|  |  |  |
| --- | --- | --- |
| Fall 2023 | CS5368 Intelligent Systems | Project 1 Report |

|  |  |
| --- | --- |
| First Name | KanvaKoushik |
| Last Name | Gopavarapu |
| Student ID | R11849414 |

Please answer the following questions and submit them through Blackboard. Be sure to submit it to the Project 0 report submission. DO NOT write the report by hand and submit a scanned version. Just write the answers in a Word document and submit it. Both Word and PDF submissions are accepted.

**Basic Questions:**

1. **What is your implementation strategy for DFS? Explain**.
   1. With the code attached below, DFS starts with the initial state of the problem. With the help of successors and the function dfs\_helper, we implement the recursive function which also checks the visited states or not. The main strategy for the implementation is to recursively explore the search tree while prioritizing deeper exploration before backtracking.

A computer screen shot of a program code

Description automatically generated

*Fig.1. DFS Implementation*

1. **What is your implementation strategy for BFS? Explain.**
   1. The BFS implementation is to explore states level by level, starting from the initial place and moving further out. It also ensures that the shallowest nodes are explored first by using a queue to maintain the order of exploration. When the goal is found, the algorithm returns the sequence of actions leading to it.

A computer screen shot of a program

Description automatically generated

*Fig.2. BFS Implementation*

1. **What is your implementation strategy for UFS? Explain.**
   1. The UCS algorithm is for solving a generic search problem. The implementation strategy is to explore the states in order of cost increase. It mainly keeps track of the lower-cost path to each state and prioritizes the states with the lowest cost to be explored first. When the goal is found, the algorithm returns the sequence of actions leading to it.

A computer screen shot of a program code

Description automatically generated

*Fig.3. UFS Implementation*

1. **What is your implementation strategy for A\*? Explain.**
   1. A\* strategy is implemented to explore the states in order of increasing combined cost and heuristic estimate, prioritizing states that are likely to lead to the goal more efficiently. It makes a queue to maintain the order of priority. When the goal is reached, it returns the sequence of actions.

A screen shot of a computer program

Description automatically generated

*Fig.4. A\* search Implementation*

1. **Explain your implementation strategy for the Corners Problem. What is your state representation?**
   1. The below code defines a search problem where Pac-Man needs to visit all four corners of a layout while avoiding walls. The state space includes Pac-Man’s position and the status of the corners. The main goal is to visit all the corners, and the function successor generates the states. This uses A\* search to find all the optimal paths to visit all the corners.  
      The state representation is a tuple ` (current\_position, corners\_status) ` where the x-coordinate gives us the position of the Pac-Man and the Y-coordinate gives us the Boolean position.

A screen shot of a computer program

Description automatically generated

*Fig.5.1. CornerProblem Implementation*

A computer screen shot of text

Description automatically generated

*Fig.5.2. CornerProblem Implementation*

1. **What are your heuristics for the Corners Problem? How did you implement it?**
   1. The Heuristics used are Manhattan distance to the nearest unvisited corner along with admissible and consistent. These heuristic guides search algorithms to explore states that are likely to lead to efficient pathfinding in the problem. It prioritizes search algorithms like A\* to explore the states that are closer to undiscovered corners.

A screenshot of a computer program

Description automatically generated

*Fig.6. Heuristic for CornerProblem Implementation*

**Advanced Questions:**

1. **Based on your implementation of DFS, what happens if you fail to push all successors to the fringe (see Get Successors**
   1. If we fail to push all the successors to the fringe, it can lead to incomplete and inaccurate results which in turn could overflow the stack due to excess memory usage. So, failing to push all the successors violates the DFS and can lead to untrustworthy search results.
2. **Does BFS give fewer actions to reach the goal compared to DFS? Explain.**
   1. Yes, BFS is more likely to find a shorter path to the goal compared to DFS as it explores all paths at each level while giving us the shortest path possible with uniform costs. It also depends on the problem we are facing or trying to solve and within which state of search it is encountered.
3. **Many paths: You have expanded some paths in all search algorithms, which of these algorithms has the least expansion, and why?**
   1. In my opinion, BFS has had optimal search results compared to other algorithms. It explores in a breadth-first manner moving from the current depth before exploring the next deeper levels.
4. **Explain how the implementation of A\* differs from UCS.** 
   1. A\* uses heuristics while prioritizing the nodes whereas UCS relies on the cost of the path and explores the nodes uniformly.
5. **Node expansion: How many nodes were expanded in Question 7?**
   1. In the food heuristic problem, 7137 nodes were expanded from implementation.

A screenshot of a computer

Description automatically generated

*Fig.7. Food Heuristic expanded nodes.*