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

Fabrice Bodson (fabbod23student.hh.se) ; Jaffar Kamil (jafkam20student.hh.se)

November 2023

Introduction

Q1: What types of search algorithms you have learned from the lecture? Please briefly introduce them here.

The objective of this lab is to make us understand how to use and how work search algorithms (random, exhaustive, greedy search, and A* with customized heuristics). To do so, we had 2 tasks :

- We had to find the shortest path from the agent's current position to the goal position.
- Secondly, we had to fin a sequence of bids for our agent to win more than 100 coins in less than 4 hands in a poker game.

For the first task, we implemented the Uninformed Search Algorithms (Random, BFS, DFS) and the Informed Search Algorithms (GBFS and A* with Euclidean, Manhattan, and personalized distances) for grids of randomly generated and fixed obstacles.

For the second task, we implemented BFS and DFS algorithms.

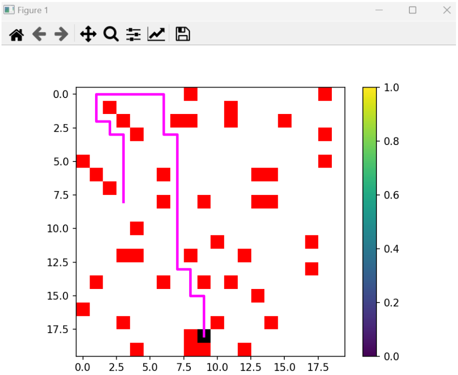
Task 1: Path Planning

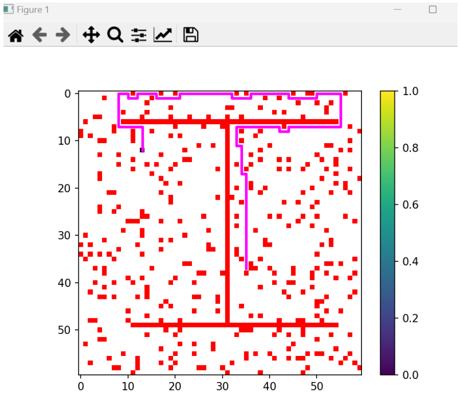
,

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.

.

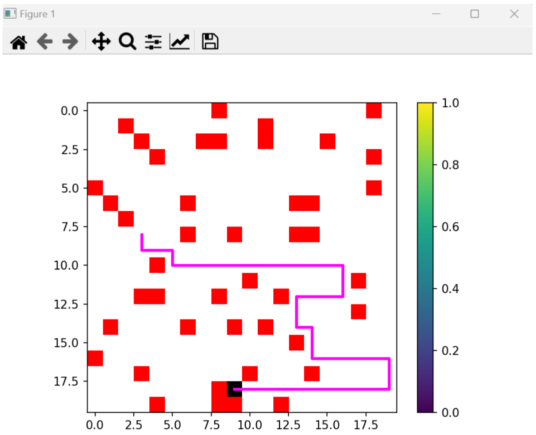
Breadth-First Search (BFS)

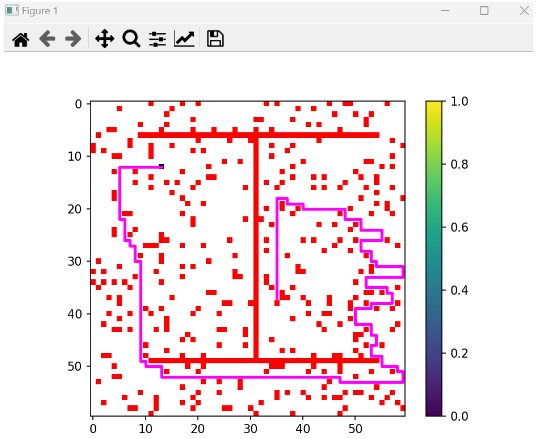




Discussion:
Did BFS find the optimal path?
How many nodes were expanded?
In this scenario, BFS did not find the optimal path. However, the effectiveness of BFS in finding the optimal path can vary depending on the specific case. Sometimes BFS does find the optimal path, and other times it does not. For this particular instance, the path cost was 37, with a total of 153 nodes visited.
As seen in the second picture, which represents the map with the predetermined obstacle, BFS does not manage to find the best way, but the path is still very optimal and good compared to the path taken by DFS. The start Start: (12, 13) – Goal: (37, 35) Path found with cost of 136, nodes visited: 1975. We visited more nodes compared to the dfs and found a better way.

Depth-First Search (DFS)





Discussion:
Did DFS find the optimal path?
How many nodes were expanded?

For DFS, in this scenario, the algorithm did not find the optimal path. This is consistent with the general behavior of DFS, as it often does not find the optimal path in most cases. In the first case, the algorithm started at (18, 9) and aimed for the goal at (8, 3), finding a path with a cost of 42 and visiting 134 nodes in the process. As seen in the second picture, which showcases the second map, DFS did find a path; however, the path is very costly and isn't optimal, with a path cost of 219 and 1406 nodes visited. The start was Start: (12, 13) – Goal: (37, 35).

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.

Implementation of Greedy and A* Search Algorithms

Heuristic Function Used

Manhattan Distance: This heuristic was implemented in both the Greedy Best-First Search and A* Search algorithms.

Algorithm Analysis

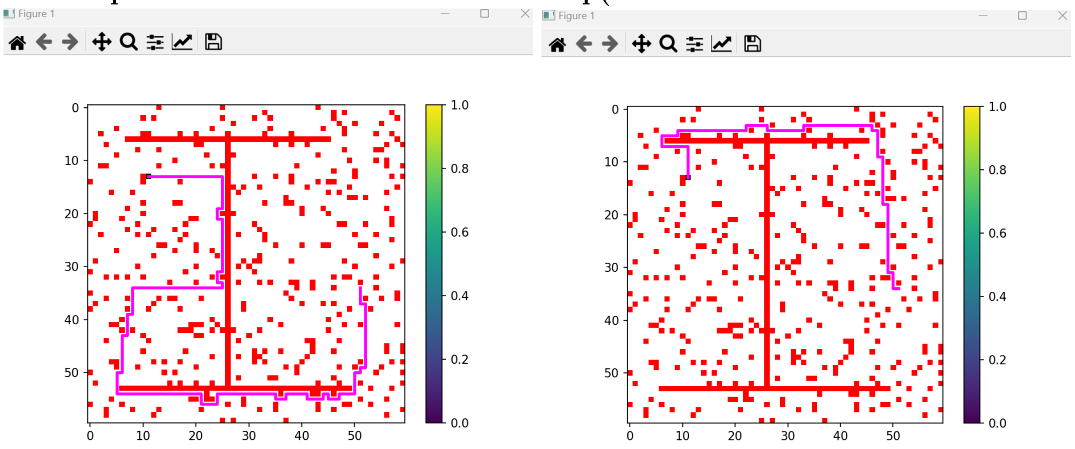
Greedy Best-First Search (GBFS):

- Utilizes the Manhattan distance to estimate the cost from the current node to the goal.
- Does not always guarantee the optimal path due to its focus on the goal rather than path cost. It is better than the BFS in both maps. However when compared to the Astar they tend to have similar performance for the first map. But when comparing them in the map with a predetermined H-obstacle the A-star has a way better optimal pathfinding. A-star regularly manages to find the best way compared to GBFS.

A* Search Algorithm:

- Combines the Manhattan distance with the actual traveled cost.
- More likely to find the optimal path by balancing traveled distance and estimated distance to the goal. The A-star algorithms always manage to fins the best path regarding the map. A* Search algorithm typically outperforms Greedy Best-First Search as it balances the cost incurred so far with the estimated cost to the goal, leading to more efficient and optimal pathfinding. In contrast, Greedy Best-First Search focuses solely on the estimated distance to the goal, often overlooking shorter or more cost-effective paths.

GBFS compared to ASTAR on H-obstacle map(Same Start and Goal



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

Search Algorithm	Number of Nodes Expanded	Average Cost	Average Path Length
DFS	1273.29	229.0	230.0
BFS	1551.25	139.05	139.05
GBFS	1434.3	120.95	121.95
A*	1379.0	100.25	101.25

Table 1: Performance comparison (Path Planning) after 20 simulations

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?

They both find a solution but not at each run of the program. It may take 5 games before getting 100 coins from the opponent.
For BFS, here is an example of an output when it finds a way :

```
BFS done
Total of hands : 1 ; Number of bidding: 16
Agent stack: 500 ; Opponent stack: 290
```

And for DFS :

```
DFS done
Total of hands : 1 ; Number of bidding: 12
Agent stack: 505 ; Opponent stack: 285
Expanded nodes: 209
```

We can see that 209 nodes were expanded in 1 hand.

Q7: Briefly describe the heuristic function implemented. How many nodes were expanded with the proposed heuristic function? Is the solution optimal?

We did not do the extra credits.

Q8: Compare different search algorithms.

See table next page.

Conclusion

This lab allowed us to get a really good understanding of the Search algorithms in different contexts. By applying them, we had the opportunity to manipulate them so they can fit our code and logic.

Search algorithms	Number of nodes expanded	Number of hands	Number of biddings
BFS	/	1 if found, otherwise 4	16
DFS	209	1 if found, otherwise 4	12

Table 2: Performance comparison (Poker Bidding)