Marifel Barbasa
Udacity AISND March 2018 Cohort
June 10, 2018

Heuristic Analysis on Planning Search within the Air Cargo Problem Domain

Applied to three air cargo problems, this report documents metrics and comparisons of five planning searches, namely breadth-first search, depth-first graph search, uniform-cost search, A* search with the ignore preconditions heuristic, and A* search with the level sum heuristic. All searches returned optimal plans except for depth-first graph search, which returned a much longer list of actions for each problem.

The problem statement in Air Cargo Problem 1 starts with C1 at SFO and C2 at JFK, with a separate plane at each airport. The optimal plan to reach the goal of C1 at JFK and C2 at SFO consists of the following actions:

Load(C1, P1, SFO)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)
Load(C2, P2, JFK)
Fly(P2, JFK, SFO)
Unload(C2, P2, SFO).

Air Cargo Problem 2 starts with C1 at SFO, C2 at JFK, and C3 at ATL, with a separate plane at each airport. The optimal plan to reach the goal of C1 at JFK, C2 at SFO, and C3 at SFO is as follows:

Load(C3, P3, ATL)
Fly(P3, ATL, SFO)
Unload(C3, P3, SFO)
Load(C2, P2, JFK)
Fly(P2, JFK, SFO)
Unload(C2, P2, SFO)
Load(C1, P1, SFO)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK).

Air Cargo Problem 3 starts with C1 at SFO, C2 at JFK, C3 at ATL, and C4 at ORD, but this time, there are only two planes: P1 at SFO and P2 at JFK. The optimal plan to reach the goal of C1 at JFK, C2 at SFO, C3 at JFK, and C4 at SFO is as follows:

Load(C2, P2, JFK)

Fly(P2, JFK, ORD)
Load(C4, P2, ORD)
Fly(P2, ORD, SFO)
Unload(C4, P2, SFO)
Load(C1, P1, SFO)
Fly(P1, SFO, ATL)
Load(C3, P1, ATL)
Fly(P1, ATL, JFK)
Unload(C3, P1, JFK)
Unload(C2, P2, SFO)
Unload(C1, P1, JFK).

The uninformed non-heuristic search algorithms tested on the problems include breadth-first search, uniform-cost search, and depth-first graph search. The first two provided optimal solutions for all three air cargo problems; depth-first graph search, however, performed the worst out of any algorithm (uninformed or informed) by providing terribly lengthy plans (in which unnecessary flights are made): 12 actions in Problem 1, 466 actions in Problem 2, and 1442 actions in Problem 3. Note that in the following problem results, the number of goal tests in a uniform-cost search was always 2 more than the number of node expansions; in addition, depth-first graph search always had 1 more goal test than node expansions. Therefore, only for breadth-first search will the number of goal tests be explicitly stated.

For Problem 1, breadth-first search and uniform-cost search were nearly even with 43 node expansions and 56 goal tests in 0.0577 seconds for breadth-first search, and 55 node expansions in 0.0701 seconds for uniform-cost search; both had optimal plans of length 6. Depth-first graph search, on the other hand, completed the fastest at 12 node expansions in 0.0166 seconds, but with a plan that was twice as long as the optimal plan.

For Problem 2, breadth-first search had 3343 node expansions and 4609 goal tests in 14.8997 seconds, while uniform-cost search took a few seconds longer to run with 4852 node expansions in 21.8292 seconds; both had optimal plans of length 9. While depth-first graph search completed the fastest at 476 node expansions in 3.8674 seconds, it produced a plan that was about 52 times longer than the optimal plan.

For Problem 3, breadth-first search had 14663 node expansions and 18098 goal tests in 81.0996 seconds, while uniform-cost search took about 22 seconds longer to run with 18235 node expansions in 103.1128 seconds; both had optimal plans of length 12. Despite depth-first graph search completing the fastest at 1511 node expansions in 25.1027 seconds, it produced a plan that was about 120 times longer than the optimal plan.

The A* search algorithms utilized the ignore preconditions and level sum heuristics, providing optimal solutions for all three air cargo problems. The runtime of A* search with the ignore preconditions heuristic was comparable to that of depth-first graph search, executing faster than breadth-first search and uniform-cost search. In contrast, A* search with the level sum heuristic always executed 4 times or more longer than the next slowest algorithm, uniform-cost search. Note that in the following problem results, the number of goal tests in A* search with these heuristics was always 2 more than the number of node expansions; therefore, the number of goal tests won't be explicitly stated.

For Problem 1, A* search with the ignore preconditions heuristic had 41 node expansions in 0.0542 seconds; therefore, the Problem 1 time elapsed was on par with breadth-first search. In comparison, A* search with the level sum heuristic executed the slowest at 11 node expansions in 0.9954 seconds.

For Problem 2, A* search with the ignore preconditions heuristic had 1450 node expansions in 6.3857 seconds, making it the second fastest of the five algorithms. In comparison, A* search with the level sum heuristic executed the slowest at 86 node expansions in 88.4016 seconds.

For Problem 3, A* search with the ignore preconditions heuristic had 5040 node expansions in 25.1043 seconds, on par with the Problem 3 runtime for depth-first graph search; both executed the fastest. In comparison, A* search with the level sum heuristic had 325 node expansions in 444.3445 seconds, making it the slowest of the five algorithms.

Overall, for all three air cargo problems, A* search with the level sum heuristic had the worst runtime, followed by uniform-cost search, breadth-first search, A* search with the ignore preconditions heuristic, and depth-first graph search. While depth-first graph search executed much faster than A* search with the ignore preconditions heuristic for Problems 1 and 2, in Problem 3, their runtimes were extremely close, with only 0.0016 seconds difference.

To conclude, the best heuristic used in these air cargo problems is the ignore preconditions heuristic because, when coupled with A* search, despite the complexity of the problem, it would execute the fastest while also providing an optimal solution. Depth-first graph search, on the other hand, would provide extremely suboptimal solutions, even if it's technically (only slightly) faster on paper; this is supported by the Lesson 9 videos on search, in that depth-first search is not guaranteed to find the shortest path and therefore not optimal, unlike breadth-first search and uniform-cost (or cheapest-first) search. The ignore preconditions heuristic was better than non-heuristic search planning methods for all problems because it

produced optimal results, unlike depth-first graph search; and, in comparison to breadth-first search and uniform-cost search, A* search with the ignore preconditions heuristic also expanded less nodes and made less goal tests, all within a shorter amount of execution time. In addition, based on the Lesson 9 videos on search, A* finds the lowest cost path if the heuristic function used is less than the true cost to the goal. Indeed, the ignore preconditions heuristic is always less than the true cost to the goal because, in the air cargo problem scenario, the minimum number of actions to achieve the end goal state (say, **Goal(At(C1, JFK) ⋀ At(C2, SFO))**) is simply the number of unmet goals (2 in the example), since for the problem scenario, there can only be one action to achieve a single goal. Therefore, A* search with the ignore preconditions heuristic is guaranteed to find an optimal solution.

TABLE 1: Air Cargo Problem 1 Search Algorithm Runtime Results

Test #	Time Elapsed (s)	Expansions	Goal Tests	New Nodes	Plan Length	Plan
	Search function #1: **breadth_first_search**					
1	0.05770876398310065	43	56	180	6	Same as Best Plan
2	0.05314574996009469	43	56	180	6	Same as Best Plan
3	0.06967585999518633	43	56	180	6	Same as Best Plan
Median	0.05770876398310065	43	56	180	6	**Best Plan:** Load(C1, P1, SFO) Load(C2, P2, JFK) Fly(P2, JFK, SFO) Unload(C2, P2, SFO) Fly(P1, SFO, JFK) Unload(C1, P1, JFK)
	Search function #3: **depth_first_graph_search**					
4	0.016622414928860962	12	13	48	12	Same as Best Plan
5	0.012734821997582912	12	13	48	12	Same as Best Plan
6	0.019051784998737276	12	13	48	12	Same as Best Plan
Median	0.016622414928860962	12	13	48	12	**Best Plan:** Fly(P1, SFO, JFK) Fly(P2, JFK, SFO) Load(C1, P2, SFO) Fly(P2, SFO, JFK) Fly(P1, JFK, SFO) Unload(C1, P2, JFK) Fly(P2, JFK, SFO) Fly(P1, SFO, JFK) Load(C2, P1, JFK) Fly(P2, SFO, JFK) Fly(P1, JFK, SFO) Unload(C2, P1, SFO)

	Search function #5: **uniform_cost_search**					
7	0.06219800398685038	55	57	224	6	Same as Best Plan
8	0.07012329809367657	55	57	224	6	Same as Best Plan
9	0.07834975700825453	55	57	224	6	Same as Best Plan
Median	0.07012329809367657	55	57	224	6	**Best Plan:** Load(C1, P1, SFO) Load(C2, P2, JFK) Fly(P1, SFO, JFK) Fly(P2, JFK, SFO) Unload(C1, P1, JFK) Unload(C2, P2, SFO)
	Search function #9: **astar_search** **h_ignore_preconditions**					
10	0.048515827977098525	41	43	170	6	Same as Best Plan
11	0.054196559940464795	41	43	170	6	Same as Best Plan
12	0.05821694293990731	41	43	170	6	Same as Best Plan
Median	0.054196559940464795	41	43	170	6	**Best Plan:** Load(C1, P1, SFO) Fly(P1, SFO, JFK) Unload(C1, P1, JFK) Load(C2, P2, JFK) Fly(P2, JFK, SFO) Unload(C2, P2, SFO)
	Search function #10: **astar_search** **h_pg_levelsum**					
13	1.0025344559689984	11	13	50	6	Same as Best Plan
14	0.9933682800037786	11	13	50	6	Same as Best Plan
15	0.9953886610455811	11	13	50	6	Same as Best Plan
Median	0.9953886610455811	11	13	50	6	**Best Plan:** Load(C1, P1, SFO) Fly(P1, SFO, JFK) Load(C2, P2, JFK) Fly(P2, JFK, SFO) Unload(C1, P1, JFK) Unload(C2, P2, SFO)
	END Table 1: Problem 1					

TABLE 2: Air Cargo Problem 2 Search Algorithm Runtime Results

Test #	Time Elapsed (s)	Expansions	Goal Tests	New Nodes	Plan Length	Plan

	Search function #1: **breadth_first_search**					
16	14.851015386055224	3343	4609	30509	9	Same as Best Plan
17	15.06198063492775	3343	4609	30509	9	Same as Best Plan
18	14.899720577988774	3343	4609	30509	9	Same as Best Plan
Median	14.899720577988774	3343	4609	30509	9	**Best Plan:** Load(C1, P1, SFO) Load(C2, P2, JFK) Load(C3, P3, ATL) Fly(P2, JFK, SFO) Unload(C2, P2, SFO) Fly(P1, SFO, JFK) Unload(C1, P1, JFK) Fly(P3, ATL, SFO) Unload(C3, P3, SFO)
	Search function #3: **depth_first_graph_search**					
19	3.867381423013285	476	477	4253	466	Same as Best Plan
20	3.8291674189968035	476	477	4253	466	Same as Best Plan
21	4.030065299011767	476	477	4253	466	Same as Best Plan
Median	3.867381423013285	476	477	4253	466	**Best Plan:** Too long to list
	Search function #5: **uniform_cost_search**					
22	21.404514878056943	4852	4854	44030	9	Same as Best Plan
23	22.679753493051976	4852	4854	44030	9	Same as Best Plan
24	21.829216588987038	4852	4854	44030	9	Same as Best Plan
Median	21.829216588987038	4852	4854	44030	9	**Best Plan:** Load(C1, P1, SFO) Load(C2, P2, JFK) Load(C3, P3, ATL) Fly(P1, SFO, JFK) Fly(P2, JFK, SFO) Fly(P3, ATL, SFO) Unload(C3, P3, SFO) Unload(C2, P2, SFO) Unload(C1, P1, JFK)
	Search function #9: **astar_search** **h_ignore_preconditions**					
25	7.420393225038424	1450	1452	13303	9	Same as Best Plan
26	6.385655596968718	1450	1452	13303	9	Same as Best Plan
27	6.203512119012885	1450	1452	13303	9	Same as Best Plan

Median	6.385655596968718	1450	1452	13303	9	**Best Plan:** Load(C3, P3, ATL) Fly(P3, ATL, SFO) Unload(C3, P3, SFO) Load(C2, P2, JFK) Fly(P2, JFK, SFO) Unload(C2, P2, SFO) Load(C1, P1, SFO) Fly(P1, SFO, JFK) Unload(C1, P1, JFK)
	Search function #10: **astar_search h_pg_levelsum**					
28	87.75279576610774	86	88	841	9	Same as Best Plan
29	88.4016493849922	86	88	841	9	Same as Best Plan
30	90.73020802694373	86	88	841	9	Same as Best Plan
Median	88.4016493849922	86	88	841	9	**Best Plan:** Load(C1, P1, SFO) Fly(P1, SFO, JFK) Load(C2, P2, JFK) Fly(P2, JFK, SFO) Load(C3, P3, ATL) Fly(P3, ATL, SFO) Unload(C3, P3, SFO) Unload(C2, P2, SFO) Unload(C1, P1, JFK)
	END Table 2: Problem 2					

TABLE 3: Air Cargo Problem 3 Search Algorithm Runtime Results

Test #	Time Elapsed (s)	Expansions	Goal Tests	New Nodes	Plan Length	Plan
	Search function #1: **breadth_first_search**					
31	76.53411770495586	14663	18098	129631	12	Same as Best Plan
32	84.59821622609161	14663	18098	129631	12	Same as Best Plan
33	81.09961764595937	14663	18098	129631	12	Same as Best Plan
Median	81.09961764595937	14663	18098	129631	12	**Best Plan:** Load(C1, P1, SFO) Load(C2, P2, JFK) Fly(P2, JFK, ORD) Load(C4, P2, ORD) Fly(P1, SFO, ATL) Load(C3, P1, ATL) Fly(P1, ATL, JFK) Unload(C1, P1, JFK) Unload(C3, P1, JFK) Fly(P2, ORD, SFO) Unload(C2, P2, SFO)

						Unload(C4, P2, SFO)
	Search function #3: **depth_first_graph_search**					
34	22.16826938802842	1511	1512	12611	1442	Same as Best Plan
35	25.10273520508781	1511	1512	12611	1442	Same as Best Plan
36	25.40063795109745	1511	1512	12611	1442	Same as Best Plan
Median	25.10273520508781	1511	1512	12611	1442	**Best Plan:** Too long to list
	Search function #5: **uniform_cost_search**					
37	99.93718774802983	18235	18237	159716	12	Same as Best Plan
38	110.02943658898585	18235	18237	159716	12	Same as Best Plan
39	103.11278278601822	18235	18237	159716	12	Same as Best Plan
Median	103.11278278601822	18235	18237	159716	12	**Best Plan:** Load(C1, P1, SFO) Load(C2, P2, JFK) Fly(P1, SFO, ATL) Load(C3, P1, ATL) Fly(P2, JFK, ORD) Load(C4, P2, ORD) Fly(P2, ORD, SFO) Fly(P1, ATL, JFK) Unload(C4, P2, SFO) Unload(C3, P1, JFK) Unload(C2, P2, SFO) Unload(C1, P1, JFK)
	Search function #9: **astar_search** **h_ignore_preconditions**					
40	25.104293714975938	5040	5042	44944	12	Same as Best Plan
41	24.796282541006804	5040	5042	44944	12	Same as Best Plan
42	25.107395399012603	5040	5042	44944	12	Same as Best Plan
Median	25.104293714975938	5040	5042	44944	12	**Best Plan:** Load(C2, P2, JFK) Fly(P2, JFK, ORD) Load(C4, P2, ORD) Fly(P2, ORD, SFO) Unload(C4, P2, SFO) Load(C1, P1, SFO) Fly(P1, SFO, ATL) Load(C3, P1, ATL) Fly(P1, ATL, JFK) Unload(C3, P1, JFK) Unload(C2, P2, SFO) Unload(C1, P1, JFK)
	Search function #10:					

	astar_search h_pg_levelsum					
43	444.3445094289491	325	327	3002	12	Same as Best Plan
44	442.0653489260003	325	327	3002	12	Same as Best Plan
45	540.9295963549521	325	327	3002	12	Same as Best Plan
Median	444.3445094289491	325	327	3002	12	**Best Plan:** Load(C2, P2, JFK) Fly(P2, JFK, ORD) Load(C4, P2, ORD) Fly(P2, ORD, SFO) Load(C1, P1, SFO) Fly(P1, SFO, ATL) Load(C3, P1, ATL) Fly(P1, ATL, JFK) Unload(C4, P2, SFO) Unload(C3, P1, JFK) Unload(C2, P2, SFO) Unload(C1, P1, JFK)
	END Table 3: Problem 3					