Hacettepe University - Computer Engineering

Artificial Intelligence CMP682 Homework

Generating Mazes and Applying Search Algorithms

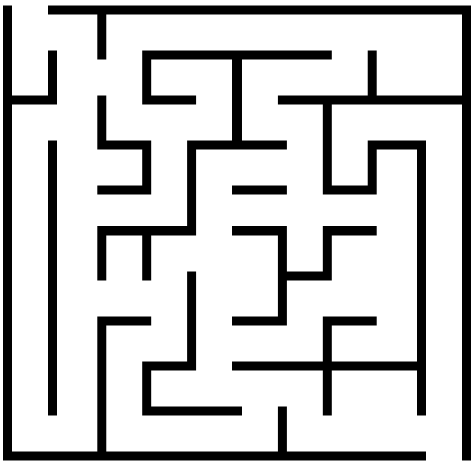
Student Name Ayça KULA

Student ID: 202237285

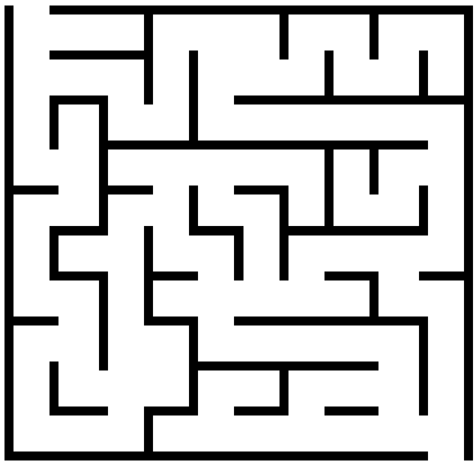
Fall 2021

1 Generate your own maze

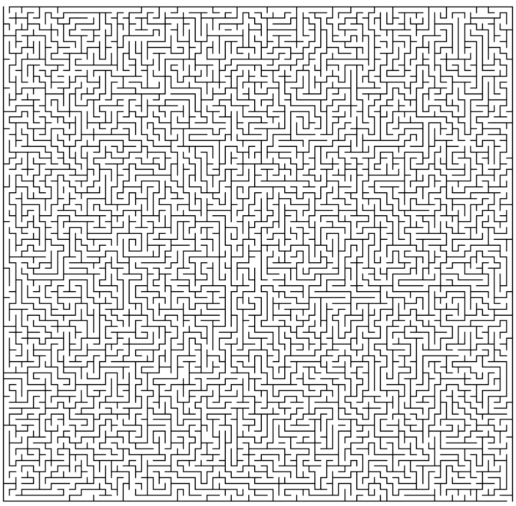
Mazes has been obtained using [1]. However, in the given reference there was a single path to the goal state. In [1] the tree was represented as a set of edges. Therefore, in order to add multiple paths the edges were modified.Random edges were added. In figures 1a, 1b, 1c and 1d it can be seen that from the start state there are many paths to the goal state.



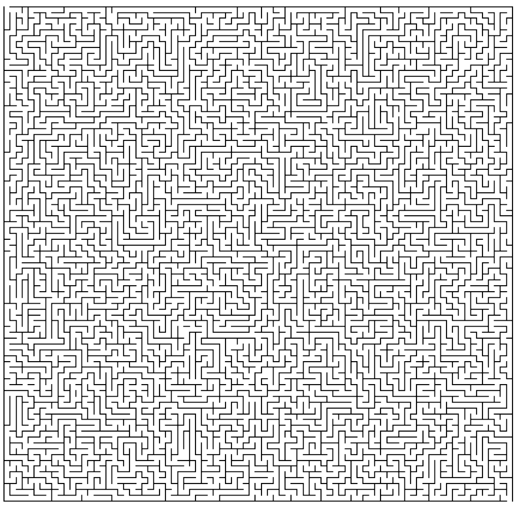
(a) First 10 x 10 Maze



(b) Second 10 x 10 Maze



(c) First 85 x 85 Maze



(d) Second 85 x 85 Maze

Figure 1: Randomly generated mazes with different sizes

2 Application of search strategies

In this part, Iterative Deepening Search, Uniform Cost Search, A* search with both Euclidean and Manhattan distance were applied. And, these search algorithms were obtained from [2]. By using each algorithm, each path was shown for each of 10x10 and 85x85 mazes.

2.1 Iterative deepening search

Iterative deep search has to give optimal solution if the step costs are equal. However, in figures 2 and 3, it can be clearly seen that the optimal solution was not found. Therefore, the algorithm does not work properly. It has to be improved for better result.

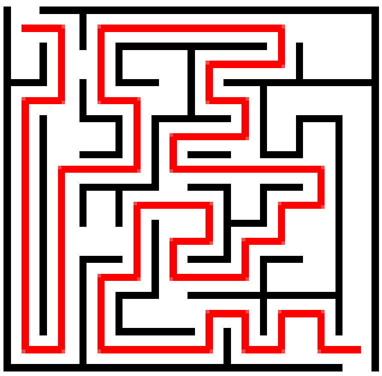


Figure 2: Iterative deepening search for the first 10x10 maze.

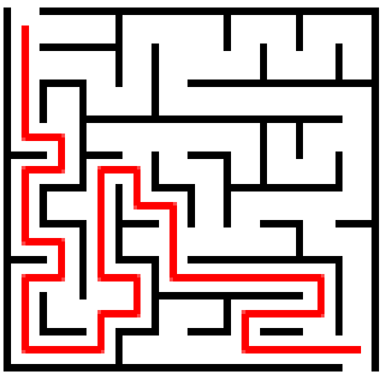


Figure 3: Iterative deepening search for the second 10x10 maze.

In figures 4 and 5 the algorithm was trying to find path for the size of 85x85 maze.

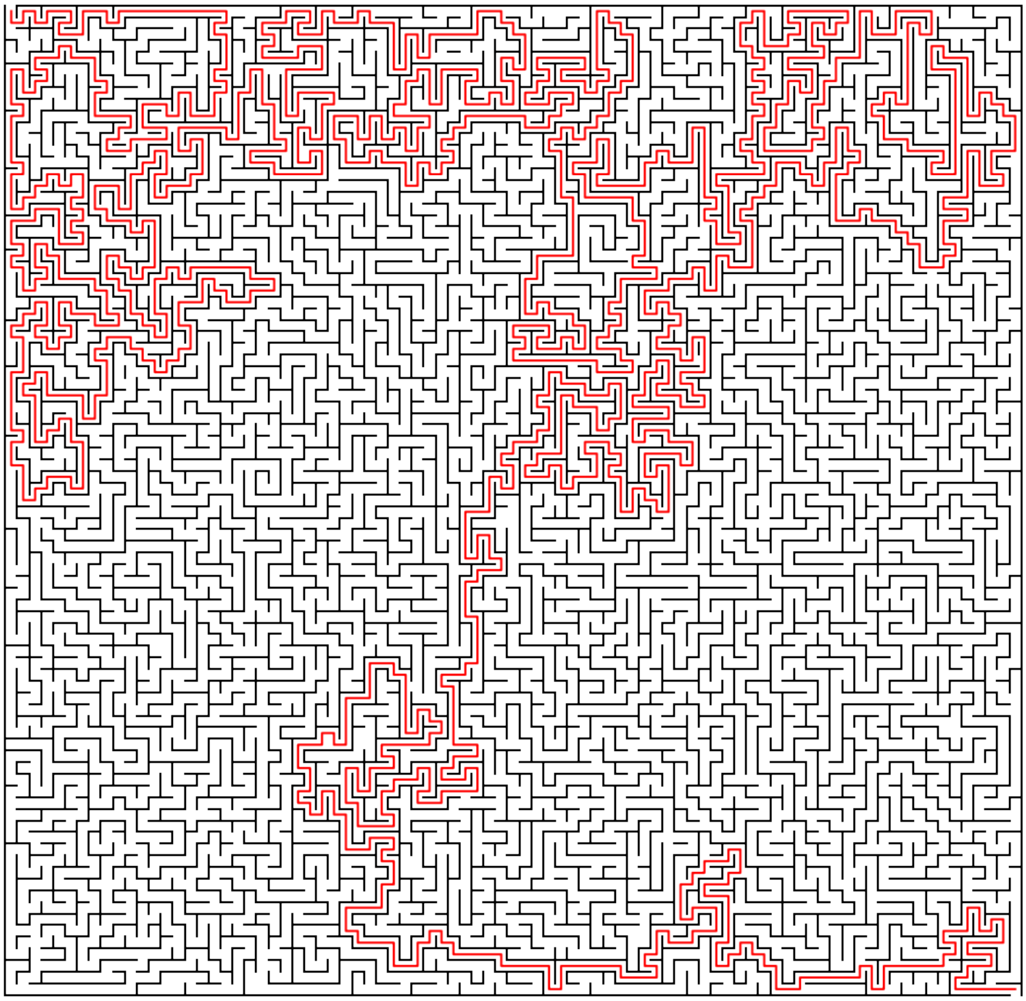


Figure 4: Iterative deepening search for the first 85x85 maze.

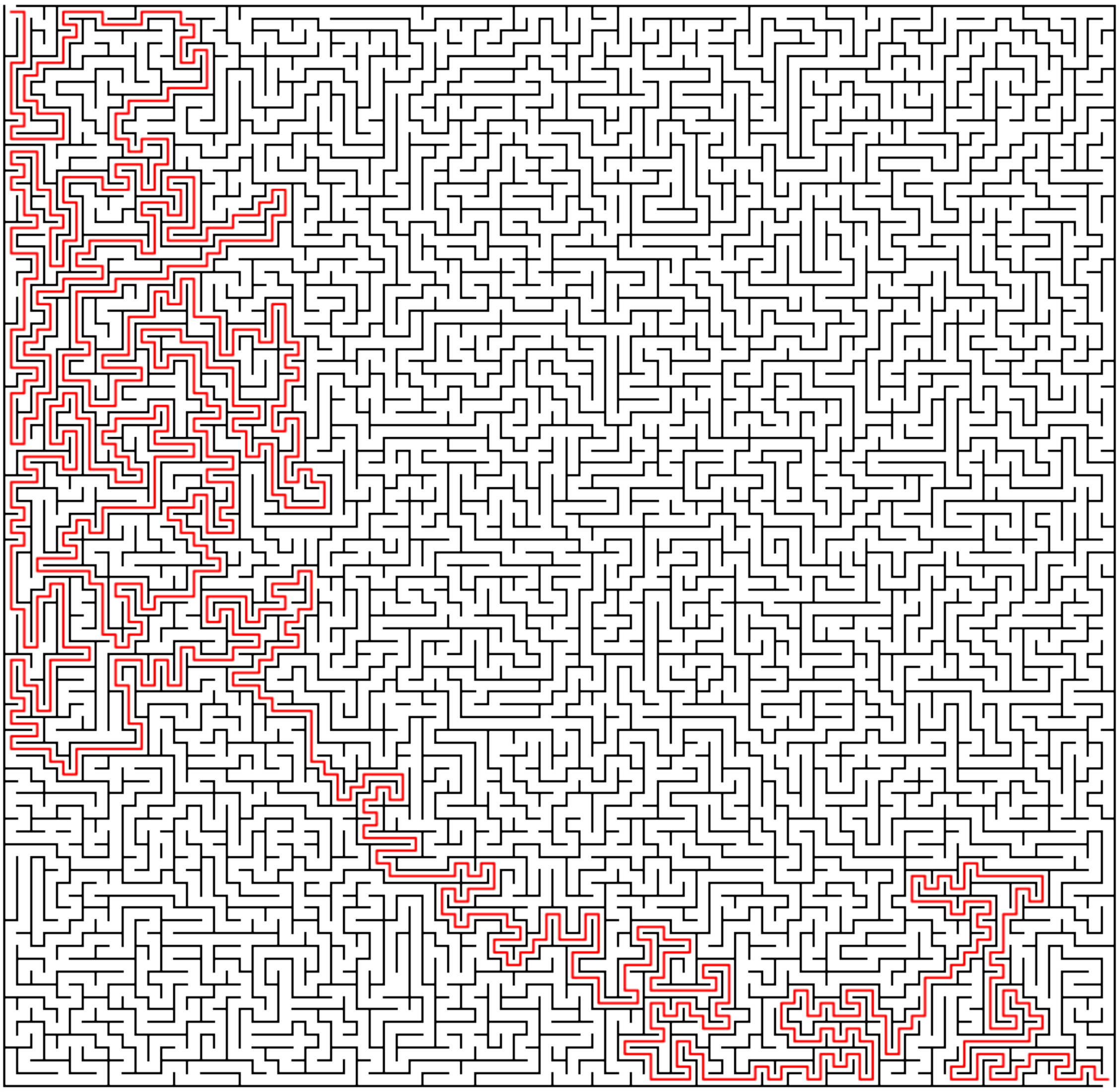


Figure 5: Iterative deepening search for the second 85x85 maze.

2.2 Uniform Cost Search

The result for the Uniform Cost Search algorithm is optimal. This situation can be seen from figures 6,7,8 and 9.

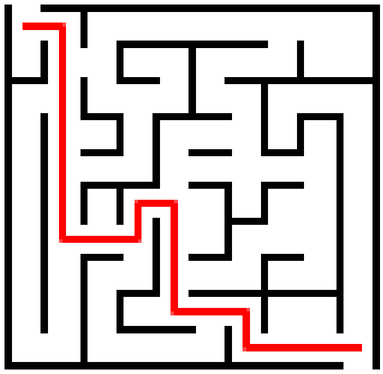


Figure 6: Uniform Cost Search for the first 10x10 maze.

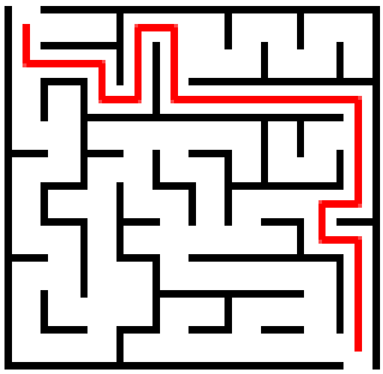


Figure 7: Uniform Cost Search for the second 10x10 maze.

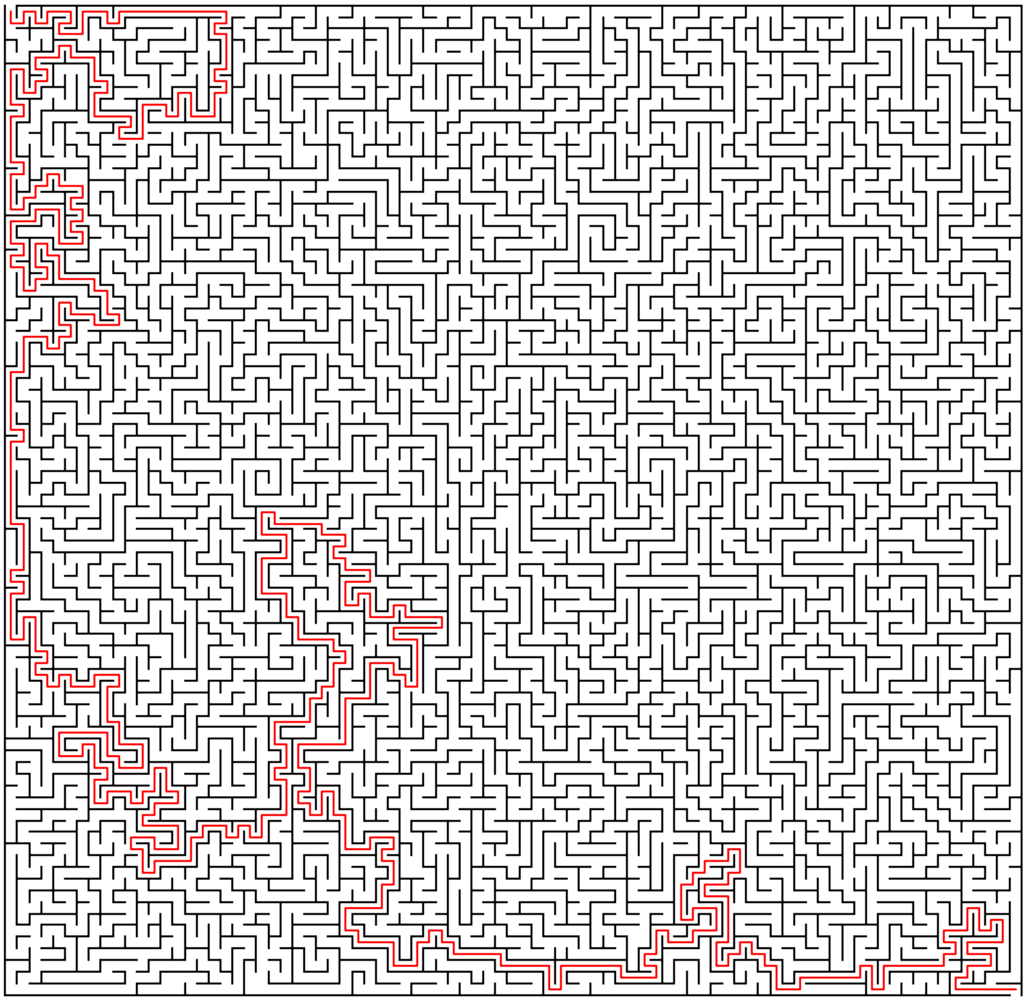


Figure 8: Uniform Cost Search for the first 85x85 maze.

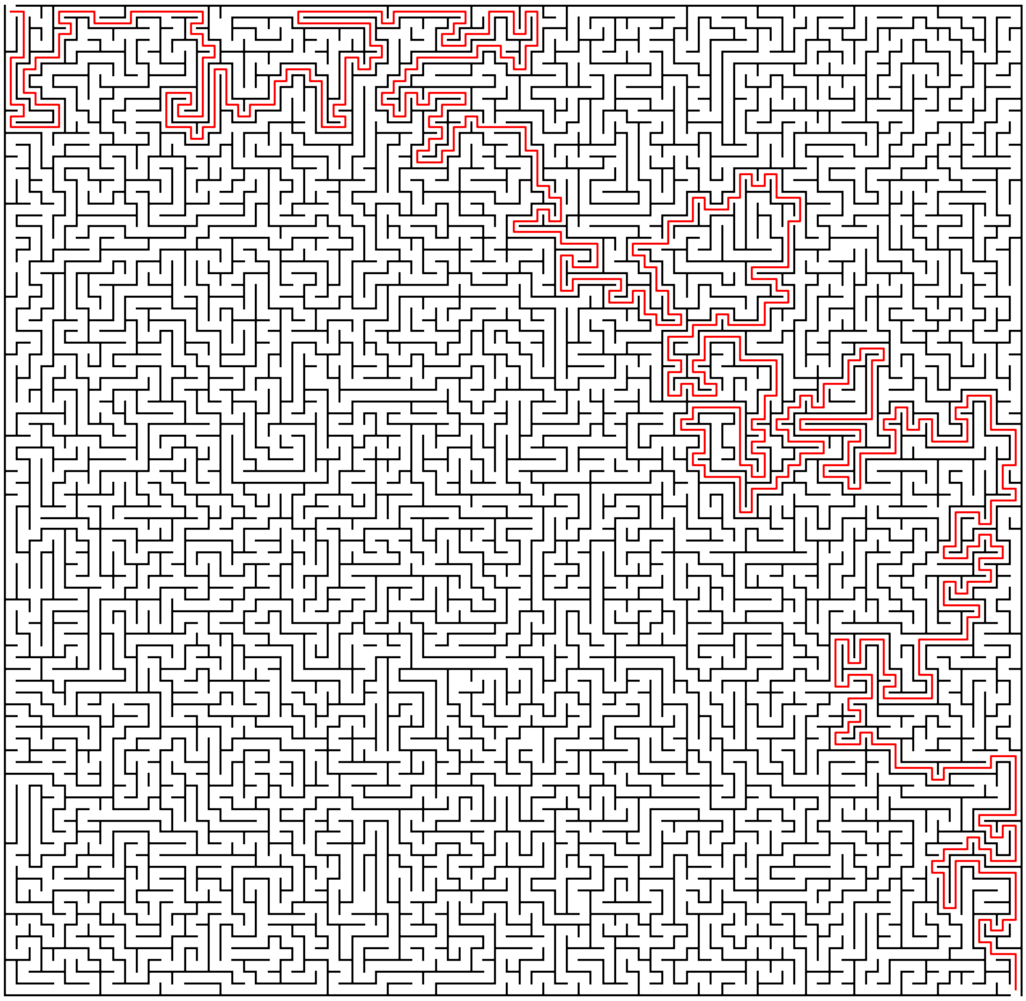


Figure 9: Uniform Cost Search for the second 85x85 maze.

2.3 A* search Using Euclidean distance

If the heuristics are admissible A* is optimal. Here in figure 10 and 11 this claim can be seen clearly. In order to find the path, the heuristic of Euclidean distance (1) is used for each edge in the maze. Euclidean distance was applied using [3].

$$\sqrt(\left((x_1 - x_2)^2 + (y_1 - y_2)^2\right) \tag{1}$$

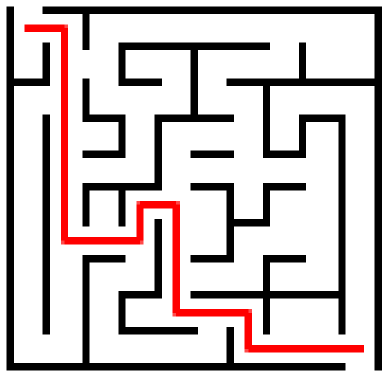


Figure 10: A* search Using Euclidean distance for the first 10x10 maze.

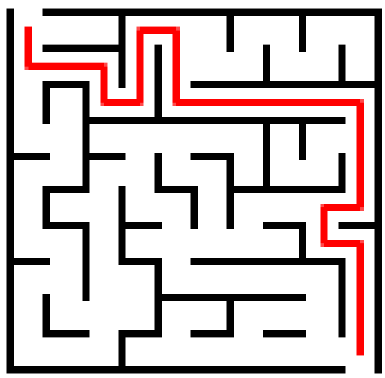


Figure 11: A* search Using Euclidean distance for the second 10x10 maze.

The optimal paths are also seen for 85x85 mazes as seen in figure 12 and 13.

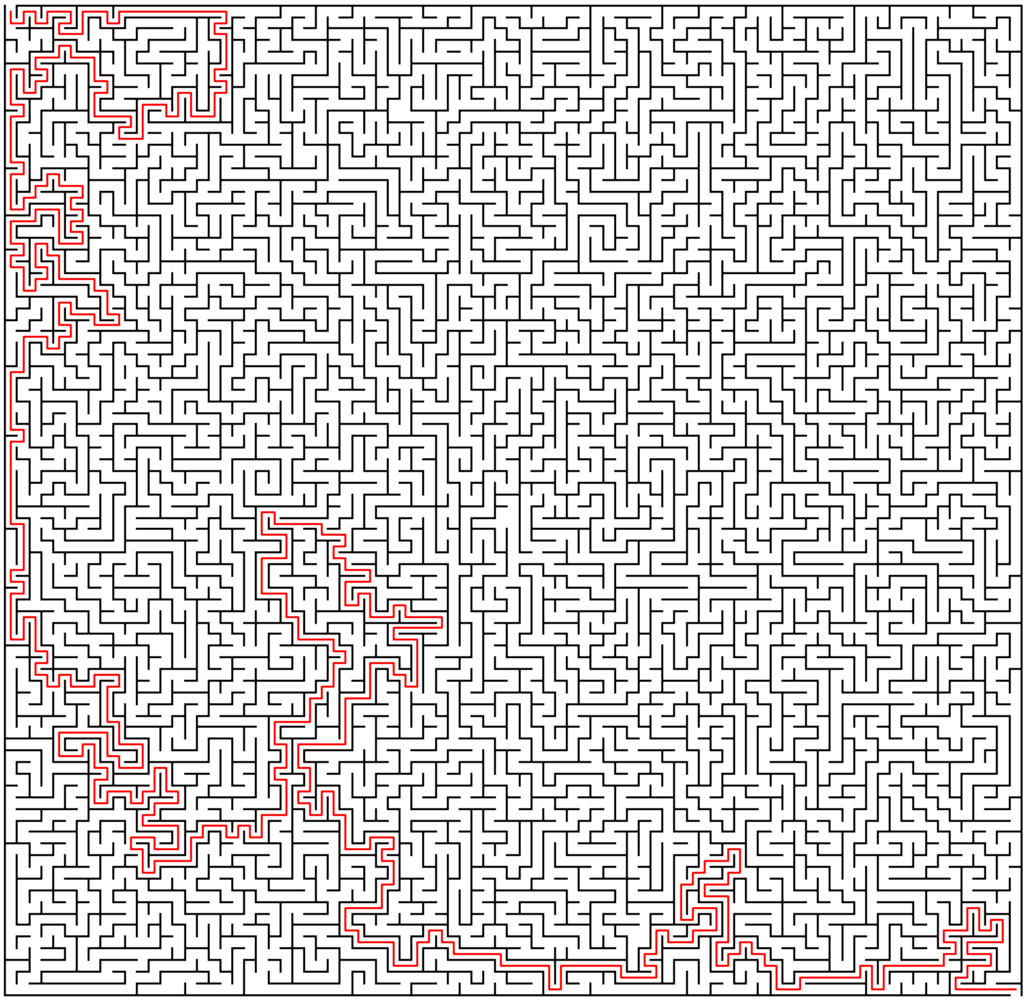


Figure 12: A* search Using Euclidean distance for the first 85x85 maze.

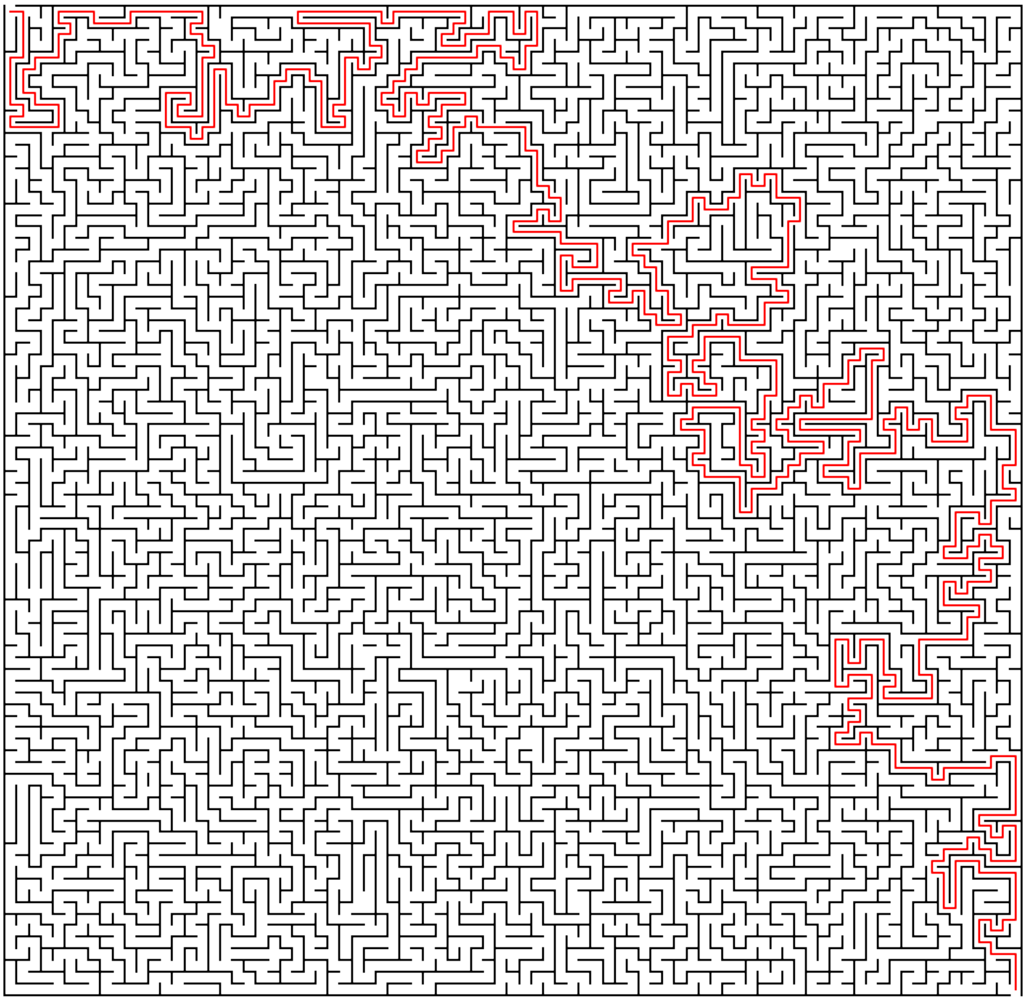


Figure 13: A* search Using Euclidean distance for the second 85x85 maze.

2.4 A* search Using Manhattan distance

Again, the optimal path is found as seen in figures 22 and 23. In order to find the path the heuristic Manhattan distance (2) can be also used. Manhattan distance was applied using [4] [5].

$$|x_1 - x_2| + |y_1 - y_2| \tag{2}$$

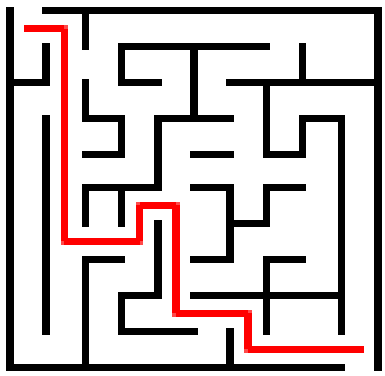


Figure 14: A* search Using Manhattan distance for the first 10x10 maze.

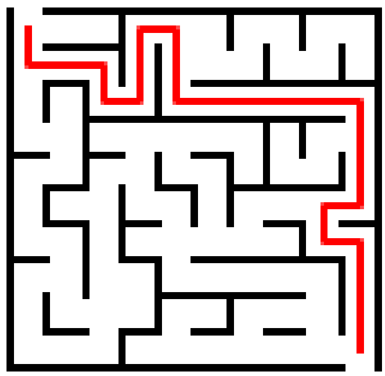


Figure 15: A* search Using Manhattan distance for the second 10x10 maze.

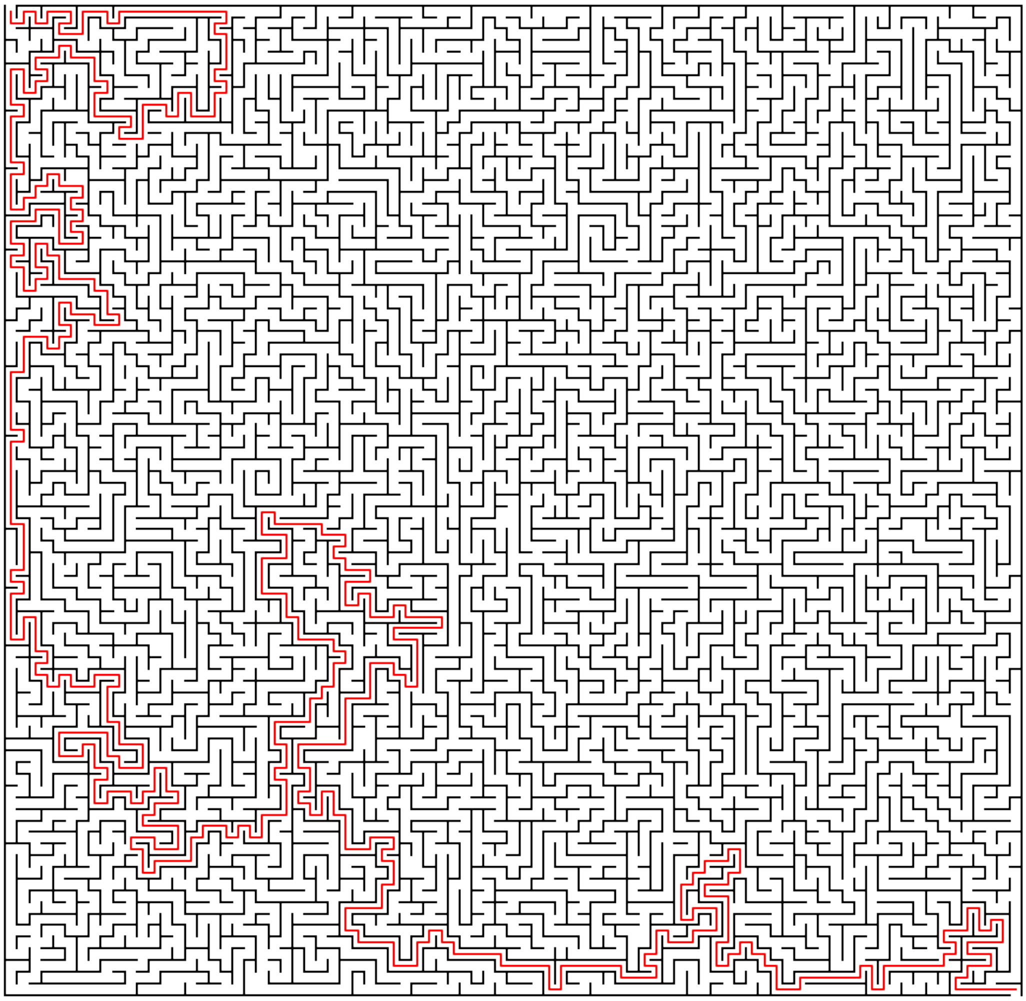


Figure 16: A* search Using Manhattan distance for the first 85x85 maze.

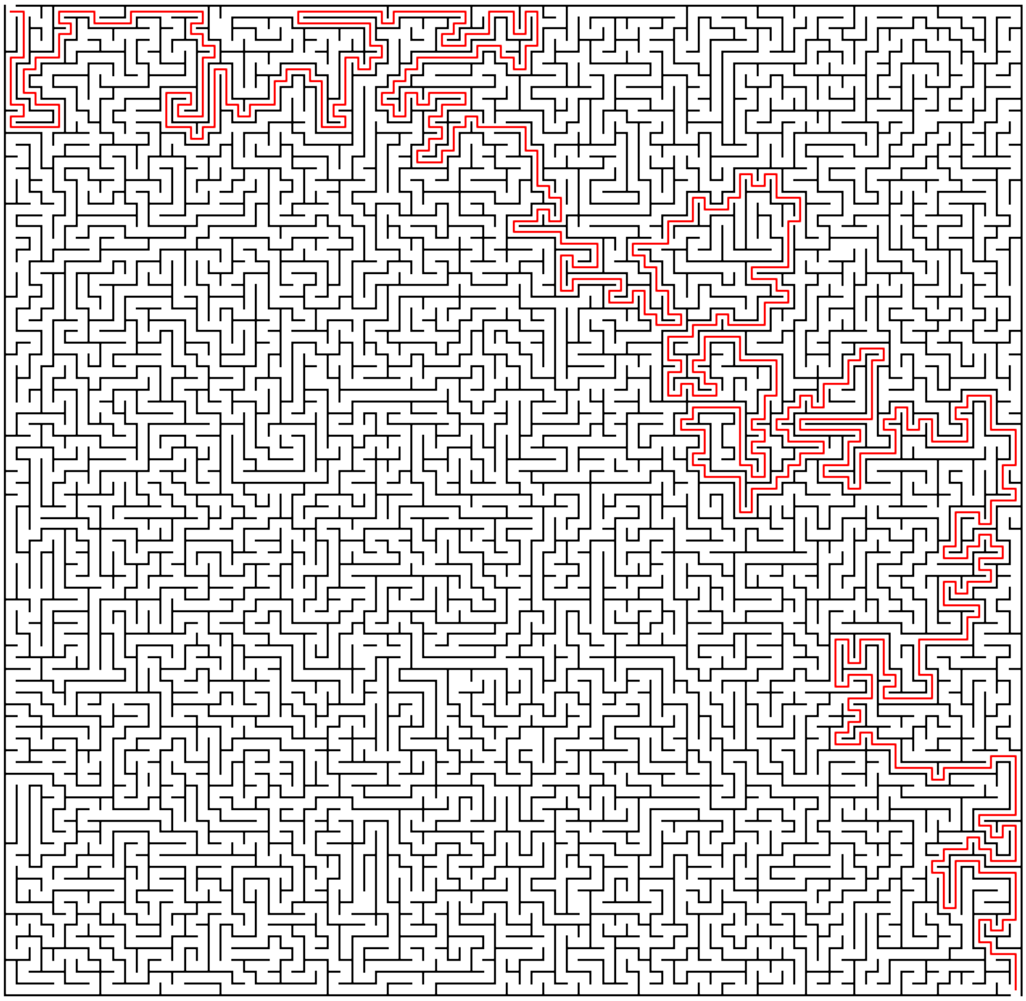


Figure 17: A* search Using Manhattan distance for the second 85x85 maze.

3 Analysis of the search strategies

From the tables below, each algorithm is compared with respect to its path length, number of expanded nodes and maximum time taken. As the maze size increases, path length, number of expanded nodes and maximum time taken increases.

In this study, we have to keep in mind that the Iterative Deepening search algorithm does not work properly. Therefore, the interpretation must be done regarding this situation. Since, 500x500 sized maze is very large the computer had an hard time solving these mazes. Especially, this can be seen from tables 7 and 8. Moreover, computer could not solve Iterative Deepening due to memory exceed. This is also related with algorithm which is not properly obtained.

Table 1: Algorithm results for the first maze of 10x10

	Length of path	Number of expanded nodes	Maximum time taken
Iterative Deepening	71	82	0.05289633700022023
Uniform Cost	21	90	0.002573739999661484
A* - Euclidean	21	79	0.001440939999611146
A* - Manhattan	21	72	0.001346466000086366
Iterative A* - Euclidean	21	79	0.009916256999986217
Iterative A* - Manhattan	21	72	0.02640870499999437

Table 2: Algorithm results for the second maze of 10 x10

	Length of path	Number of expanded nodes	Maximum time taken
Iterative Deepening	39	75	0.011728497000149218
Uniform Cost	25	89	0.00024512799973308574
A* - Euclidean	25	63	0.001750736000303732
A* - Manhattan	25	61	0.0024657909998495597
Iterative A* - Euclidean	25	63	0.01266693400020813
Iterative A* - Manhattan	25	61	0.01410218499995608

Table 3: Algorithm results for the first maze of 100x100

	Length of path	Number of expanded nodes	Maximum time taken
Iterative Deepening	1367	3808	5.776390059000278
Uniform Cost	265	9910	0.049114611000277364
A* - Euclidean	265	9547	0.14179569799989622
A* - Manhattan	265	8978	0.06741356900010942
Iterative A* - Euclidean	265	9547	3.189633918000254
Iterative A* - Manhattan	265	8978	3.6847190179996687

Table 4: Algorithm results for the second maze of 100 x100

	Length of path	**Number of expanded nodes**	**Maximum time taken**
Iterative Deepening	1197	5308	4.483825237000019
Uniform Cost	273	9999	0.032916887999817845
A* - Euclidean	273	9585	0.05123791300002267
A* - Manhattan	273	9105	0.04949112199983574
Iterative A* - Euclidean	273	9585	3.3974570759996823
Iterative A* - Manhattan	273	9105	4.155829610000183

Table 5: Algorithm results for the first maze of 85 x 85

	Length of path	**Number of expanded nodes**	**Maximum time taken**
Iterative Deepening	1561	2277	6.584827843999847
Uniform Cost	627	5638	0.10174117999986265
A* - Euclidean	627	5093	0.08065886599979422
A* - Manhattan	627	4827	0.08273000600001978
Iterative A* - Euclidean	627	5093	3.5873550299997987
Iterative A* - Manhattan	627	4827	3.9898510660000284

Table 6: Algorithm results for the second maze of 85 x 85

	Length of path	**Number of expanded nodes**	**Maximum time taken**
Iterative Deepening	1059	3256	2.2437255809995804
Uniform Cost	711	6716	0.024057443999936368
A* - Euclidean	711	6317	0.07224127200015573
A* - Manhattan	711	6153	0.0791031670000848
Iterative A* - Euclidean	711	6317	4.845320435999838
Iterative A* - Manhattan	711	6153	5.225856864999969

Table 7: Algorithm results for the first maze of 500 x 500

	Length of path	**Number of expanded nodes**	**Maximum time taken**
Iterative Deepening	-	-	-
Uniform Cost	3815	248474	1.3726657330003036
A* - Euclidean	3815	247109	1.3774991079990286
A* - Manhattan	3815	246960	1.779015903999607
Iterative A* - Euclidean	3815	247109	1474.2887173030003
Iterative A* - Manhattan	3815	246960	1677.608753652

Table 8: Algorithm results for the second maze of 500 x 500

	Length of path	**Number of expanded nodes**	**Maximum time taken**
Iterative Deepening	-	-	-
Uniform Cost	3471	248153	0.9655129970001326
A* - Euclidean	3471	247375	1.663772889000029
A* - Manhattan	3471	246992	1.8358616050009005
Iterative A* - Euclidean	3471	247375	1208.010327854
Iterative A* - Manhattan	3471	246992	1378.1752052800002

The optimal path was found from Uniform Cost, A* with Euclidean heuristics, A* with Manhattan heuristics, Iterative A* with Euclidean heuristics and Iterative A* with with Manhattan heuristics. However, Iterative Deepening search algorithm could not found the optimal path due to the improperly obtained algorithm. Also, the expanded nodes were increased using uniform cost search algorithm. Moreover, when we compare A* - Manhattan and Iterative A* -Manhattan, we see that they have the same number of expanded nodes and length of path. However, the maximum time taken has been increased for iterative A* search algorithm. The same situation can be seen for A* - Euclidean and Iterative A* -Euclidean. Time taken has been increased mostly in Iterative A* -Manhattan.

4 Iterative Deepening A* search

It is a variant of Iterative Deepening Depth-First Search that uses the idea of a heuristic to find only necessary nodes [6][7]. The algorithm has been obtained from [2].This algorithm uses less memory as it does not keep information and forgets after it reaches a certain depth and start over [8].It finds the optimal solution as seen from the tables given above. This algorithm is slower because of repeating the exploring of explored nodes and more processing power and time is necessary compared with A* [8]. The maximum time taken has been increases with respect to A* as seen in all tables given above.

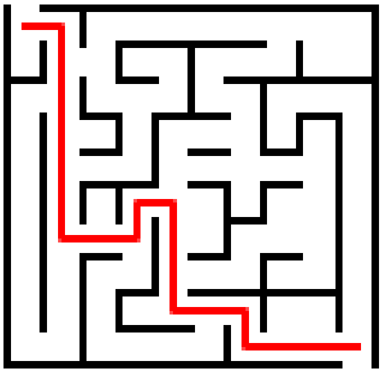


Figure 18: Iterative deepening A* search using Euclidean distance for the first 10x10 maze.

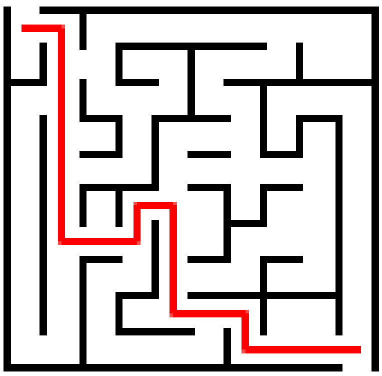


Figure 19: Iterative deepening A* search using Manhattan distance for the first 10x10 maze.

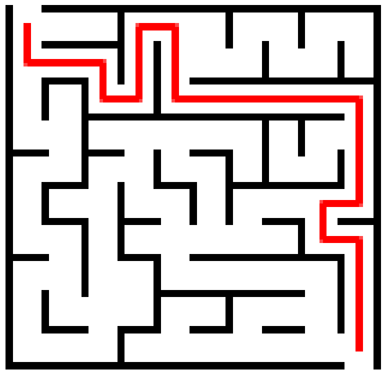


Figure 20: Iterative deepening A* search using Euclidean distance for the second 10x10 maze

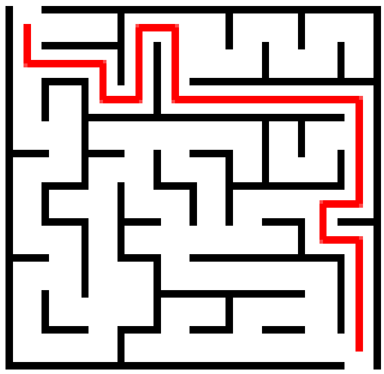


Figure 21: Iterative deepening A* search using Manhattan distance for the second 10x10 maze.

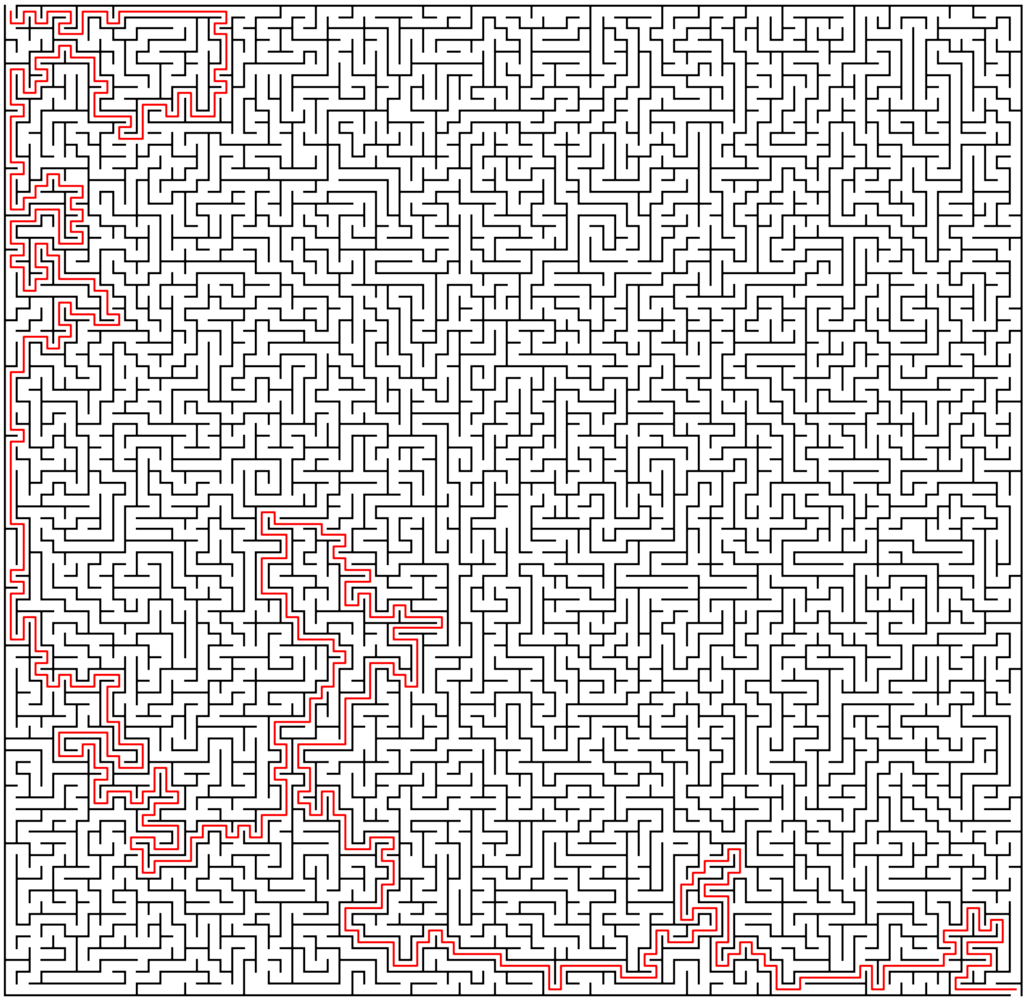


Figure 22: Iterative deepening A* search using Euclidean distance for the first 85x85 maze.

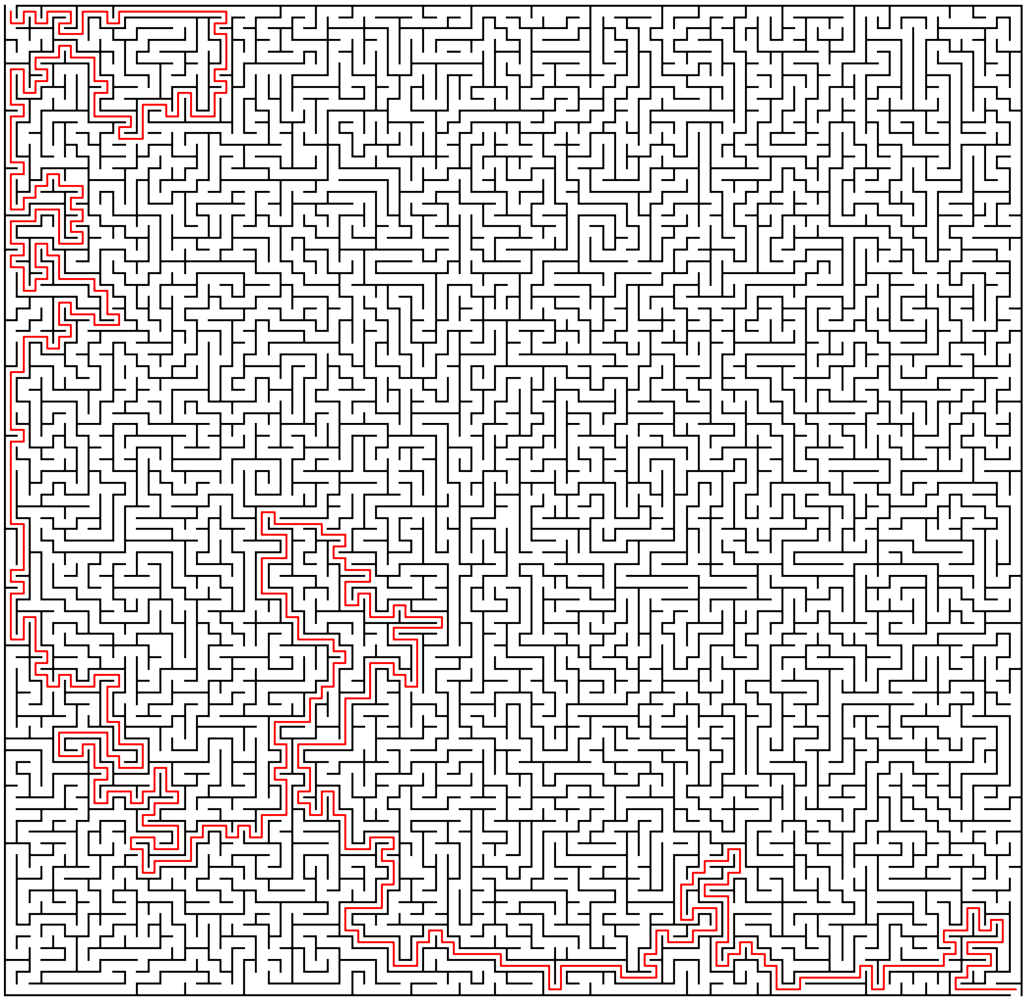


Figure 23: Iterative deepening A* search using Manhattan distance for the first 85x85 maze.

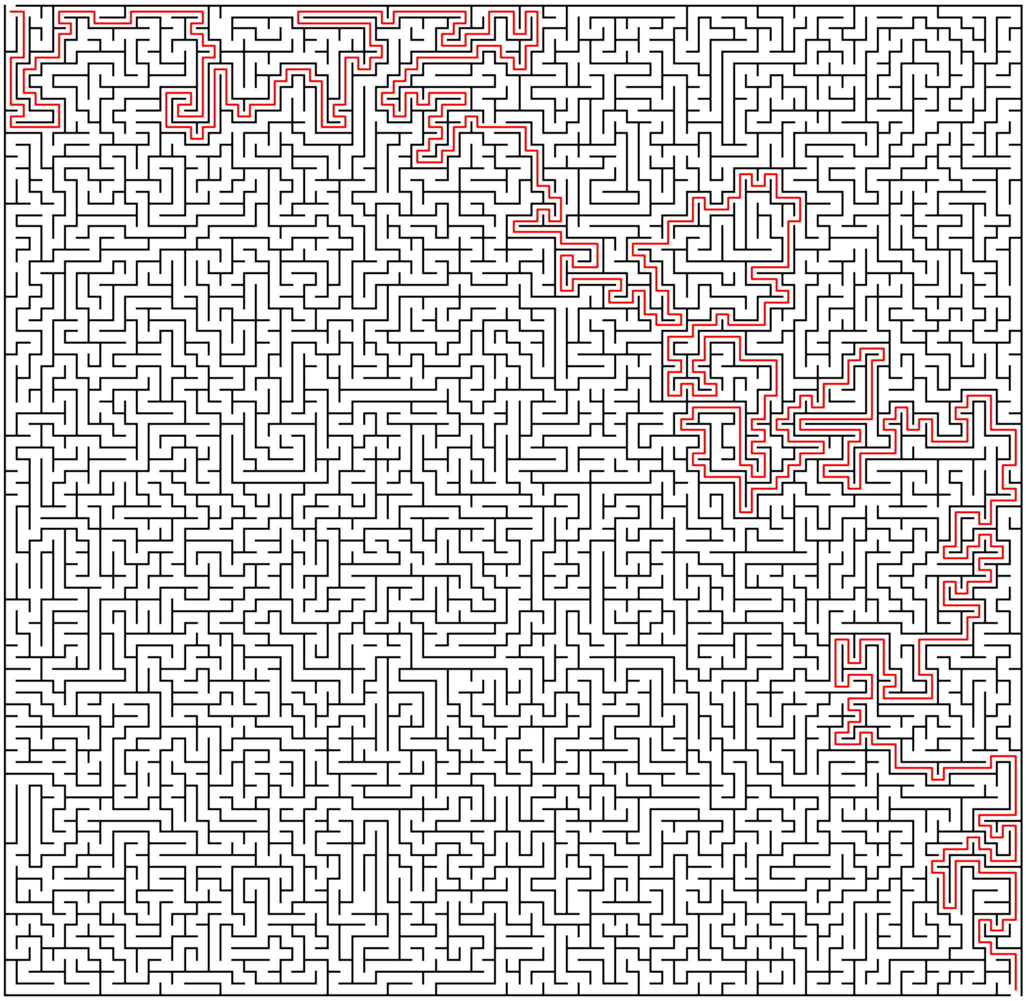


Figure 24: Iterative deepening A* search using Euclidean distance for the second 85x85 maze

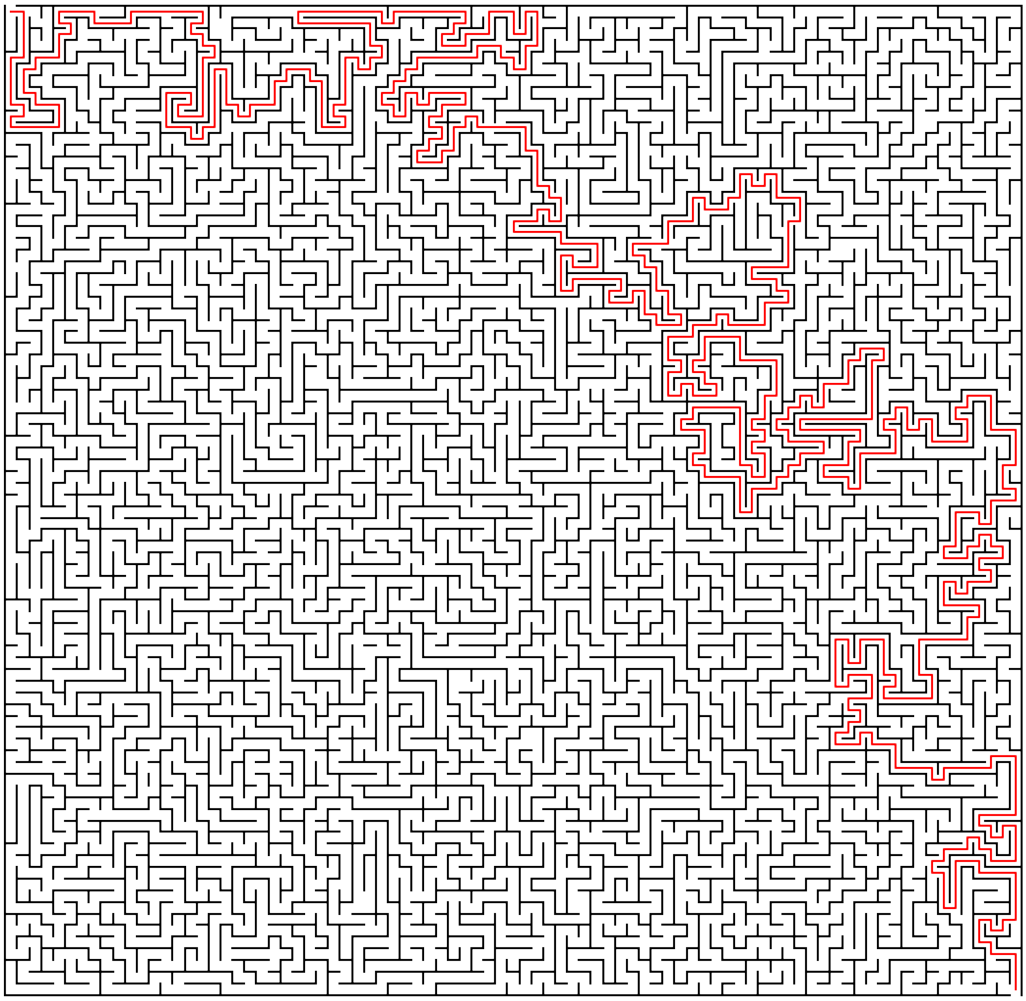


Figure 25: Iterative deepening A* search using Manhattan distance for the second 85x85 maze.

References

[1] Peter Norvig. Google colaboratory. URL https://colab.research.google.com/github/norvig/pytudes/blob/main/ipynb/Maze.ipynb#scrollTo=LARC9yrp9UY5.

[2] Atikur Rahman Chitholian. chitholian/ai-search-algorithms: This is an educational repository containing implementation of some search algorithms in artificial intelligence. URL https://github.com/chitholian/AI-Search-Algorithms.

[3] URL https://xlinux.nist.gov/dads/HTML/euclidndstnc.html.

[4] URL https://xlinux.nist.gov/dads/HTML/manhattanDistance.html.

[5] Maximum manhattan distance between a distinct pair from n coordinates, Jan 2021. URL https://www.geeksforgeeks.org/maximum-manhattan-distance-between-a-distinct-pair-from-n-coordinates/.

[6] Claas Meiners. Iterative deepening a star in all languages. URL https://www.algorithms-and-technologies.com/iterative_deepening_a_star.

[7] Richard E. Korf, Michael Reid, and Stefan Edelkamp. Time complexity of iterative-deepening-a. *Artificial Intelligence*, 129(1-2):199–218, 2001. doi: 10.1016/s0004-3702(01)00094-7.

[8] Ida-star(ida*) algorithm in general, Apr 2016. URL https://algorithmsinsight.wordpress.com/graph-theory-2/ida-star-algorithm-in-general/.