Results

After completing the algorithm for Planning, I have obtained the following results:

Cargo Problem 1	Expansions	Goal Tests	New Nodes	Plan Length	Time elapsed in s
Breadth-first-search	43	56	180	6	1.39
Depth-first-search	21	22	84	20	0.02
Greedy best first graph search with h1	7	9	28	6	0.007
A* with h_1	55	57	224	6	0.07
A* with h_ignore_preconditions	41	43	170	6	0.07
A* with h_pg_levelsum	41	43	170	6	4.15

All plans provided by the different algorithms achieve a goal state, but the plans with length 6 are the optimal ones. One such example is (from A* search with ignore pre-conditions heuristic):

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

For Cargo Problem 1, the problem is simple and therefore the state space is not large 2^2 x (2+2)^2 = 64, therefore it can be easily traversed by standard informed search algorithms. In particular the greedy best first graph search with h_1 is very fast and expand the fewest amount of nodes. However it is only because the characteristic of the problem that enables it to perform well. A* with the level sum heuristic is taking a long time to compute the heuristic, and therefore is the slowest among all other algorithms to find a goal.

Cargo Problem 2	Expansions	Goal Tests	New Nodes	Plan Length	Time elapsed in s
Breadth-first-search	3190	4380	28279	9	16.76
Depth-first-search	1172	1173	9578	200	4.73
Greedy best first graph search with h1	620	622	5355	21	4.56
A* with h_1	4548	4550	40338	9	56.9
A* with h_ignore_preconditions	1424	1426	12572	9	17.8
A* with h_pg_levelsum	1118	1120	9760	9	2200.3

All plans provided by the different algorithms achieve a goal state, but the plans with length 9 are the optimal ones. One such example is (from A* search with ignore pre-conditions heuristic):

Load(C2, P2, JFK)
Fly(P2, JFK, ATL)
Load(C3, P2, ATL)
Fly(P2, ATL, SFO)
Unload(C3, P2, SFO)
Load(C1, P1, SFO)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)

With Cargo Problem 2, there are significantly more states: 3^3 x (3 + 3) ^ 3 = 5832
Again here the A* search with Level Sum heuristic is taking a long time to compute the heuristic function and takes 2200 seconds to reach a goal state, it is 100 times slower than A* with ignore pre-conditions heuristic. Here the fastest algorithm to achieve an optimal plan is Breadth-first-search. It is executing faster than the A* with ignore pre-conditions heuristic.

Cargo Problem 3	Expansions	Goal Tests	New Nodes	Plan Length	Time elapsed in s
Breadth-first-search	14663	18098	129631	12	146.9
Depth-first-search	408	409	3364	392	2.54
Greedy best first graph search with h1	5578	5580	49150	22	239.6
A* with h_1	18223	18225	159618	12	617.3
A* with h_ignore_preconditions	5118	5120	45650	12	137.6
A* with h_pg_levelsum	N/A	N/A	N/A	N/A	N/A

All plans provided by the different algorithms achieve a goal state, but the plans with length 12 are the optimal ones. One such example is (from A* search with ignore pre-conditions heuristic):

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)

With Cargo Problem 3, there are even more more states: 4^2 x (4 + 2) ^ 4 = 20736
Again here the A* search with Level Sum heuristic is taking a long time to compute the heuristic function and is not finishing within a reasonable timeframe. The heuristic implementation may be optimised to enable it to finish.
Here the fastest algorithm to achieve an optimal plan is A* with ignore pre-conditions heuristic. It is executing faster than the non-heuristic search planning methods.

There is an inherent trade-off between the computing cost of calculating the heuristic and the number of nodes to be explored to reach the goal. In Problem 1 and 2, the search space was rather limited so we could afford to explore unnecessary nodes to search for a solution. However for Problem 3, to have a simple enough to calculate heuristic enables to reduce the number of nodes explored and therefore can reach the goal faster than other non-heuristic search methods.