**Planning Problem – Heuristic Analysis**

Rob Straker

**Optimal Path Length**

For this analysis, it’s important to understand the optimal path lengths for each of the three air cargo planning problems given. Following are representative plans that use the shortest possible (optimal) paths to reach the desired goal states:

|  |  |  |
| --- | --- | --- |
| **Problem 1** | **Problem 2** | **Problem 3** |
| Load (C1, P1, SFO) | Load (C1, P1, SFO) | Load (C1, P1, SFO) |
| Fly (P1, SFO, JFK) | Fly (P1, SFO, JFK) | Fly (P1, SFO, ATL) |
| Unload (C1, P1, JFK) | Unload (C1, P1, JFK) | Load (C3, P1, ATL) |
| Load (C2, P2, JFK) | Load (C2, P2, JFK) | Fly (P1, ATL, JFK) |
| Fly (P2, JFK, SFO) | Fly (P2, JFK, SFO) | Unload (C1, P1, JFK) |
| Unload (C2, P2, SFO) | Unload (C2, P2, SFO) | Unload (C3, P1, JFK) |
|  | Load (C3, P3, ATL) | Load (C2, P2, JFK) |
|  | Fly (P3, ATL, SFO) | Fly (P2, JFK, ORD) |
|  | Unload (C3, P3, SFO) | Load (C4, P2, ORD) |
|  |  | Fly (P2, JFK, SFO) |
|  |  | Unload (C2, P2, SFO) |
|  |  | Unload (C4, P2, SFO) |
| **Plan Length = 6** | **Plan Length = 9** | **Plan Length = 12** |

**Planning Search Metrics**

The performance of the uninformed and informed planning searches is measured in terms of several metrics, two of which relate to computer memory usage (node expansions, goal tests), and two of which relate to computational speed (time elapsed, optimality).

**Metrics for Problem 1**

Performance scores are based on rankings of 1, 2 or 3 within each metric category as follows, with best search strategy shaded in green, worst in red:

**RANKING 1 2 3**

# Node Expansions <50 50-500 >500

# Goal Tests <50 50-500 >500

Time Elapsed <0.1 s 0.1-1 s >1 s

Optimality =6 7-25 >25

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Search Algorithm** | **# Node Expansions** | **# Goal Tests** | **Time Elapsed** | **Optimality** (plan length) | **SCORE** |
| 1 | Breadth First | 43 | 56 | 0.033 s | 6 | 5 |
| 2 | Breadth First Tree | 1458 | 1459 | 1.000 s | 6 | 9 |
| 3 | Depth First Graph | 21 | 22 | 0.015 s | 20 | 5 |
| 4 | Depth Limited | 101 | 271 | 0.098 s | 50 | 8 |
| 5 | Uniform Cost | 55 | 57 | 0.039 s | 6 | 6 |
| 6 | Recursive Best First | 4229 | 4230 | 2.951 s | 6 | 10 |
| 7 | Greedy Best First Graph | 7 | 9 | 0.005 s | 6 | 4 |
| 8 | A\* with h1 | 55 | 57 | 0.044 s | 6 | 6 |
| 9 | A\* with Ignore Preconditions | 41 | 43 | 0.045 s | 6 | 4 |
| 10 | A\* with Level Sum | 11 | 13 | 0.512 s | 6 | 5 |

**Metrics for Problem 2**

Performance scores are based on rankings of 1, 2 or 3 within each metric category as follows, with best search strategy shaded in green, worst in red:

**RANKING 1 2 3**

# Node Expansions <100 100-1000 >1000

# Goal Tests <100 100-1000 >1000

Time Elapsed <1 s 1-5 s >5 s

Optimality =9 10-50 >50

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Search Algorithm** | **# Node Expansions** | **# Goal Tests** | **Time Elapsed** | **Optimality** (plan length) | **SCORE** |
| 1 | Breadth First | 3343 | 4609 | 14.208 s | 9 | 10 |
| 2 | Breadth First Tree | Incomplete, as runtime exceeded 10 minutes. | | | | |
| 3 | Depth First Graph | 624 | 625 | 3.505 s | 619 | 9 |
| 4 | Depth Limited | Incomplete, as runtime exceeded 10 minutes. | | | | |
| 5 | Uniform Cost | 4853 | 4855 | 12.223 s | 9 | 10 |
| 6 | Recursive Best First | Incomplete, as runtime exceeded 10 minutes. | | | | |
| 7 | Greedy Best First Graph | 35 | 37 | 0.085 s | 15 | 5 |
| 8 | A\* with h1 | 4853 | 4855 | 13.656 s | 9 | 10 |
| 9 | A\* with Ignore Preconditions | 1428 | 1430 | 4.742 s | 9 | 9 |
| 10 | A\* with Level Sum | 114 | 116 | 50.915 s | 9 | 8 |

**Metrics for Problem 3**

Performance scores are based on rankings of 1, 2 or 3 within each metric category as follows, with best search strategy shaded in green, worst in red:

**RANKING 1 2 3**

# Node Expansions <500 500-5000 >5000

# Goal Tests <500 500-5000 >5000

Time Elapsed <10 s 10s-50s >50 s

Optimality =12 13-50 >50

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Search Algorithm** | **# Node Expansions** | **# Goal Tests** | **Time Elapsed** | **Optimality** (plan length) | **SCORE** |
| 1 | Breadth First | 14120 | 17673 | 107.968 s | 12 | 10 |
| 2 | Breadth First Tree | Incomplete, as runtime exceeded 10 minutes. | | | | |
| 3 | Depth First Graph | 292 | 293 | 1.234 s | 288 | 6 |
| 4 | Depth Limited | Incomplete, as runtime exceeded 10 minutes. | | | | |
| 5 | Uniform Cost | 16961 | 16963 | 53.390 s | 12 | 10 |
| 6 | Recursive Best First | Incomplete, as runtime exceeded 10 minutes. | | | | |
| 7 | Greedy Best First Graph | 4193 | 4195 | 13.157 s | 32 | 8 |
| 8 | A\* with h1 | 16961 | 16963 | 53.021 s | 12 | 10 |
| 9 | A\* with Ignore Preconditions | 4444 | 4446 | 16.053 s | 12 | 7 |
| 10 | A\* with Level Sum | 237 | 239 | 152.778 s | 13 | 7 |

**Performance Comparison**

**Uninformed (non-Heuristic) Searches**

Across four dimensions (# node expansions, # goal tests, time elapsed, plan length), the highest performing search algorithm was **Greedy Best First Graph Search** for problems 1 and 2, and **Depth First Search** for Problem 3. The poorest performing was **Recursive Best First Search** for Problem 1, while **Breadth First Tree Search**, **Depth Limited Search** and **Recursive Best First Search** all failed to run under the 10-minute limit for Problems 2 and 3.

**Greedy Best First Graph Search** was the fastest algorithm across Problem 1 and 2, while **Depth First Graph Search** was fastest for Problem 3.

Optimal path length was only achieved by **Breadth First Search** and **Uniform Cost Search** across all 3 problems within the 10-minute runtime limit.

The only algorithms to achieve optimal plan length for all 3 problems were **Breadth First Search** and **Uniform Cost Search**.

**Informed (Heuristic) Searches**

Across four dimensions (# node expansions, # goal tests, time elapsed, plan length), the highest performing search algorithms was **A\* Search with Ignore Preconditions** for Problems 1 and 3, and **A\* Search with Level Sum** for Problems 2 and 3. The poorest performing was **A\* Search with h1** for all Problems.

**A\* Search with Ignore Preconditions** was the fastest algorithm across Problems 2 and 3, and a close second to **A\* Search with h1** for Problem 1.

Optimal path length was achieved by all techniques for all problems, except for **A\* Search with Level Sum** which took 13 steps rather than 12 for Problem 3.

**Best Non-Heuristic and Heuristic Searches**

Following is a comparison of the ranking across all searches for only those uninformed and informed searches which achieve optimal path length across each of the 3 problems:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Search Algorithm** | **Problem 1** | | | **Problem 2** | | | **Problem 3** | | |
| **Nodes** | **Tests** | **Time** | **Nodes** | **Tests** | **Time** | **Nodes** | **Tests** | **Time** |
| 1 | Breadth First | 5 | 5 | 3 | 5 | 5 | 6 | 5 | 7 | 6 |
| 5 | Uniform Cost | 6 | 6 | 4 | 6 | 6 | 4 | 6 | 5 | 5 |
| 8 | A\* with h\_1 | 6 | 6 | 6 | 6 | 6 | 5 | 6 | 5 | 4 |
| 9 | A\* with Ignore Preconditions | 2 | 2 | 8 | 4 | 4 | 3 | 4 | 4 | 3 |

**A\* Search with Ignore Preconditions** appears to have consistently performed the best across all 3 problems. It used the least memory (fewest node expansions and goal tests) across every problem, and was fastest for all but problem 1 (8th). However, it did not perform better than all non-heuristic search methods for all problems. **Greedy Best First Search** worked best for Problem 1, while there was no single method that worked best (fewest node expansions and goal tests, and fastest) across Problems 2 and 3.

However, if optimal path length is not a consideration, then **Greedy Best First Graph Search** appears to be the most consistent performer, using the least memory across all but Problem 3 (3rd), and being the fastest for all but Problem 3 (2nd).

These results raise a few questions. For instance, why does **A\* Search with Ignore Preconditions** consistently run faster than **A\* Search with Level Sum**? The primary reason is that Level Sum is much more accurate than the Ignore Preconditions heuristic[[1]](#endnote-1). Russell and Norvig point out that Ignore Preconditions adds more edges to the graph, making it easier to find a path to the goal.

Also, why does **Depth First Graph Search** fail to find an optimal path? Depth First Search always expands the deepest node, although it is complete because it will eventually expand every node.i This approach will be suboptimal, as it will lead to longer paths if goal nodes are discovered deep in one area of the graph, regardless of whether a shorter path to another goal node is available elsewhere.

1. Stuart Russell and Peter Norvig, *Artificial Intelligence: A Modern Approach, 3/e* (New York: Prentice Hall, 2011), Kindle Edition, Chapter 10, Bibliographical and Historical Notes. [↑](#endnote-ref-1)