Project Proposal

Project Title

Solving the 8-Puzzle Problem Using Artificial Intelligence Search Algorithms

Introduction

The 8-Puzzle problem is a classical problem in Artificial Intelligence used to demonstrate the concept of state-space search. It consists of a 3×3 grid with eight numbered tiles and one empty space. The goal is to reach a predefined goal state by sliding tiles into the empty space using valid moves.

This project focuses on solving the 8-Puzzle problem using different AI search algorithms and comparing their performance.

Project Objectives

To understand state-space representation in AI.

To implement uninformed and informed search algorithms including BFS, DFS, UCS, IDS, and A*.

To compare algorithms based on efficiency and resource usage.

Algorithms Used and Evaluation

1. Breadth First Search (BFS)

Advantages:

Guarantees finding the shortest solution path.

Complete algorithm (will always find a solution if one exists).

Disadvantages:

Very high memory consumption.

Slow for large search spaces.

Time Complexity: O(b^d)

Memory Complexity: O(b^d)

2. Depth First Search (DFS)

Advantages:

Requires low memory.

Simple and easy to implement.

Disadvantages:

Does not guarantee the shortest solution.

Can get stuck in deep or infinite paths.

Time Complexity: O(b^m)

Memory Complexity: O(b × m)

3**. Uniform Cost Search (UCS)**

Advantages:

Guarantees optimal solution.

Suitable for problems with varying action costs.

Disadvantages:

High memory usage.

Slower than BFS when all step costs are equal.

Time Complexity: O(b^(C*/ε))

Memory Complexity: O(b^(C*/ε))

4. A* Search Algorithm

Advantages:

Very efficient compared to uninformed search algorithms.

Guarantees optimal solution when using an admissible heuristic.

Disadvantages:

Performance depends heavily on heuristic quality.

High memory consumption.

Time Complexity: O(b^d)

Memory Complexity: O(b^d)

5. Iterative Deepening Search (IDS)

Advantages:

Combines the benefits of BFS and DFS.

Uses low memory like DFS.

Guarantees finding the optimal solution.

Disadvantages:

Repeats node expansions.

Slightly slower than BFS due to repeated searches.

Time Complexity: O(b^d)

Memory Complexity: O(b × d)

Tools and Technologies

Programming Language: Python

Data Structures: Queue, Stack, Priority Queue

Expected Outcomes

Breadth First Search (BFS): Expected to always find the shortest solution path, but with very high memory consumption due to storing all frontier states.

Depth First Search (DFS): Expected to use minimal memory and run fast in some cases, but may return a non-optimal solution or fail in deep search spaces.

Uniform Cost Search (UCS): Expected to guarantee an optimal solution by expanding the lowest-cost paths first, with high memory and time requirements.

Iterative Deepening Search (IDS): Expected to find an optimal solution like BFS while using much less memory, at the cost of repeated state expansions.

A Search Algorithm:* Expected to provide the fastest and most efficient solution when using an admissible heuristic, with moderate to high memory usage.

Conclusion

This project demonstrates the effectiveness of different Artificial Intelligence search algorithms in solving the 8-Puzzle problem. The inclusion of Iterative Deepening Search (IDS) highlights how optimal solutions can be achieved while maintaining low memory usage. By comparing time and memory complexities, the project clearly shows the trade-offs between uninformed and informed search strategies and emphasizes why heuristic-based algorithms such as A* are preferred for complex problems.