

Jack Gallagher

Babak Forouraghi

Artificial Intelligence

2/27/2024

Assignment 4

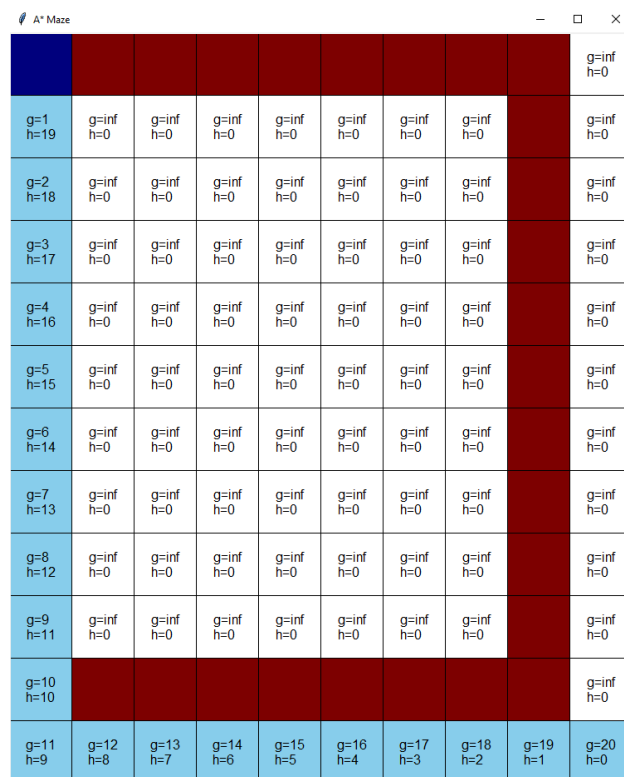
Problem 1

Objective: Modify AStarMaze to compare the behaviors of the Greedy Best-First and A* search algorithms.

A* Algorithm:

Code difference: $\text{new_g} = \text{current_cell.g} + 1$

Image of Maze:



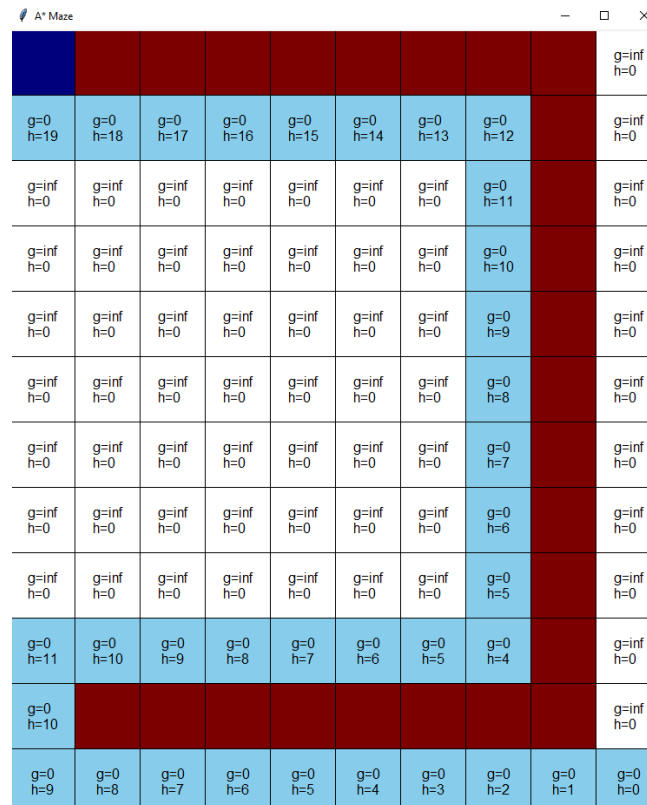
A* Maze visualization showing a 10x10 grid. The start cell (0,0) is blue, and the goal cell (9,9) is red. The path from start to goal is highlighted in blue. The grid contains g and h values for each cell.

	0	1	2	3	4	5	6	7	8	9
0										g=inf h=0
1	g=1 h=19	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
2	g=2 h=18	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
3	g=3 h=17	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
4	g=4 h=16	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
5	g=5 h=15	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
6	g=6 h=14	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
7	g=7 h=13	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
8	g=8 h=12	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
9	g=9 h=11	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
10	g=10 h=10									g=inf h=0
11	g=11 h=9	g=12 h=8	g=13 h=7	g=14 h=6	g=15 h=5	g=16 h=4	g=17 h=3	g=18 h=2	g=19 h=1	g=20 h=0

Greedy Algorithm:

Code difference: `new_g = 0`

Image of Maze:



A* Maze									
									g=inf h=0
g=0 h=19	g=0 h=18	g=0 h=17	g=0 h=16	g=0 h=15	g=0 h=14	g=0 h=13	g=0 h=12		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=0 h=11		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=0 h=10		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=0 h=9		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=0 h=8		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=0 h=7		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=0 h=6		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=0 h=5		g=inf h=0
g=0 h=11	g=0 h=10	g=0 h=9	g=0 h=8	g=0 h=7	g=0 h=6	g=0 h=5	g=0 h=4		g=inf h=0
g=0 h=10									g=inf h=0
g=0 h=9	g=0 h=8	g=0 h=7	g=0 h=6	g=0 h=5	g=0 h=4	g=0 h=3	g=0 h=2	g=0 h=1	g=0 h=0

Analysis:

The change that needed to be made to the a* algorithm code to make it into a greedy algorithm was to set `g=0`. This is due to `g` representing actual cost which the Greedy algorithm does not take into account when pathing through the maze. This leads the Greedy Algorithm to make the best move at its current position not accounting for the future moves ahead as it leans solely on heuristic. On the other hand, the a* algorithm finds the fastest path utilizing both heuristics and actual cost. The more thorough pathing from the a* algorithm as seen in the images above gives it the faster path to the targeted goal.

Problem 2

Objective: Repeat the above experiment but this time: Use the Euclidean Distance heuristic.

The agent is allowed to make diagonal moves (i.e., NE, NW, SE, SW) in addition to the usual N, S, E, and W moves. The moves are made randomly and not in any specific order.

Notable code changes: Put directions in a variable directions and added a

`random.shuffle(directions)` to randomize the move order, added in (1,1) (-1,1) (-1,-1) (1,-1) and

```
swapped the manhattan distance with euclidean distance math.sqrt((pos[0] - self.goal_pos[0]) **
2 + (pos[1] - self.goal_pos[1]) ** 2)
```

A* Algorithm:

A* Maze								-	<input type="checkbox"/>	
										g=inf h=0
g=inf h=0	624847486597	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0			g=inf h=0
g=inf h=0	g=11 h=11.40175425099130	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0			g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=3 h=10.0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0			g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=8.602325267042020	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0			g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=7.21102550927970	g=inf h=0	g=inf h=0	g=inf h=0			g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0.0710678118654755	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0			g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=7.21102550927970	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0			g=inf h=0
g=inf h=0	g=inf h=7.615773105863909	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0			g=inf h=0
g=inf h=8.246211251235820		g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0			g=inf h=0
5385138137417										g=inf h=0
g=inf h=0	g=11 h=8.0	g=12 h=7.0	g=13 h=6.0	g=14 h=5.0	g=15 h=4.0	g=16 h=3.0	g=17 h=2.0	g=18 h=1.0	g=19 h=0.0	

Greedy Algorithm:

[illegible]

Analysis: After applying the changes noted above we get the expected change of the maze having the ability to go in diagonal directions. This led to both mazes utilizing the diagonal almost immediately in attempts to cut the time down. Notably again the greedy algorithm did not take into account the future moves when running through the maze leading itself into a corner and having to wrap back around the wall. The a* algorithm, now utilizing the diagonal, is still able to find its way to the finish in the minimum number of moves. The only notable difference between the two algorithms even with the changes was simply the greedy algorithm's disregard for g.

Problem 3

Objective: Explain how different values of weight on $g(n)$ and $h(n)$ affect the A* algorithm's behavior.

Notable Code Change: added multiplication weights into the $f(n) = g(n) + h(n)$:

```
self.cells[new_pos[0]][new_pos[1]].f = new_g * 1 + self.cells[new_pos[0]][new_pos[1]].h * 1
```

NOTE: The bolded 1s are the weights and they were changed with each test.

Table:

α	β	Observed
1	1	Stays the same as no weight
1	5	Went right 5 spaces before going down
5	1	Stays the same as no weight
2	5	Only goes right 3 times before going down
8	40	Goes over right 5 times before going down
100	40	Same as above
3	4	Goes over right 1 time before going down

1, 5:

A* Maze

									g=inf h=0
g=1 h=19	g=2 h=18	g=3 h=17	g=4 h=16	g=5 h=15	g=6 h=14	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=7 h=13	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=8 h=12	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=9 h=11	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=10 h=10	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=11 h=9	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=12 h=8	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=13 h=7	g=inf h=0	g=inf h=0		g=inf h=0
g=19 h=11	g=18 h=10	g=17 h=9	g=16 h=8	g=15 h=7	g=14 h=6	g=inf h=0	g=inf h=0		g=inf h=0
g=20 h=10									g=inf h=0
g=21 h=9	g=22 h=8	g=23 h=7	g=24 h=6	g=25 h=5	g=26 h=4	g=27 h=3	g=28 h=2	g=29 h=1	g=30 h=0

8, 40:

A* Maze

									g=inf h=0
g=1 h=19	g=2 h=18	g=3 h=17	g=4 h=16	g=5 h=15	g=6 h=14	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=7 h=13	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=8 h=12	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=9 h=11	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=10 h=10	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=11 h=9	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=12 h=8	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=13 h=7	g=inf h=0	g=inf h=0		g=inf h=0
g=19 h=11	g=18 h=10	g=17 h=9	g=16 h=8	g=15 h=7	g=14 h=6	g=inf h=0	g=inf h=0		g=inf h=0
g=20 h=10									g=inf h=0
g=21 h=9	g=22 h=8	g=23 h=7	g=24 h=6	g=25 h=5	g=26 h=4	g=27 h=3	g=28 h=2	g=29 h=1	g=30 h=0

100,40:

A* Maze

									g=inf h=0
g=1 h=19	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
g=2 h=18	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
g=3 h=17	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
g=4 h=16	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
g=5 h=15	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
g=6 h=14	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
g=7 h=13	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
g=8 h=12	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
g=9 h=11	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
g=10 h=10									g=inf h=0
g=11 h=9	g=12 h=8	g=13 h=7	g=14 h=6	g=15 h=5	g=16 h=4	g=17 h=3	g=18 h=2	g=19 h=1	g=20 h=0

3,4:

A* Maze

									g=inf h=0
g=1 h=19	g=2 h=18	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=3 h=17	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=4 h=16	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=5 h=15	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=6 h=14	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=7 h=13	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=8 h=12	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
g=inf h=0	g=9 h=11	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
g=11 h=11	g=10 h=10	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0	g=inf h=0		g=inf h=0
g=12 h=10									g=inf h=0
g=13 h=9	g=14 h=8	g=15 h=7	g=16 h=6	g=17 h=5	g=18 h=4	g=19 h=3	g=20 h=2	g=21 h=1	g=22 h=0

Analysis:

After testing out a bunch of different weights I came to the conclusion that for this particular maze boosting the weight higher for alpha did not seemingly change the outcome of the journey through the maze. When increasing the beta though you could see more attention being given to the heuristics as it would end up going right a couple of times before the actual cost would kick in. Looking at the data that I had collected the beta/alpha rounded up was always the amount of times the maze would move over to the right making me believe that the weight always ends up being the beta divided by the alpha as the indicator of weight. In conclusion though, however much weight you added to the beta would alter the A* algorithm's path to favor heuristics.