CPCS-331 Artificial Intelligence (I)

AI Group Project: Solving Maze Problems
Formal Analysis

Made by:

Group CS1 AE

Mahmued Alardawi - 2135209

Ahmad Aljedaani - 2136071

Abdullah Emad Almashharawi – 2136141

Abdullah Abed Alharbi - 2135999

Introduction

The problem consists of a randomly generated maze with a starting point and an end point, and we need to find the path from the initial state to the goal state. We will be utilizing two agents to solve this problem. The first agent will solve the maze using Depth-First search approach (DFS), while the second agent will use Breadth-first search approach (BFS).

For the sake of simplicity, we will use a predetermined maze pattern as an assumption. Regardless, the agents can solve any randomly generated maze that has a real solution.

We will use Java to program both the maze and the agents. As for the maze, we will use an adjacency matrix to represent the graph. We will use GUI to display the maze, with green representing starting state, and red representing goal state. While white for the unmarked states, and black for the illegal states (walls). The agents will start at the initial state, and mark their path with green, till they reach the goal state.

Environment Type

Since the agents can only move one step at a time, the environment type would be partially observable. Not only that, but the maze is also randomly generated. Therefore, it is stochastic. However, the maze does not change while the agent is finding the path, so it is static. And there are a limited number of possible moves, so it is also discrete. And finally, since the agent moves one step at a time towards the goal state, it will not be episodic, but sequential.

To conclude, the maze is a partially observable, stochastic, static, discrete, sequential environment.

PEAS

Performance Measure:

How short the path found by the agent, how fast can the agent find a path.

Environment:

Unmarked states, illegal states (walls), initial state, goal state.

Actuators:

Mark a state, move from one of the four vertices surrounding it.

Sensors:

Looking at the four surrounding vertices and each of their states.