Heuristic analysis

Fangjun Shi

Air Cargo Problem 1

Search Algorithms	Expansions	Goal Tests	New Nodes	Plan length	Time elapsed
breadth_first_search	43	56	180	6	0.0383
breadth_first_tree_search	1458	1459	5960	6	1.2293
depth_first_graph_search	12	13	48	12	0.0102
depth_limited_search	101	271	414	50	0.1174
uniform_cost_search	55	57	224	6	0.0534
recursive_best_first_search h_1	4229	4230	17029	6	3.5606
greedy_best_first_graph_search h_1	7	9	28	6	0.0082
astar_search h_1	55	57	224	6	0.0506
astar_search h_ignore_preconditions	41	43	170	6	0.0482
astar_search h_pg_levelsum	11	13	50	6	0.8163

Problem 1 is easy to solve, and each algorithm gives the right result at a reasonable time. In non-heuristic algorithm, *greedy_best_first_graph_search* outperforms the other algorithm, it took the least amount of expansions, goal tests, new nodes and plan length. In heuristic algorithm, *astar_search* with *h_ignore_preconditions* took the least amount of time and got the best result.

Optimal Plan

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

Air Cargo Problem 2

Search Algorithms	Expansions	Goal Tests	New Nodes	Plan length	Time elapsed
breadth_first_search	3343	4609	30509	9	10.2402
breadth_first_tree_search	N/A	N/A	N/A	N/A	N/A
depth_first_graph_search	582	583	5211	575	3.9232

Search Algorithms	Expansions	Goal Tests	New Nodes	Plan length	Time elapsed
depth_limited_search	N/A	N/A	N/A	N/A	N/A
uniform_cost_search	4853	4855	44041	9	14.8303
recursive_best_first_search h_1	N/A	N/A	N/A	N/A	N/A
greedy_best_first_graph_search h_1	998	1000	8982	15	3.1173
astar_search h_1	4853	4855	44041	9	15.9054
astar_search h_ignore_preconditions	1450	1452	13303	9	5.5553
astar_search h_pg_levelsum	86	88	841	9	74.7839

Problem 2 is more complicated. *breadth_first_tree_search*, depth_limited_search and recursive_best_first_search could not find a plan in reasonable time. In non-heuristic algorithm, *breadth_first_search* outperforms the other algorithm. In heuristic algorithm, *astar_search* with *h_ignore_preconditions* took the least amount of time and got the best result.

Optimal Plan

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

Air Cargo Problem 3

Search Algorithms	Expansions	Goal Tests	New Nodes	Plan length	Time elapsed
breadth_first_search	14663	18098	129631	12	57.4634
breadth_first_tree_search	N/A	N/A	N/A	N/A	N/A
depth_first_graph_search	627	628	5176	596	4.1443
depth_limited_search	N/A	N/A	N/A	N/A	N/A
uniform_cost_search	18234	18236	159707	12	69.1089
recursive_best_first_search h_1	N/A	N/A	N/A	N/A	N/A
greedy_best_first_graph_search h_1	5605	5607	59360	22	21.8499
astar_search h_1	18234	18236	159707	12	70.9522

Search Algorithms	Expansions	Goal Tests	New Nodes	Plan length	Time elapsed
astar_search h_ignore_preconditions	5040	5042	44944	12	23.2782
astar_search h_pg_levelsum	325	327	3002	12	377.7148

Problem 3 is also complicated. *breadth_first_tree_search*, *depth_limited_search* and *recursive_best_first_search* could not find a plan in reasonable time too. In non-heuristic algorithm, *breadth_first_search* outperforms the other algorithm. In heuristic algorithm, *astar_search* with *h_ignore_preconditions* got the best result.

Optimal Plan

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)

Summary

The main reason why the majority of current systems favor forward search is the fact that backward search uses state sets rather than individual states makes it harder to come up with good heuristics. But forward search is prone to exploring irrelevant actions and planning problems often have large state spaces. Even the relatively small problem instance is hopeless without an accurate heuristic. Accordingly, neither forward nor backward search is efficient without a good heuristic function. The ignore preconditions heuristic drops all preconditions from actions. Every action becomes applicable in every state, and any single goal fluent can be achieved in one step. (AIMA 10.2)

Therefore, *astar_search* with *h_ignore_preconditions* was the best heuristic used in these problems.