The optimal plans for the problems are:

Problem 1 (Path length 6):

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 (Path length 9):

Load(C3, P3, ATL)
Fly(P3, ATL, SFO)
Unload(C3, P3, SFO)
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 3 (Path length 10):

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

Comparison of non-heuristic search result metrics:

Breadth first search-

	Optimality	Time Elapsed (seconds)	Number of node expansion
Problem 1	Yes	0.028	43
Problem 2	Yes	12.979	3343
Problem 3	Yes	46..525	7320

Depth first graph search-

	Optimality	Time Elapsed	Number of node expansion
Problem 1	No	0.013	21
Problem 2	No	3.258	624
Problem 3	No	1.028	273

Uniform cost search-

	Optimality	Time Elapsed	Number of node expansion
Problem 1	Yes	0.034	55
Problem 2	Yes	42.244	4852
Problem 3	Yes	301.432	12558

Depth first graph search appears to be fast and expands fewer nodes but it really just gives the first solution it finds which is non optimal. If the problem were to find any solution or to just detect whether there is a solution then it would have worked great.

For finding optimal solutions (paths), breadth first search does a better job than uniform cost search in respect of time consumed as well as the number of nodes expanded.

Hence, breadth first search seems to be the best choice for a non heuristic search within the given constraints.

Comparison of heuristic search result metrics:

A* search with h_ignore_preconditions-

	Optimality	Time Elapsed (seconds)	Number of node expansion
Problem 1	Yes	0.042	41
Problem 2	Yes	14.542	1506
Problem 3	Yes	32.842	2036

A* search with h_pg_levelsum-

	Optimality	Time Elapsed	Number of node expansion
Problem 1	Yes	1.385	11
Problem 2	Yes	212.398	86
Problem 3	Yes	661.029	189

The first heuristic which ignores the preconditions is no doubt faster than the level sum heuristic as it is computationally expensive to compute the level sum, but if you look closely at the number of nodes expanded, level sum does it in an extremely low number.

On comparing the heuristic search with non heuristic ones, even the first heuristic is better than the breadth first search (which was the best non heuristic search). These comparisons are based on the number of nodes expanded parameter. Even considering the time elapsed, heuristic search performs better (the first one).

Heuristic search provides a solution by expanding fewer nodes as it computes intelligently which nodes may lead to the solution and propagating the estimated right path in the planning graph.