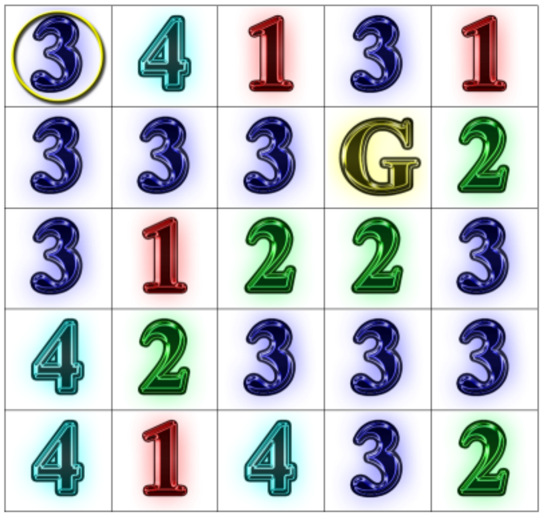
Optimization Search



3	4	1	3	1
3	3	3	G	2
3	1	2	2	3
4	2	3	3	3
4	1	4	3	2

Rook Jumping Maze is a puzzle problem of the sort that one might find in a newspaper alongside a crossword puzzle or a Sudoku puzzle. In a Rook Jumping Maze, start at the circled square in the upper-left corner and find a path to the goal square marked "G". From each numbered square, one may move that exact number of squares horizontally or vertically in a straight line. For more information on Rook Jumping Mazes, see http://cs.gettysburg.edu/~tneller/rjmaze/.

Suppose we want to build a system that generates Rook Jumping Mazes from scratch. That is, the system starts with an empty grid and must determine which number to put in each cell. The system must also determine which cell should be the start and which cell should be the goal.

One way to build a Rook Jumping Maze generator is to use optimization search: hill-climbing search, simulated annealing, or a genetic algorithm. An optimization search algorithm starts from an arbitrary state and uses an evaluation function to determine the "quality" of a potential solution.

1. Describe a function that generates a starting state.

2. Describe as many evaluation functions as you can. An evaluation function will take a potential solution (in this case a grid of numbers, one of which is marked as a start and one that is marked with a "G") and returns a real number that reflects the quality of the solution. Be as precise as possible, but you don't have to write code.

For example, one criterion for a good Rook Jumping Maze might be that it is not impossible to solve. How would one determine whether a maze is impossible to solve? Another criterion might be the difficulty of the maze. How might one measure

the difficulty of a maze? See
http://modelai.gettysburg.edu/2010/rjmaze/evaluation2.html for some other ideas.

3. Describe a neighbor function. This function takes a potential solution and generates a neighbor solution. This function would be used in blind hill climbing or simulated annealing to produce the next state. In a genetic algorithm this function would be the mutation operation.