Definitions

The optional plan consists in a plan with the right length (6, 9 and 12) that
take the less time to compute.

The timeout annotation on Time columns means that the search takes more than 600
seconds to compute, so I kill the search and move to the next one.

The best approach consists in a unique search strategy for all problems, even if
this search strategy is not the optimal for one Problem.

Optimal Plans

Problem 1

Solving Air Cargo Problem 1 using greedy_best_first_graph_search with h_1...

Expansions	Goal Tests	New Nodes	Plan length	Time (sec.)
7	9	28	6	0.0046

- Load(C1, P1, SFO)
- Load(C2, P2, JFK)

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

Problem 2

Solving Air Cargo Problem 2 using astar_search with h_ignore_preconditions...

Expansions	Goal Tests	New Nodes	Plan length	Time (sec.)
1450	1452	13303	9	5.7446

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

Problem 3

Solving Air Cargo Problem 2 using h_ignore_preconditions...

Expansions	Goal Tests	New Nodes	Plan length	Time (sec.)
5040	5042	44944	12	20.32

- 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)

Non-heuristic search result metrics

breadth_first_search

Problem	Expansions	Goal Tests	New Nodes	Plan Length	Time (sec.)
Problem 1	43	56	180	6	0.0252
Problem 2	-	-	-	-	timeout

Problem 3	14663	18098	129631	12	118.1575

breadth_first_tree_search

Problem	Expansions	Goal Tests	New Nodes	Plan Length	Time (sec.)
Problem 1	1458	1459	5960	6	0.7996
Problem 2	-	-	-	-	timeout
Problem 3	-	-	-	-	timeout

depth_first_graph_search

Problem	Expansions	Goal Tests	New Nodes	Plan Length	Time (sec.)
Problem 1	12	13	48	12	0.0073
Problem 2	582	583	5211	575	3.30
Problem 3	627	628	5176	596	4.16

depth_limited_search

		Goal	New	Plan	Time

Problem	Expansions	Tests	Nodes	Length	(sec.)
Problem 1	12	13	48	12	0.0073
Problem 2	222719	2053741	2054119	50	926.8643
Problem 3	-	-	-	-	timeout

Heuristic search

greedy_best_first_graph_search with h_1

Problem	Expansions	Goal Tests	New Nodes	Plan Length	Time (sec.)
Problem 1	7	9	28	6	0.0042
Problem 2	990	992	8910	15	2.4041
Problem 3	5614	5616	49429	22	19.9140

astar_search with h_1

Problem	Expansions	Goal Tests	New Nodes	Plan Length	Time (sec.)
Problem 1	55	57	224	6	0.0321

Problem 2	4852	4854	44030	9	20.8326
Problem 3	18235	18237	159716	12	62.5677

astar_search with h_pg_levelsum

Problem	Expansions	Goal Tests	New Nodes	Plan Length	Time (sec.)
Problem 1	11	13	50	6	1.3315
Problem 2	86	88	841	9	250.6339
Problem 3	-	-	-	-	timeout

astar_search with h_ignore_preconditions

Problem	Expansions	Goal Tests	New Nodes	Plan Length	Time (sec.)
Problem 1	41	43	170	6	0.0228
Problem 2	1450	1452	13303	9	5.7446
Problem 3	5040	5042	44944	12	20.3272

Best approach

The breadth_* and depth_* search strategies results in too long plans with many
timeouts this is due the fact that on each expansion we increase the frontier of
the problem (in depth or in breadth) so our search space increase much more than the
needed.

The h_pg_levelsum expands into few nodes but take more time to compute, giving a
timeout on Problem 3. This is due to the fact that the h_pg_levelsum expands the
use of planning graph to search the planning space.

The technique that worked with more or less good time results and expansions
for the three problems is the *h_ignore_preconditions*, so this is the best choice.