Artificial Intelligence(Assignment 1)

NAME: ABHINAV GUPTA
ID : 2015B4A70602P

In the given problem we have to remove matchsticks from a given configuration of matchsticks to create given number of squares in the end. These Squares can be of any size that is 2x2 or 3x3 etc. In the end, every matchstick is an edge to a unit square and no overlapping square exist at any point of time.

There are five python files namely:
Initial_state: to generate Initial State
Goal_state: to generate all possible goalstates
Functions: file contains BFS and DFS functions and all other functions required for performing bfs and dfs such as addchildnode(), createroot() and next_state().
GUI2: this file is for making GUI it takes 3 argument size,state and path and show the result using turtle library
main_AI: This is the driver file you will have to run this to run the complete code

General Information:
I have numbered cells from 0 to size*size-1 and I have stored matchstick positions in list of Size (2*size)*(size+1) if there exists a stick at position say pos then list[pos]=1 else 0 I have numbered array such that all horizontal sticks get numbered first and then we go for vertical sticks

In initial state I am using random function to get random integer between 0 and (size*size) to get the cell number and then I am taking one more random number for size of the square to be formed on that cell if it is not visited earlier now using checkpossible() function I am checking whether this square can be formed on that cell or not if not then loop continues else we store the new cell covered in the set().This goes on until square covered becomes equal to area tobe covered.

In goal state I am generating all possible goal states with number of squares 3 or less.

In GUI2 I have written a code to show the initial state and how it changes using turtle library it takes three arguments size, state and path. Path stores the action path returned by bfs that is making goal state from initial state.I am using turtle in this example.

In Functions I have written functions like create_root(), add_child() and next_state() which are necessary for performing BFS and DFS. In BFS and DFS what I am doing is I am performing standard bfs and dfs and if current node’s state matches with any of the goal state then I am returning the action path. To get action path I am storing parent of each

Artificial Intelligence(Assignment 1)

node in dictionary.To optimize DFS I have defined a separate mapping. My BFS is relatively slow as compared to DFS.

main_AI is the driver file you have to copy all files in a single folder and then just run this driver file to see the output. You can run dfs or bfs from main_AI

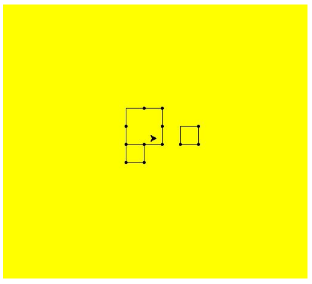
WORKING :

I am showing example for BFS on a random initial state

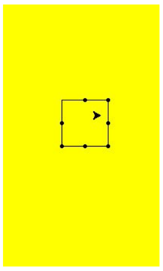
Initial State: [1,1,0,0,0,0,0,1,1,1,0,1,1,0,0,0,0,0,0,0,1,1,1,1,0,0,1,0,1,1,0,0,0,1,0,0,0,1,0,0]
Initial cells occupied: {0,1,4,5,7,8}

Final state: [1,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0]
Final cells Occupied: {0,1,4,5}

GUI:
Initial state :



Final state:



END

Artificial Intelligence(Assignment 1)

****
****