Artificial Intelligence Laboratory 2: A* Search Algorithm
DT8042 HT22, Halmstad University

Santhikrishna Reji and Elliot Giron-Michel, sanrej23@student.hh.se and ellgir23@student.hh.se,
Masters in Information Technology,Exchange students

November 2023

Introduction

The objective of this lab is to ...

Q1: What types of search algorithms you have learned from the lecture? Please briefly introduce them here.
First, we discussed the Random Search algorithm, which explores the search space randomly without adhering to a specific pattern. Then we used breadth search and depth search. BFS explores a node's neighbors before moving on to next nodes, while DFS explores as deep as possible before moving back. We also covered the Greedy Search algorithm, which selects the path that looks most promising based on specific heuristics. Finally, we looked at informed search algorithms such as A* with Euclidean and Manhattan distances. A* is an algorithm that combines the advantages of a greedy and uniform cost search using an evaluation function that takes into account the current cost and an estimate of the remaining cost to reach the solution. These different algorithms offer distinct approaches to solving search problems, each with their own advantages and disadvantages depending on the specific context of the task at hand.

Task 1: Path Planning

For task 1...

Q2: Implement and apply uninformed search algorithms, e.g. random search, BFS, and DFS. Do they find the optimal path on both maps? How many nodes were expanded? Please include a few example plots of the grid maps with the evaluation values of each cell and the path found.
The random did not find an optimized path and browses a lot of nodes.
The BFS finds an optimized path and browses a lot of nodes.
The DFS did not find an optimized path and browses few nodes

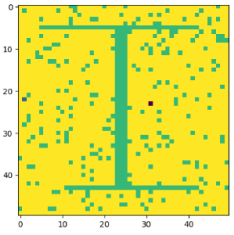

Figure 1 : fig:original map 50x50

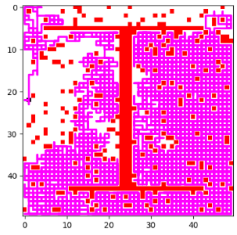

Figure 2 : random algo
24341 nodes explored

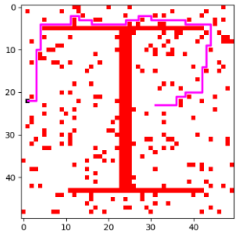

Figure 3 : BFS algo
51632 nodes explored

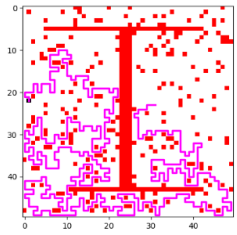

Figure 4 : DFS algo
978 nodes explored

Q3: Implement and apply greedy and A* search algorithm. What heuristic function have you implemented? Do both algorithms find the optimal path on both maps? Please provide examples.

The greedy did not find an optimized path and browses few nodes.
The A* euclidian and manhattan find an optimized path and browses a lot of nodes.
My A* heuristic almost finds an optimized path and browses few nodes. Instead of going from point A to point B, it creates intermediate destinations. In this case it looks at the position of the final destination and creates 2 intermediate destinations. On the y axis of departure and arrival, as close as possible to the exterior wall. A more optimized version would place these 2 points on the corners of the H

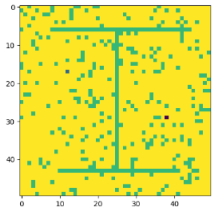

Figure 5 : fig:original map 50x50

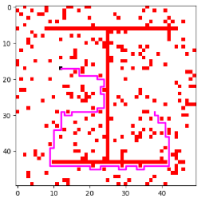

Figure 6 : greedy algo
604 nodes explored

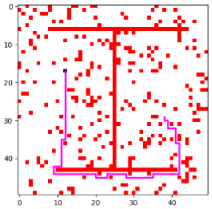

Figure 7 : A* euclidian algo
1363 nodes explored

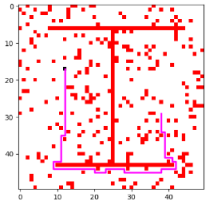

Figure 8 : A* manhattan algo
1277 nodes explored

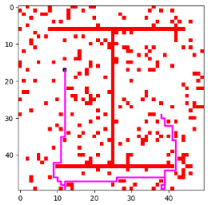

Figure 9 : A* heuristic algo
502 nodes explored

Q4: Compare different search algorithms. For each search algorithm, run the experiment multiple times (e.g. 20) and fill in the following table.

Search algorithms	Number of nodes expanded	Time consumed	Optimum count
Random	= 51750.0 ± 42250.0	= 0.11 ± 0.09	= 51750.0 ± 42250.0
BFS	= 31000.0 ± 14000.0	= 0.1 ± 0.04	= 75.0 ± 19.0
DFS	= 1177.0 ± 686.0	= 0.1 ± 0.08	= 574.5 ± 273.5
Greedy	= 477.0 ± 232.0	= 0.3125 ± 0.2875	= 113.5 ± 43.5
A* Heuristic	= 532.5 ± 269.5	= 0.102 ± 0.098	= 98.0 ± 16.0

Table 1: Performance comparison (Path Planning) (map 50x50)

Task 2: Poker Bidding

Q5: Implement and apply BFS and DFS algorithms. What have you observed? Do all of them found a solution? How many nodes were expanded?
For both algorithms we took into account the hands of the 2 players for the next 4 games, the simple version runs through all the nodes and the optimized version removes unnecessary branches
The simple BFS finds an optimized sequence of actions and browses a lot of nodes, between 460 and 1052.
The optimized BFS finds an optimized sequence of actions and browses few nodes, between 49 and 206.
The simple DFS finds an optimized sequence of actions and browses far too much nodes, between 2500 and 10470.
The optimized DFS finds an optimized sequence of actions and browses a lot of nodes, between 590 and 2200.

Q7: Briefly describe the heuristic function implemented. How many nodes were expanded with the proposed heuristic function? Is the solution optimal?
Each hand is played independently one after the other, only the state of the simulated field is shared. For a hand the best score of the nodes visited and its depth are saved. If a node is encountered with a score lower then the best score divided by its best depth, these branches will not be explored.

Q8: Compare different search algorithms.

Search algorithms	Number of nodes expanded	Number of hands	Number of biddings
BFS optimized	$= 756.0 \pm 296.0$	$= 3 \pm 1$	4
DFS optimized	$= 127.5 \pm 78.5$	$= 3 \pm 1$	4
BFS simple	$= 6485.0 \pm 3985.0$	$= 3 \pm 1$	4
DFS simple	$= 1395.0 \pm 805.0$	$= 3 \pm 1$	4

Table 2: Performance comparison (Poker Bidding)

Conclusion

Guided algorithms show better performance on particular cases. Regarding the bfs and dfs algo. The bfs will on average go through more nodes but will have a better result. The dfs will go through fewer nodes but has a greater risk of not having a perfect result.