### **Comparison of Search Algorithms for Solving the 8-Puzzle Problem**

#### **Introduction**

The 8-Puzzle problem is a classic sliding puzzle that consists of a 3x3 grid with 8 numbered tiles and one empty space. The goal is to arrange the tiles in a specific order, typically from 1 to 8, by sliding tiles into the empty space. Various search algorithms can be applied to solve this problem, and in this report, we compare the performance of four different algorithms: **Breadth-First Search (BFS)**, **Depth-First Search (DFS)**, **Uniform Cost Search (UCS)**, and **Iterative Deepening Search (IDS)**. The comparison focuses on the number of nodes visited, memory consumption, time taken, and the path cost of each algorithm.

#### **Comparison of Results**

* **BFS**:
  + BFS is an optimal algorithm that explores all possible states level by level. For this particular problem, BFS visited only 4 nodes, consumed minimal memory (4 units), and solved the puzzle very quickly with a time of 0.00015 seconds. However, BFS might struggle with memory usage in larger or more complex problems, though it was efficient in this case.
* **DFS**:
  + DFS explores the deepest nodes first, which may lead to unnecessary exploration of irrelevant branches. This algorithm visited an extremely high number of nodes—181,204—consumed significant memory (42,946 units), and took 0.638 seconds to complete the puzzle. Although it eventually reached the goal, its performance was much less efficient compared to BFS and UCS.
* **UCS**:
  + UCS is an optimal algorithm that explores nodes based on their path cost, ensuring the lowest-cost solution is found. For this specific instance, UCS visited 5 nodes, consumed minimal memory (4 units), and solved the puzzle almost instantly with a time of 0.000099 seconds. UCS achieved a path cost of 2, which is optimal for this puzzle. This makes UCS the most efficient algorithm in terms of both time and resources.
* **IDS**:
  + IDS combines the benefits of DFS and BFS by performing depth-limited searches iteratively. IDS solved the puzzle with 8 nodes visited, minimal memory usage (3 units), and a quick time of 0.000073 seconds. However, the path cost of IDS was 3, which is higher than that of UCS (2). IDS performed well in terms of memory usage but still lagged behind UCS in terms of path cost.

#### **Conclusion**

After comparing the performance of the algorithms, **Uniform Cost Search (UCS)** stands out as the best choice for solving the 8-Puzzle problem in this case. UCS achieved the goal with minimal nodes visited, optimal memory usage, and the lowest path cost (2). Additionally, it was the fastest algorithm, solving the puzzle almost instantly. While BFS is fast and efficient for small problems, UCS is more reliable for finding the optimal solution in terms of both time and cost, making it the preferred algorithm for solving the 8-Puzzle problem.

In summary, UCS offers a balance of efficiency, optimality, and low resource consumption, making it the best algorithm for this puzzle.

| Algorithm | Nodes Visited | Memory COnsumed | Time Taken (s) | Path Cost |
| --- | --- | --- | --- | --- |
| BFS | 4 | 4 | 0.00015 | 3 |
| DFS | 181,204 | 42,946 | 0.638 | 295 |
| UCS | 5 | 4 | 0.00099 | 2 |
| IDS | 8 | 3 | 0.00073 | 3 |