Heuristic Analysis

Results for the search problems:

Air Cargo Problem 1	**breadth_first_search**	**breadth_first_tree_search**	**depth_first_graph_search**	**depth_limited_search**	**uniform_cost_search**	**recursive_best_first_search with h_1**	**greedy_best_first_graph_search with h_1**	**astar_search with h_1**	**astar_search with h_ignore_preconditions**	**astar_search with h_pg_levelsum**
Expansions	43	1458	12	101	55	4229	7	55	41	11
Goal Tests	56	1459	13	271	57	4230	9	57	43	13
New Nodes	180	5960	48	414	224	17029	28	224	170	50
Plan length	6	6	12	50	6	6	6	6	6	6
Time elapsed in seconds	0.0337	1.0688	0.0101	0.1073	0.0433	3.0741	0.0070	0.0437	0.0433	1.1338

Optimal Plan: For this problem the greedy_best_first_graph_search_with_h1 is most optimal as it expands very few nodes in comparison to rest and also takes very less time to execute, while also giving the best plan length.

Plan:

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

Air Cargo Problem 2	**breadth_first_search**	**depth_first_graph_search**	**uniform_cost_search**	**greedy_best_first_graph_search with h_1**	**astar_search with h_1**	**astar_search with h_ignore_preconditions**
Expansions	3343	582	4853	998	4853	1450
Goal Tests	4609	583	4855	1000	4855	1452
New Nodes	30509	5211	44041	8982	44041	13303
Plan length	9	575	9	21	9	9
Time elapsed in seconds	9.293419	4.03291	22.675072	3.153881	13.15169	4.833652

Optimal Plan: For this problem the depth_first_graph_search is most optimal as it expands very few nodes in comparison to rest and also takes lesser time to execute, while also giving the best plan length.

Plan:

Load(C2 P2 JFK)
Load(C1 P1 SFO)
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)

Air Cargo Problem 3	breadth_first_search	depth_first_graph_search	greedy_best_first_graph_search with h_1	astar_search with h_1	astar_search with h_ignore_preconditions	astar_search with h_pg_levelsum
Expansions	14663	627	5578	Took more than 10 mins to execute	5040	Took more than 10 mins to execute
Goal Tests	18098	628	5580		5042	
New Nodes	129631	5176	49150		44944	
Plan length	12	596	22		12	
Time elapsed in seconds	50.320166	3.69332	18.04512		18.69254	

Optimal Plan: For this problem the greedy_best_first_graph_search with h_1 and astar_search with h_ignore_preconditions both seem optimal as they execute in reasonable amount of time while giving best plan length but expands large number of nodes.

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)

The depth_first_graph_search executes faster but the plan length is too large compared to rest.

We can observe here that breadth first runs successfully for all the problems and give optimal plan length. But the node expansion is very large in case of complex problems like problem 3. Problem 1 which is less complex the algorithm performs well. The A* search with h_ignore_preconditions performs better for complex problem. In terms of time it is almost similar in these 3 problems but the nodes expansion and the goal test reduces considerably in complex cases. The A* search with h_ignore_preconditions out performs other algorithms in expansions, new nodes, and time elapsed,

As stated in section 2.3 Discrete Optimal Planning of ‘Planning Algorithms’ by Steven M. La Valle

“With nearly all optimization problems, there is the arbitrary, symmetric choice of whether to define a criterion to minimize or maximize. If the cost is a kind of energy or expense, then minimization seems sensible, as is typical in robotics and control theory. If the cost is a kind of reward, as in investment planning or in most AI books, then maximization is preferred.”

If we take “Plan length” and “Time elapsed in seconds” as our cost, we should minimize both of these cost. Breadth First Search gives lowest cost in all cases but increases the “Time elapsed in seconds” in complex cases. The A* search with h_ignore_preconditions minimizes both these cost even for complex problem, hence we can consider it to be an optimal planning algorithm for the above cases.