CSA2001 - Fundamentals of AI and ML Autonomous Delivery Agent Project Report

1. Introduction

This project implements an autonomous delivery agent that navigates a 2D grid environment to deliver packages.
The agent operates under constraints such as static obstacles, varying terrain costs, and dynamic moving obstacles.
The goal is to evaluate different search strategies — uninformed (BFS, UCS), informed (A*), and local search (Hill Climbing) — for pathfinding efficiency.

2. Environment Model

The environment is modeled as a 2D grid where each cell can represent free space, an obstacle, or terrain with varying movement costs.
- '0' represents normal terrain with cost 1
- Digits >1 represent higher terrain costs
- '#' represents impassable obstacles
- 'S' and 'G' denote the start and goal positions

The agent moves in four directions (up, down, left, right).

3. Agent Design and Algorithms

The agent is rational: it chooses actions that maximize delivery efficiency while minimizing path cost and time.
Implemented planners include BFS, UCS, A*, and Hill Climbing. Each algorithm has different trade-offs in terms of optimality, runtime, and adaptability.

4. Experimental Results

The algorithms were tested on four maps: small, medium, large, and dynamic. Results are summarized below:

map	algo	path_len	cost	nodes	time
small.txt	bfs	18	17	89	0.0
small.txt	ucs	18	17	279	0.0015060901641845
small.txt	astar	18	17	100	0.0

small.txt	hill	18	17	18	0.0
medium.txt	bfs	39	38	230	0.0
medium.txt	ucs	39	38	691	0.0010268688201904
medium.txt	astar	39	38	196	0.0010035037994384
medium.txt	hill	39	38	39	0.0
large.txt	bfs	59	58	849	0.0010066032409667
large.txt	ucs	59	58	3074	0.0030274391174316
large.txt	astar	59	58	1349	0.004504919052124
large.txt	hill	59	58	59	0.0
dynamic.txt	bfs	28	27	200	0.0
dynamic.txt	ucs	28	27	673	0.0
dynamic.txt	astar	28	27	293	0.0010504722595214
dynamic.txt	hill	28	27	28	0.0

5. Analysis

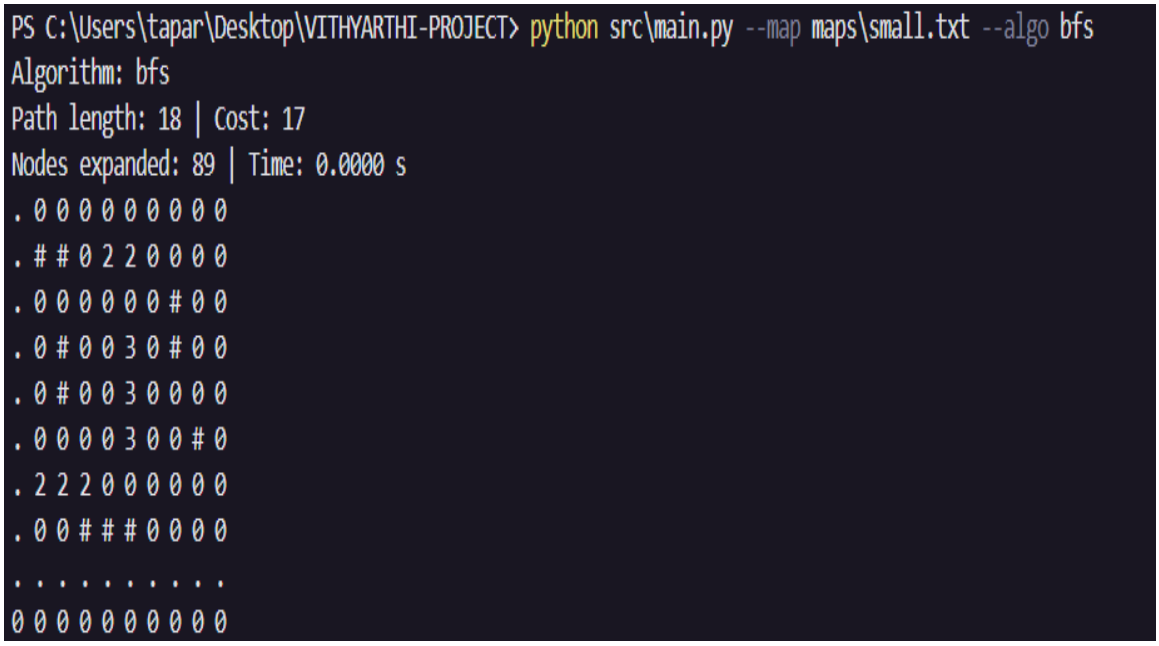
From the experimental results:
- BFS always finds shortest paths in steps but expands many nodes, especially on large maps.
- UCS ensures optimal cost paths but at the expense of much higher node expansions and time.
- A* significantly reduces nodes expanded and runtime while still delivering optimal cost paths, proving efficiency.
- Hill Climbing is extremely fast with minimal nodes expanded but risks suboptimal or failed solutions in some scenarios.

6. Dynamic Replanning

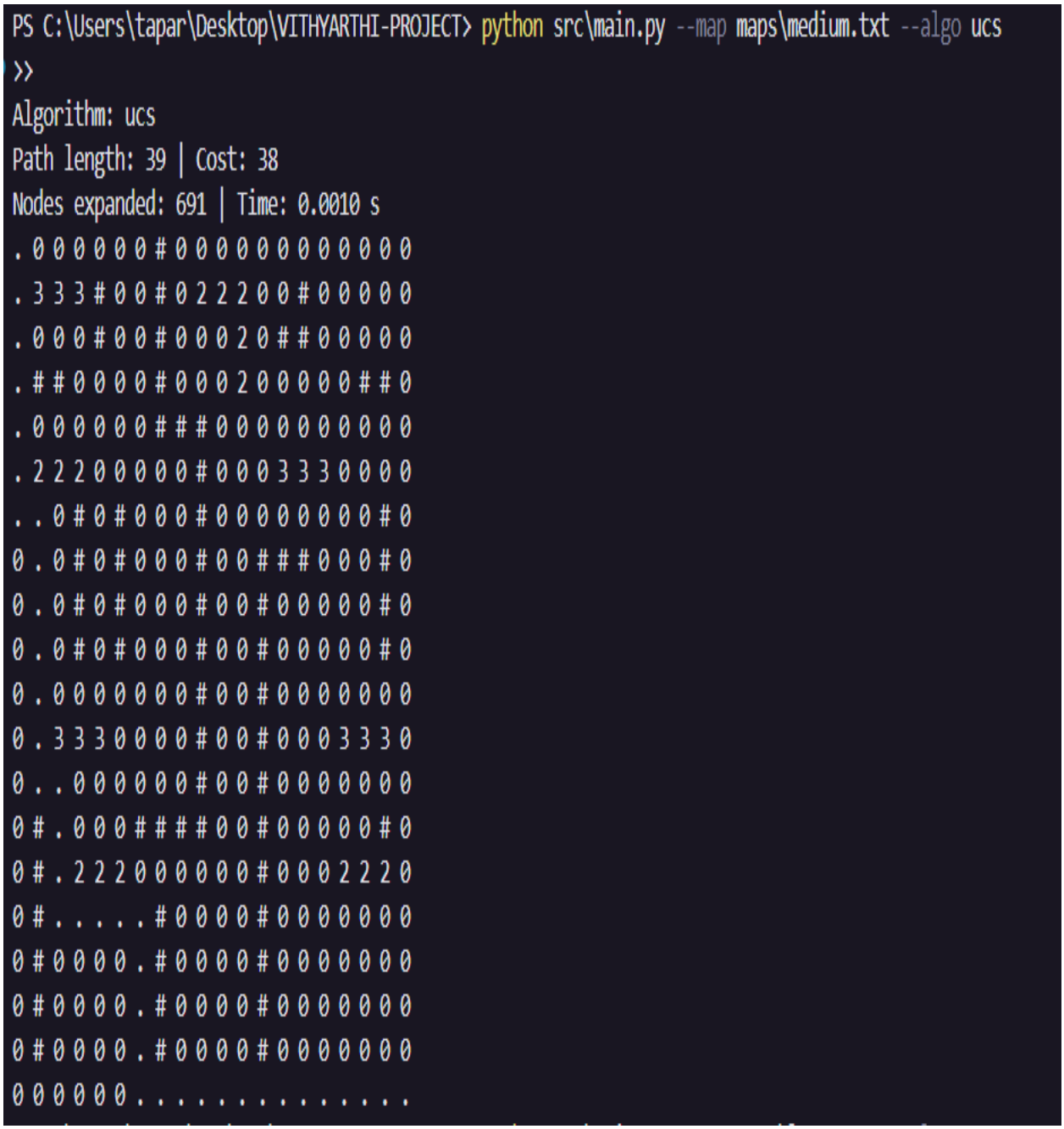
A dynamic scenario was tested where obstacles appeared after initial planning.
The agent detected blocked paths and replanned using the chosen algorithm. Logs were generated in 'logs/dynamic_replan.log' and snapshots captured intermediate steps.
This demonstrates adaptability in real-world conditions with unpredictable traffic or moving obstacles.

7. Output Screenshots

BFS RUN ON SMALL MAP:



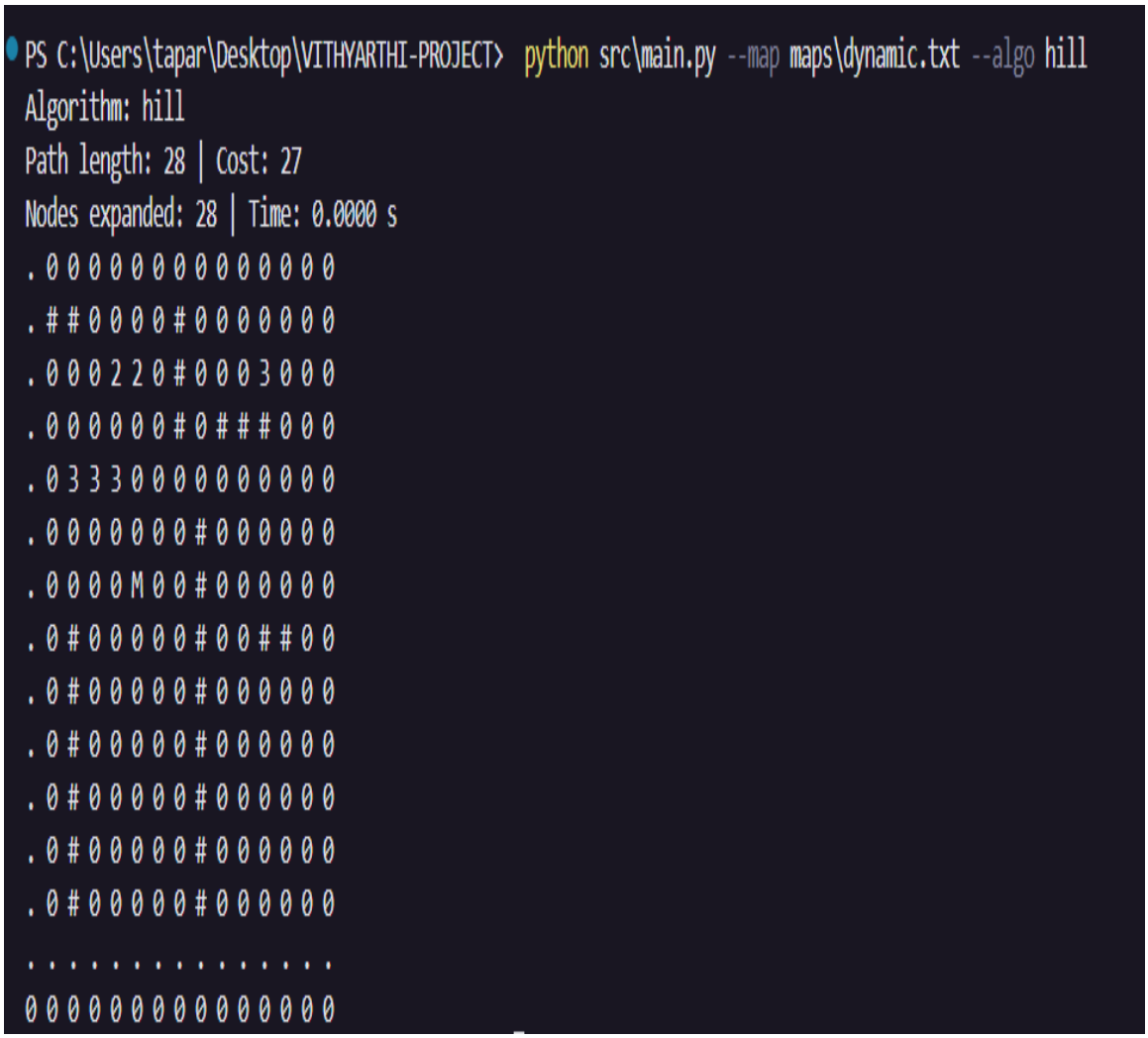
UCS RUN ON MEDIUM MAP:



A* RUN ON LARGE MAP:

```
PS C:\Users\tapar\Desktop\VITHYARTHI-PROJECT> python src\main.py --map maps\large.txt --algo astar
>>
Algorithm: astar
Path length: 59 | Cost: 58
Nodes expanded: 1349 | Time: 0.0030 s
. . . . . . . # 0 0 0 0 0 0 0 0 0 0 # 0 0 0 0 0 0 0 0 0 0 0
0 0 # 0 2 0 . . . . . 3 0 0 0 # 0 2 0 0 0 0 0 # 0 3 0 0 0 0
0 2 0 0 0 0 0 # 0 0 . . . 2 0 0 0 0 3 0 0 0 0 0 0 0 0 0 2 0
0 0 0 0 # 0 0 0 0 0 0 0 . . . . . . # 0 0 0 3 0 0 0 0 # 0 0
0 0 0 0 0 0 # 0 2 0 0 0 # 0 0 0 0 . . . . . 3 0 0 0 0
0 # 0 0 0 0 0 0 0 0 0 0 0 # 0 0 0 0 0 0 0 2 0 . . # 0 0 0
0 0 0 0 2 0 0 0 # 0 0 0 0 0 0 0 0 0 0 0 0 0 0 # 0 . . . 2 0
0 0 0 0 0 0 # 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 # 0 . . .
0 0 # 0 0 0 0 0 0 0 0 # 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 # .
0 0 0 0 0 0 0 0 # 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 .
0 # 0 0 0 0 0 0 0 0 0 0 0 0 0 0 # 0 0 0 0 0 # 0 0 0 0 0 .
0 0 0 2 0 0 # 0 0 0 0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 .
0 0 0 0 0 0 0 0 # 0 0 0 0 0 0 0 0 0 # 0 0 0 0 0 0 0 0 0 .
0 0 # 0 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 # 0 0 0 0 0 0 0 0 .
0 0 0 0 0 0 # 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 .
0 0 0 0 0 0 0 0 # 0 0 0 0 0 0 0 0 # 0 0 0 0 0 0 0 0 0 .
0 0 0 0 0 0 # 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 # 0 0 .
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 # 0 0 0 0 0 0 0 0 0 0 0 .
0 0 0 # 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 # .
0 0 0 0 0 0 0 0 0 0 0 # 0 0 0 0 0 0 0 0 0 0 0 0 0 0 .
0 0 0 0 0 # 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 # 0 0 0 0 .
0 0 0 0 0 0 0 0 0 0 0 0 0 # 0 0 0 0 0 0 0 0 0 0 0 0 .
0 # 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 # 0 .
0 0 0 0 0 0 0 0 0 0 0 0 # 0 0 0 0 0 0 0 0 0 0 0 0 .
0 0 0 0 # 0 0 0 0 0 0 0 0 0 0 0 0 # 0 0 0 0 0 0 0 .
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 # 0 0 0 0 .
0 0 0 0 0 0 0 0 # 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 .
0 0 0 0 0 # 0 0 0 0 0 0 0 0 0 0 0 0 0 0 # 0 0 .
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 .
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0Â 0Â 0Â 0Â 0Â 0Â 0Â 0Â .
```

HILL CLIMBING RUN ON DYNAMIC MAP:



8. Conclusion

The study shows that no single algorithm is universally best:
- BFS is simple but inefficient in large/costly maps.
- UCS is optimal but computationally expensive.
- A* provides the best balance of cost and efficiency for static maps.
- Hill Climbing excels in fast, adaptive replanning for dynamic environments.