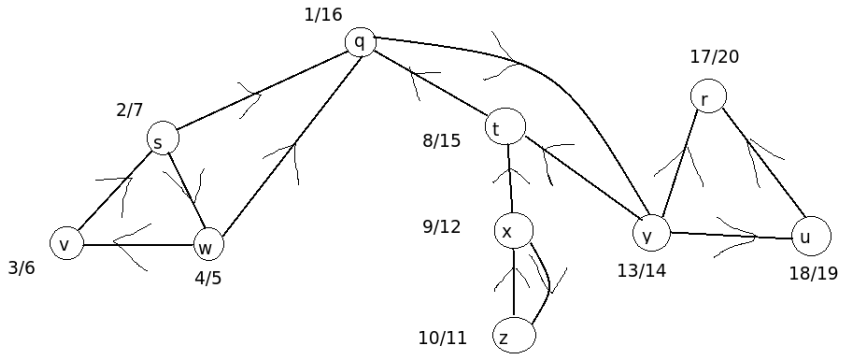
3.

$O(|V|)$ is not enough for detecting the any cycle in the undirect graph. The preparation stage, that is setting "visit" and "parent" properly, take $O(|V|)$, since every node in the graph needs to be visit. The for loop in the algorithm needs to be execute exactly one times in the worst case, (a tree). Therefore, the time complexity here is $O(|E|)$. And the total time is $O(|E|) + O(|V|)$. So $O(|V|)$ is not enough.

4.



Reverse



5.

First of all, I will define some status of node. 1. white: not visited
2. gray: visited, but it has unvisited neighbour.
3. black: visited, and not unvisited neighbour.

In order to prove, need to show for any edge (u, v), u turns black after v.
1. If v is black, then obviously u turns black after v.
2. if v is gray, then the graph is not a DAG. Since from v, there exists another path to u, plus the edge (u, v).
3. If v is white, then it will be inserted into the stack after u. When poped the stack, v will come out before u.