Assignment 1:
Solving Color-Maze Puzzle using Search

Due Sunday, 9 April, 11:30pm

Color-Maze puzzle is a single-agent grid-game played on a rectangular board that includes a maze. Initially, the agent is located on a single maze cell. The agent can move in four cardinal directions: up, down, right or left. Once a direction is chosen, the agent moves in that direction until it reaches a wall at once, and colors all the cells it travels over. Once a cell is colored, its color does not change. The goal is to color all the cells of the maze by moving the agent over them, while minimizing the total distance traveled by the agent. This game is available at:

`https://www.mathplayground.com/logic_color_maze`.

Please implement in Python

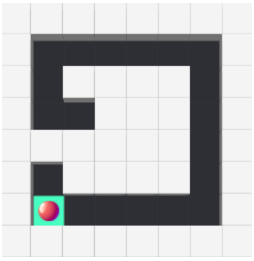
1. a Uniform Cost Search (UCS) algorithm, and
2. an A* search algorithm

to solve the following version of the Color-Maze puzzle.

Input rectangular game board with a single agent is represented as a grid where each grid cell is uniquely marked with 0, X or S:

- 0 denotes the cells that are empty and that need to be colored,
- X denotes the cells occupied by the walls, and
- S denotes the cell occupied by the agent.

For instance, in Figure 1 (right) describes the game board depicted in Figure 1 (left).



0	0	0	0	0	0
0	X	X	X	X	0
0	0	X	X	X	0
X	X	X	X	X	0
0	X	X	X	X	0
S	0	0	0	0	0

Figure 1: An example puzzle (left) and its representation in the given format (right).

Output is a sequence of legal states and moves that illustrates how the agent colors all the cells such that the total distance traveled by the agent is minimized.

For instance, Figure 2 (resp. Figure 3) shows step by step how the agent can color the cells of a given maze, where the total distance traveled by the agent is 6 (resp. 5). In both solutions, the agent takes the same number of moves but the total distance traveled by the agent is smaller in Figure 3. Therefore, the sequence of states and actions illustrated in Figure 3 should be returned as the output.

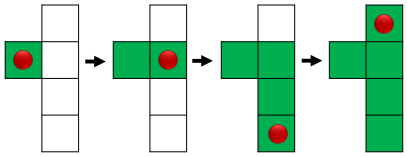


Figure 2

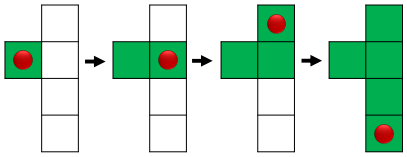


Figure 3

What to do The assignment consists of five parts:

1. (10 points) Model the Color-Maze puzzle as a search problem: Specify the states, successor state function, initial state, goal test, and step cost function.[1]
2. (20 points, provided that part 1 is completed) Implement in Python the UCS algorithm studied in class,[2]to solve the Color-Maze puzzle.
3. (10 points, provided that part 1 is completed) Extend your search model for Color-Maze to apply A* search: Find a heuristic function and prove that it is admissible.[1]
4. (30 points, provided that part 3 is completed) Implement in Python the A* search algorithm studied in class,[2]to solve the Color-Maze puzzle.
5. (30 points) Evaluate your UCS and A* implementations experimentally by 15 Color-Maze puzzle instances of 3 difficulty levels (i.e., 5 easy, 5 normal, 5 difficult instances), on a game board of size 10×10.[3] Summarize the results of your experiments in a table that shows, for each search algorithm, for each puzzle instance,

   - the number of cells in the maze, the difficulty level,
   - the total distance traveled by the agent,
   - the total number of expanded nodes,
   - the CPU time, the memory consumption.

   Discuss these results comparing UCS and A* search: In the table, what do you observe about the scalability of these two algorithms? How do these algorithms explore the search space? Are these observations surprising or expected? Please explain why.

[1]Note that the step cost function and the heuristic function return positive numbers $\geq \epsilon > 0$.

[2]Many UCS and A* implementations are available online. Note that they may not match the algorithms covered in class. So, make sure that your own implementations match the ones taught in class.

[3]Note that you need to clarify how you define the difficulty level: How do you describe an easy instance? A normal instance? A difficult instance? It is preferable to present one example for each difficulty level in your report, to better explain your formulation of difficulty levels. In the demos, you are expected to show one example for each difficulty level.

Submit

- A pdf copy of a description of
  - your formulations of the Color-Maze puzzle (i.e., search model and heuristic function), and
  - experimental evaluation of your UCS and A* search implementations on the Color-Maze instances (i.e., table and discussion).[4]
- A zip file containing the following:
  - Your Python code for each algorithm, with comments describing your solution.
  - Several test boards you created for the fifth part of the assignment, and the corresponding solutions found by your implementations.

In each one of the deliverables above, please include your name and student id.

Demos You are expected to make a demo of your implementations so that we can grade parts 2, 4, and 5 of the assignment. The demos are planned for the week following the deadline and will be scheduled later on.

[4] You can use Overleaf for editing in LaTeX: `https://www.overleaf.com/`.