Evaluation Report

Snake Puzzle

Group Members:

Muhammad Ahmad(21L-5617)

Nimra Amer(21L-5609)

Section: BDS-6A

We implemented three search algorithms:

- BFS (21L-5617)
- Greedy Best First Search (21L-5609)
- A* (21L-5617)

Implementation Details:

A*:

The implementation details are as follows:

- **Agent Class:**

The Agent class defines a basic structure for agents. It includes a method SearchSolution which currently returns an empty list.

- **AgentSnake_AStar:**

This class inherits from the Agent class and overrides the SearchSolution method.

It utilizes the A* search algorithm to find the shortest path from the snake's head to the food in a maze.

1. The astar_search function performs the A* search using a priority queue.
2. The heuristic function calculates the Manhattan distance heuristic between the current node and the goal.
3. The neighbors function finds valid neighboring positions for the snake to move.

4. The convert_path_to_plan function converts the path obtained from the search into a plan (sequence of actions) for the snake to reach the food.

- **Main Functionality:**

The SearchSolution method in AgentSnake_AStar calculates the path using A* search and converts it into a plan for the snake to reach the food.

- **Additional Notes:**
  1. The code assumes a grid-based maze environment where the snake can move up, down, left, or right.
  2. It considers obstacles represented by -1 in the maze.
  3. The directions are represented as follows: 0 (up), 3 (right), 6 (down), and 9 (left).
  4. The A* algorithm uses the Manhattan distance heuristic to estimate the distance from the current node to the goal.
  5. The heuristic function takes into account the maze object as an additional parameter.

Greedy Best First Search:

The implementation details are as follows:

- **Agent Class:**

The Agent class defines a basic structure for agents. It includes a method SearchSolution which currently returns an empty list.

- **AgentSnake_GBFS:**

This class inherits from the Agent class and overrides the SearchSolution method.

It uses a greedy best-first search algorithm to find the shortest path from the snake's head to the food in a maze.

1. The heuristic function calculates the Manhattan distance between the current node and the goal (food position).
2. The neighbor's function finds valid neighboring positions for the snake to move.
3. The greedy_best_first_search function performs the search using a priority queue.
4. The convert_path_to_plan function converts the path obtained from the search into a sequence of actions (directions) for the snake to follow.

- **Main Functionality:**

The SearchSolution method in AgentSnake_GBF calculates the path using the greedy best-first search algorithm and converts it into a plan for the snake to reach the food.

- **Additional Notes:**

The code assumes a grid-based maze environment where the snake can move up, down, left, or right.

It considers obstacles represented by -1 in the maze.

The directions are represented as follows: 0 (up), 3 (right), 6 (down), and 9 (left).

BFS:

The implementation details are as follows:

- **Agent Class:**

The Agent class defines a basic structure for agents. It includes a method SearchSolution which currently returns an empty list.

- **BFS_Algorithm:**

This class inherits from the Agent class and overrides the SearchSolution method.

It utilizes the Breadth-First Search algorithm to find the shortest path from the snake's head to the food in a maze.

1. The bfs_search function performs the BFS using a queue.
2. The neighbor's function finds valid neighboring positions for the snake to move.
3. The convert_path_to_plan function converts the path obtained from the search into a plan (sequence of actions) for the snake to reach the food.

- **Main Functionality:**

The SearchSolution method in BFS_ALgorithm calculates the path using BFS and converts it into a plan for the snake to reach the food.

- **Additional Notes:**

The code assumes a grid-based maze environment where the snake can move up, down, left, or right.

It considers obstacles represented by -1 in the maze.

The directions are represented as follows: 0 (up), 3 (right), 6 (down), and 9 (left).

Results:

We ran the code for all the algorithms for approximately 5 minutes and got the following results:

- Using A*, the snake scored 1600 points
- Using Greedy Best First Search, the snake scored 1670 points
- Using BFS, the snake scored 1600 points

It is concluded that Greedy Best First Search performed the best and caused the snake to score the best in comparison to both the other algorithms at the same time.
