Optimal plans

Problem 1
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P2, JFK, SFO)
Unload(C2, P2, SFO)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)

Problem 2
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Load(C3, P3, ATL)
Fly(P2, JFK, SFO)
Unload(C2, P2, SFO)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)
Fly(P3, ATL, SFO)
Unload(C3, P3, SFO)

Problem 3
Load(C2, P2, JFK)
Fly(P2, JFK, ORD)
Load(C4, P2, ORD)
Fly(P2, ORD, SFO)
Load(C1, P1, SFO)
Fly(P1, SFO, ATL)
Load(C3, P1, ATL)
Fly(P1, ATL, JFK)
Unload(C4, P2, SFO)
Unload(C3, P1, JFK)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)

Results on uninformed search

I used three uninformed search algorithm to solve the planning problem - breadth first graph search, depth first graph search and uniform cost search.
All three searches found solutions to all three problems, but not all of them were optimal.
BFS and UCS found optimal solutions, but BFS was faster in finding solution. This is due to the fact that it terminates as it finds the first solution, luckily these problems do not have path costs

and the shortest expansion solutions are optimal. UCS unnecessarily expanded more nodes in each problem to be sure that the solutions are optimal.
DFS performs fastest in finding solution but the solution is not guaranteed to be optimal.

Problem 1

	Expansions	Goal test	New nodes	Time	Plan length
breadth_first_search	43	56	180	0.035247838	6
depth_first_graph_search	21	22	84	0.016468541	24
uniform_cost_search	55	57	224	0.045131716	6

Problem 2

	Expansions	Goal test	New nodes	Time	Plan length
breadth_first_search	3343	4609	30509	15.67767396	9
depth_first_graph_search	624	625	5602	4.018900404	619
uniform_cost_search	4852	4854	44030	47.95937673	9

Problem 3

	Expansions	Goal test	New nodes	Time	Plan length
breadth_first_search	14663	18098	129631	110.9151079	12
depth_first_graph_search	408	409	3364	1.969298796	392
uniform_cost_search	18234	18236	159707	404.1868242	12

Results on informed search

I used four informed search algorithms to solve the planning problem - Greedy best first search, A* search with H_1 heuristic, A* search with ignore preconditions heuristic, A* search with planning graph levelsum heuristic.
GBFS performs fastest in finding solutions but the solutions are not guaranteed to be optimal. A* searches find optimal solutions. The performance varies on the heuristics used. The better the heuristic function - the fewer nodes are expanded during search, but the time taken to compute complex heuristics may be much longer. The levelsum heuristic expanded very few nodes to find the optimal solution but when considering time measurements it performed worst.

Problem 1

	Expansions	Goal test	New nodes	Time	Plan length
greedy_best_first_graph_search with h_1	7	9	28	0.007126611999	6
astar_search h_1	55	57	224	0.047033779	6
astar_search h_ignore_preconditions	41	43	170	0.051459556	6
astar_search h_pg_levelsum	11	13	50	1.831630364	6

Problem 2

	Expansions	Goal test	New nodes	Time	Plan length
greedy_best_first_graph_search with h_1	990	992	8910	8.493362685	21
astar_search h_1	4852	4854	44030	54.55604284	9
astar_search h_ignore_preconditions	1506	1508	13820	15.19730821	9
astar_search h_pg_levelsum	86	88	841	171.9431099	9

Problem 3

	Expansions	Goal test	New nodes	Time	Plan length
greedy_best_first_graph_search with h_1	5605	5607	49360	108.8572857	22
astar_search h_1	18234	18236	159707	406.099976	12
astar_search h_ignore_preconditions	5118	5120	45650	93.65506085	12
astar_search h_pg_levelsum	408	410	3758	1470.701208	12

Informed vs uninformed search

After analyzing the results of informed and uninformed searches the fastest algorithms in time perspective were the ones that do not find optimal solutions - DFS and GBFS. If we compare algorithms that are guaranteed to find optimal solutions[1] - the best algorithms in the fewest nodes expanded is the A* search with levelsum heuristic, however the time taken to compute the heuristic every time may be long enough to not to use this algorithm to solve these

[1] According to Chapter 11, Russell, Stuart J.; Norvig, Peter (2009). Artificial Intelligence: A Modern Approach (3rd ed.). Upper Saddle River, New Jersey: Prentice Hall.

problems. Best algorithm in time perspective is A* search with ignore preconditions heuristic. It is guarantied to find optimal solution, its heuristic is good enough not to expand too many nodes, and computing its heuristic is not time consuming. I find it to be optimal search algorithm in solving these cargo problems.