Homework 1

Here is the definition of Color-Maze Puzzle as a Search Problem:

- States: Configurations of the board with each cell able to be: Agent cell := “S”, Unvisited Cell:=”0”, Wall Cell := “X”, and Visited Cell := “1”.
- Successor function: Movement of the ball. When a direction is chosen, all cells in that direction until a wall become visited, and the last cell becomes the Agent Cell.
- Initial state: The input of the unsolved board.
- Goal test: Checks if the board has any unvisited cells. If there are none, the goal is reached.
- Step cost function: The distance traveled by the ball per move. Distance is equal to the number of cells the ball moves over.

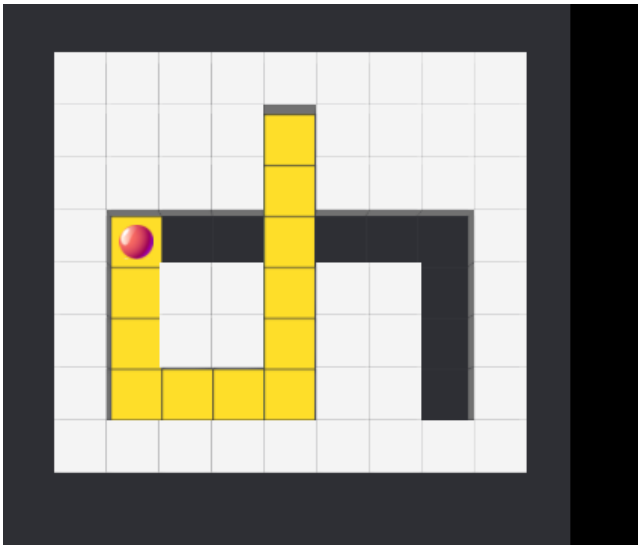
Heursitic function h1

The heuristic function h1 is defined as follows: the cost to reach the goal equals the number of unvisited cells on the board multiplied by 5.

Admissibility:

This heuristic function is not admissible because there are cases where the actual cost to reach a node n is less than what the heuristic function h1 estimates.

For example:



Sample level from the game Color Maze Puzzle

In this state (let’s call it state1) h1 guesses the cost to getting to the goal as 48.

(8 unvisited cells X 6= 48)

However the actual cost of getting towards the goal (the h* function) is 9. (The distance the ball moves when the LEFT AND DOWN moves are executed in sequence)

Since h1(state1)>h*(state1); heuristic function h1 is not admissible.

Heursitic function h2

The Heursitic function h2 is defined as follows: The cost to goal is equal to the number of unvisited cells on the board.

Admissibility:

This function is admissible because one single move can only lower the number of the unvisited cells less then or equal to its single due to the nature of the game. The reason is a move can go over already visited cells. This means the true cost of getting to a goal from a node n (or in other words h*(n)) will always be higher or equal to the value h2(n). Therefore its proven that h2 is an admissible heuristic function.

Monotonicity:

One single move can only lower the number of the unvisited cells less then or equal to its single due to the nature of the game. Because it only lowers the number of zeroes by going over cells and not all cells have to be 0. some cells could have been 1 due to a move that has been played before.

h2(n1) is the number of 0s on the board. cost(n1→n2) is the distance made by the move. h2(n2) is the number of zeroes after the move

therefore:

$h2(n1)-h2(n2) \leq c(n1 \rightarrow n2)$

This means h2 is monotonic.

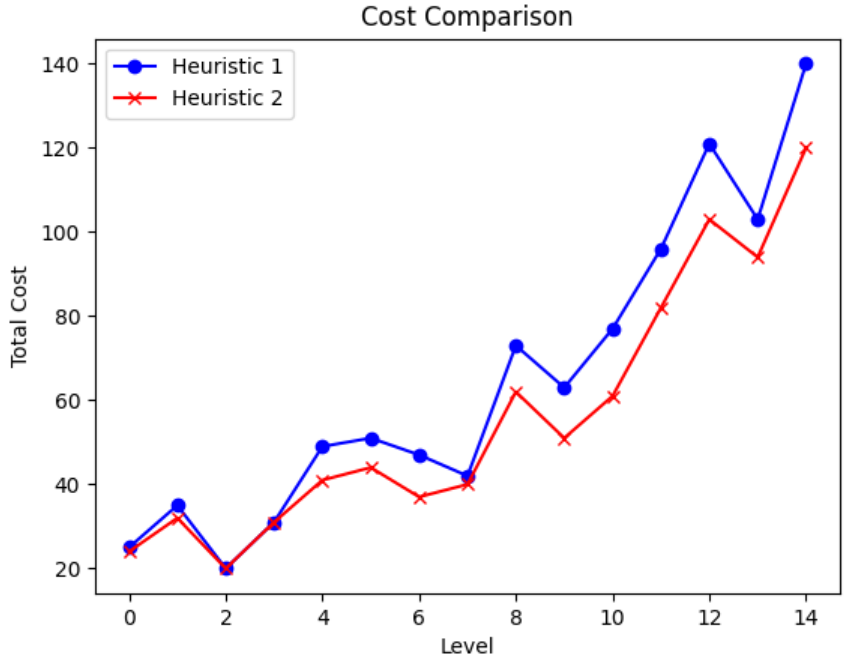
Benchmarking Results

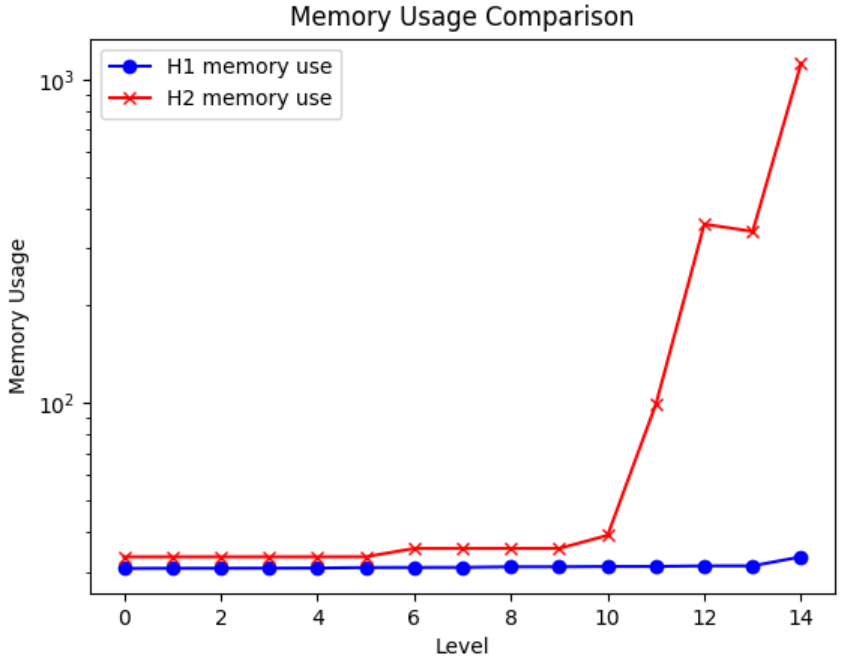
Here are the results of CPU time, memory usage and the total distance travelled for each level.

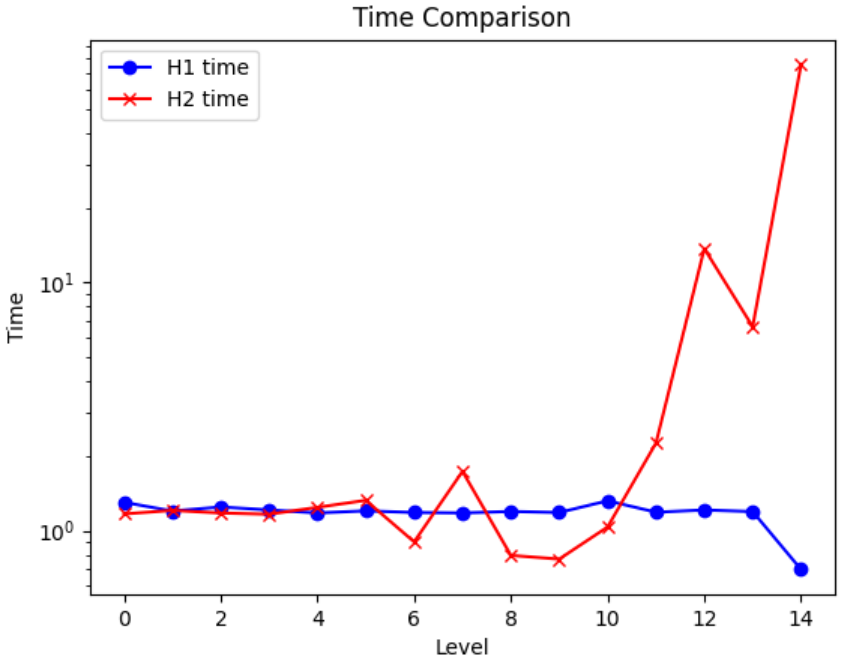
- The algorithm performs optimally at each level when using heuristic 2.
- Every cost result, indicated by the total distance travelled, is less with heuristic 2 than with heuristic 1.
- This outcome is anticipated, as heuristic 2 is admissible while heuristic 1 is not.
- The algorithm that uses heuristic 2 demonstrates exponential growth in space and time complexity. This is consistent with the A* algorithm's expected complexity of O(b^d), where 'b' represents the branching factor and 'd' is the total depth of the solution.
- In contrast, heuristic 1 does not follow this pattern exactly. It runs more quickly but does not provide optimal results.
- The cause for this result might be the non-optimality of heuristic 1 since it causes the A* search algorithm to behave closer to a greedy best-first search algorithm, as it overestimates the actual costs of nodes.
- This overestimation leads to more aggressive exploration of the search space and results in easier-to-find but non-optimal solutions.

The Table and Graphs

Level	Cell Count	H1 Cost	H1 Time	H1 Memory	H2 Cost	H2 Time	H2 Memory
easy_level1.txt	42	25	1.236555576324463	31.078125	24	1.4014368057250977	33.7265625
easy_level2.txt	42	35	1.2478806972503662	31.109375	32	1.7249207496643066	33.73828125
easy_level3.txt	25	20	1.2888941764831543	31.109375	20	1.8162462711334229	33.73828125
easy_level4.txt	42	31	1.5498640537261963	31.12890625	31	1.7118909358978271	33.73828125
easy_level5.txt	42	49	1.6290569305419922	31.15625	41	1.3698441982269287	33.74609375
normal_level1.txt	49	51	1.5535387992858887	31.28515625	44	0.9486324787139893	33.75
normal_level2.txt	42	47	1.5226330757141113	31.30078125	37	1.0114965438842773	35.796875
normal_level3.txt	56	42	1.648881196975708	31.30078125	40	1.2694416046142578	35.796875
normal_level4.txt	56	73	1.4908523559570312	31.39453125	62	1.033308982849121	35.80859375
normal_level5.txt	56	63	1.5048975944519043	31.3984375	51	1.1984663009643555	35.81640625
hard_level1.txt	64	77	1.4446113109588623	31.44921875	61	1.161177635192871	39.2265625
hard_level2.txt	108	96	1.637122392654419	31.50390625	82	3.0256733894348145	100.0234375
hard_level3.txt	88	121	1.4945533275604248	31.62890625	103	19.416344165802002	357.16015625
hard_level4.txt	108	103	1.4675366878509521	31.62890625	94	10.699127197265625	342.17578125
hard_level5.txt	81	140	0.8671703338623047	33.7265625	120	86.83239960670471	1118.5546875




Memory Usage Comparison
H1 memory use
H2 memory use
Memory Usage
10^3
10^2
0
2
4
6
8
10
12
14
Level


Time Comparison
H1 time
H2 time
Time
10^1
10^0
0
2
4
6
8
10
12
14
Level