Project 01: Searching

Delivery System

Group 5 - Report

Table of Contents

[1. Group Introduction 3](#_heading=h.30j0zll)

[2. Overview 3](#_heading=h.1fob9te)

[2.1 Purpose, Scope & Brief Description 3](#_heading=h.3znysh7)

[2.2 Construction 3](#_heading=h.2et92p0)

[2.3 Work assignment 4](#_heading=h.tyjcwt)

[2.4 Completion rate for project requirements 4](#_heading=h.1devjnbmjmc5)

[3. Implementation & GUI 5](#_heading=h.3dy6vkm)

[3.1 Level 1 (Basic level) 5](#_heading=h.1t3h5sf)

[3.2 Level 2 (Time limitation) 5](#_heading=h.7p6mxa75ulmr)

[3.3 Level 3 (Fuel limitation) 5](#_heading=h.bntfmavhzj8k)

[3.4 Level 4 (Multiple agents) 6](#_heading=h.zgkpgtthlasu)

[3.5 Graphic User Interface (GUI) 6](#_heading=h.h83iejm4hlxf)

[4. Test Case 7](#_heading=h.2s8eyo1)

[4.1 Test 01 (20x20 grid) 7](#_heading=h.17dp8vu)

[4.2 Test 02 (20x20 grid) 15](#_heading=h.3rdcrjn)

[4.3 Test 03 (10x10 grid) 23](#_heading=h.31vni04j0w90)

[4.4 Test 04 (25x25 grid) 31](#_heading=h.6wdreaz8kncz)

[4.5 Test 05 (30x30 grid) 39](#_heading=h.ih925c3mkw5s)

[5. Demo 46](#_heading=h.dd5s59yzbjb1)

Project 01 - Report

# Group Introduction

* This project was built, developed, and completed by a group of 4 members:
  + 22127273 - Phan Hải Minh - **Team Leader**.
  + 22127084 - Mai Đức Duy.
  + 22127130 - Nguyễn Trần Minh Hoàng.
  + 22127250 - Trần Thành Long.
* Group number: 5.

# Overview

## Purpose, Scope & Brief Description

* **Purpose:** Develop a comprehensive processing program to find the shortest path from the starting point to the endpoint of one or multiple delivery routes according to different levels (constraints):
  + **Level 1:** Find the basic shortest path (presenting DFS, BFS, UCS, GBFS, A\* algorithms) for one delivery route.
  + **Level 2:** Find the shortest and fastest path (with time constraints) for one delivery route.
  + **Level 3:** Find the shortest and fastest path with an additional fuel level constraint (the fuel level must always be higher than 0 throughout the journey) for one delivery route.
  + **Level 4:** Find the shortest and fastest path that meets fuel requirements for multiple delivery routes.
* **Scope & Brief Description:**
  + The search algorithms are implemented on a grid consisting of m rows and n columns representing the map of a city:
    - A cell on the grid can represent either an open area (vehicles can easily move through) or a city building (vehicles cannot move into this cell). Some cells on the grid may indicate starting points (S1, S2,...) and endpoints (G1, G2) of certain delivery routes.
    - Moving between cells on the grid is considered as traveling on a road. At any position on the grid, there are up to four movement options: move up, move down, move left, and move right.
  + The project's task is to find the optimal path that meets the constraints at each required level.

## Construction

* **Programming Language:** Python.
* Including a GUI to visually describe the operation of the search algorithms that are built and presented to address the requirements (learn how to use it in demo video).

## Work assignment

| **STT** | **Description** | **Due Date** | **Responsibility** | **%Complete** |
| --- | --- | --- | --- | --- |
| 1 | Discuss the project requirements and divide the work among the team members. | 08/07/2024 | 22127273 | 100% |
| 2 | Implement Level 1 | 21/07/2024 | 22127273 | 100% |
| 3 | Implement Level 2 | 21/07/2024 | 22127273 | 100% |
| 4 | Implement Level 3 | 24/07/2024 | 22127273  22127250 | 100% |
| 5 | Implement Level 4 | 28/07/2024 |  | 0% |
| 6 | Construct & test GUI | 24/07/2024 | 22127084  22127130 | 100% |
| 6 | Perform preliminary testing of the project to ensure the program operates stably, without conflicts or minor errors. | 27/07/2024 | 22127084  22127130  22127250  22127273 | 100% |
| 7 | Discuss and propose suitable test cases (e.g., special cases, larger input data, etc.). | 27/07/2024 | 22127084  22127130  22127273 | 100% |
| 8 | Prepare the report document and complete it. | 27/07/2024 | 22127250 | 100% |
| 9 | Create a demo video | 28/07/2024 | 22127084  22127273 | 100% |
| 10 | Submission | 28/07/2024 | 22127273 | 100% |

## 

## Completion rate for project requirements

| **STT** | **Requirements** | **Responsibility** | **%Complete** |
| --- | --- | --- | --- |
| 1 | Finish Level 1 successfully | 22127273 | 100% |
| 2 | Finish Level 2 successfully | 22127273 | 100% |
| 3 | Finish Level 3 successfully. | 22127273  22127250 | 100% |
| 4 | Finish Level 4 successfully. |  | 0% |
| 5 | Graphical User Interface (GUI). | 22127084  22127130 | 100% |
| 6 | Generate at least 5 test cases for each level with different attributes. Describe them in the experiment section of your report. Videos to demonstrate each test case. | 22127084  22127130  22127273 | 100% |
| 7 | Report algorithms, and experiment with some reflection or comments. | 22127084  22127250  22127273 | 100% |

# 

# Implementation & GUI

## Level 1 (Basic level)

* **Requirement(s) & Description:** Find the shortest path for one delivery from the starting point (S) to the endpoint (G) based on the information provided from the input file. Sequentially use the following search algorithms to produce results and perform comparisons in terms of accuracy and execution time between the algorithms:
  + Blind search algorithm: Breadth-first search (BFS), Depth-first search (DFS), Uniform-cost search (UCS).
  + Heuristic search algorithm: Greedy Best-First search (GBFS), A\* search.
* **Implementation:**
  + Blind search algorithm:
    - BFS:
    - DFS:
    - UCS:

**→ Conclusion:**

* + Heuristic search algorithm:
    - GBFS:
    - A\*:

**→ Conclusion:**

## Level 2 (Time limitation)

* **Requirement(s) & Description:**
  + Find the shortest and fastest path for one delivery from the starting point (S) to the endpoint (G) based on the information provided from the input file, including the following additional information (via GUI):
    - A time limit indicating the maximum allowed total time for one delivery. When a vehicle moves through a cell, it will consume 1 unit of time.
    - The locations of toll stations in the city, and the additional time cost incurred if passing through these stations.
  + Sequentially use the following search algorithms to produce results and perform comparisons in terms of accuracy and execution time between the algorithms: BFS, DFS, UCS, GBFS and A\*.
* **Implementation:**

## Level 3 (Fuel limitation)

* **Requirement(s) & Description:**
  + Find the shortest and fastest path for one delivery from the starting point (S) to the endpoint (G) based on the information provided from the input file, including the following additional information (via GUI):
    - Each vehicle used for a delivery has a fixed fuel level, and moving through each cell will consume 1 fuel point. A path is considered valid if, at any point along the path, the fuel level is never negative.
    - The locations of fuel stations in the city, where a vehicle passing through these stations will have its fuel replenished to the maximum limit.
  + Sequentially use the following search algorithms to produce results and perform comparisons in terms of accuracy and execution time between the algorithms: BFS, DFS, UCS, GBFS and A\*.
* **Implementation:**

## Level 4 (Multiple agents)

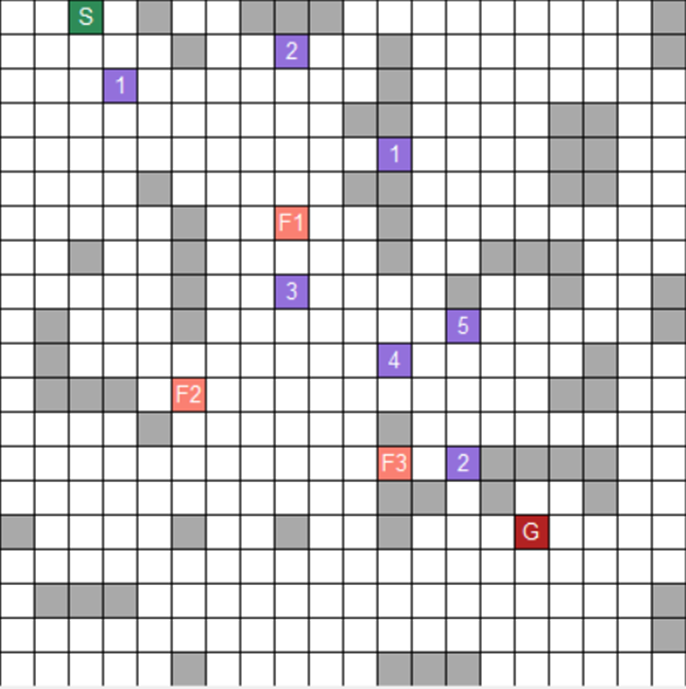
* **Requirement(s) & Description:**
  + Similar to the requirement(s) at Level 3, but the scope of the problem will be extended to multiple agents:
    - The movements are turn-based.
    - There is a maximum limit of 9 other vehicles on the map, with the Starting location numbered from S1 to S9, and the Goal location numbered from G1 to G9.
    - At each turn, if a vehicle does not have a destination, it will randomly generate a valid destination on the map (i.e., not on obstacles, or other vehicles); and then, the vehicle will make its move.
    - Once a vehicle completes its trip, it immediately generates a new trip to continue its operation. Two vehicles are not allowed to occupy the same cell at the same time. If a vehicle encounters an obstacle, like another vehicle, it can choose to wait or move in a different direction.
    - All vehicles aim to optimize their paths effectively.
  + Sequentially use the following search algorithms to produce results and perform comparisons in terms of accuracy and execution time between the algorithms: BFS, DFS, UCS, GBFS and A\*.
* **Implementation:** None.

## Graphic User Interface (GUI)

* **Requirement(s):**
  + The program is recommended to have a graphical interface to show the process step by step.
  + Each vehicle should use a different color and be printed separately for each vehicle.
* **Description & Guide:** Watch in demo video (check title 5).

# Test Case

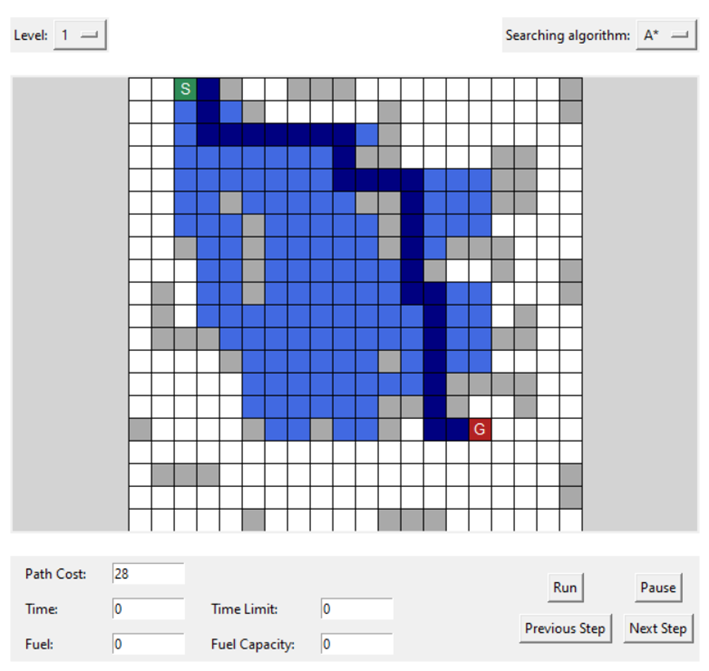
## Test 01 (20x20 grid)



* This is the map for this test case at level 3. This test case is saved in the file “*input\_1.txt*”.
* With this test case, the start node is at cell (0, 2), and the goal node is at cell (15, 15). This case also has 7 toll booths and 3 gas stations. The time limit is 35 and the fuel limit is 15.

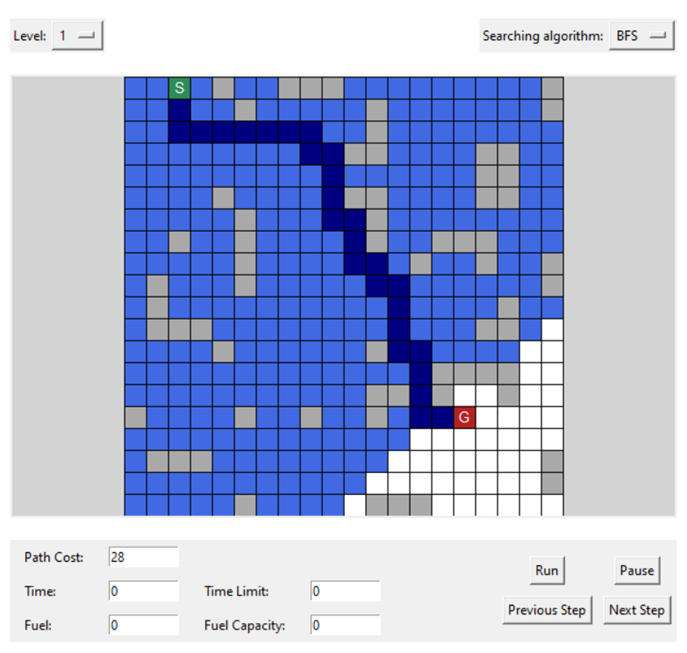
### 

### **4.1.1** A\* Search Result



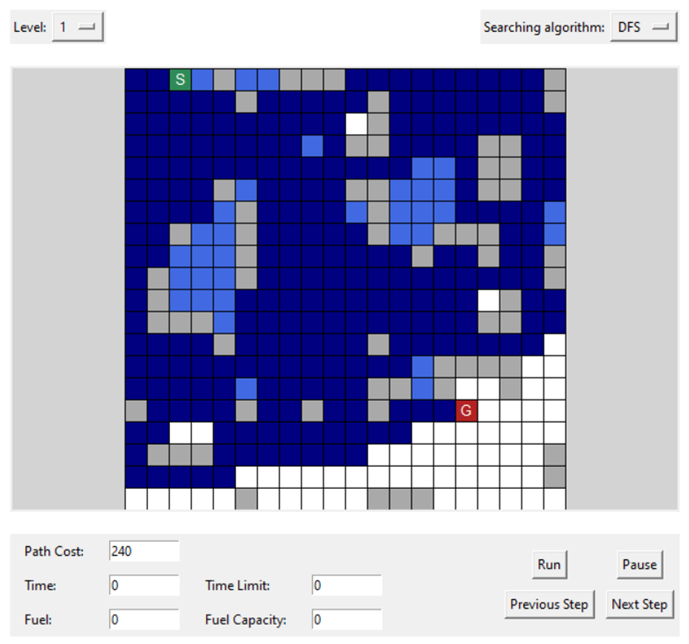
* This is the path that A\* algorithm found. The path cost is 28.

### **4.1.2** Breadth - First Search Result



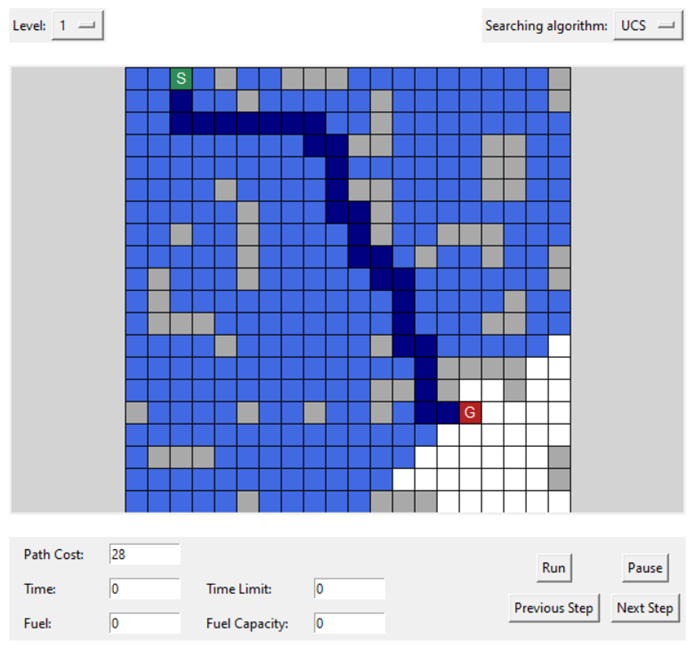
* This is the path found using BFS. The path cost is 28, the same as A\*.

### **4.1.3** Depth - First Search Result



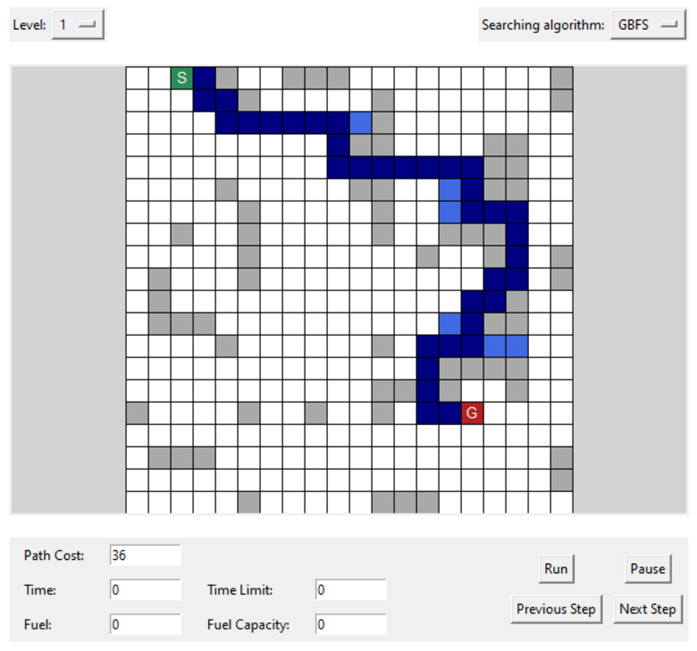
* With DFS, the path cost is 240. It shows the disadvantage of DFS. The path found is very long and time consuming compared to BFS and A\*.

### **4.1.4** Uniform - Cost Search Result



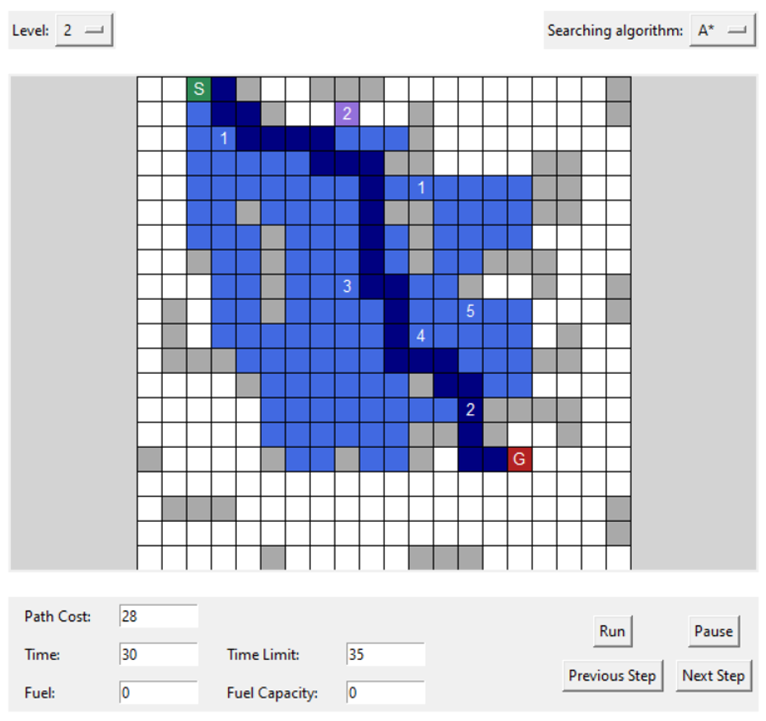
* The path found by UCS is the cost-optimal path with the cost of 28.

### **4.1.5** Greedy Best First Search Result



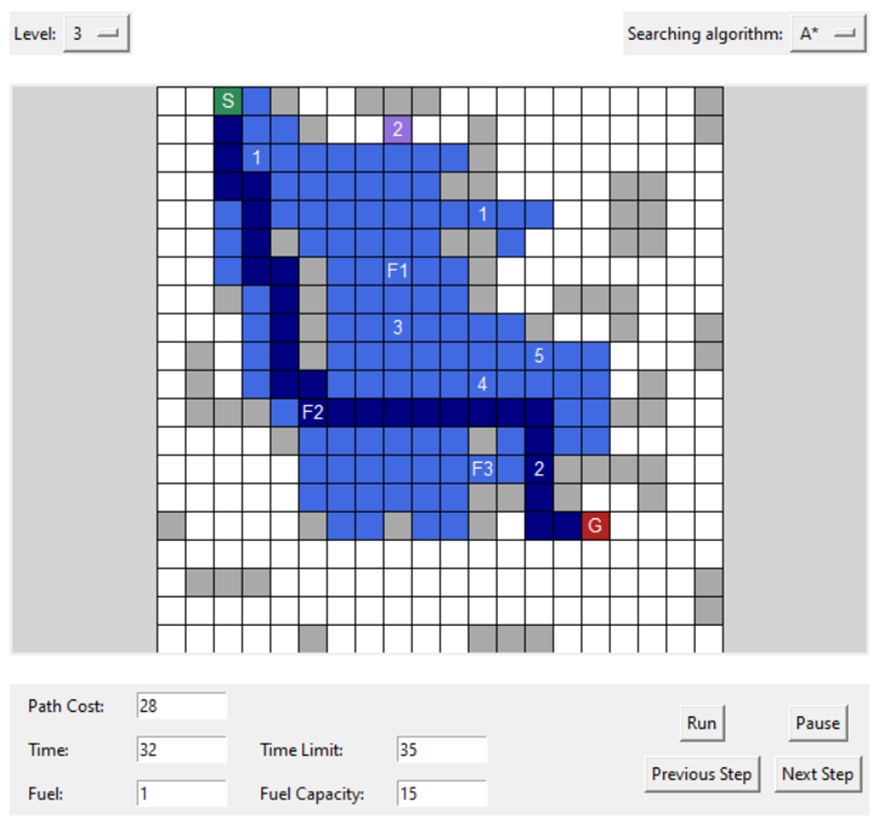
* This is the path found by GBFS with the path cost of 36.

### **4.1.6** Level 2 Result



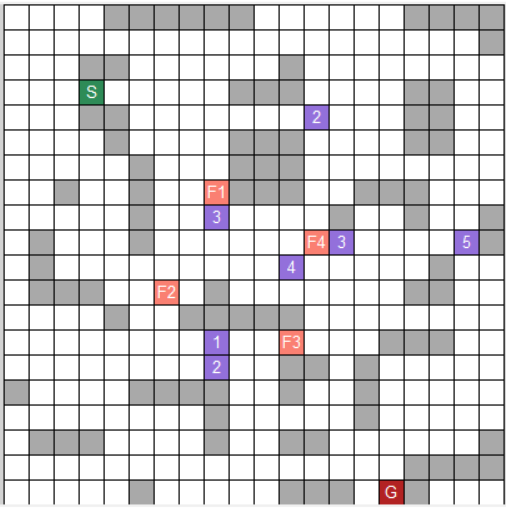
* This is the result for level 2 with time limitation. It uses A\* for its search algorithm, but the path is different from the path in level 1 A\*. Because if it follows the same path as in level 1, it will reach the time limit.
* The path cost is 28, the same as in level 1. The time needed is 30. This is acceptable because the time limit is 35.

### **4.1.7** Level 3 Result



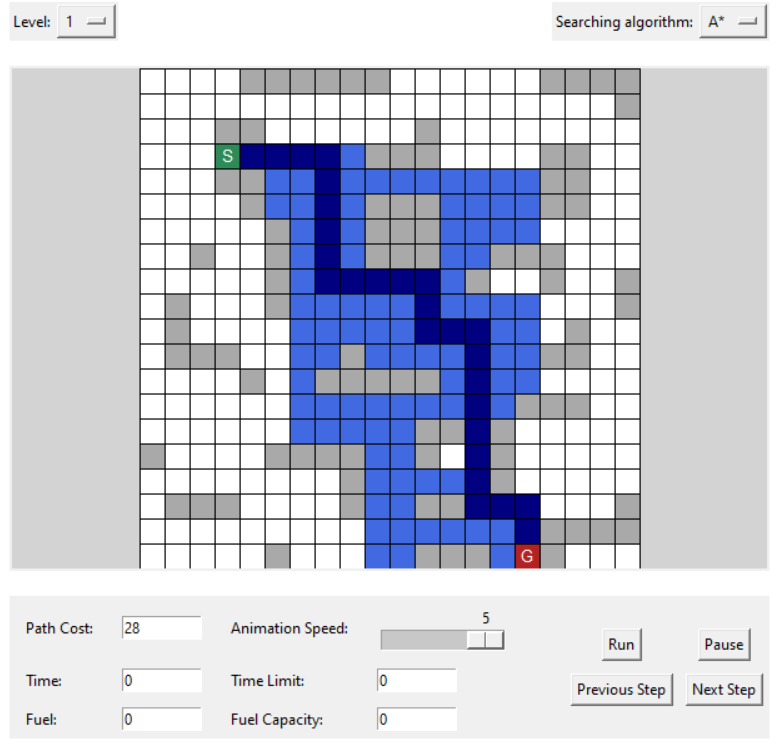
* With level 3, the search algorithm is A\*, but the path is now completely different from the path in level 2. Because if it follows the same path, the fuel will not be enough.
* The path cost is 28, and the delivery time is 32, still acceptable. The remaining fuel is 1.

## Test 02 (20x20 grid)



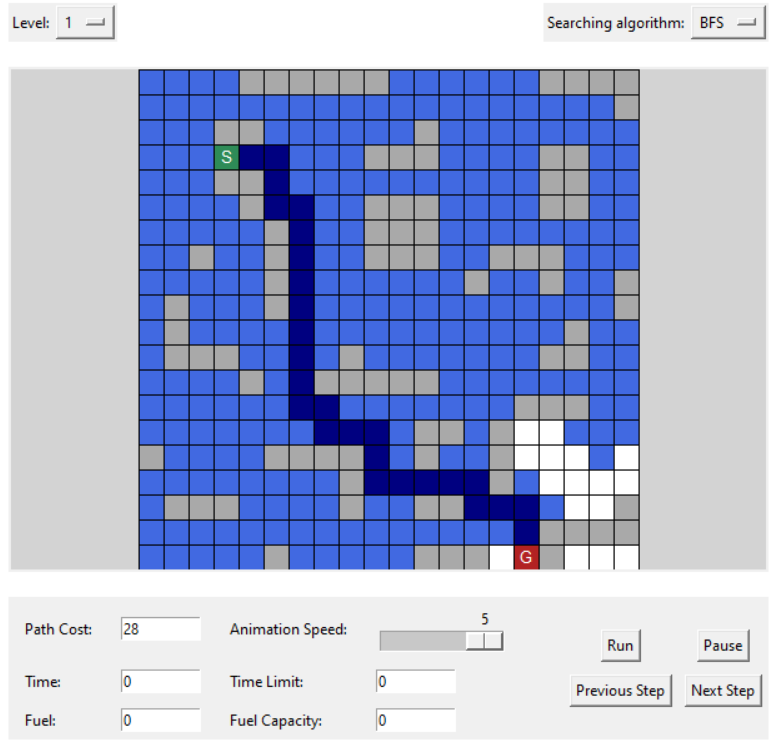
* Here is the map at level 3 for test case 2. It is stored in the file “*input\_2.txt*”.
* With this test case, the start node is at cell (3, 3), and the goal node is at cell (19, 15). This case also has 7 toll booths and 4 gas stations. The time limit is 35 and the fuel limit is 18.

### ***4.2.1*** *A\* Search Result*



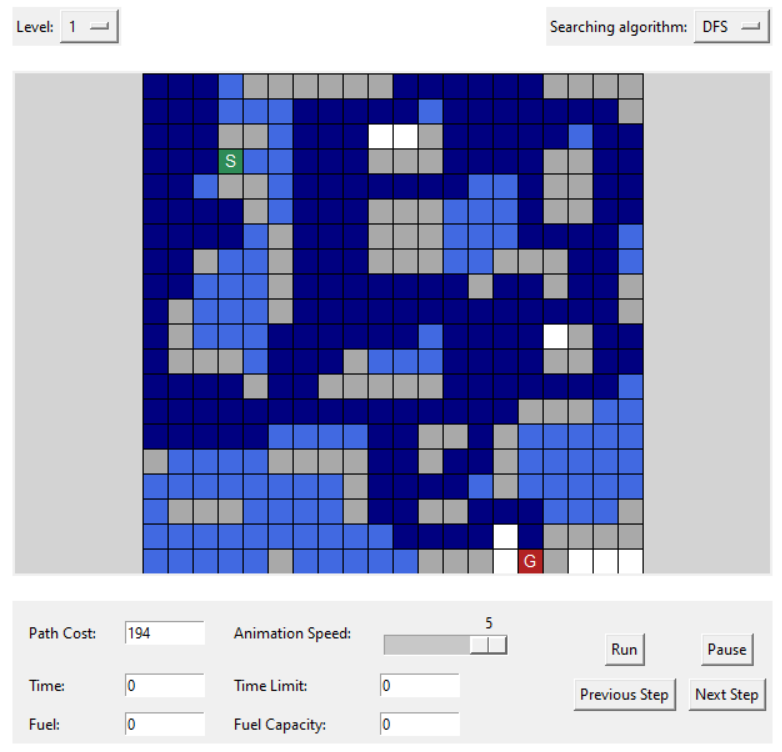
* This is the path that A\* algorithm found. The path cost is 28.

### **4.2.2** Breadth - First Search Result



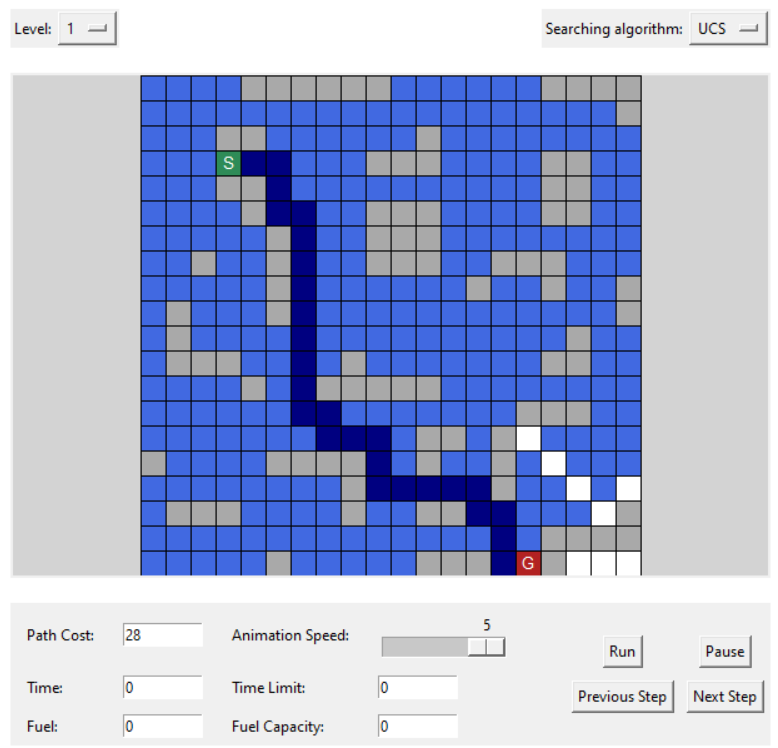
* This is the path found using BFS. The path cost is 28, the same as A\*.

### **4.2.3** Depth - First Search Result



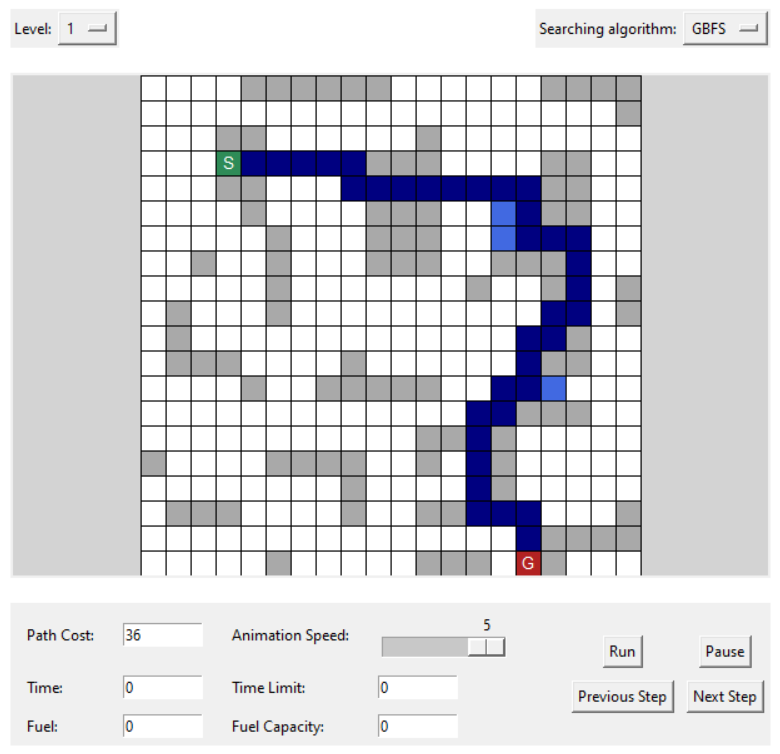
* The path found with DFS in this test case is somewhat similar to test case 1. DFS has to expand a great amount of cells, and the path found is long and time consuming compared to BFS and A\*.

### **4.2.4** Uniform - Cost Search Result



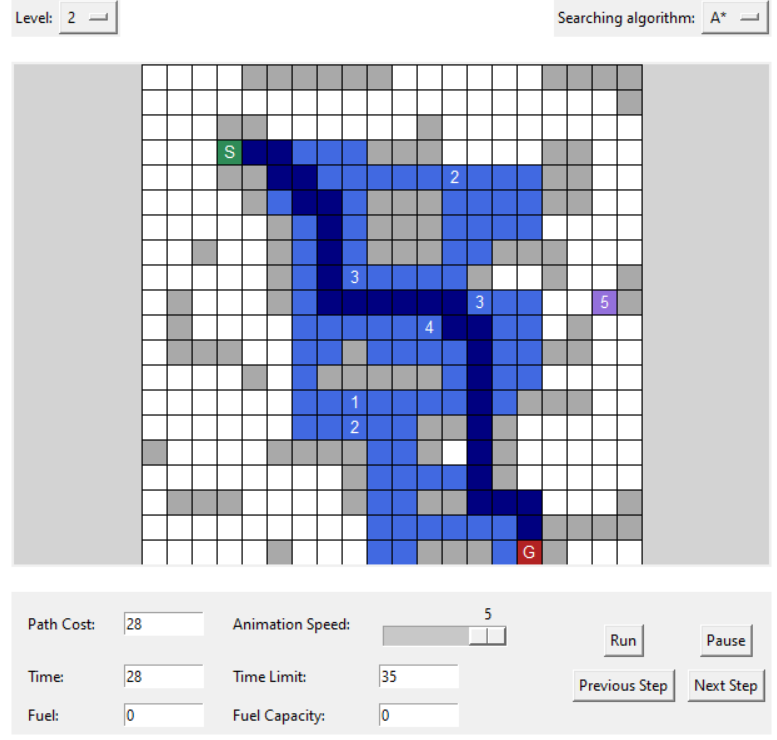
* UCS expands quite a lot of cells, but the path found is cost-optimal with the cost of 28.

### **4.2.5** Greedy Best First Search Result



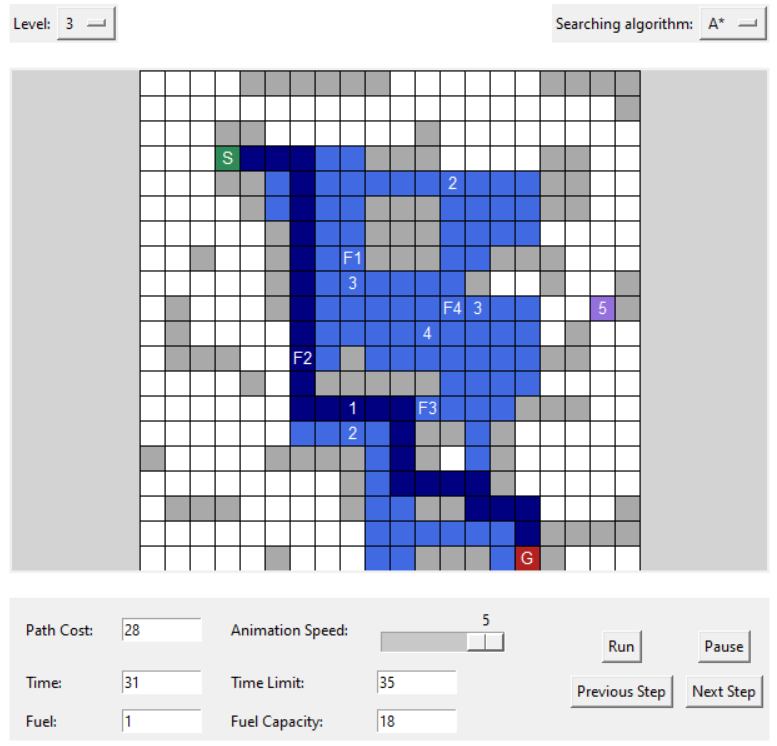
* In this test case, GBFS expands quite a few cells, but the path found is not cost-optimal. However, this algorithm takes the least time to run and expands the fewest cells compared to the other algorithms mentioned above.

### **4.2.6** Level 2 Result



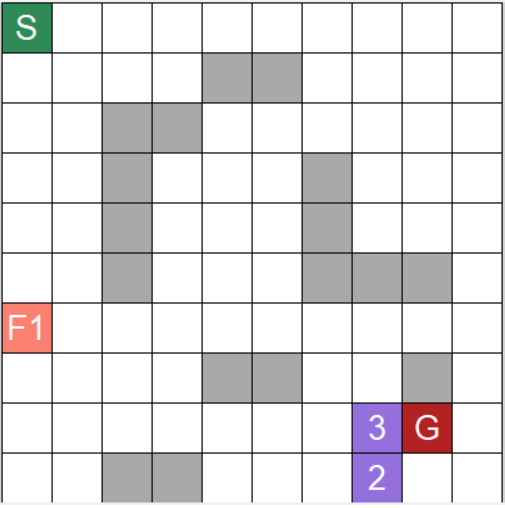
* The path found in level 2 is quite similar to the path found by A\* in level 1. The difference is that it avoids the toll booth to lower the delivery time.
* The delivery time is 28, which is acceptable compared to the time limit of 35.

### **4.2.7** Level 3 Result



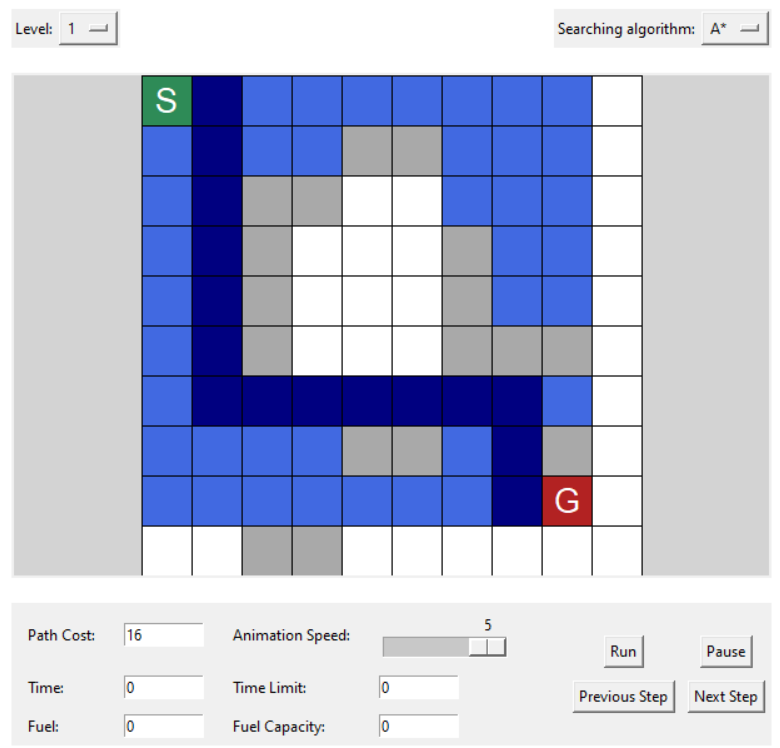
* The path that level 3 chooses is completely different from the path in level 2. Because the fuel will not be enough if it takes the same path as in level 2.
* The path cost is 28, the delivery time is 31 and the remaining fuel is 1.

## Test 03 (10x10 grid)

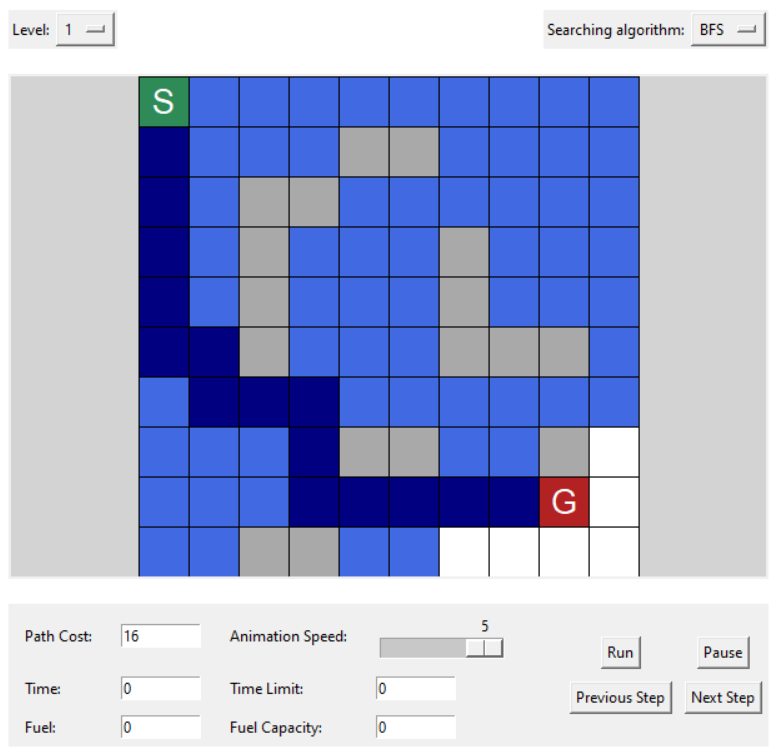


* This is the map at level 3 for test case 3. It is stored in the file “*input\_3.txt*”.
* With this test case, the start node is at cell (0, 0), and the goal node is at cell (8, 8). This case also has 2 toll booths and 1 gas station. The time limit is 15 and the fuel limit is 8.

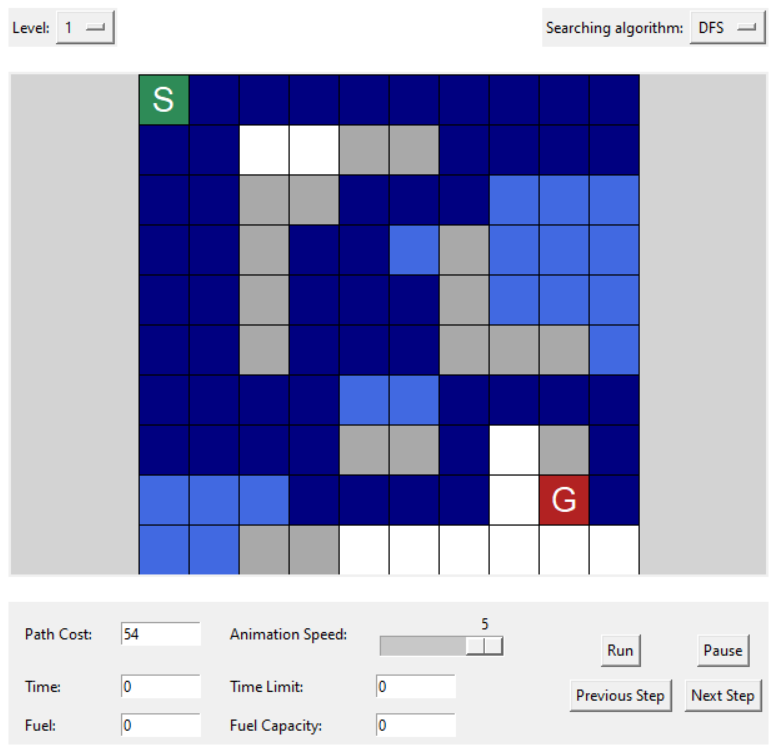
### **4.3.1** A\* Search Result



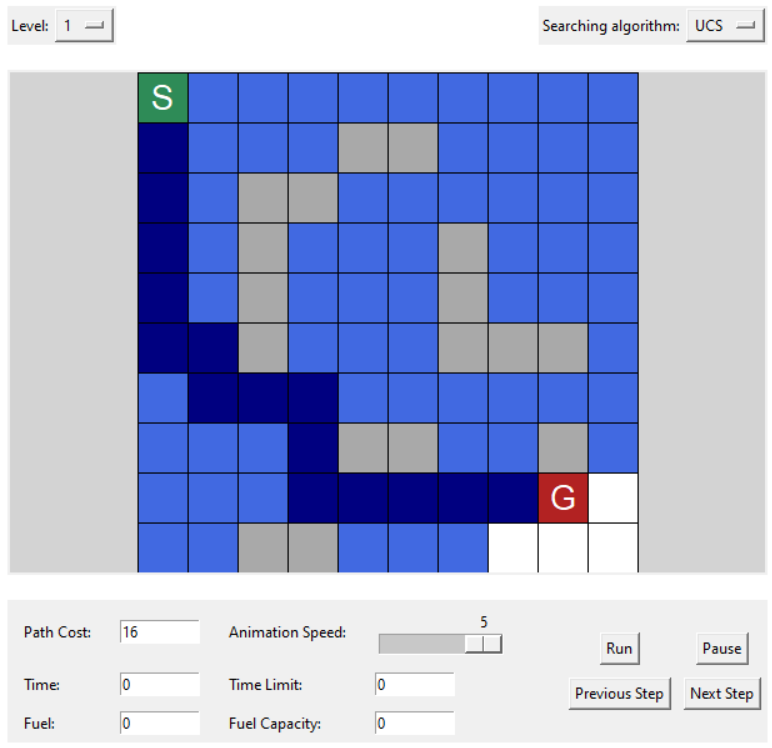
### **4.3.2** Breadth - First Search Result



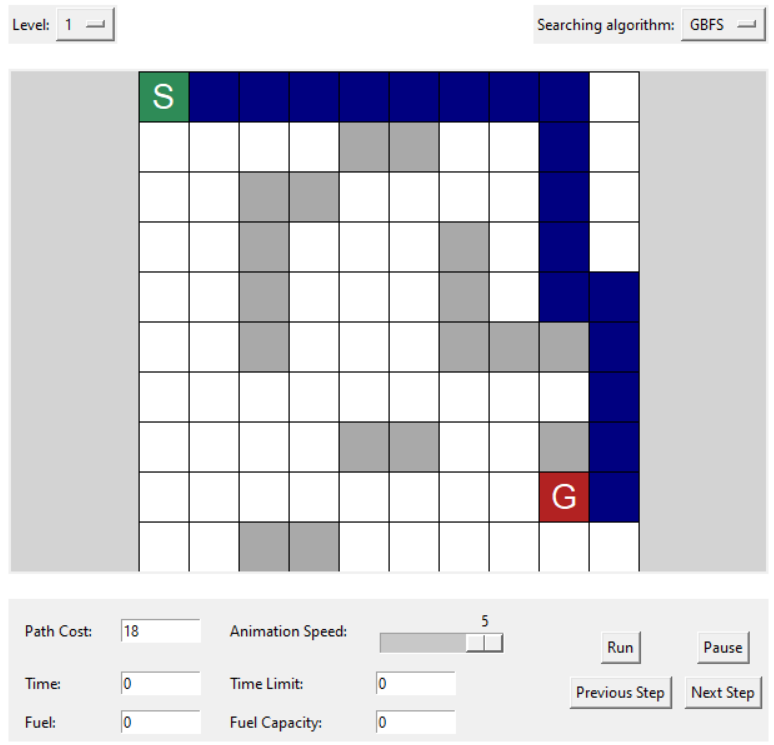
### **4.3.3** Depth - First Search Result



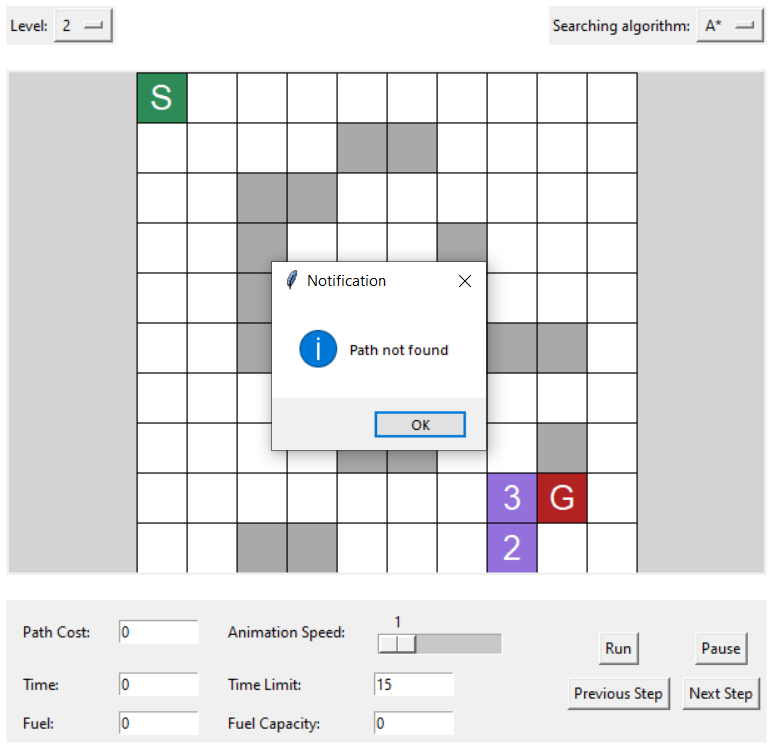
### **4.3.4** Uniform - Cost Search Result



### **4.3.5** Greedy Best First Search Result

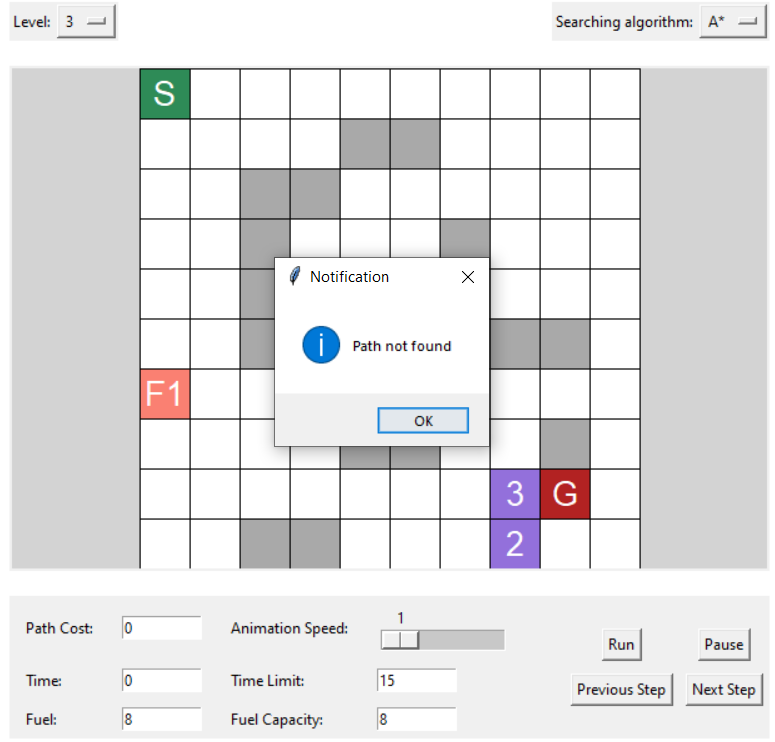


### **4.3.6** Level 2 Result



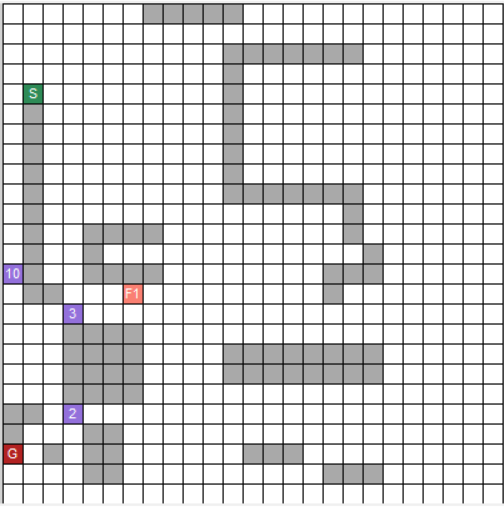
* With level 2 of this test case, there is no path that satisfies the time limit.

### **4.3.7** Level 3 Result



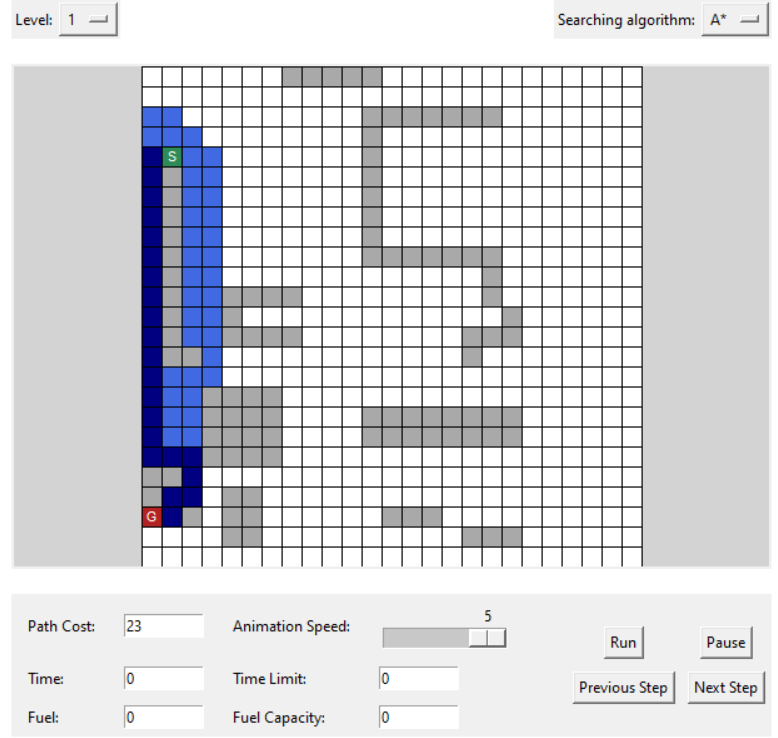
* Similar to level 2, level 3 of this test case has no path that satisfies the requirement.

## Test 04 (25x25 grid)

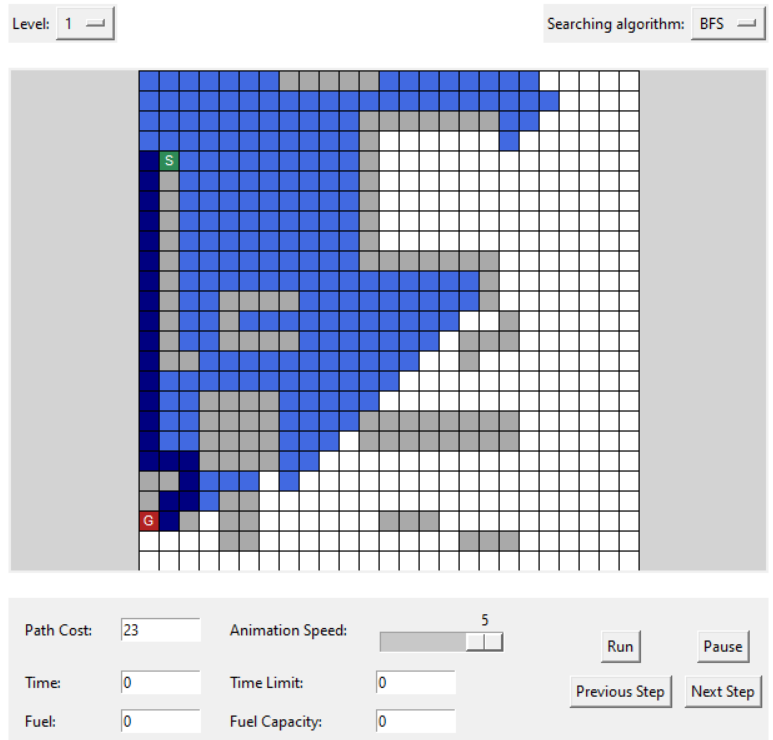


* This is the map at level 3 for this test case. It is stored in the file “*input\_4.txt*”.
* With this test case, the start node is at cell (4, 1), and the goal node is at cell (22, 0). This case also has 3 toll booths and 1 gas station. The time limit is 35 and the fuel limit is 20.

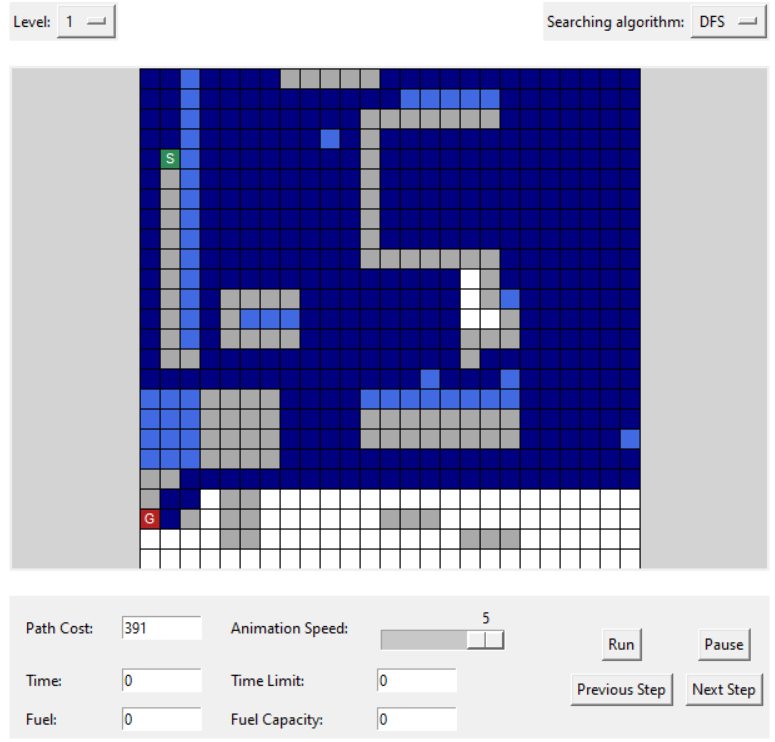
### **4.4.1** A\* Search Result



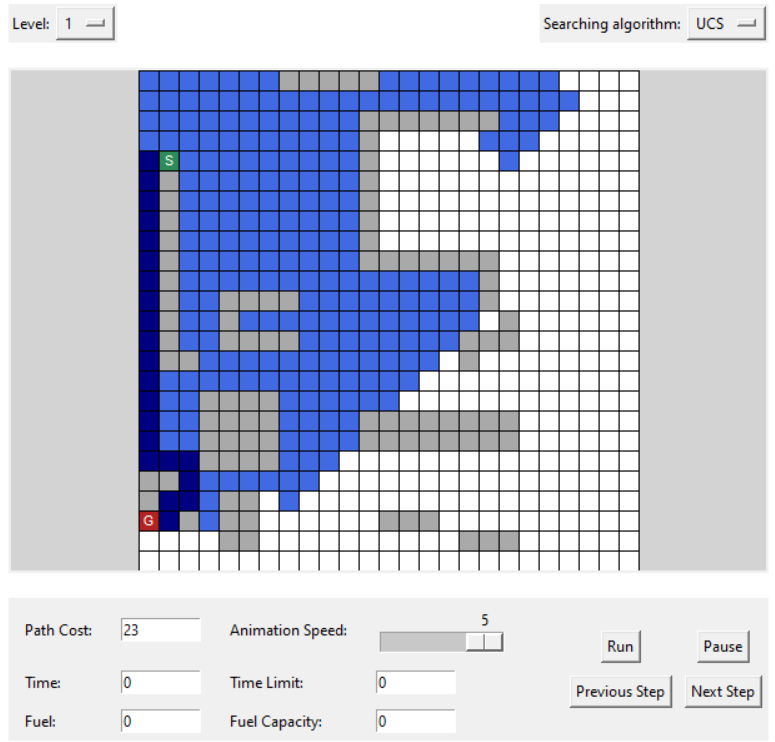
### **4.4.2** Breadth - First Search Result



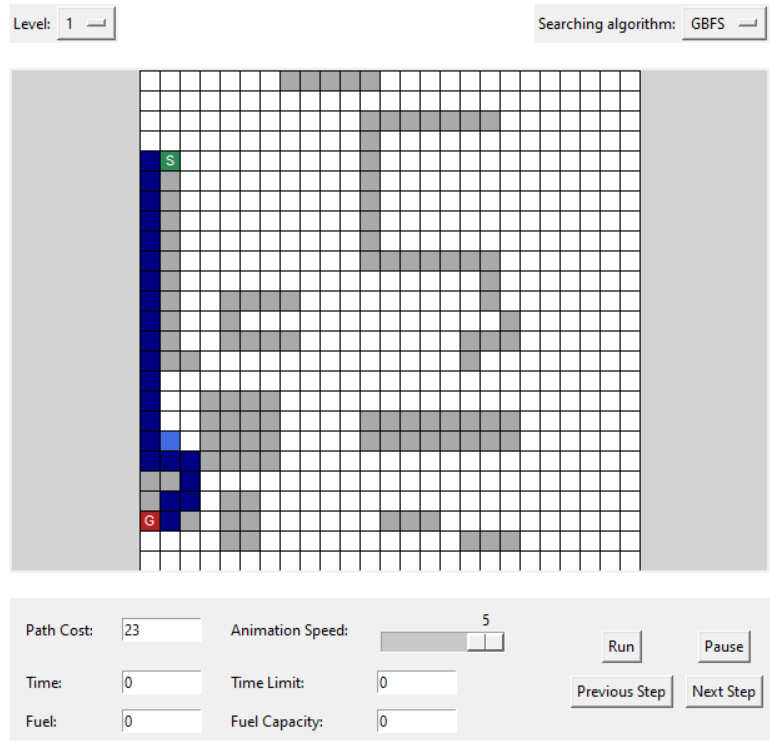
### **4.4.3** Depth - First Search Result



### **4.4.4** Uniform - Cost Search Result

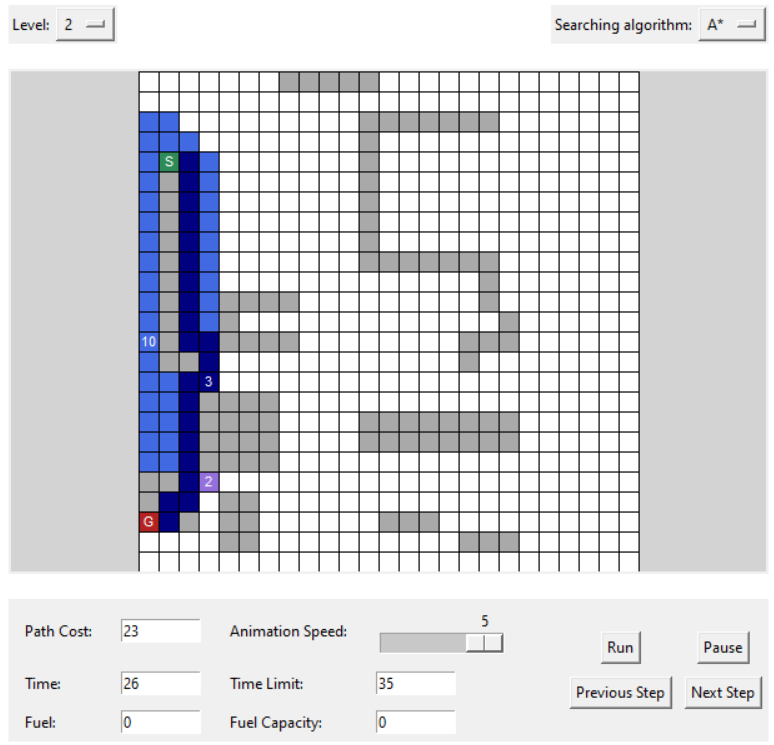


### **4.4.5** Greedy Best First Search Result



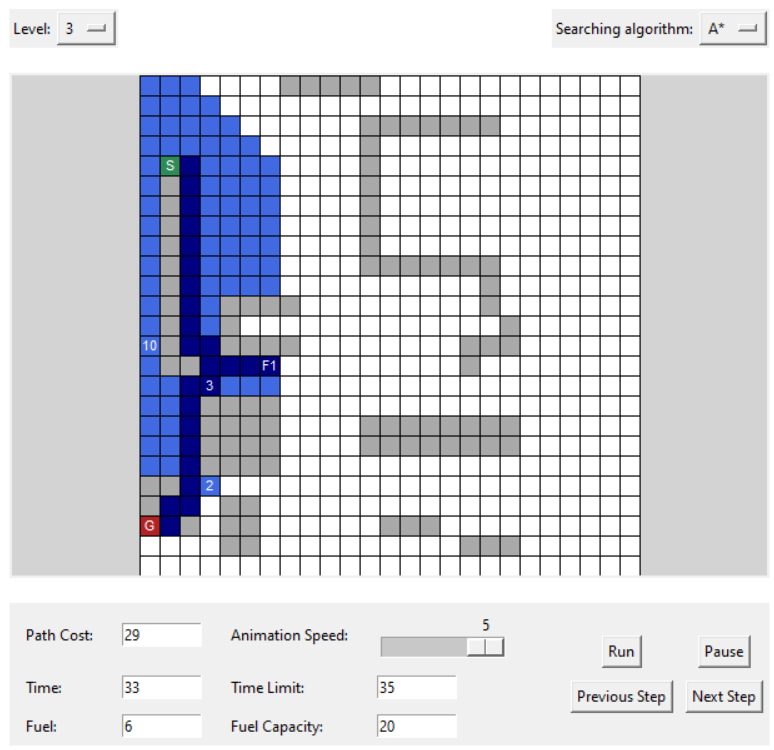
* For this test case, GBFS has the best performance among the search algorithms. It returns the cost-optimal path and expands the fewest cells.

### **4.4.6** Level 2 Result



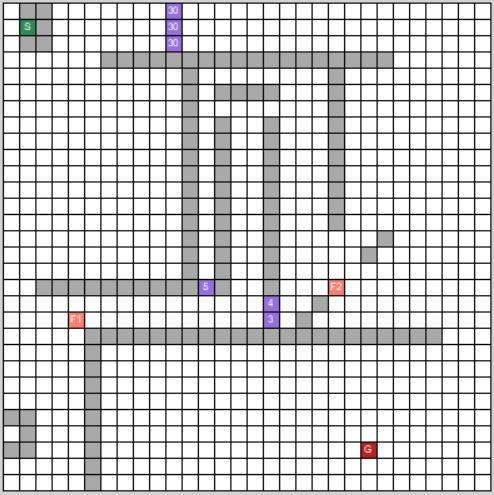
* In this test case, the program chooses a different path from level 1. This path is also cost-optimal and satisfies the time limit.

### **4.4.7** Level 3 Result



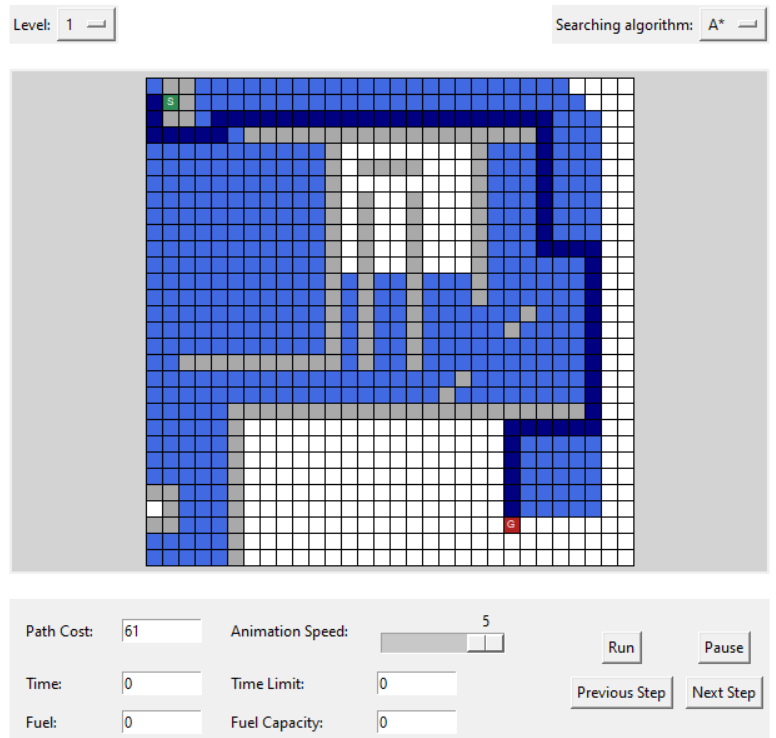
* In this level, because the fuel is not enough to go straight to the goal, the agent has to go to the gas station first, then return to the path to the goal.

## Test 05 (30x30 grid)

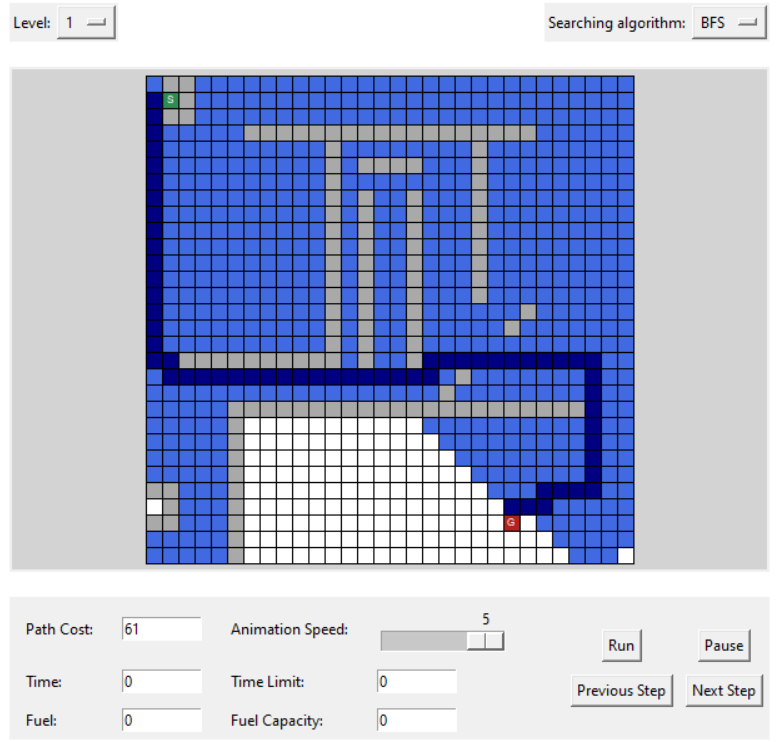


* This is the map at level 3 for this test case. It is stored in the file “*input\_5.txt*”.
* With this test case, the start node is at cell (1, 1), and the goal node is at cell (27, 22). This case also has 6 toll booths and 2 gas stations. The time limit is 80 and the fuel limit is 35.

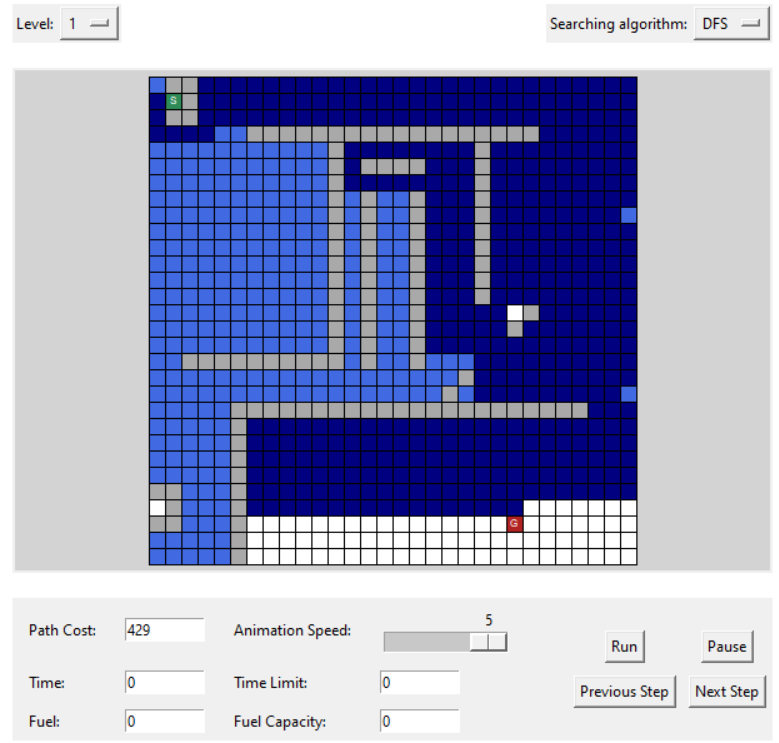
### **4.5.1** A\* Search Result



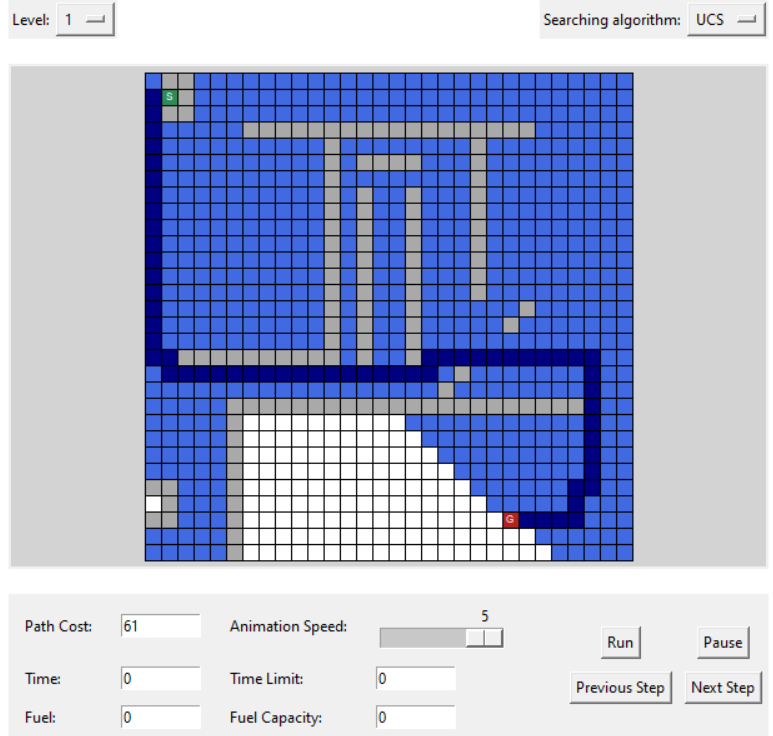
### **4.5.2** Breadth - First Search Result



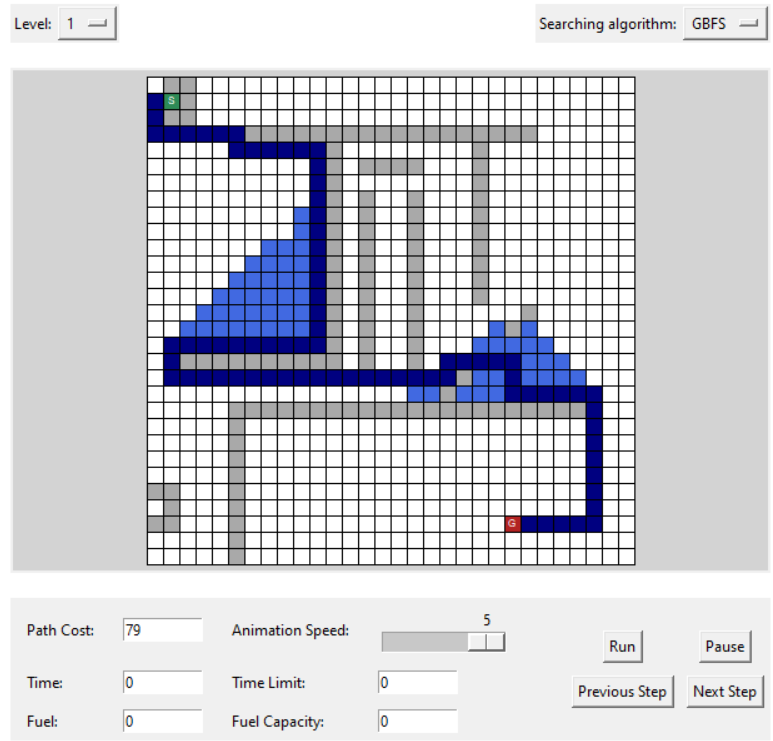
### **4.5.3** Depth - First Search Result



### **4.5.4** Uniform - Cost Search Result

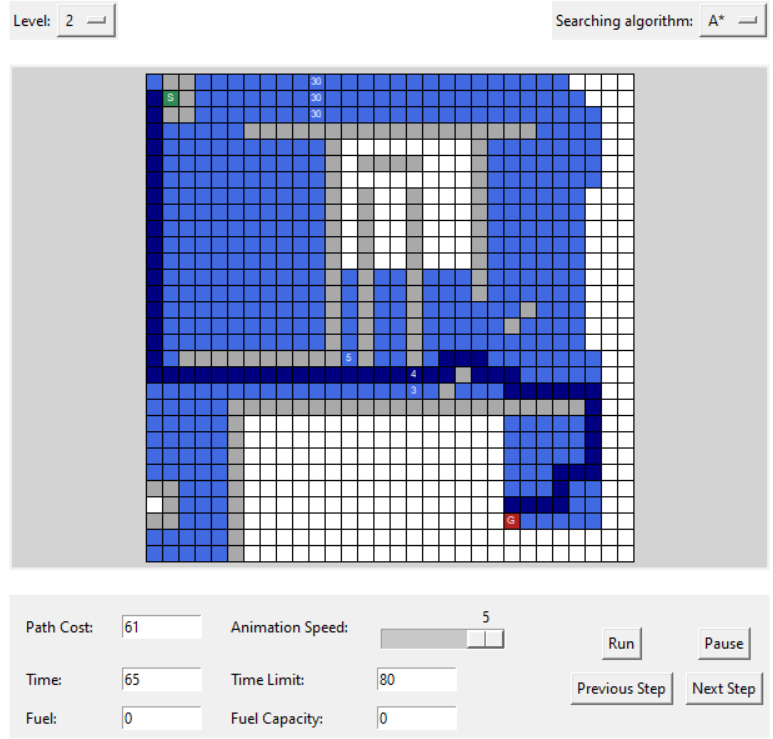


### **4.5.5** Greedy Best First Search Result



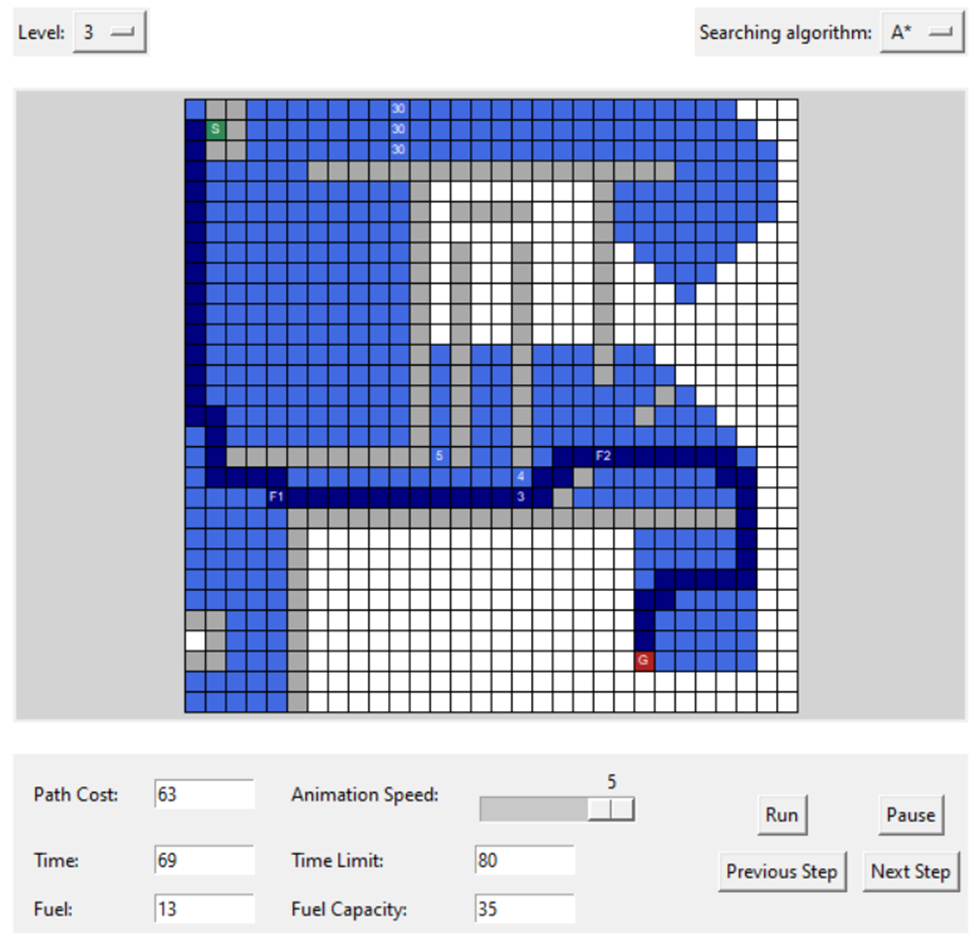
* Although it may not give the cost-optimal path, GBFS is the algorithm that expands the fewest cells among the 5 search algorithms: A\*, BFS, DFS, UCS, GBFS.

### **4.5.6** Level 2 Result



* The path in level 2 is completely different from the path generated by A\* in level 1. It cannot follow the same path as in level 1 because that path would not satisfy the time limit.

### **4.5.7** Level 3 Result



* The path in level 3 is nearly the same as in level 2. However, in this test case, the run-time in level 3 is much longer than in level 2 because it has to expand the same cell repeatedly.

# Demo

* GUI guide & all test-cases (include sample test) demonstration: [Demo video](https://youtu.be/jYYZUnpD_8E?si=RutK4jkajpyFvgcB).