Assignment 1: Solving Bloxorz using Search

Due Tuesday, 23 October, 11pm

Bloxorz is a game where the goal is to drop a $1 \times 2 \times 1$ block through a hole in the middle of a board without falling off its edges. This game is available at

`http://www.coolmath-games.com/0-bloxorz/index.html`.

Please implement in Python

- a Uniform Cost Search (UCS) algorithm, and
- an A* search algorithm

to solve the following version (stage 2) of this puzzle.

Input of the problem The input game board of size $m \times n$ is represented by a matrix of size $m \times n$ where

- `O` denotes safe tiles: the block can stand on these anytime;
- `X` denotes empty tiles: the block may never touch an empty tile, even if half of the block is on a safe tile;
- `S` denotes the tile(s) occupied by the block: if the block is in the vertical orientation then there is one tile labeled `S`, otherwise (if the block is in the horizontal orientation) there are two adjacent tiles labeled `S`;
- `G` denotes the goal tile: the block needs to be on it (vertically) in order to fall into the goal.

For instance, the matrix below represents the game board depicted in Figure 1.

```
O O O X X X X X X X
O O O O O O X X X X
O O O S O O O O O X
X O O S O O O O O O
X X X X X O O G O O
X X X X X X O O O X
```

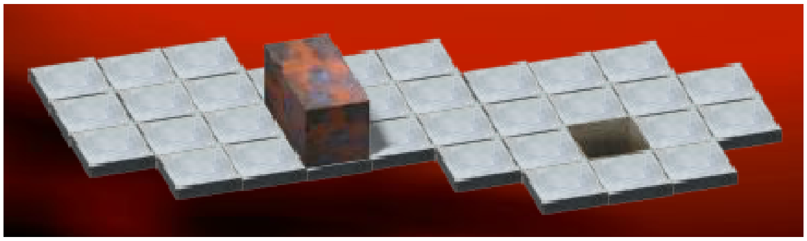

Figure 1: A game board for Bloxorz.

Output of the problem A shortest sequence of legal states that navigate the block from its given initial location into the goal.

What to do The assignment consists of five parts:

1. (10 points) Model the puzzle as a search problem: specify the states, successor state function, initial state, goal test, and step cost.
2. (30 points, provided that the first part is completed) Implement in Python a UCS algorithm to solve the puzzle.
3. (10 points, provided that the first part is completed) Extend your search model for A* search: Find a heuristic function and prove that it is admissible.
4. (30 points, provided that the third part is completed) Implement in Python an A* search algorithm to solve the puzzle.
5. (20 points) Compare your UCS and A* codes on some sample puzzle instances. Construct a table that shows, for each instance, time and memory consumption. Discuss the results of these experiments: Are the results surprising or as expected? Please explain.

Submit

- A pdf copy of a description of your solutions (search model and heuristic function), and experimental evaluation (table and discussion).
- Your Python code for each algorithm, with comments describing your solution.
- Several test boards you created for the fifth part of the assignment.

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

Late policy Late assignments are handled based on a system of "grace days": you begin the term with two grace days, an assignment handed in from one minute to 24 hours late uses up one grace day, and 24:01 to 48 hours late uses up two grace days.