Maze Solving Agents

Enrique Calderón
Ramírez del Prado
Tepal Hansel
Luis Ugartechea
Universidad Nacional Autónoma de México
Facultad de Ingeniería
September 03, 2025

Objectives

- Implement two agent approaches: one reactive (simple reflex agent) and one goal-based (goal-based agent).
- Compare their results in terms of steps and ability to reach the goal.

Problem Statement and Objectives

Goal: Solve a maze represented as a matrix.

Maze definition:

- `1` = wall
- `0` = free path
- `S` = start
- `M` = goal

Simple Reflex Agent Implementation

- Our `simple_reflex_agent` function starts at the initial position (`S`) and tries to reach the goal (`M`).
- At each step, it checks the four possible directions (right, down, left, up) in order, and moves to the first valid, unvisited cell (`0` or `M`).
- The agent keeps track of its path and marks visited cells with a dot (`.`), except for the goal.
- If no valid moves are available, the agent stops and reports that it is stuck.
- This approach is fast and simple, but may fail in mazes where the path is not straightforward, as it does not backtrack or plan ahead.

Simple reflex agent example

Successful Case

```
Initial Maze:    Final Result:
1 1 1 1 1 1      1 1 1 1 1 1
1 S 0 0 M 1      1 . . . M 1
1 1 1 0 1 1      1 1 1 0 1 1
1 1 1 0 1 1      1 1 1 0 1 1
1 1 1 1 1 1      1 1 1 1 1 1
```

Failed Case (Gets Stuck)

```
Initial Maze:      Agent gets stuck:
1 1 1 1 1 1 1      1 1 1 1 1 1 1
1 S 0 0 0 0 1      1 S . . . . 1
1 1 0 1 1 1 1      1 1 0 1 1 1 1
1 1 M 1 1 1 1      1 1 M 1 1 1 1
1 1 1 1 1 1 1      1 1 1 1 1 1 1
```

Goal-Based Agent (BFS)

Implementation

- The `goal_based_agent` function uses Breadth-First Search (BFS) to find the shortest path from the start (`S`) to the goal (`M`).
- It systematically explores all possible paths by expanding nodes level by level, using a queue to keep track of paths to explore.
- The agent remembers visited positions to avoid revisiting them and to ensure it finds the shortest route.
- When the goal is reached, the path is marked in the maze, and then returns the path taken for the solution, also returns the number of steps.
- This approach guarantees that if a solution exists, it will be found and will be optimal in terms of the number of steps.

Goal-Based Agent (BFS)

Example Successful Case

```
1 1 1 1 1 1 1 1
1 S 0 0 0 0 0 1
1 0 1 1 1 1 0 1
1 0 1 1 1 1 0 1
1 0 1 1 1 1 0 1
1 0 1 1 1 1 1 1
1 0 0 0 0 0 M 1
1 1 1 1 1 1 1 1
```

Goal-Based Agent (BFS)

Example Failed Case

```
1 1 1 1 1 1 1 1
1 S 0 0 0 0 0 1
1 0 1 1 1 1 0 1
1 0 1 1 1 1 0 1
1 0 1 1 1 1 0 1
1 1 1 1 1 1 1 1
1 0 0 0 0 0 M 1
1 1 1 1 1 1 1 1
```

Goal-Based Agent (BFS)

Example Special Case

```
1 1 1 1 1 1 1 1
1 S 0 0 0 0 0 1
1 0 1 1 1 1 0 1
1 0 1 1 1 1 0 1
1 0 1 1 1 1 0 1
1 0 1 1 1 1 0 1
1 0 0 0 0 0 M 1
1 1 1 1 1 1 1 1
```

Results

Initial Maze

```
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 S 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 1
1 M 0 0 0 0 0 0 0 0 0 0 0 0 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

Results

With Simple Reflex Agent

```
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 S . . . . . . . . . . . . 1
1 0 0 0 0 0 0 0 0 0 0 0 0 . 1
1 0 0 0 0 0 0 0 0 0 0 0 0 . 1
1 0 0 0 0 0 0 0 0 0 0 0 0 . 1
1 0 0 0 0 0 0 0 0 0 0 0 0 . 1
1 0 0 0 0 0 0 0 0 0 0 0 0 . 1
1 0 0 0 0 0 0 0 0 0 0 0 0 . 1
1 0 0 0 0 0 0 0 0 0 0 0 0 . 1
1 0 0 0 0 0 0 0 0 0 0 0 0 . 1
1 0 0 0 0 0 0 0 0 0 0 0 0 . 1
1 0 0 0 0 0 0 0 0 0 0 0 0 . 1
1 M . . . . . . . . . . . . 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

Results

With Goal-Based Agent (BFS)

```
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 S 0 0 0 0 0 0 0 0 0 0 0 0 1
1 . 0 0 0 0 0 0 0 0 0 0 0 0 1
1 . 0 0 0 0 0 0 0 0 0 0 0 0 1
1 . 0 0 0 0 0 0 0 0 0 0 0 0 1
1 . 0 0 0 0 0 0 0 0 0 0 0 0 1
1 . 0 0 0 0 0 0 0 0 0 0 0 0 1
1 . 0 0 0 0 0 0 0 0 0 0 0 0 1
1 . 0 0 0 0 0 0 0 0 0 0 0 0 1
1 . 0 0 0 0 0 0 0 0 0 0 0 0 1
1 . 0 0 0 0 0 0 0 0 0 0 0 0 1
1 . 0 0 0 0 0 0 0 0 0 0 0 0 1
1 M 0 0 0 0 0 0 0 0 0 0 0 0 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```