Automated Maze Solver: Implementing DFS, BFS, and A* Algorithms for Random Maze Generation and Navigation

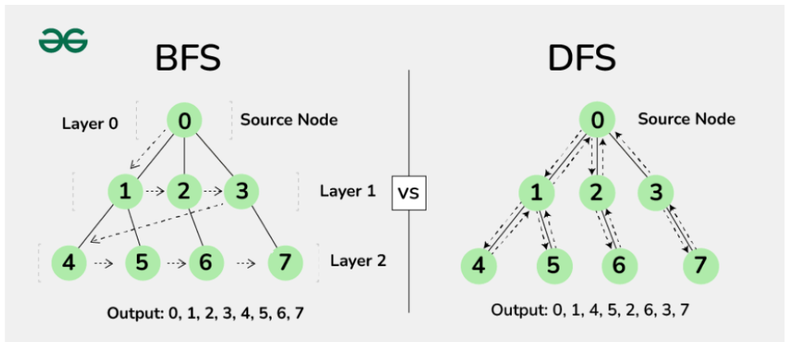
ABSTRACT

This project develops an automated maze-solving system using DFS, BFS, and A* algorithms, comparing their efficiency in terms of time, memory usage, and path optimality. It addresses real-world pathfinding challenges, highlighting each algorithm's strengths and trade-offs: DFS for memory efficiency, BFS for optimal paths, and A* for balancing speed and optimality.

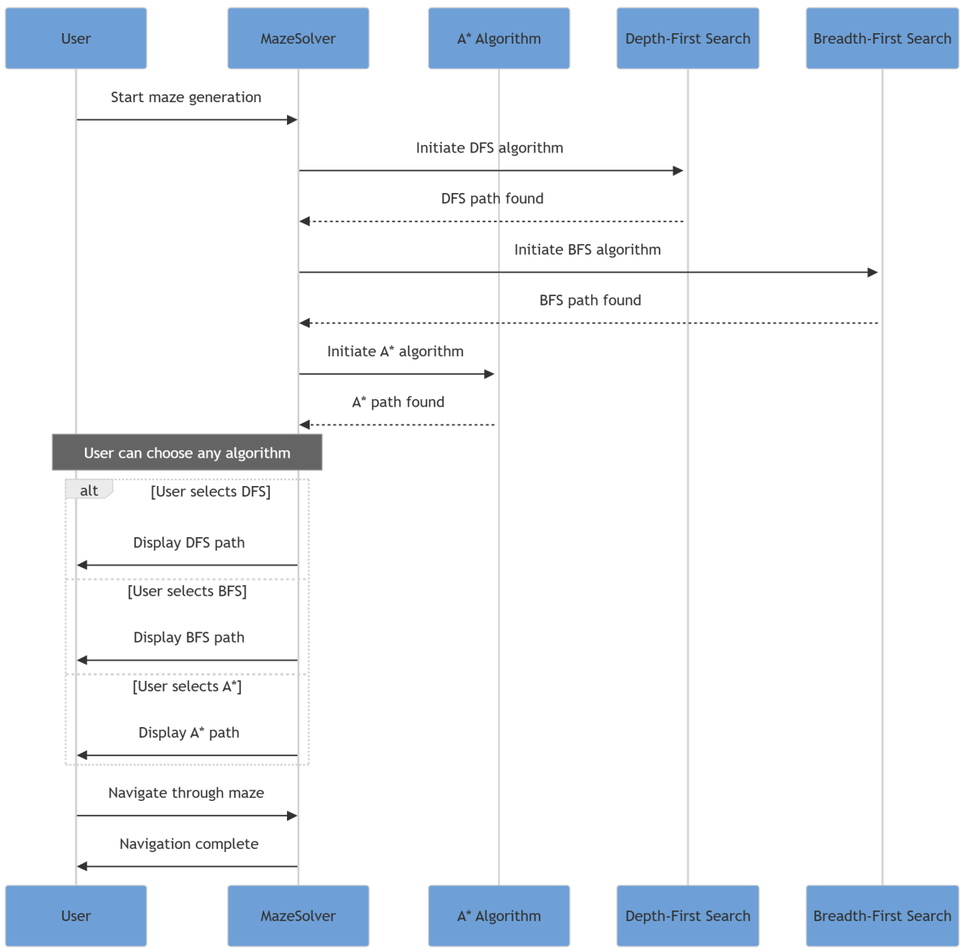
PROBLEM STATEMENT

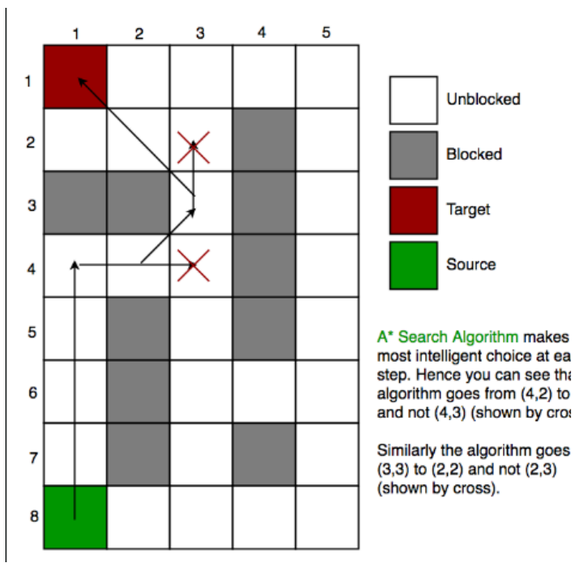
The problem addressed in this study is determining the most effective algorithm—Depth First Search (DFS), Breadth First Search (BFS), or A*—for solving randomly generated mazes of varying complexities. Each algorithm has trade-offs in terms of speed, memory consumption, and path optimality

ALGORITHMS



Depth-First Search (DFS): Explores deeply before backtracking, memory-efficient but may not find the shortest path.
Breadth-First Search (BFS): Explores level-by-level, guarantees the shortest path but is memory-intensive.
A Search:* Uses heuristics to balance speed and optimality, effective for complex mazes but depends on the heuristic used.





RESULT

DFS:

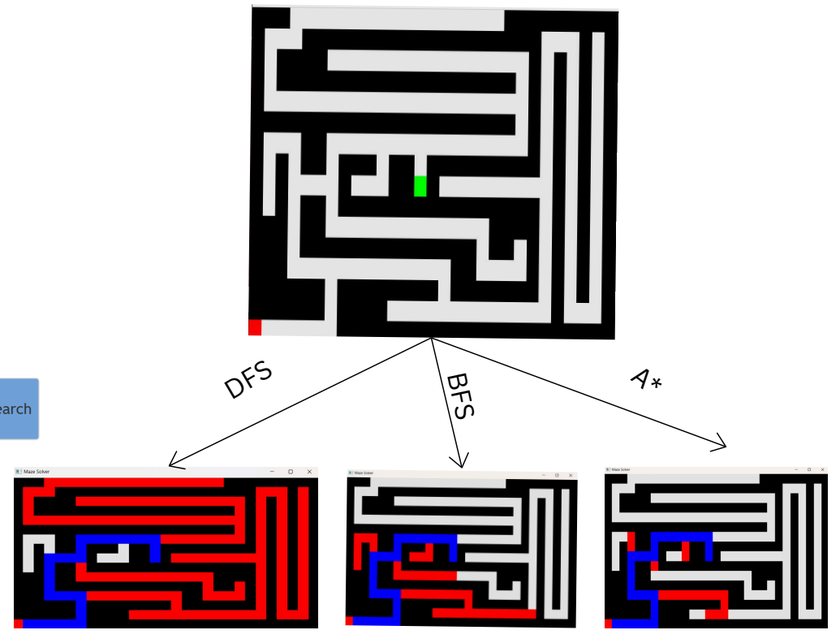
- Time complexity: O(V+E). Explores blindly, often covering unnecessary areas, leading to inefficiency in complex mazes.

BFS:

- Time complexity: O(V+E). Guarantees the shortest path but explores all possible paths, making it slower in large or dense mazes.

A*:

- Time complexity: O(E). Uses heuristics to prioritize nodes, reducing unnecessary exploration and making it faster in practice.



CONCLUSION

- DFS is memory-efficient but inefficient for complex mazes, often failing to find the shortest path.
- BFS guarantees the shortest path but is resource-intensive and slow for larger mazes.
- A* outperforms both by balancing exploration and optimality, making it ideal for real-world and computationally complex pathfinding problems.

REFERENCES: Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to Algorithms (3rd ed.).