Planning Heuristics Analysis

The machine I tested these algorithms on was an MSI Gaming Laptop, with 4 Intel(R) Core(TM) i7-6700HQ CPU @ 2.60GHz cores.

Task 1: Experiment and document metrics for non-heuristic planning solution searches

In this section the problems air_cargo_1, air_cargo_2 and air_cargo_3 will be analyzed in terms of non-heuristic search. The search types I chose were breadth_first_search, breadth_first_tree_search, and depth_first_graph_search.

Problem	Search Type	Node Expansions	Goal Tests	Time Elapsed (s)	Plan Length
air_cargo_1	breadth_first	43	56	0.101	6
air_cargo_1	breadth_first_tree_search	1458	1459	3.40	6
air_cargo_1	depth_first_graph_search	21	22	0.083	20
air_cargo_2	breadth_first	3346	4612	35.32	9
air_cargo_2	breadth_first_tree_search	Didn’t finish	Didn’t finish	> 10 mins	Didn’t finish
air_cargo_2	depth_first_graph_search	107	108	0.94	105
air_cargo_3	breadth_first	14120	17673	230	12
air_cargo_3	breadth_first_tree_search	Didn’t finish	Didn’t finish	> 10 mins	Didn’t finish
air_cargo_3	depth_first_graph_search	292	293	3.72	288

Task 2: Experiment and document: metrics of A* searches with these heuristics

In this section the problems air_cargo_1, air_cargo_2 and air_cargo_3 will be analyzed in terms of heuristic search. The search types I chose were astar_search h_1, astar_search h_ignore_preconditions, and astar_search h_pg_levelsum. The results appear in the table below.

Problem	**Search Type**	**Node Expansions**	**Goal Tests**	**Time Elapsed (s)**	**Plan Length**
air_cargo_1	astar_search h_1	55	57	0.125	6
air_cargo_1	astar_search h_ignore_preconditions	41	43	0.100	6
air_cargo_1	astar_search h_pg_levelsum	8	10	0.431	6
air_cargo_2	astar_search h_1	4853	4855	44.36	9
air_cargo_2	astar_search h_ignore_preconditions	1450	1452	12.52	9
air_cargo_2	astar_search h_pg_levelsum	17	19	12.81	9
air_cargo_3	astar_search h_1	18223	18225	201.36	12
air_cargo_3	astar_search h_ignore_preconditions	5040	5042	54.103	12

air_cargo_3	astar_search h_pg_levelsum	18	20	23.47	14

Analysis

For air_cargo_1, the non-heuristic searches breadth_first and breadth_first_tree_search produced optimal results in a short amount of time. The depth_first_graph_search search did not produce an optimal result. All of the heuristic searches produced optimal results, and the runtime was pretty comparable to the non-heuristic searches.

For air_cargo_2, the non-heuristic search breadth_first produced an optimal result. The depth_first_graph_search finished but produced a poor result (non-optimal) while the astar_search h_ignore_preconditions didn't finish in a reasonable amount of time. The heuristic searches did much better, and all three searches produced optimal plans. The searches astar_search h_ignore_preconditions and astar_search h_pg_levelsum both finished in just over 12 seconds, while the astar search h1 tool almost 4x longer.

For air_cargo_3, the only search which produced an optimal result was breadth_first, but it took 230 seconds. The breadth_first_tree_search didn't finish, while the depth_first_graph_search finished very quickly but produced a very poor result (non-optimal). As for the heuristic searches, astar_search h_1 and astar_search h_ignore_preconditions both produced optimal plans, and the astar_search h_ignore_preconditions ran much faster than astar_search h_1 (54 seconds versus 201 seconds). The astar_search h_pg_levelsum ran in 23 seconds, but produced a slightly less optimal plan than the other two (plan length of 14 vs 12).

Overall it seems that for easier planning problems like air_cargo_1, heuristic and non-heuristic search produce optimal plans and run in approximately the same time. As the planning problem becomes more difficult, heuristic search really outperforms non-heuristic search. For air_cargo_2 and air_cargo_3, breadth_first_tree_search didn't solve the problem in a reasonable amount of time. The A* heuristic searches excelled at the harder air_cargo_3 problem, with only the level_sum heuristic failing to provide an optimal solution, though the solution it provided would have likely been acceptable since it only had 2 extra actions compared to the optimal plan. One key element of this heuristic is that it expands orders of magnitude fewer nodes than the other two algorithms, and this makes it a great choice for some planning problems.

The choice of search algorithm to solve planning problems depends on the complexity of the planning problem and the runtime requirements of the user. Experimentation and multiple approaches are recommended.