CIS 521 Homework #2: Search

February 6, 2014
Due: Thursday, February 20, 2014

Unlike the last HW, it is unlikely that you can finish this problem set if you start soon before the deadline. ***Make sure to start early!***

1. (10 points) Do 3.6 (a) and (d) in AIMA, 3rd edition. For the problem formulations, just specify what the state space is, the initial state, goal state, and how to transition between states. A high-level description is okay. Please be concise.

2. (10 points) Do 3.15 in AIMA, 3rd edition.

3. (10 points) Do 3.25 in AIMA, 3rd edition.

4. (30 pts) The 8-Puzzle

Implement the successor function for the 8-puzzle. Generate all the successors at once by copying and then editing an earlier state of the 8-puzzle data structure. Using iterative deepening depth-first search, write a demo function `eightPuzzleDemo(startState)` that shows the puzzle being solved. The function should print out the initial state, the transitions and the solved puzzles and the transitions in an easy-to-understand format. Clearly indicate how to run these demos. Turn in your source code and a sentence or two of what you did.

5. (40 points) A Robotic World

(The following problem is based on Problem 3.7 from the book.) Imagine you are a robot, though a very unusual one: you're one-dimensional (a point) and you live in a two-dimensional world. You are bounded in by a fence that is a square of side length 10 centered on the origin. The world also contains many obstacles which are convex polygons (see the illustration on pg. 114 of AIMA). Your job, given two points, is to get from one to the other by the shortest path possible. You can go along the sides of the polygons, but you cannot go where two polygons meet (that is, if two polygons share a vertex, you may not go "through" that vertex to go between them).

(a) Suppose that the state space for this problem consists of all positions *(x; y)* in the plane, where each pair *(x;y)* constitutes one state. How many states are there? How many possible paths are there between any two points?

(b) Explain briefly why the shortest path from one polygon vertex to any other in the scene must consist of straight line segments joining some of the vertices in the polygons. Define a good state space now. How large is this state space?

(c) Describe, in general terms, how to determine the successors of a state.

(d) Write a program to simulate the robot: that is, you will be given a set of obstacles and two points, and you must determine the path between them. Implement this using three search algorithms: both breadth-first and depth-first (to find any path) and A* search to find the shortest path. The path found should be output as a tuple of *(x; y)* pairs (which are themselves tuples).

We are providing you with some useful utilities and a number of test cases in the directory contained in `hw2_kit.zip`. The file `robotUtilities.py` contains some helper methods that you might find useful. In particular, you are required to use the read function there to read in input files. The function `pathIntersectsAnyLines` will also be useful - it takes a start point of a line segment, its ending point, and a list of line segments each in the form ((0,0),(0,1))) and determines if the line intersects any of the line segments in the list.

The file `guihelp.py` contains helper functions for drawling lines and obstacles which might help you in visualization, but you are not required to use it. Note that it requires `Tkinter`. A few example files are provided: `square_in_the_middle`, `triangle_in_the_middle`, `tall_triangle_in_the middle`, and `big_triangles`.

The following code fragment gives an example of using the code in `guihelp.py`:

```
c = Canvas(height=250, width=300)
drawboard(c, read('big_triangles'))
c.pack()
```