# Heuristic Analysis of Air Cargo Planning Problem

Yanfei Wu

This project implemented progression search algorithms to solve deterministic logistics planning problems for an Air Cargo transport system using a planning search agent. The search result metrics for uninformed non-heuristic search (breadth-first, depth-first, etc.) and domain-independent heuristic search (A* search with “ignore preconditions” and “level sum” heuristics) are compared in terms of optimality, time elapsed, number of node expansions for three particular air cargo planning problems. The comparison results are summarized and optimal plans are provided in this report.

### The Problems

The three air cargo planning problems the search algorithms are tested on are listed below.

Problem 1:

```
Init(At(C1, SFO) ∧ At(C2, JFK)
	∧ At(P1, SFO) ∧ At(P2, JFK)
	∧ Cargo(C1) ∧ Cargo(C2)
	∧ Plane(P1) ∧ Plane(P2)
	∧ Airport(JFK) ∧ Airport(SFO))
Goal(At(C1, JFK) ∧ At(C2, SFO))
```

Problem 2:

```
Init(At(C1, SFO) ∧ At(C2, JFK) ∧ At(C3, ATL)
	∧ At(P1, SFO) ∧ At(P2, JFK) ∧ At(P3, ATL)
	∧ Cargo(C1) ∧ Cargo(C2) ∧ Cargo(C3)
	∧ Plane(P1) ∧ Plane(P2) ∧ Plane(P3)
	∧ Airport(JFK) ∧ Airport(SFO) ∧ Airport(ATL))
Goal(At(C1, JFK) ∧ At(C2, SFO) ∧ At(C3, SFO))
```

Problem 3:

```
Init(At(C1, SFO) ∧ At(C2, JFK) ∧ At(C3, ATL) ∧ At(C4, ORD)
	∧ At(P1, SFO) ∧ At(P2, JFK)
	∧ Cargo(C1) ∧ Cargo(C2) ∧ Cargo(C3) ∧ Cargo(C4)
	∧ Plane(P1) ∧ Plane(P2)
	∧ Airport(JFK) ∧ Airport(SFO) ∧ Airport(ATL) ∧ Airport(ORD))
Goal(At(C1, JFK) ∧ At(C3, JFK) ∧ At(C2, SFO) ∧ At(C4, SFO))
```

### The Result Metrics

The result metrics for non-heuristic search and heuristic search for problem 1, 2, and 3 are summarized in Table 1, 2, and 3, respectively. The optimal plans are highlighted in green.

For problem 1, as shown in Table 1, except for two non-heuristic search algorithms (depth first graph search and depth limited search) which did not find the optimal solution, all the rest search algorithms including six non-heuristic search algorithms (breadth first search, breadth first tree search, uniform cost search, recursive best first search, greedy best first graph search, and A* search), and two heuristic search algorithms (A* search with two custom heuristic functions, i.e., "ignore preconditions" and "level sum"), found the solution in an optimal way with reasonable speed. Recursive best first search and breadth first tree search had significantly more node expansions compared to the rest, and recursive best first search is also the slowest amongst all. ***Overall, the non-heuristic search function, greedy best first graph search, showed the best performance when solving problem 1.***

**Table 1** Result metrics of non-heuristic and heuristic search for air cargo planning problem 1.

| Search Function | Expansions | Goal Tests | New Nodes | Plan Length | Search Time (s) | Optimal |
|---|---|---|---|---|---|---|
| **Breadth First Search** | 43 | 56 | 180 | 6 | 0.030 | Yes |
| **Breadth First Tree Search** | 1458 | 1459 | 5960 | 6 | 0.902 | Yes |
| **Depth First Graph Search** | 21 | 22 | 84 | 20 | 0.015 | No |
| **Depth Limited Search** | 101 | 271 | 414 | 50 | 0.090 | No |
| **Uniform Cost Search** | 55 | 57 | 224 | 6 | 0.035 | Yes |
| **Recursive Best First Search** | 4229 | 4230 | 17023 | 6 | 2.63 | Yes |
| **Greedy Best First Graph Search** | **7** | **9** | **28** | **6** | **0.005** | **Yes** |
| **A* Search** | 55 | 57 | 224 | 6 | 0.037 | Yes |
| **A* Search h_ignore_preconditions** | 41 | 43 | 170 | 6 | 0.038 | Yes |
| **A* Search h_pg_levelsum** | 11 | 13 | 50 | 6 | 0.918 | Yes |

For problem 2, as shown in Table 2, the metrics for breadth first tree search, depth limited search, and recursive best first search were not recorded because the run time exceeded 10 minutes. Among the non-heuristic search algorithms, breadth first search, uniform cost search, and A* search found the optimal solution with similar number of node expansions and search time, whereas depth first graph search and greedy best first graph search did not find the solution in an optimal way. The two heuristic search algorithms also found the optimal solution but with much fewer node expansions. A* search with “level sum” had the least node expansions (about 1/20 of that of A* search with “ignore preconditions” heuristic). However, it took significantly longer than the rest of the algorithms (over 30 times more than “ignore preconditions”). ***Overall, the heuristic search function, A* search with “ignore preconditions” heuristic, had the best performance for problem 2, especially when speed is more of a concern than memory.***

**Table 2** Result metrics of non-heuristic and heuristic search for air cargo planning problem 2.

| Search Function | Expansions | Goal Tests | New Nodes | Plan Length | Search Time (s) | Optimal |
|---|---|---|---|---|---|---|
| **Breadth First Search** | 3343 | 4609 | 30509 | 9 | 13.066 | Yes |
| **Breadth First Tree Search** | - | - | - | - | - | - |
| **Depth First Graph Search** | 624 | 625 | 5602 | 619 | 3.256 | No |
| **Depth Limited Search** | - | - | - | - | - | - |
| **Uniform Cost Search** | 4853 | 4855 | 44041 | 9 | 10.953 | Yes |
| **Recursive Best First Search** | - | - | - | - | - | - |
| **Greedy Best First Graph Search** | 998 | 1000 | 8982 | 21 | 2.231 | No |
| **A* Search** | 4853 | 4855 | 44041 | 9 | 11.346 | Yes |
| **A* Search h_ignore_preconditions** | **1450** | **1452** | **13303** | **9** | **4.033** | **Yes** |
| **A* Search h_pg_levelsum** | 86 | 88 | 841 | 9 | 144.655 | Yes |

*Note*: Metrics of Breadth First Tree Search, Depth Limited Search, and Recursive Best First Search with h_1 were not recorded because the search time exceeded 10 minutes.

The result metrics for problem 3 are shown in Table 3. Again, the search time with breadth first tree search, depth limited search, and recursive best first search exceeded 10 minutes and their result metrics were not recorded. Similar to the results for problem 2, the three non-heuristic search algorithms, breadth first search, uniform cost search, and A* search found the optimal solution with similar number of node expansions and search time, while depth first search and greedy best first graph search did not achieve optimality. The two heuristic search algorithms also found the optimal solution but with much fewer node expansions. A* search with “level sum” heuristic had the least node expansions (1/15 of that of A* search with "ignore preconditions") but it was significantly slower than all other search algorithms (50 times slower than A* search with "ignore preconditions"). ***Overall, the heuristic search function, A* search with "ignore preconditions" heuristic, had the best performance in solving problem 3, especially if speed is concerned more.***

**Table 3** Result metrics of non-heuristic and heuristic search for air cargo planning problem 3.

| Search Function | Expansions | Goal Tests | New Nodes | Plan Length | Search Time (s) | Optimal |
|---|---|---|---|---|---|---|
| **Breadth First Search** | 14663 | 18098 | 129631 | 12 | 93.995 | Yes |
| **Breadth First Tree Search** | - | - | - | - | - | - |
| **Depth First Graph Search** | 408 | 409 | 3364 | 392 | 1.618 | No |
| **Depth Limited Search** | - | - | - | - | - | - |
| **Uniform Cost Search** | 18223 | 18225 | 159618 | 12 | 49.317 | Yes |
| **Recursive Best First Search** | - | - | - | - | - | - |
| **Greedy Best First Graph Search** | 5578 | 5580 | 49150 | 22 | 14.553 | No |
| **A* Search** | 18223 | 18225 | 159618 | 12 | 50.333 | Yes |
| **A* Search h_ignore_preconditions** | **5040** | **5042** | **44944** | **12** | **15.342** | **Yes** |
| **A* Search h_pg_levelsum** | 325 | 327 | 3002 | 12 | 777.921 | Yes |

*Note*: Metrics of Breadth First Tree Search, Depth Limited Search, and Recursive Best First Search with h_1 were not recorded because the search time exceeded 10 minutes.

**Table 4** Optimal plans for air cargo problem 1, 2, and 3.

| Problem 1 | Problem 2 | Problem 3 |
|---|---|---|
| `Load(C1, P1, SFO)`<br>`Load(C2, P2, JFK)`<br>`Fly(P1, SFO, JFK)`<br>`Fly(P2, JFK, SFO)`<br>`Unload(C1, P1, JFK)`<br>`Unload(C2, P2, SFO)` | `Load(C3, P3, ATL)`<br>`Fly(P3, ATL, SFO)`<br>`Unload(C3, P3, SFO)`<br>`Load(C2, P2, JFK)`<br>`Fly(P2, JFK, SFO)`<br>`Unload(C2, P2, SFO)`<br>`Load(C1, P1, SFO)`<br>`Fly(P1, SFO, JFK)`<br>`Unload(C1, P1, JFK)` | `Load(C2, P2, JFK)`<br>`Fly(P2, JFK, ORD)`<br>`Load(C4, P2, ORD)`<br>`Fly(P2, ORD, SFO)`<br>`Unload(C4, P2, SFO)`<br>`Load(C1, P1, SFO)`<br>`Fly(P1, SFO, ATL)`<br>`Load(C3, P1, ATL)`<br>`Fly(P1, ATL, JFK)`<br>`Unload(C3, P1, JFK)`<br>`Unload(C2, P2, SFO)`<br>`Unload(C1, P1, JFK)` |

**Summary**

To summarize, among the ten search functions, three non-heuristic search functions (breadth first search, uniform cost search, and A* search) and two heuristic search functions (A* search with “ignore preconditions” and “level sum” heuristics) all found the optimal solution for the three air cargo problems. Based on the comparisons of optimality, search time, number of node expansions, greedy best first graph search is found to be the best for problem 1 and A* with “ignore preconditions” heuristic is found to be the best for both problem 2 and 3. The optimal plans for these three problems obtained from the best search function for each problem are shown in Table 4.

The results of this project also suggest that as the complexity of the problem increases, the advantages of using heuristic search algorithms also become more significant. For simpler problems, however, heuristic based search does not necessarily outperform non-heuristic based search algorithms, as demonstrated in air cargo problem 1 in which greedy best first graph search was actually the best.

In addition to advantages of optimality, speed, and memory compared to non-heuristic search, heuristic search also affords the flexibility to adjust the balance between search speed and node expansions by customizing heuristic functions. For example, in this case, the two custom heuristics, “ignore preconditions” and “level sum”, both resulted in A* search to find an optimal solution for an problem, but A* search with “ignore preconditions” heuristic won in speed whereas A* search with “level sum” won in node expansions.