Air Cargo Planning Heuristic Analysis
Eduardo Carrasco Jr

This review summarizes the metrics for three searches, and three A* heuristic searches. I try to compare and contrast the different solutions for three air cargo problems of increasing complexity and look at the benefits of a heuristic based solution for complex problem.

Type	Expansion	Goal Test	New Nodes	Plan Length	Time (seconds)
Breath-First	43	56	180	6	0.060
Depth-first Graph	12	13	48	12	0.017
Breath-First-Tree-Search	1458	1459	5960	6	1.945
Uniform Cost Search	55	57	224	6	0.078
A* h_1	55	57	224	6	0.075
A* h_ignore_preconditions	41	43	170	6	0.061
A* h_pg_levelsum	11	13	50	6	0.924

Figure 1: Summary of Problem 1

For first problem, I had two airports, two planes and two cargo units.
For a simple problem like this, we see that all searches perform reasonable. The A* search with the planning graph performs efficiently in terms of expansions and goal tests, but takes 10x more computation time to complete. The best solution for simpler problems is therefore just a breath-first-search.

Type	Expansion	Goal Test	New Nodes	Plan Length	Time (seconds)
Breath-First	3343	4609	30509	9	25.047
Depth-first Graph	582	583	5211	575	5.007
Uniform Cost Search	4852	4854	44030	9	28.435
A* h_1	4852	4854	44030	9	29.669
A* h_ignore_preconditions	1450	1452	13303	9	8.657
A* h_pg_levelsum	86	88	841	9	151.339

Figure 2: Summary table for problem 2

For the second problem, we have three airports, three planes, and three cargo units. Because of the added complexity of the problem, we now see the limitations of a breath-first-search. Depth-first search is still good in terms of completion time, but doesn’t come anywhere near to an optimal solution. However, we see that a planning graph heuristic is still much slower, though we start to see benefits of a heuristic-based planning approach. The best solution here is the A* search with the ignore-preconditions heuristic. It should be noted that in the first problem I tried 7 algorithms, in this one I couldn’t include Breath-First-Tree-Search due to the algorithm never having finished. This assignment did for warn me that this issue may come up.

Air Cargo Planning Heuristic Analysis
Eduardo Carrasco Jr

Type	Expansion	Goal Test	New Nodes	Plan Length	Time (seconds)
Breath-First	14663	18098	129631	12	152.080
Depth-first Graph	627	628	5176	596	5.195
Uniform Cost Search	18235	18237	159716	12	124.885
A* h_1	18235	18237	159716	12	120.736
A* h_ignore_preconditions	5040	5042	44944	12	37.295
A* h_pg_levelsum	318	320	2934	12	790.300

For the third problem, I had a much harder problem to tackle than the previous two. We have four airports, two planes, and four cargo units. This is a exponentially more complex problem, and we see that breath-first search is no longer optimal. Depth-first Graph search still finds a solution very quickly, but is nowhere near optimal again. The planning heuristic ran for more than 60 minutes I had to shut down the program for overheating my mac book pro. The best solution here is the A* search with the ignore-preconditions heuristic.

Optimal Solutions:

The following table summarizes the optimal solutions for each problem, given that we are looking at both Plan Length and Time as consideration for best solution:

Problem	Time (seconds)	Best Search Type	Action Sequence
Problem 1	0.060	Breath-First	1. Load(C2, P2, JFK) 2. Load(C1, P1, SFO) 3. Fly(P2, JFK, SFO) 4. Unload(C2, P2, SFO) 5. Fly (P1, SFO, JFK) 6. Unload(C1, P1, JFK)
Problem 2	8.657	A* -h_ignore_preconditions	1. Load(C3, P3, ATL) 2. Fly(P3, ATL, SFO) 3. Unload(C3, P3, SFO) 4. Load(C1, P1, SFO) 5. Fly(P1, SFO, JFK) 6. Unload(C1, P1, JFK) 7. Load(C2, P2, JFK) 8. Fly(P2, JFK, SFO) 9. Unload(C2, P2, SFO)
Problem 3	37.295	A* -h_ignore_preconditions	1. Load(C2, P2, JFK) 2. Fly(P2, JFK, ORD) 3. Load(C4, P2, ORD) 4. Fly(P2, ORD, SFO) 5. Unload(C4, P2, SFO) 6. Load(C1, P1, SFO) 7. Fly(P1, SFO, ATL) 8. Load(C3, P1, ATL) 9. Fly(P1, ATL, JFK) 10. Unload(C3, P1, JFK) 11. Unload(C1, P1, JFK) 12. Unload(C2, P2, SFO)

Conclusion:

Air Cargo Planning Heuristic Analysis
Eduardo Carrasco Jr

Breath First Search found an optimal solution more quickly than the A* searches for problem 1, while A* search with the ignore preconditions heuristic found an optimal solution the quickest for problems 2 and 3. This suggests that an uninformed search can be more efficient for simple problems. As complexity increases, informed searches tend to outperform, and their higher computational costs start to pay off.

With regards to memory requirements, informed searches consistently expanded fewer nodes than uninformed searches, reflecting the value of the heuristics applied.

For simple problems, BFS is a good choice, while for more complex problems, A*search with the ignore preconditions heuristic is preferable. If minimizing memory requirement is the absolute priority and compute power is abundant, A* with the level sum heuristic would be preferable across all three problems.

Overall, we see that A* -h_ignore_preconditions is good for medium to large complex problems. It can an optimal solution, without an excessive running time and still got second place for the first problem. As the complexity ramps up, Depth-First Graph is the fastest for nearly all the problems but the Plan Length is too large complex as the difficulty of the problem grows. Extrapolating from the data, we may assume that the ignore-preconditions heuristic would perform adequately for medium to high complexity problems.

Reference:

1. Stuart Russell, Peter Norvig (2014), Artificial Intelligence: A Modern Approach (Third Edition)