CS5400 Spring 2019
Puzzle 4: Bonus 1

William Lorey
wwlytc@mst.edu

Contents

Heuristic Design

For this bonus assignment, two heuristics were created. The first, Heuristic 1, determines the Manhattan distance between the zeroth wriggler's end that is closest to the goal square and the goal square. Where the end (either head or tail) with the smaller Manhattan distance is the closer end to the goal square. This heuristic was chosen because it optimistically approximates the number of moves it would take for the wriggler to reach a goal state in the absolute best case - where no obstacles block its way.

Heuristic 2 computes the number of obstacles between the zeroth wriggler and the goal square, where obstacles consist of segments from other wrigglers as well as walls. To count the number of obstacles, the wriggler's end closest to the goal square was found by computing the Manhattan distance of both ends. Then, a subset of the board was iterated through with the upper left corner of the sub-board being the wriggler's end closest to the goal state and the lower right corner being the goal state. This heuristic is less optimistic than Heuristic 1 because it takes into account what must move out of the wriggler's way for the goal state to be reached while not including the moves it will take the wriggler to make it to the goal state once its path is clear.

Heuristic Performance Comparison

For the four given puzzles, the effective branching factor (b^*) was computed for both Heuristic 1 and Heuristic 2. The equation for the effective branching factor, as implemented in `main.py`, was derived as follows:

$$N + 1 = 1 + b^* + (b^*)^2 + ... + (b^*)^d = \frac{1 - (b^*)^{d+1}}{1 - b^*}$$

$$(N + 1)(1 - b^*) = 1 - (b^*)^{d+1}$$

$$(b^*)^{d+1} - (N + 1)b^* + N = 0$$

Where N is the total number of nodes generated by the search algorithm and d is the depth at which the goal node was found. The N and d values were substituted into the above polynomial to solve for b^* for each run of the search algorithm. Table 1 shows the experimentally determined b^* values.

Table 1: Effective Branching Factors (b^*) for Heuristic 1 and Heuristic 2

	b^* for Heuristic 1 (Manhattan distance)	b^* for Heuristic 2 (number of obstacles)
puzzle1	1.19916	1.19916
puzzle2	1.50096	1.46472
puzzle3	1.12217	1.12079
puzzle4	1.13851	1.13662

From the table, it can be seen that Heuristic 2 produced consistently lower effective branching factors than Heuristic 1, except for puzzle1 where the b^* values for both algorithms were equal. Although the sample size for this experiment was relatively small, it can be determined that Heuristic 2 is a more effective heuristic than Heuristic 1 for the given problem spaces because its effective branching factor was for the most part smaller than that of Heuristic 1.