Brittany Klose
AI

Assignment 4: Star Maze

Problem 1:

The first problem is a modification of a given code example ‘AStarMaze’ to compare the behaviors of the Greedy Best-First and A* search algorithms. The maze configuration is modified to visually observe differences in the optimum paths generated by the two algorithms.

Below displays the results which can also be viewed in their labeled image files. On the left shows the Greedy Best First Search and on the right shows the path for A*. As can be seen, the A* path of 19 squares is more optimal over the GBF path of 27 squares. This test emphasizes that A* is a better path finding algorithm as It is always optimal and complete whereas GreedyBF is not always optimal as it picks the local optima at each stage, only focusing on the heuristic. While it does find the goal it takes a longer path than the A*.

Figure 1.1

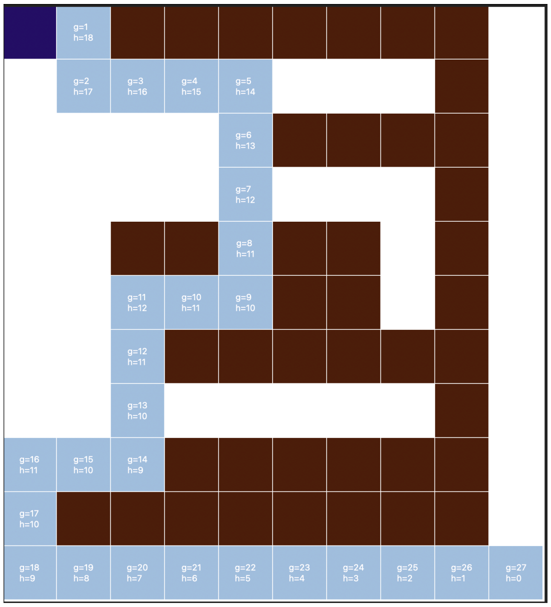
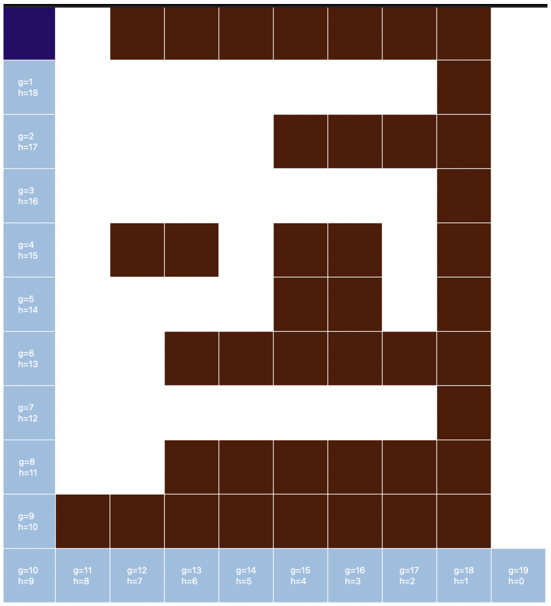


Figure 2.1



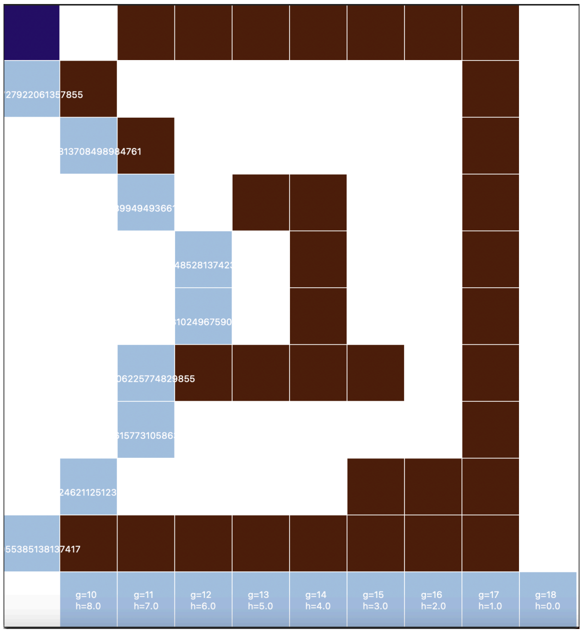
Problem 2:

The same experiment is repeated but this time using 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. And the moves are made randomly and not in any specific order. You can view the file marked problem 2 to see the actual code as well as the images marked with euclidean distance.

Two photos for the euclidean distance are included in the repository, though the same results were received both GBF and A* when switching to euclidean seen below. Despite switching up the layout of the maze a few times their paths remained the same. This is likely because the euclidean distance algorithm always goes in a straight line. Thus, it is not a great distance algorithm to use especially to show a comparison between A* and GBF. The inclusion of diagonal moves offers an essentially straight line halfway to the goal so it was not surprising the graphs were the same.

Figure 1.2



Problem 3:

The evaluation function in AstarMaze is defined as f(n) = g(n) + h(n). A weighted version of the function can be defined as:

f(n) = α * g(n) + β * h(n) where α, β >= 0

This problem evaluates how different values of α and β affect the A* algorithm's behavior. Results are documented below and all photos are included in HW4images folder.

α	β	**Observed Behavior**
2	2	Behaved the same as original behavior where values aren’t adjusted. Shown below in figure 1.2
3	4	Figure 2.2. Behaved more like GBF taking a slightly longer path.
5	7	Behaved the same as previous values where a=3, B=4. See figure 2.2
13	9	Behaved the same as the original behavior.
9	5	Behaved the same as the original behavior.
2	11	Least optimal behavior. Acted more like GBF taking a longer path.

Based on the results in the table above it appears that a change in the value of α has very little effect on the algorithm. If the value of α is > = β there is no change to the maze path. Whereas when $\beta > \alpha$ the path through the maze is less optimal and gets less so as the value of β increases. The original algorithm and the input values of β and α that gave the same result had a path cost of 19. When β and α values were changed to 3 & 4 the path cost increased to 21 and when β and α values were changed to 2 & 11 the path cost increased to 27 making it the least optimal path found in the tests; supporting the theory that as the value of β increases the maze path the algorithm finds is less optimal.

Brittany Klose
AI

Original Figure 1.3

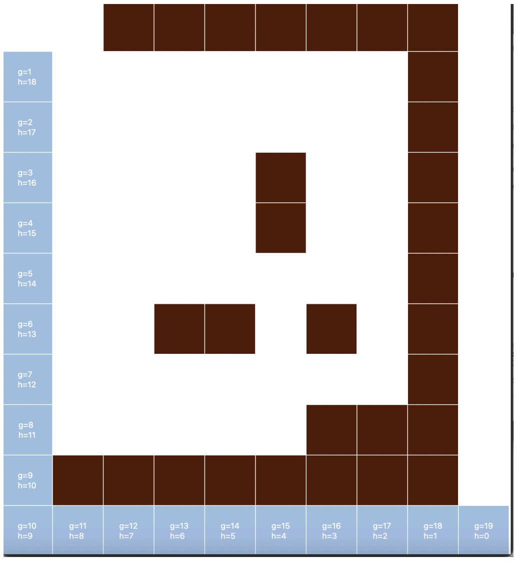


Figure 2.3

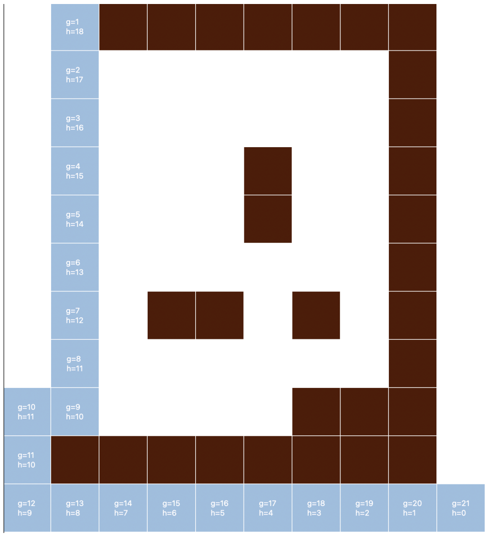
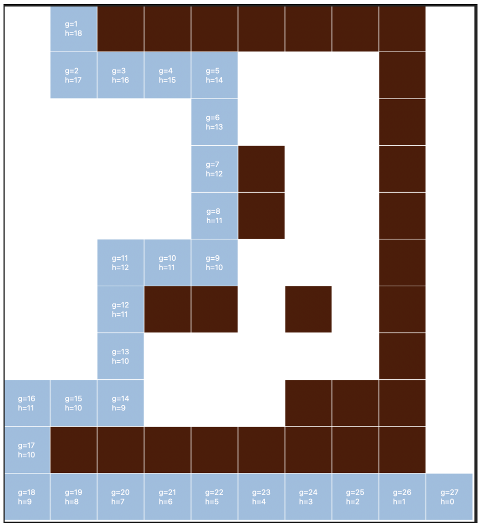


Figure 3.3



β can be considered the algorithm's bias towards states that are closer to goal. Run the algorithm for various values of the bias to determine what changes, if any, are observed in the optimum path. Include screenshots of the path for each specific value along with your explanation.

Brittany Klose
AI

β	Observed Behavior
2	Less optimal than original. Figure 4.3
4	Significantly less optimal than original and when value was 2. Figure 5.3
7	Same behavior as previous value of 4, see Figure 5.3.
11	Least optimal behavior taking the longest path resembling more of a GBF approach. Figure 6.3

As stated above it was found that only changing the value of β altered the algorithm's behavior. The images below help demonstrate how the path increases in cost as the value of β increases. In the first figure where β is 2 the cost g= 23. When β is 4 the cost g=27. And when β is 11 g=31.

Figure 4.3

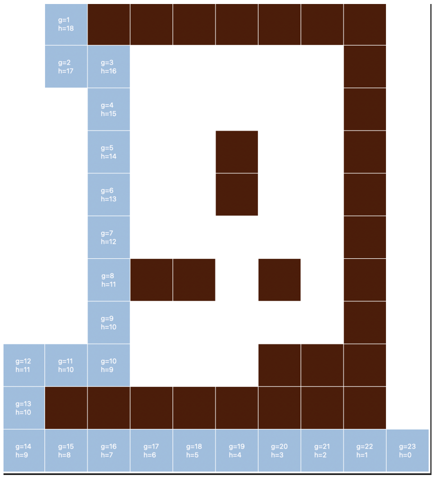


Figure 5.3

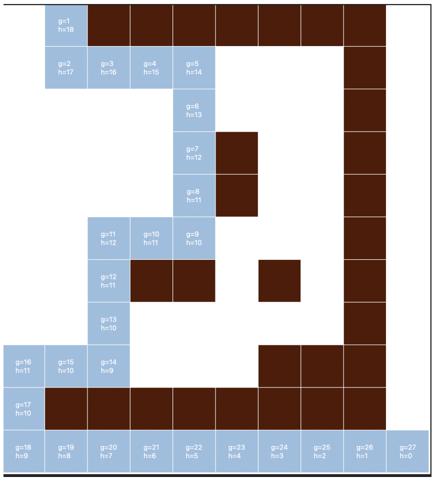
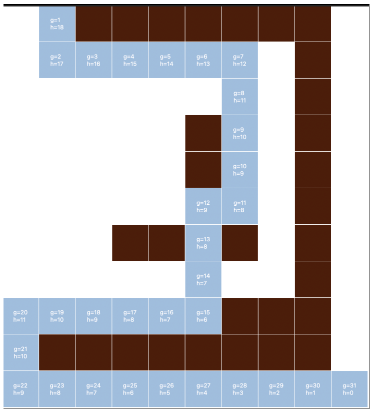


Figure 6.3


g=1 h=18
g=2 h=17
g=3 h=16
g=4 h=15
g=5 h=14
g=6 h=13
g=7 h=12
g=8 h=11
g=9 h=10
g=10 h=9
g=12 h=9
g=11 h=8
g=13 h=8
g=14 h=7
g=20 h=11
g=19 h=10
g=18 h=9
g=17 h=8
g=16 h=7
g=15 h=6
g=21 h=10
g=22 h=9
g=23 h=8
g=24 h=7
g=25 h=6
g=26 h=5
g=27 h=4
g=28 h=3
g=29 h=2
g=30 h=1
g=31 h=0