Depth-limited works similarly to depth-first search. The difference here is that depth-limited search has a pre-defined limit up to which it can traverse the nodes. Depth-limited search solves one of the drawbacks of DFS as it does not go to an infinite path.

DLS ends its traversal if any of the following conditions exits.

**Standard Failure**

It denotes that the given problem does not have any solutions.

**Cut off Failure Value**

It indicates that there is no solution for the problem within the given limit.

Now, consider the same example.

Let’s take A as the start node and C as the goal state and limit as 1.

The traversal first starts with node A and then goes to the next level 1 and the goal state C is there. It stops the traversal.

The path of traversal is:

A —-> C

If we give C as the goal node and the limit as 0, the algorithm will not return any path as the goal node is not available within the given limit.

If we give the goal node as F and limit as 2, the path will be A, C, F.

DLS code

graph = {

'A' : ['B','C'],

'B' : ['D', 'E'],

'C' : ['F', 'G'],

'D' : [],

'E' : [],

'F' : [],

'G' : []

}

def DLS(start,goal,path,level,maxD):

print('nCurrent level-->',level)

path.append(start)

if start == goal:

print("Goal test successful")

return path

print('Goal node testing failed')

if level==maxD:

return False

print('nExpanding the current node',start)

for child in graph[start]:

if DLS(child,goal,path,level+1,maxD):

return path

path.pop()

return False

start = 'A'

goal = input('Enter the goal node:-')

maxD = int(input("Enter the maximum depth limit:-"))

print()

path = list()

res = DLS(start,goal,path,0,maxD)

if(res):

print("Path to goal node available")

print("Path",path)

else:

print("No path available for the goal node in given depth limit")