‹ Back

Python | DFS

reddddddd ★ 148 Last Edit: June 12, 2019 9:42 PM 1.1K VIEWS

21

In DFS approach, we procced with the regular cycle detection approach. If there is a cycle, then there are infinite number of paths, hence we return false. Also when the DFS path ends, we check if we are at destination to check if return value is `true` are `false` . We also make use of optimizations to ensure paths once checked are not checked again.

```
from collections import defaultdict
class Solution:
    def leadsToDestination(self, n: int, edges: List[List[int]], source: int, destination: int) -> bool:
        g = defaultdict(set)
        visited = defaultdict(int)
        for [x,y] in edges:
            g[x].add(y)

        def dfs(node):
            if visited[node] == 1:
                return True
            elif visited[node] == -1:
                return False
            elif len(g[node]) == 0:
                return node==destination
            else:
                visited[node] = -1
                for child in g[node]:
                    if not dfs(child):
                        return False
                visited[node] = 1
                return True

        return dfs(source)
```