

AI Assignment-2 Report

Team -14

Tabish Khalid Halim ,

200020049

Anand Hegde ,

200020007

Department of Computer Science, IIT Dharwad

January 2, 2022

Contents

1	Approach	1
2	Variables used in Python Program	2
3	Functions created in Python Program	3
4	Pseudo Code	4
5	Graphical Analysis	5
6	Results	6
7	References	7

1 Approach

The basic approach we used for the A.I assignment is to use the concept of Heuristic Search Algorithms and apply BFS {Best First Search} and Hill-climbing.

State Space

Start & Goal Node

- **Start Node**

2 Variables used in Python Program

- **dfs_stop** : tells when dfs to stop.
- **goaldfs** : target state to achieve for DFS.
- **goaldfid** : target state to achieve for DFID.
- **statesdfs** : No. of states explored during DFS traversal .
- **statesdfid** : No. of states explored during DFID traversal .
- **DFIDstop** : to break out of recursion .
- **visited** : Variable created to store the set of visited vertices .
- **parent** : Tuple to store the parent of each node . It is used for finding path .
- **graph_input** : This stores the input given in as a list of lists .
- **m** : No. of rows in the Maze .
- **n** : No. of columns in the Maze .
- **states** : Variable to store no. of states explored .
- **pathlength** : Variable to store length of the path .

3 Functions created in Python Program

- **goal_state(i,j,graph_input)** : This function determines whether the coordinate (i,j) is the end goal for the PacMan or not .
- **move_gen(i,j,graph_input)** : This function's task is returning all possible moves available to the PacMan , if the adjacent block has a space (' ') or astrik('*') , funtion returns its coordinates .
- **DFSUtil(v, visited,parent,graph_input,open_list)** : This is the recursive DFS Utility function .
- **DFS(graph_input,v=(0,0))** : The function to do DFS traversal. It uses recursive DFSUtil()- dfs utility function .
- **DFID(graph_input, depth,v=(0,0))** : The function to do DFID traversal. It uses recursive DFSUtil()- DFS Utility Function.
- **DFIDUtil(v, visited,parent,graph_input, depth)** : This is recursive DFID- utility function .
- **dfid(graph_input,v=(0,0))** : This is the Main DFID function- which calls DFID- which is dfs version for DFID . The extra thing is the depth here.
- **bfs(graph_input,s=(0,0))** : This function is used to perform BFS .
- **searchmethod(bdd,graph_input)** : Simple function to deal with the case wise operation to perform BFS, DFS or DFID as per the requirement

4 Pseudo Code

The main pseudo code used in our assignment is as follows :

move_gen function

```
def move_gen(i,j,graph_input):
    global open_list
    templist=[]
    if(i<n-1):
        if((graph_input[i+1][j]==' ' or '**') and ((i+1 , j) not in open_list)):
            templist.append((i+1 , j))
    if(i>0):
        if(graph_input[i-1][j]==' ' or '**') and ((i-1 , j) not in open_list):
            templist.append((i-1,j))
    if(j<n-1):
        if(graph_input[i][j+1]==' ' or '**') and ((i , j+1) not in open_list):
            templist.append((i,j+1))
    if(j>0):
        if(graph_input[i][j-1]==' ' or '**') and ((i+1 , j) not in open_list):
            templist.append((i,j-1))
    return templist
```

goal_state function

```
def goal_state(i,j,graph_input):
    if(graph_input[i][j]=='*')
        return True
    else :
        return True
```

5 Graphical Analysis

Based on observation , we have plotted the following .

6 Results

The following conclusions have been made after evaluation of my program :

1. :)

Conclusion

Dont know .

7 References

- <http://geeksforgeeks.com>
- <https://wikipedia.org>
- <https://stackoverflow.com>