Takahiro Ishikawa
AIND: Implement a Planning Search

Heuristic Analysis

Non-heuristic planning solution

Results of number of expansion required, number of goal tests, new nodes, time elapsed, plan length and optimal solution are below,

1. Air Cargo Problem 1

	Expansion	Goal Tests	New Nodes	Time Elapsed	Plan Length	Optimal Solution
breadth_first_search	43	56	180	0.0413	6	TRUE
breadth_first_tree_search	1458	1459	5960	1.2215	6	TRUE
depth_first_graph_search	21	22	84	0.0201	20	FALSE
depth_limited_search	101	271	414	0.1198	50	FALSE
uniform_cost_search	55	57	224	0.0497	6	TRUE

2. Air Cargo Problem 2

	Expansion	Goal Tests	New Nodes	Time Elapsed	Plan Length	Optimal Solution
breadth_first_search	3063	4274	25442	15.2185	9	TRUE
breadth_first_tree_search	-	-	-	-	-	-
depth_first_graph_search	82	83	511	0.2080	77	FALSE
depth_limited_search	-	-	-	-	-	-
uniform_cost_search	4296	4398	36069	13.0588	9	TRUE

- : Stopped search because of taking longer than 10 min

3. Air Cargo Problem 3

	Expansion	Goal Tests	New Nodes	Time Elapsed	Plan Length	Optimal Solution
breadth_first_search	14663	18098	129631	127.6693	12	TRUE
breadth_first_tree_search	-	-	-	-	-	-
depth_first_graph_search	408	409	3364	2.2969	392	FLASE
depth_limited_search	-	-	-	-	-	-
uniform_cost_search	18151	18153	159038	67.9362	12	TRUE

- : Stopped search because of taking longer than 10 min

For all problems, breadth_first_search and uniform_cost_search provide optimal solution, but other metrics are not minimum values. On the other hand, depth_first_graph_search don't provide optimal solution, but other metrics are minimum values. These means breath_first_search and uniform_cost_search can find optimal solution because the search radiate in all directions first, but using more space consumption, and depth_first_graph can reach goal at not of deep location because the search advances along one branch, but few space consumption [1].

A* searches with heuristic

Results of number of expansion required, number of goal tests, new nodes, time elapsed, plan length and optimal solution are below,

1. Air Cargo Problem 1

	Expansion	Goal Tests	New Nodes	Time Elapsed	Plan Length	Optimal Solution
A* search h_1	55	57	224	0.0564	6	TRUE
A* search h_ignore_preconditions	41	43	170	0.0595	6	TRUE
A* search h_pg_levelsum	11	13	50	0.9291	6	TRUE

2. Air Cargo Problem 2

	Expansion	Goal Tests	New Nodes	Time Elapsed	Plan Length	Optimal Solution
A* search h_1	4396	4398	36069	12.6857	9	TRUE
A* search h_ignore_preconditions	1294	1296	10927	4.7433	9	TRUE
A* search h_pg_levelsum	252	254	2054	195.1564	9	TRUE

3. Air Cargo Problem 3

	Expansion	Goal Tests	New Nodes	Time Elapsed	Plan Length	Optimal Solution
A* search h_1	18151	18153	159038	66.2141	12	TRUE
A* search h_ignore_preconditions	5038	5040	44926	20.4705	12	TRUE
A* search h_pg_levelsum	-	-	-	-	-	-

- : Stopped search because of taking longer than 10 min

h_pg_levelsum, h_ignore_preconditions and h_1 are in ascending order for expansion, goal tests and new nodes. h_ignore_preconditions, h_1 and h_pg_levelsum are in ascending order for time elapsed. Few expansion, goal tests and new nodes mean using few space. For example, h_pg_levelsum uses few space but it takes a long time elapsed. A* search with heuristic perform better as the problem complexity increased [1].

The Best heuristic

The best heuristic is h_ignore_preconditions. Because this space consumption is lower than maximum one and time elapsed is the shortest of three heuristics. Non-heuristic providing optimal solution is not good because of using more space consumption for example breath_first_search.

Optimal Plans by using the best heuristic (h_ignore_preconditions)

1. Air Cargo Problem 1

	Optimal Plan
1	Load(C1, P1, SFO)
2	Fly(P1, SFO, JFK)
3	Load(C2, P2, JFK)
4	Fly(P2, JFK, SFO)
5	Unload(C1, P1, JFK)
6	Unload(C2, P2, SFO)

2. Air Cargo Problem 2

	Optimal Plan
1	Load(C2, P2, JFK)
2	Fly(P2, JFK, ATL)
3	Load(C3, P2, ATL)
4	Fly(P2, ATL, SFO)
5	Load(C1, P1, SFO)
6	Fly(P1, SFO, JFK)
7	Unload(C1, P1, JFK)
8	Unload(C2, P2, SFO)
9	Unload(C3, P2, SFO)

3. Air Cargo Problem 3

	Optimal Plan
1	Load(C2, P2, JFK)
2	Fly(P2, JFK, ORD)
3	Load(C4, P2, ORD)
4	Fly(P2, ORD, SFO)
5	Load(C1, P1, SFO)
6	Fly(P1, SFO, ATL)
7	Load(C3, P1, ATL)
8	Fly(P1, ATL, JFK)
9	Unload(C1, P1, JFK)
10	Unload(C2, P2, SFO)
11	Unload(C3, P1, JFK)
12	Unload(C4, P2, SFO)

Reference

[1] Stuart J. Russell, Peter Norvig (2010), Artificial Intelligence: A Modern Approach (3rd Editions).