INFO-F-311: Artificial Intelligence - Project 1: Search

Your Name

1 Preamble

This report outlines the implementation of artificial intelligence techniques based on graph search techniques like Breadth-First Search (BFS), Depth-First Search (DFS), and the A* algorithm.

The primary languages and tools used are Python 3.10 and Poetry.

1.1 The Problems

1. **SimpleSearchProblem**: The goal is to reach the exit of the environment with multiple agents.
2. **CornerSearchProblem**: The agents must pass through all four corners of the environment before reaching an exit.
3. **GemSearchProblem**: The agents need to collect all gems in the environment before reaching an exit.

2 SimpleSearchProblem

2.1 Problem Modeling

This section describes the is_goal_state and get_successors methods.

2.2 Breadth-First Search

The Breadth-First Search algorithm is implemented in `search.py` via the `bfs` function.

2.3 Depth-First Search

The Depth-First Search algorithm is implemented in `search.py` via the `dfs` function.

2.4 A* Search

The A* algorithm is implemented with Manhattan distance as the heuristic.

3 CornerSearchProblem

3.1 Problem Modeling

The problem aims to pass through all four corners of the grid.

3.2 Heuristic

A consistent heuristic more efficient than Manhattan distance is developed.

4 GemSearchProblem

4.1 Problem Modeling

The problem aims to collect all gems in the environment.

4.2 Heuristic

A consistent heuristic more efficient than Manhattan distance is developed.

5 Results

5.1 Path Size Comparison

Comparison of the path sizes found by BFS, DFS, and A* on level 3.

5.2 Node Expansion Comparison

Comparison of the number of nodes expanded in BFS, DSF, and A* during the solution of level 3.

$\in$

6 Understanding the Code

The codebase includes utility functions for calculating distances, an abstract SearchProblem class, and concrete problem classes like SimpleSearchProblem, CornerSearchProblem, and GemSearchProblem.

6.1 Key Methods

- `is_goal_state()`: Determines if a state is the goal state.
- `get_successors()`: Generates possible successor states from the current state.
- `heuristic()`: Calculates the heuristic value for the A* algorithm.

7 Heuristics Development

heuristic for CornerSearchProblem and GemSearchProblem.

8 Optimizations

- Use a priority queue for the A* algorithm.
- Cache heuristic values to avoid redundant calculations.

9 Tool Usage

Explanation of the use of tools like ChatGPT in the project.