CS2180 Artificial Intelligence Lab (Jan-May 2023)

Department of Computer Science and Engineering
Indian Institute of Technology Palakkad

Assignment 3: Search (Given: 14 Feb 2023, Due: 7 Mar 2023)

General instructions

- Solutions are to be typed in the `.ipynb` file provided and uploaded in the lab course page in Moodle on the due date.
- Your code should be well commented and should be compatible with python3.
- For this assignment, you are allowed to import the libraries `random` and `queue` of python3. No other libraries may be imported.

1 Generate Maze

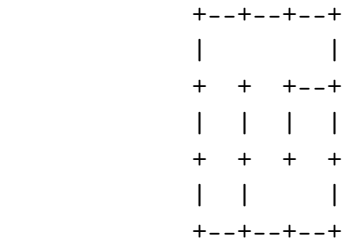
A maze can be visualized as an arrangement of cells in a rectangular $m \times m$ grid with walls between some pairs of cells.

(a) Write a program that uses randomized depth-first search to generate a maze. The randomized depth-first search procedure is as follows: Starting from a given cell (say (0,0)), this algorithm produces a path that visits each cell in the grid according to the following recursive procedure.

- If the current cell C has a neighbouring cell that is not yet visited, chose one of such cells (say C') at random and remove the wall between these two cells. Repeat with C' as the current cell.
- If all neighbouring cells of the current cell C are already visited, then backtrack to the last cell $\widehat{C}$ with a neighbouring cell $\widehat{C}'$ that is not yet visited and repeat with $\widehat{C}'$ as the current cell.

A sample output of a 3×3 maze is along with the graph adjacency representation is as shown below. The bottom left corner of the maze is $(0, 0)$ and top right corner of the maze is $(2, 2)$.

```
Node (0, 0): (0, 1),
Node (1, 0): (1, 1), (2, 0),
Node (2, 0): (1, 0), (2, 1),
Node (0, 1): (0, 2), (0, 0),
Node (1, 1): (1, 0), (1, 2),
Node (2, 1): (2, 0),
Node (0, 2): (0, 1), (1, 2),
Node (1, 2): (1, 1), (0, 2), (2, 2),
Node (2, 2): (1, 2),
```



(b) Write a program to do DFS on a $m \times m$ maze given in adjacency representation to find a route from the source cell $(0, 0)$ to the destination cell $(m-1, m-1)$. Output the route and the number of cells explored.

(c) Write a program to do BFS on a $m \times m$ maze given in adjacency representation to find a route from the source cell $(0, 0)$ to the destination cell $(m-1, m-1)$. Output the route and the number of cells explored.

(d) Write a program to do A* search on a $m \times m$ maze given in adjacency representation to find a route from the source cell $(0, 0)$ to the destination cell $(m-1, m-1)$. Use the Manhattan heuristic for A* search. The Manhattan distance between two cells of the maze (i, j) and (k, ℓ) where $i, j, k, \ell \in \{0, 1 \ldots, m-1\}$ is $|i-k| + |j-\ell|$. Output the route and the number of cells explored.

2 Sliding Blocks

Consider the sliding blocks puzzle where we are given a 3×3 grid of blocks with each block containing a unique integer between 0 and 8. An example configuration is given below.

1 2 3

7 8 5

0 6 4

At each step, the block containing 0 can swap places with an adjacent block.

(a) Use A* search to start from any given initial configuration and reach the goal configuration

1 2 3

4 5 6

7 8 0

with the sum of Manhattan distances of the blocks from their goal positions as the heuristic. The Manhattan distance of a pair of blocks occupying the integer n at the locations (i, p) and (j, q) (where $i, j, p, q \in \{0, \ldots, 8\}$ and $n \in \{0, \ldots, 8\}$) is given by \$ |i-j| + |p-q|\$. Print the total number of steps taken to reach the goal and the blocks configuration at each step.

(b) Repeat part (a) with the following alternative heurisitics.

1. the number of misplaced blocks
2. the Manhattan distance of the "0" block alone instead of the sum

Compare the performance of the three heuristics using the size of the explored list as the measure.