Snake Game with Search Algorithms

Introduction

This document provides an overview of our implementation of the Snake game with three different search algorithms: A*, Greedy Best First Search, and Breadth-First Search. The goal of this assignment is to compare the performance of these algorithms in solving the Snake puzzle.

Implementation Details

Files

- **`main.py`**: Contains the main game loop and integration of search algorithms.
- **`AgentSnake.py`**: Defines the **`AgentSnake`** class with methods for each search algorithm.
- **`AStar.py`**, **`GreedyBestFirst.py`**, **`BreadthFirst.py`**: Implementations of the A*, Greedy Best First Search, and Breadth-First Search algorithms, respectively.
- **`State.py`**: Defines the **`SnakeState`** class representing the game state.
- **`View.py`**: Implements the **`SnakeViewer`** class for displaying the game.

Search Algorithms

1. **A***: A* algorithm is an informed search algorithm that uses a heuristic to find the path to the goal state with the lowest cost.
2. **Greedy Best First Search**: Greedy Best First Search is also an informed search algorithm that selects the path which appears to be the best at that moment.
3. **Breadth-First Search**: Breadth-First Search is an uninformed search algorithm that explores all the nodes at the present depth before moving on to the nodes at the next depth level.

Running the Game

1. Upload all Python files (**`main.py`**, **`AgentSnake.py`**, **`AStar.py`**, **`GreedyBestFirst.py`**, **`BreadthFirst.py`**, **`State.py`**, **`View.py`**) to Vs Code.
2. Run the **`main.py`** file to start the game with the specified search algorithm.

Experimental Results

We ran the game with different maps for 5 minutes each and observed the scores and performance of each search algorithm. Here are the results:

- A* Algorithm:
  - Map 1(Maze1): Score = 830
  - Map 2(Maze2): Score = 1300
  - Map 3(Maze): Score = 1530
  - Map 4(Maze0): Score = 1600
- Greedy Best First Search:
  - Map 1(Maze1): Score = 860
  - Map 2(Maze2): Score = 1170
  - Map 3(Maze): Score = 1400
  - Map 4(Maze0): Score = 1420
- Breadth-First Search:
  - Map 1(Maze1): Score = 900
  - Map 2(Maze2): Score = 1320
  - Map 3(Maze): Score = 1610
  - Map 4(Maze0): Score = 1560

1. **A* Algorithm**: A* is a best-first search algorithm that uses a heuristic to guide its search. It is generally considered the best-performing algorithm in terms of finding the optimal path to the goal. In our experiments, A* achieved scores of 830, 1300, 1530, and 1600 on the four maps, respectively. These scores indicate that A* performed reasonably well, but not exceptionally better than the other algorithms.
2. **Greedy Best First Search:** Greedy Best First Search is an algorithm that always expands the node that is closest to the goal according to some heuristic. It is not guaranteed to find the optimal path, but it can be faster than A* because it does not consider the full cost of reaching a node. In our results, Greedy Best First Search achieved scores of 860, 1170, 1400, and 1420 on the four maps. These scores suggest that Greedy Best First Search did not perform as well as A* or Breadth-First Search.
3. **Breadth-First Search:** Breadth-First Search explores all nodes at the present depth before moving on to nodes at the next depth level. It is complete and optimal for finding the shortest path in unweighted graphs. In our results, Breadth-First Search achieved scores of 900, 1320, 1610, and 1560 on the four maps. These scores indicate

that Breadth-First Search performed the best among the three algorithms on the given maps.

4. Conclusion

Based on our experiments, we observed that the Breadth-First Search algorithm performed the best in terms of score to reach the goal state. It consistently achieved higher scores on all maps compared to the other two algorithms. A* algorithm also performed well but did not consistently outperform Breadth-First Search. Greedy Best First Search, while sometimes producing good results, often resulted in suboptimal paths and lower scores. Overall, the choice of search algorithm depends on the specific requirements of the game and the desired balance between optimality and computational cost. In this context, Breadth-First Search appears to be the most reliable choice for the snake game.