

Heuristics Analysis

Here are the statistics for the various Plan search implementations:

Air Cargo Problem 1:

Search Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time
breadth_first_search	43	56	180	6	0.02661
depth_first_graph_search	21	22	84	20	0.01244
uniform_cost_search	55	57	224	6	0.06125
recursive_best_first_search h_1	4229	4230	17023	6	4.1631
greedy_best_first_graph_se arch h_1	7	9	28	6	0.00689
astar_search h_1	55	57	224	6	0.133
astar_search h_ignore_preconditions	41	43	170	6	0.0620
astar_search h_pg_levelsum	11	13	50	6	11.8528

Optimal Length : 6

Preferred optimal 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)

Air Cargo Problem 2:

Search Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time
breadth_first_search	3343	4609	30509	9	45.1548
depth_first_graph_search	624	625	5602	619	11.66935
uniform_cost_search	4852	4854	44030	9	132.0314
recursive_best_first_search h_1	*	*	*	*	*
greedy_best_first_graph_search h_1	990	992	8920	17	27.7241
astar_search h_1	4852	4854	44030	9	146.3674
astar_search h_ignore_preconditions	1506	1508	13820	9	49.48725
astar_search h_pg_levelsum	86	88	841	9	475.7886

* too long to fit in

Optimal Plan Length: 9

Optimal 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(C1, P1, JFK)

Unload(C2, P2, SFO)

Air Cargo Problem 3:

Search Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time
breadth_first_search	14663	18098	129631	12	240.1591
depth_first_graph_search	408	409	3364	392	3.8626
uniform_cost_search	18235	18237	159716	12	1091.995
recursive_best_first_search h_1	*	*	*	*	*
greedy_best_first_graph_search h_1	5614	5616	49429	22	253.84279
astar_search h_1	18235	18237	159716	12	1115.49439
astar_search h_ignore_preconditions	5118	5120	45650	12	260.283
astar_search h_pg_levelsum	408	410	3758	12	2374.5656

* too long to fit in

Optimal Plan Length: 12

Optimal 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(C1, P1, JFK)

Unload(C2, P2, SFO)

Inference:

Interference from above search run results:

Breadth first search: It provides optimal plan. But consumes more time. As the complexity of the problem increases it is not feasible since it gets quite slow.

Depth first Search graph: As per the findings it produces higher plan length than optimal, even though it produces results in less amount of time and not optimal at all and it's not guaranteed to produce shortest path

Uniform Cost search and A* Search with h1: Here we see that both have same number of expanded nodes, goal test and nodes are created further they use almost same amount of time to generate results. According to AIMA code - Uniform cost search uses `best_first_graph_search` and A* search is best-first graph search with $f(n) = g(n) + h(n)$. ($h(1)$ in this case). Even though, both produce optimal plan, in terms of node expansion and time consumed, it performs poorly compared to Breadth first search. This is mainly because once goal is found in breadth first search it doesn't expand unlike uniform cost search plan

Recursive best first search h1: Very slow, to generate result. Even though it produces ideal plan it's very time consuming, it's because of recursion of expanding same node over again.

Greedy best first search with h1: It doesn't return optimal plan. Even though it provides result with less expansion than any of the above searches in less time. For example for problem 2 it produces a plan with length 17 instead of optimal 9. Greedy best-first search is accomplished by specifying $f(n) = h(n)$. (`best_first_graph_search`) **A* with h_ignore precondition and h_pg_levelsum:**

`h_ignore precondition`, heuristic estimates the minimum number of actions that must be carried out from the current state in order to satisfy all of the goal conditions by ignoring the preconditions required for an action to be executed. `h_pg_levelsum`, refers to The sum of the level costs of the individual goals (admissible if goals independent). *Both A* with h_ignore precondition and h_pg_levelsum produces ideal plan.* It's better than any above searches in terms of node expansion, goal test and new node.

Comparing A* h_ignore precondition and h_pg_levelsum, h_ignore precondition produces plan optimal plan with less time, even though h_pg_sum expansion of node is less. This is mainly due to heuristic calculation duration produced by h_pg_sum.

Conclusion:

From the above analysis we find that though best optimal heuristic planning search for the Aircargo problem is A* with h_ignore precondition. As mentioned in analysis, h_ignore precondition produce significantly quicker than A* with h_pg_levelsum, even though node expansion of h_pg_levelsum is minimal, as calculation timing of h_pg_levelsum is expensive. Best non heuristic optimal planning search is breadth first search.