Optimal Plans and Search Results for Problems

Problem 1

Search	Expansions	Goal Tests	New Nodes	Plan length	Time	Optimal
breadth_first_search	43	56	180	6	0.034926917 00012074	Y
breadth_first_tree_search	1458	1459	5960	6	0.942795344 9994675	Y
depth_first_graph_search	21	22	84	20	0.016273938 99980234	N
depth_limited_search	101	271	414	50	0.095486995 99690735	N
uniform_cost_search	55	57	224	6	0.040744327 99825445	Y
recursive_best_first_search h_1	4229	4230	17023	6	2.869758317 9964568	Y
greedy_best_first_graph_search h_1	7	9	28	6	0.006893670 004501473	Y
astar_search h_1	55	57	224	6	0.040410083 99508428	Y
astar_search h_ignore_preconditions	41	43	170	6	0.041996167 004981544	Y
astar_search h_pg_levelsum	11	13	50	6	0.981356162 999873	Y

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

In this case, All search models could find solution except depth-depend searches. And greedy_best_first_graph_search h_1 model found the optimal solution by shortest time.

Problem 2

Search	Expansions	Goal Tests	New Nodes	Plan length	Time	Optimal
breadth_first_search	3343	4609	30509	9	18.07571472300333	Y
breadth_first_tree_search	N/A it takes longer than 10 minutes to run					
depth_first_graph_search	624	625	5602	619	3.80788694700459	N
depth_limited_search	N/A it takes longer than 10 minutes to run					
uniform_cost_search	4852	4854	44030	9	13.032695085996238	Y
recursive_best_first_search h_1	N/A it takes longer than 10 minutes to run					
greedy_best_first_graph_search h_1	990	992	8910	15	2.726652552 9994396	N
astar_search h_1	4852	4854	44030	9	12.491919120999228	Y
astar_search h_ignore_preconditions	1450	1452	13303	9	5.486802405001072	Y
astar_search h_pg_levelsum	86	88	841	9	188.99900394400174	Y

Optimal Plan
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)**

In this case, astar_search h_ignore_preconditions model made a best performance. Some models took longer than 10 minutes. So, they couldn't finish search.

Problem 3

Search	Expansions	Goal Tests	New Nodes	Plan length	Time	Optimal
breadth_first_search	14663	18098	129631	12	146.735627916100305	Y
breadth_first_tree_search	N/A it takes longer than 10 minutes to run					
depth_first_graph_search	408	409	3364	392	2.4674300180049613	N
depth_limited_search	N/A it takes longer than 10 minutes to run					
uniform_cost_search	18235	18237	159716	12	86.16590857499978	Y
recursive_best_first_search h_1	N/A it takes longer than 10 minutes to run					
greedy_best_first_graph_search h_1	5614	5616	49429	22	21.626514317002147	N
astar_search h_1	18235	18237	159716	12	91.02503750899632	Y
astar_search h_ignore_preconditions	5040	5042	44944	12	24.72490674499568	Y
astar_search h_pg_levelsum	N/A it takes longer than 10 minutes to run					

Optimal Plan
Load(C1, P1, SFO) **Load(C2, P2, JFK)** **Fly(P2, JFK, ORD)** **Load(C4, P2, ORD)** **Fly(P1, SFO, ATL)** **Load(C3, P1, ATL)** **Fly(P1, ATL, JFK)** **Unload(C1, P1, JFK)** **Unload(C3, P1, JFK)** **Fly(P2, ORD, SFO)** **Unload(C2, P2, SFO)** **Unload(C4, P2, SFO)**

In this case, as same as previous problem, astar_search h_ignore_preconditions model made a best performance. Because this problem was more complex than other, a lot of models took longer time than before.

Conclusion

breadth_first_search, uniform_cost_search, astar_search h_1 and astar_search h_ignore_preconditions always found the optimal solution. In simple problem, breadth_first_search and uniform_cost_search could find a solution efficiently. But, when the problem gets complicated, star_search made a best performance.

And depth-depend searches always made small expansions and also these took short time. But, depth-first search does not guarantee optimality. It makes sense. According to our video lessons from AIND, it can't ensure that the solution will be the shortest path. Because, when the depth-first search find a solution, it finished searching although other path could be optimal solution.

So, Considering expansions and time, In this problems, choosing astar_search h_ignore_preconditions was best choice. But, actually, astar_search h_pg_levelsum is more powerful than astar_search h_ignore_preconditions. But, it took long time. So, I could find a solution in problem 3. I think it may be due to a complexity of this model. In conclusion, astar_search h_ignore_preconditions is a universal model.