AIND Planning Project
Carlos Arreaza

Part 1: Results

Part 1

Problem	Search Type	Expansions	Goal Tests	New Nodes	Path Length	Time Elapsed (sec)
P1	1 (breadth_first_search)	43	56	180	6	0.05729
P1	2(breadth_first_tree_search)	1458	1459	5960	6	1.0833
P1	3(depth_first_graph_search)	12	13	48	12	0.018636
P1	4 (depth_limited_search)	101	271	414	50	0.1459
P1	5 (uniform_cost_search)	55	57	224	6	0.065807
P2	1 (breadth_first_search)	3346	4612	30534	9	9.3459
P2	2(breadth_first_tree_search)					Took more than 10 min
P2	3(depth_first_graph_search)	1124	1125	10017	1085	8.3001199
P2	4 (depth_limited_search)					Took more than 10 min
P2	5 (uniform_cost_search)	4853	4855	4401	9	14.88
P3	1 (breadth_first_search)	14120	17673	124926	12	50.77
P3	2(breadth_first_tree_search)					Took more than 10 min
P3	3(depth_first_graph_search)	677	678	5608	660	3.469243
P3	4 (depth_limited_search)					Took more than 10 min
P3	5 (uniform_cost_search)	18235	18237	159716	12	70.4954

Part 2: Results

Part 2

Problem	Search Type	Expansions	Goal Tests	New Nodes	Path Length	Time Elapsed (sec)
P1	8 (astar_search wth h_1_	55	57	224	6	0.0691
P1	9 (astar_search with h_ignore_prec	41	43	170	6	0.05502
P1	10 (astar_search with h_pg_levels	11	13	50	6	0.9864
P2	8 (astar_search wth h_1_	4853	4855	44041	9	12.814
P2	9 (astar_search with h_ignore_prec	1450	1452	13303	9	3.78135
P2	10 (astar_search with h_pg_levels	86	88	841	9	76.088
P3	8 (astar_search wth h_1_	18235	18237	159716	12	62.511
P3	9 (astar_search with h_ignore_prec	5040	5042	44944	12	16.95
P3	10 (astar_search with h_pg_levels	318	320	2934	12	417.12

Part 3: written analysis

1) Provide an optimal plan for Problems 1, 2, and 3.

Optimal plan for problem 1:
Load(C1, P1, SFO)
Fly(P1, SFO, JFK)
Load(C2, P2, JFK)

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

Optimal plan for problem 2:
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Load(C3, P3, ATL)
Fly(P1, SFO, JFK)
Fly(P2, JFK, SFO)
Fly(P3, ATL, SFO)
Unload(C3, P3, SFO)
Unload(C2, P2, SFO)
Unload(C1, P1, JFK)

Optimal plan for problem 3:
Load(C2, P2, JFK)
Fly(P2, JFK, ORD)
Load(C4, P2, ORD)
Fly(P2, ORD, SFO)
Load(C1, P1, SFO)
Fly(P1, SFO, ATL)
Load(C3, P1, ATL)
Fly(P1, ATL, JFK)
Unload(C4, P2, SFO)
Unload(C3, P1, JFK)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)

2) Compare and contrast non-heuristic search result metrics (optimality, time elapsed, number of node expansions) for Problems 1,2, and 3. Include breadth-first, depth-first, and at least one other uninformed non-heuristic search in your comparison; Your third choice of non-heuristic search may be skipped for Problem 3 if it takes longer than 10 minutes to run, but a note in this case should be included.

The 3 non-heuristic search to be compared here are: breadth-first-search, depth-first-graph-search, and uniform-cost-search. The best of the 3 search algorithms for all 3 problems is **breadth-first-search**. It took less time than the other 2, it provided a more optimal plan (problem 1: length 6 instead of 12, problem 2: plan length 9, instead of 1085, and problem 3: plant length 12, instead of 660). Breath-first-search had an overall less node expansion than uniform-cost-search, but higher than depth-first-graph-search. The problem with depth-first-graph-search is that it provided a longer path.

3) Compare and contrast heuristic search result metrics using A* with the "ignore preconditions" and "level-sum" heuristics for Problems 1, 2, and 3.

The direct comparison of A* with ignore preconditions and level-sum heuristics can be shown in the following table:

Problem	Search Type	Expansions	Goal Tests	New Nodes	Path Length	Time Elapsed (sec)
P1	9 (astar_search with h_ignore_prec	41	43	170	6	0.05502
P1	10 (astar_search with h_pg_levels	11	13	50	6	0.9864
P2	9 (astar_search with h_ignore_prec	1450	1452	13303	9	3.78135
P2	10 (astar_search with h_pg_levels	86	88	841	9	76.088
P3	9 (astar_search with h_ignore_prec	5040	5042	44944	12	16.95
P3	10 (astar_search with h_pg_levels	318	320	2934	12	417.12

4) What was the best heuristic used in these problems? Was it better than non-heuristic search planning methods for all problems? Why or why not?

The best heuristic search algorithm was A* with ignore preconditions heuristic. It took significantly less amount of time to complete (almost an order of magnitude less than A* with level-sum heuristic), and had less node expansions than A* with h_1 heuristic.