Artificial Intelligence Nanodegree
Project 3: Implement a Planning Search

The results of the implementation of various non-heuristic and heuristic planning search algorithms for the three different air cargo problem is summarized in Table 1 below.

Table 1: Summary of Planning Search Results

		Non-Heuristic Search			Heuristic Search	
		Breadth First Search	Depth First Graph Search	Uniform Cost Search	A-star Search Ignore Preconditions	A-star Search Level-sum Heuristics
Air Cargo Problem 1	Plan Length	6	20	6	6	6
	Time Elapse (s)	0.02	0.01	0.03	0.02	1.45
	Expansions	43	21	55	41	11
	Goal Tests	56	22	57	43	13
	New Nodes	180	84	224	170	50
Air Cargo Problem 2	Plan Length	6	183	6	6	6
	Time Elapse (s)	1.54	0.60	11	0.60	24
	Expansions	477	184	1462	155	13
	Goal Tests	1110	185	1464	157	15
	New Nodes	4317	1663	13408	1411	234
Air Cargo Problem 3	Plan Length	12	392	12	12	Did not finish
	Time Elapse (s)	102	1.6	393	83	
	Expansions	14663	408	17882	5114	
	Goal Tests	18098	409	17884	5116	
	New Nodes	129631	3364	156769	45610	

Optimal Plans

The optimal plan for Problem 1 is:
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P2, JFK, SFO)

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

The optimal plan for Problem 2 is:
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P2, JFK, SFO)
Unload(C2, P2, SFO)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)

The optimal plan for Problem 3 is:
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P1, SFO, ATL)
Load(C3, P1, ATL)
Fly(P2, JFK, ORD)
Load(C4, P2, ORD)
Fly(P2, ORD, SFO)
Fly(P1, ATL, JFK)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)
Unload(C3, P1, JFK)
Unload(C4, P2, SFO)

Non-Heuristic Search Results

First, we compare the non-heuristic search results. From the results above, we can observe that breadth first search and uniform cost search were able to find the optimal plan in all three problems, whereas depth first graph search results in plan that are significantly longer. However, depth first graph search was able to find a plan within the shortest amount of time, due to the lower number of nodes explored. Even though breadth first search and uniform cost search were both able to find the optimal plan, breadth first search was able to find the plan faster in all three problems due to the lower number of node expansions needed.

Based on these observations, the following recommendations can be made:

1. If the goal is to find the optimal plan, the breadth first search algorithm is the best of the three options as it is able to find the optimal plan in the shortest amount of time.
2. If the goal is just to find a feasible plan (i.e. optimality is not a concern), the depth first graph search is the best of the three options as it is able to find a plan in the shortest amount of time.

Heuristic Search Results

Next, we compare the heuristic search results. Overall, A* search with the ignore preconditions heuristics were able to find a plan significantly faster than A* search with level sum heuristic. A* search with level sum heuristic did not find a plan within 10 minutes for problem 3. Based on the results of problem 1 and 2, we can observe that both heuristics were able to find the optimal plan for both problems. It is interesting to note that A* search with the ignore preconditions heuristic was able to find a plan faster, while exploring a greater number of nodes. This suggests that the a significant amount of processing time for the level-sum heuristics is spent on determining the best nodes to expand to instead of actually exploring those nodes.

Overall Best Heuristic

The overall best heuristic is the A* search with ignore preconditions heuristics as it was able to find the optimal solution in the shortest amount of time. If the goal is to find the optimal plan, A* search with ignore preconditions heuristics is better than the non-heuristic search planning methods tested for all problems as it is able to find the optimal plan in the shortest amount of time. However, if the goal is just to find a feasible plan, the depth first graph search non-heuristic algorithm can find a feasible plan faster than the heuristic algorithm.