Heuristics Analysis

Optimal Plans:

Problem 1:

Load(C1, P1, SFO)

Fly(P1, SFO, JFK)

Unload(C1, P1, JFK)

Load(C2, P2, JFK)

Fly(P2, JFK, SFO)

Unload(C2, P2, SFO)

Problem 2:

Load(C1, P1, SFO)

Fly(P1, SFO, JFK)

Unload(C1, P1, JFK)

Load(C2, P2, JFK)

Fly(P2, JFK, SFO)

Unload(C2, P2, SFO)

Load(C3, P3, ATL)

Fly(P3, ATL, SFO)

Unload(C3, P3, SFO)

Problem 3:

Load(C1, P1, SFO)

Fly(P1, SFO, ATL)

Load(C3, P1, ATL)

Fly(P1, ATL, JFK)

Unload(C3, P1, JFK)

Unload(C1, P1, JFK)

Load(C2, P2, JFK)

Fly(P2, JFK, ORD)

Load(C4, P2, ORD)

Fly(P2, ORD, SFO)

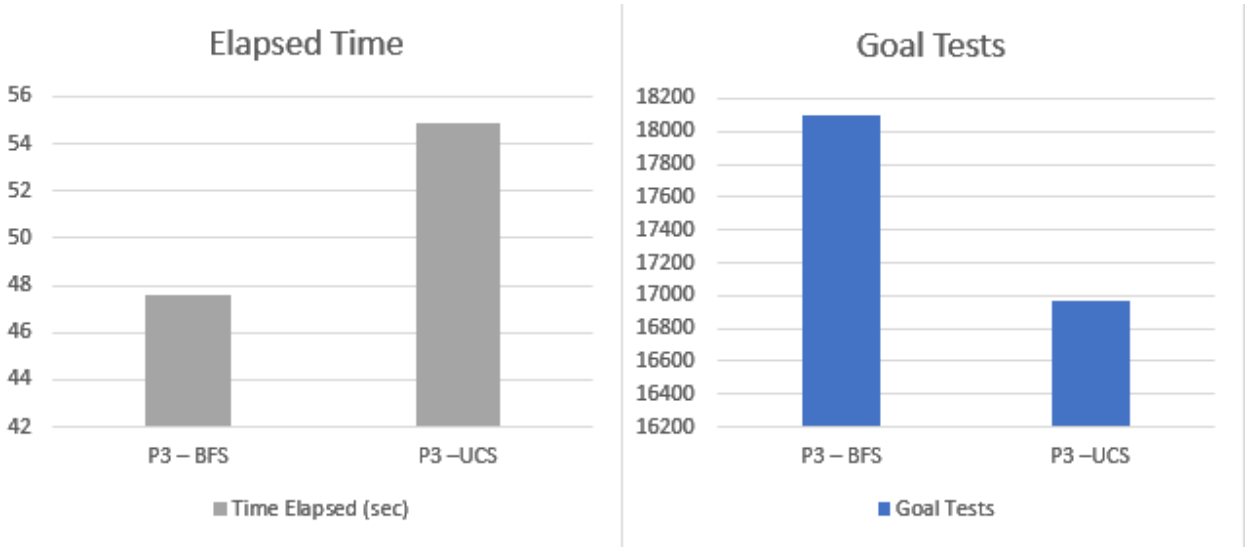
Unload(C4, P2, SFO)

Unload(C2, P2, SFO)

Uniformed Search

The best non-heuristic search was *BFS*. *DFGS* performed better on all other metrics but produced sub-optimal solutions with a length two order of magnitudes larger than the optimal plan length.

On elapsed time and node expansions *BFS* performed better than *UCS*. *UCS* performed marginally better than *BFS* on the goal tests metric.



Heuristic Search – A* with levelsum vs ignore preconditions

The best heuristic for all of the problems was *ignore preconditions*, it performed better on both the optimality and time metrics.

Level sum could produce either optimal or sub-optimal solutions for problem 3 depending on the order of literals in the initial state. For problems 1 and 2 it produced an optimal solutions and required less node expansions and goal tests but required an order of magnitude more time than *ignore preconditions* to complete.

Heuristic vs Non-heuristic Search

Comparing heuristic to non-heuristic search techniques, *A* search* with *ignore preconditions* heuristic performed best overall for all problems. It required roughly a quarter of the goal tests and node expansions and a third of the time of *BFS*.

DFGS does not find optimal solutions because it searches each path to its end and terminates once any solution, that is, a path that reaches all goals is found. So only if the first possible solution happens to be optimal it would reach it.

The graph search version of A* is optimal given ignore preconditions heuristic is consistent. In any case the use of a good heuristic provides enormous savings compared to uninformed search such as BFS.

The metrics table for the informed and uninformed searches shows that *A** with either heuristic dominates any uninformed search method by any criteria.

Comparing informed searches we see *levelsum* dominates *ignore preconditions* by node expansions and goal tests but is being dominated by the time complexity criteria.

Ignore preconditions iterates the goals and for each looks up its state, that is ***O(goals x log states)*** python list index operation is *O(log n)*. Level sum iterates the goals and for each iterates planning graph state levels. That is **O(goals x levels x states)** making *ignore preconditions* an order of magnitude faster.

*A** expands nodes with minimal *f(n) = g(n) + h(n)*, it is optimal yet its space complexity is still prohibitive but much better than *BFS* having exponential space complexity.

Complete Metrics Table

	Plan Length	Optimal	Time Elapsed (sec)	Node Expansions	Goal Tests	New Nodes
P1 – BFS	6	Yes	.027	43	56	180
P1 –DFGS	12	No	.009	12	13	48
P1 –UCS	6	Yes	.034	55	57	224
P2 – BFS	9	Yes	8.72	3343	4609	30509
P2 –DFGS	466	No	2.44	476	477	4253
P2 –UCS	9	Yes	11.9	4604	4606	41828
P3 – BFS	12	Yes	47.6	14663	18098	129631
P3 –DFGS	1442	No	13.9	1511	1512	12611
P3 –UCS	12	Yes	54.9	16963	16965	149136
P1 – A* h1	6	Yes	.034	55	57	224
P1 – A* ignore_precond	6	Yes	.026	41	43	170
P1 – A* pg_levelsum	6	Yes	.896	11	13	50
P2 – A* h1	9	Yes	11.5	4604	4606	41828
P2 – A* ignore_precond	9	Yes	3.54	1310	1312	11979
P2 – A* pg_levelsum	9	Yes	93.9	94	96	925
P3 – A* h1	12	Yes	52.9	16963	149136	16965
P3 – A* ignore_precond	12	Yes	14.6	4444	4446	39227
P3 – A* pg_levelsum	12	Yes	434	314	316	2894