AI Lab

Assignment 2: *SEARCH with Edge Costs*

Note:

1. From here onwards, any code that you write will be helpful in the examinations. Please protect your codes and do not share your codes. The TAs/portal do a random check on plagiarism. If any part of the code is copied, even if it the standard code of a search, the TA will regard it as copied. We are not following the policy of the last year where copying the base codes was allowed. For the same reasons it may be risky to take codes from online repository or your seniors' repository.
2. Any form of copying will be attract negative marks and penalties
3. You can code inside lab as well as outside lab after the normal working hours, however the code should be your own
4. Any magnitude of verbal discussion outside lab is allowed. No codes must be exchanged
5. Please protect your own codes. Copying will more severely affect the person whose codes are taken.
6. The TAs/portal itself will check for plagiarism.
7. If your performance is very good during the semester and poor in the mid-sem and end-sem, it would be assumed that you have copied and all marks would be zeroed.
8. In the exams you will be allowed to carry your own codes, written apriori, but not of anybody else.

Consider that there a bit map, where every location is either 0 (navigable) or 1 (non-navigable). For ease, consider the X axis is the vertical axis and Y axis is the horizontal axis, with origin as the top left corner. Both the axes start with 0. The point robot can move through the navigable areas. The robot can travel to any 1 of the 8 neighboring grids with a cost function given by the Eucledian distance of the edge. The preference of actions are: right, down-right diagonal, down, down-left diagonal, left, up- left diagonal, up, up-right diagonal.

Input: The input starts with the size of the grid map as 2 numbers *m* and *n*, separated by a space. Then there are exactly *m* lines with *n* integers each, representing the map. The next input is in the next line, the number of queries *q*. Then there are exactly *q* lines, each line with 4 integers, the first two denote the source and the last 2 denote the goal.

Sample Input:

```
4 4
0 0 0 0
0 1 1 0
0 1 1 1
0 0 0 0
1
0 0 2 0
```

Output: For each query, print a variable number of lines, each with 2 integers representing the solution path, starting from the source state to the goal state.

Sample Output:

```
0 0
1 0
2 0
```

1. Solve the problem using Uniform Cost Search.
2. Solve the problem using A* Algorithm.
3. Solve the problem with the constraint that the point robot can only go from one grid to the other grid if the entire area between the two grids is obstacle free. So in the following map, no solution is possible between the top-left and bottom right corner. Similarly the transition shown by the number 2 in the next map is regarded as illegal.

0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
1	1	1	0	0	0
1	1	1	0	0	0
1	1	1	0	0	0

0	0	0	0	0	0
0	0	0	0	0	0
0	0	2	0	0	0
1	1	1	2	0	0
1	1	1	0	0	0
1	1	1	0	0	0

4. Solve the problem such that the robot needs to visit the 4 corners (0,0), (0,*n*-1), (*m*-1,*n*-1) and (*m*-1,0) in the same order.
5. Solve the problem such that the robot needs to visit any one of the 4 corners (0,0), (0,*n*-1), (*m*-1,*n*-1) and (*m*-1,0). (What is the heuristic function of use?).
6. Solve the problem such that the robot needs to visit the 4 corners (0,0), (0,*n*-1), (*m*-1,*n*-1) and (*m*-1,0) in any order. The corners are mentioned as per their preference in this question.
7. Consider that there are 2 point robots. Both the robots can only move rectilinearly, without diagonal movements. Plan the robots such that they meet at one common point using A* algorithm. All movements of both robots take a unit time. You need to minimize the catching time. All movements of the first robot are preferred, over the movements of the second robot. (What is the heuristic function of use?). In the above input description, interpret the goal as the coordinates of the 2nd robot. The path will comprise of the coordinates of the 1st robot and then the coordinates of the 2nd robot in the same line.
8. Solve the problem using Bidirectional using Uniform Cost Search at both search trees.