Heuristic Analysis:

Plan search run stats:

1. Problem: Air Cargo Problem 1

Air Cargo Problem 1					
	Expansions	Goal Tests	New Nodes	Time	Plan length
Breadth first search	43	56	180	0.05780	6
Breadth first tree search	1458	1459	5960	1.31080	6
Depth first graph search	21	22	84	0.02447	20
Depth limited search	101	271	414	0.14215	50
Uniform cost search	55	57	224	0.05246	6
Recursive best first search with h1	4229	4230	17023	3.99315	6
Greedy best first graph search with h1	7	9	28	0.00862	6
A* search with h1	11	13	50	1.09854	6
A* search with heuristic ignore preconditions	41	43	170	0.05769	6
A* search with heuristic pg levelsum	11	13	50	1.13266	6

Optimum Path length : 6

Load(C1, P1, SFO)

Load(C2, P2, JFK)

Fly(P1, SFO, JFK)

Fly(P2, JFK, SFO)

Unload(C1, P1, JFK)

Unload(C2, P2, SFO)

From computational point of view it is the easiest problem. Greedy best first graph search with h1 is the optimum for project 1.

2. Problem: Air Cargo Problem 2

Air Cargo Problem 2					
	Expansions	Goal Tests	New Nodes	Time	Plan length
Breadth first search	3343	4609	30509	18.2717	9
Breadth first tree search	Inf	Inf	Inf	Inf	inf
Depth first graph search	624	625	5602	4.95970	619
Depth limited search	Inf	Inf	Inf	Inf	Inf
Uniform cost search	4852	4854	44030	17.5086	9
Recursive best first search with h1	Inf	Inf	Inf	Inf	inf
Greedy best first graph search with h1	990	992	8910	3.4957	21
A* search with h1	4852	4854	44030	17.3719	9
A* search with heuristic ignore preconditions	1450	1452	13303	6.24356	9
A* search with heuristic pg levelsum	86	88	841	100.1790	9

*Inf : More than 10 minutes

Optimum Path length : 9

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)

This problem is a little hard, it could not solve all the result within the 10 minute time frames. A* search with heuristic ignore preconditions outshines here.

3. Problem: Air Cargo Problem 3

Air Cargo Problem 3					
	Expansions	Goal Tests	New Nodes	Time	Plan length
Breadth first search	14663	18098	129631	161.4511	12
Breadth first tree search	Inf	Inf	Inf	Inf	inf
Depth first graph search	408	409	3364	2.50742	392
Depth limited search	Inf	Inf	Inf	Inf	Inf
Uniform cost search	18234	18236	159707	84.48874	12
Recursive best first search with h1	Inf	Inf	Inf	Inf	inf
Greedy best first graph search with h1	5605	5607	49360	28.23314	22
A* search with h1	18234	18236	159707	91.63914	12
A* search with heuristic ignore preconditions	5040	5042	44944	26.31628	12
A* search with heuristic pg levelsum	325	327	3002	540.2048	12

*Inf: more than 10 minutes

Optimum Path length : 12

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)

A* search with heuristic ignore preconditions outshines here.

Analysis:

BFS expands all the nodes of the search graph before going deeper. It considers the shortest path first. But it grows slower as the search space grows bigger. DFS goes in depth first, not optimal. UCS is said to find the path with the cheapest total cost. It expands the node with the lowest cost because of internal BFS. A* search includes heuristic functions(calls BFS internally).

For heuristic search methods,

A* works by going to the path which has the minimum value of the function (f)

f = g + h

g(path) = (path cost)

h is the estimated distance goal,

h1 always returns 1 for the estimated distance to goal, not very useful

Given the above results, ignore preconditions performs good, but it may not always be the case.

for problem 1, the search space is small, 2^{12}, so non heuristic methods are still somewhat efficient, but as the search space grows, for problem 2 and problem 3, 2^{27} and 2^{32} respectively, problem 2 can go with heuristic and non heuristic approached because of medium size search spacce, it is better to use the heuristic methods, because they narrow down the search space significantly, depending upon the heuristic function employed, because they go towards the goal state without going through the whole search space, in this process the result may or may not always be the best but they are affordable/better in terms of time and space and provide a real world solution.

Heuristic ignore precondition outperforms in the current scenario, it 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.

Level Sum is costly and slow because this uses a planning graph representation of the problem search state space to estimate the sum of all actions that must be carried out from the current state in order to satisfy each individual goal condition.

comparing level sum and ignore preconditions, ignore preconditions works much faster, because level-sum requires more computational power than ignore preconditions. ignore preconditions only current search space state level, which is finite while level sum goes through multiple search space state levels.

Conclusion:

If the search space is small, it even non-heuristic methods are effective. for larger search space , non-heuristic searches started to take a lot of time. As the search space grows, heuristic search becomes important.

References:

1. Norwig, P. and Russel, Artificial Intelligence: A modern Approach. 3rd edition