Rush Hour

Eckel, TJHSST AI1, Fall 2022

Background & Explanation

Rush Hour is a famous brainteaser. See how the game works here https://www.youtube.com/watch?v=HI0rlp7tiZ0 (the first minute only). You can also use https://www.thelogicgame.com/games/unblockmepack/index.html if it isn't blocked, or get Unblock Me for free from either Google or Apple's app store. I like the app the best; it tells you the number of moves in the solution in challenge mode, which is great for testing!

This is a game to which basic search algorithms clearly apply. The challenge is how to store game states and model moves so that children can be generated and a search tree formed.

Required Task

Model Rush Hour with code that can input a particular starting state and find the shortest solution (ie, solution with fewest moves). Specifically:

- The board is always 6x6; the exit is always where you see it in the video; vehicles can only be 1x2 or 1x3; the car that needs to get to the exit specifically is always 1x2.
- One move is sliding any individual car some number of spaces. If a car can move one, two, or three spaces from where it is right now, all three of those moves should determine different children of the current state. This matches how the challenge mode of the Unblock Me app counts moves; if my explanation is unclear, experiment with that until it makes sense to you and you can code to match.
- I am not giving you a particular input or storage format. Figure out one that makes enough sense to you that I could show you a picture of a start state and you could input it into your program without excessive difficulty.
- Find a way to display board states that are recognizable and easy to follow. (Ie, do not just use "X" to represent the location of any blocking car because then it's impossible to tell where one blocking car ends and another begins. I should be able to tell the pieces apart.)
- Find the shortest path from the given initial state to the end and display every board state along the solution path as well as the solution's length. Just like sliding puzzles, the length of the solution is the number of **moves**.

Specification

I will check this assignment in class. (If this is the last assignment in Unit 1a that you do, it's ok if you show it to me on the first day of Unit 1b.)

This assignment is **complete** if:

- You can successfully input a state that I will show you.
- Your code finds a solution that is the correct number of moves.
- You can display every board state along the way so that I can follow the solution easily.