East West University

LAB-03

**Course Title**: Artificial Intelligence

**Course Code**: CSE366

**Section**: 02

**Submitted To**

Dr. Raihan Ul Islam

Associate Professor

Department of Computer Science & Engineering

East West University

**Submitted By**

Tanjila Akter

Student ID: 2022-1-60-078

Semester: Fall 2024

Department Of Computer Science and Engineering (CSE)

**Introduction**

We compare the performance of **Breadth-First Search (BFS), Depth-First Search (DFS)**, and **Uniform-Cost Search (UCS)** on different maze sizes (Tiny, Medium, and Big). The performance metrics include the number of nodes expanded and the execution time.

**1. Performance of BFS**

Breadth-First Search (BFS) explores the shallowest nodes first. It guarantees finding the shortest path in terms of the number of edges.

|  |  |  |
| --- | --- | --- |
| **Maze** | **Nodes Expanded** | **Execution time (μs)** |
| Tiny | 15 | 12655 |
| Medium | 269 | 16753 |
| Big | 620 | 16783 |

**Command Use:**

python pacman.py -l tinyMaze -p SearchAgent -a fn=bfs

python pacman.py -l mediumMaze -p SearchAgent -a fn=bfs

python pacman.py -l bigMaze -p SearchAgent -a fn=bfs

**2. Performance of DFS**

Depth-First Search (DFS) explores the deepest nodes first. It does not guarantee finding the shortest path but is often faster in finding a solution.

|  |  |  |
| --- | --- | --- |
| **Maze** | **Nodes Expanded** | **Execution time (μs)** |
| Tiny | 16 | 1001 |
| Medium | 270 | 5947 |
| Big | 621 | 13182 |

**Command Use:**

python pacman.py -l tinyMaze -p SearchAgent -a fn=dfs

python pacman.py -l mediumMaze -p SearchAgent -a fn=dfs

python pacman.py -l bigMaze -p SearchAgent -a fn=dfs

**3. Performance of UCS**

**Uniform-Cost Search (UCS)** expands the node with the lowest total cost first. It guarantees finding the optimal solution but can be computationally expensive.

|  |  |  |
| --- | --- | --- |
| **Maze** | **Nodes Expanded** | **Execution time (μs)** |
| Tiny | 16 | 1000 |
| Medium | 270 | 11010 |
| Big | 621 | 19119 |

**Command Use:**

python pacman.py -l tinyMaze -p SearchAgent -a fn=ucs

python pacman.py -l mediumMaze -p SearchAgent -a fn=ucs

python pacman.py -l bigMaze -p SearchAgent -a fn=ucs

**Overall Comparison**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm** | **Maze Size** | **Node Expanded** | **Execution time (μs)** | **Score** |
| BFS | Tiny | 15 | 12655 | 502 |
| DFS | Tiny | 15 | 1001 | 500 |
| UCS | Tiny | 15 | 1000 | 502 |
| BFS | Medium | 269 | 16753 | 442 |
| DFS | Medium | 146 | 5947 | 380 |
| UCS | Medium | 269 | 11010 | 442 |
| BFS | Big | 620 | 16783 | 300 |
| DFS | Big | 390 | 13182 | 300 |
| UCS | Big | 620 | 19119 | 300 |

**Maze Compare**

**Tiny Maze**

* All algorithms expand the same number of nodes (15), as the problem is straightforward and small.

 BFS and UCS are slightly faster than DFS.

 BFS and UCS achieve the highest scores (502), likely due to optimal path selection.

**Medium Maze**

* DFS explores fewer nodes compared to BFS and UCS, resulting in faster execution but lower scores.

 BFS and UCS expand the same number of nodes and have similar execution times, showing consistent behavior.

 Scores are highest for BFS and UCS, suggesting they find better-quality paths than DFS.

**Big Maze**

 BFS and UCS expand the same number of nodes, with UCS executing faster than BFS.

* DFS explores significantly fewer nodes, with faster execution, but the score remains the same for all three algorithms.
* The constant score of 300 indicates that all algorithms find a path, but the quality is equal across all approaches.

**Efficiency**

* DFS is generally faster than BFS and UCS, especially in larger mazes, because it focuses on depth and doesn't expand as many nodes.
* UCS tends to have longer execution times compared to DFS but is often faster than BFS due to prioritization of cost.

**Effectiveness**

* BFS and UCS often achieve higher scores in smaller mazes, indicating they prioritize optimal or near-optimal paths.
* In larger mazes, all algorithms achieve similar scores, but DFS's lower node expansion may be beneficial for memory efficiency.

**Suitability**

* BFS and UCS are better for finding optimal solutions, while DFS is suitable for quick explorations where optimality is not a priority.

**Conclusion**

BFS and UCS are optimal and perform similarly in terms of node exploration and scores, making them suitable for finding high-quality solutions. DFS is faster and explores fewer nodes, but it sacrifices solution quality, especially in larger mazes. The choice of algorithm depends on the trade-off between speed, memory, and solution optimality.