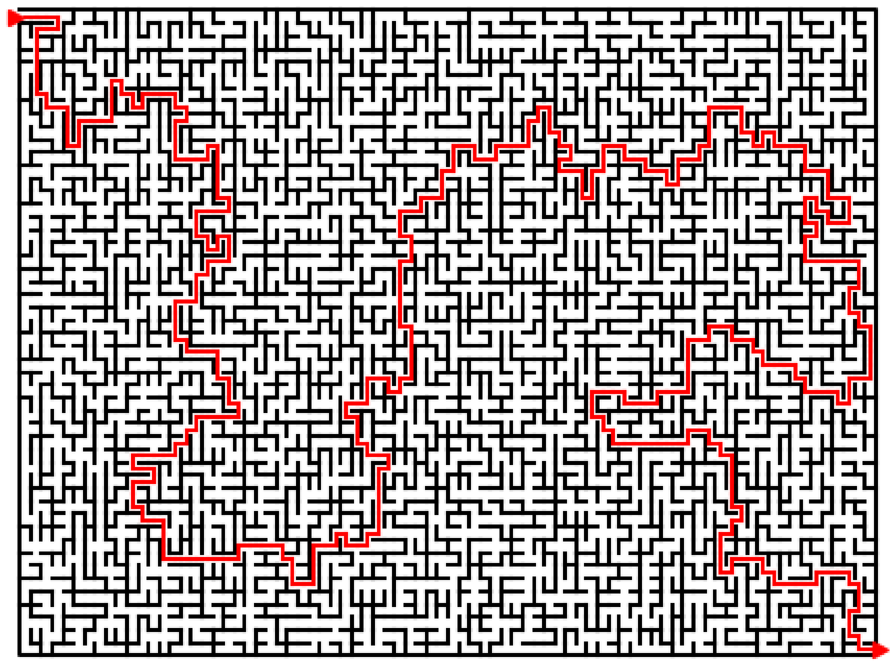
Faster Maze Solver

CSCI112, Lab 11

March 21, 2023



File names: Names of files, functions, and variables, when specified, must be EXACTLY as specified. This includes simple mistakes such as capitalization.

Individual work: All work must be your own. Do not share code with anyone other than the instructor and teaching assistants. This includes looking over shoulders at screens with the code open. You may discuss ideas, algorithms, approaches, *etc.* with other students but NEVER actual code. Do not use code written by anyone else, in the class or from the internet.

Documentation: Each file should begin with a docstring that includes your name, the class number and name, the lab number, and a short description of the lab, as well as documentation pertinent to that particular file.

Speeding up search with heuristics: When your depth-first solver from last week was deciding which node to explore next, it had no guidance at all from the maze itself. A human will look at the maze and use spatial reasoning to make reasonable choices. Not infallible choices, just reasonable ones.

We are going to give our maze solver a bit of intelligence by using a priority queue for the gray cells instead of a stack. Each node's priority will be determined by the Manhattan distance from the goal. Manhattan distance between two points is defined as follows:

$$\text{manhattan}((x_1, y_1), (x_2, y_2)) = |x_1 - x_2| + |y_1 - y_2|$$

where $|x|$ is the absolute value of x.

Our heuristic is that nodes that are *closer* to the goal are more likely to be on the solution path than nodes that are *farther*. We will accordingly use a min queue for the cells on the gray list, and use the cell closest to the goal each time.

Timing: Run some experiments to see how much you've speeded up the search. In lieu of actual timing, you can compare the number of nodes added to the gray list before termination. The heuristic should reduce this.