AIND - Planning Analysis

Antonio Oro

April 2017

1 Overview

Planning Search: The following documents the results from the implemented functions for the deterministic logistic planning problems for an air cargo transport system. Three different air cargo problems were defined in PDDL (Planning Domain Definition Language) as follows:

air_cargo_p1: Init(At(C1, SFO) $\wedge$ At(C2, JFK) $\wedge$ At(P1, SFO) $\wedge$ At(P2, JFK) $\wedge$ Cargo(C1) $\wedge$ Cargo(C2) $\wedge$ Plane(P1) $\wedge$ Plane(P2) $\wedge$ Airport(JFK)$\wedge$ Airport(SFO))

Goal(At(C1, JFK) $\wedge$ At(C2, SFO))

air_cargo_p2: Init(At(C1, SFO) $\wedge$ At(C2, JFK) $\wedge$ At(C3, ATL) $\wedge$ At(P1, SFO) $\wedge$ At(P2, JFK) $\wedge$ At(P3, ATL) $\wedge$ Cargo(C1) $\wedge$ Cargo(C2) $\wedge$ Cargo(C3) $\wedge$ Plane(P1) $\wedge$ Plane(P2) $\wedge$ Plane(P3) $\wedge$ Airport(JFK) $\wedge$ Airport(SFO)$\wedge$ Airport(ATL))

Goal(At(C1, JFK) $\wedge$ At(C2, SFO) $\wedge$ At(C3, SFO))

air_cargo_p3: Init(At(C1, SFO) $\wedge$ At(C2, JFK) $\wedge$ At(C3, ATL) $\wedge$ At(C4, ORD) $\wedge$ At(P1, SFO) $\wedge$ At(P2, JFK) $\wedge$ Cargo(C1) $\wedge$ Cargo(C2) $\wedge$ Cargo(C3) $\wedge$ Cargo(C4) $\wedge$ Plane(P1) $\wedge$ Plane(P2) $\wedge$ Airport(JFK) $\wedge$ Airport(SFO) $\wedge$ Airport(ATL) $\wedge$ Airport(ORD))

Goal(At(C1, JFK) $\wedge$ At(C3, JFK) $\wedge$ At(C2, SFO) $\wedge$ At(C4, SFO))

2 Results

2.1 Optimal Plans

Optimal plans could be found for all three Air Cargo Problems. The optimal plan lengths, as can be seen in table 1 are 6, 9 and 10 for the problems 1 to 3.

	Path
Problem 1	Load(C1, P1, SFO)
	Load(C2, P2, JFK)
	Fly(P1, SFO, JFK)
	Fly(P2, JFK, SFO)
	Unload(C1, P1, JFK)
	Unload(C2, P2, SFO)
Problem 2	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)
Problem 3	Load(C1, P1, SFO)
	Fly(P2, JFK, ATL)
	Load(C3, P2, ATL)
	Fly(P1, SFO, JFK)
	Unload(C1, P1, JFK)
	Fly(P2, ATL, ORD)
	Load(C4, P2, ORD)
	Fly(P2, ORD, SFO)
	Unload(C3, P2, SFO)
	Unload(C4, P2, SFO)

Table 1: Optimal paths for Air Cargo Problems 1,2 and 3

2.2 Metrics Analysis

The following table collected the results of different experiments, both for non-heuristic planning searches as well as heuristic searches.

In table 2 the *na* entries refer to methods that exceed a 10 min computation time frame and could therefore not be computed.

Comparison of non-heuristic search results: Breadth First Search and Uniform Cost Search both reach optimal solutions for all three cargo problems. Depth First Search though did not find optimal paths, but was faster then both other methods when comparing elapsed times. But why is DFS so much faster than the other methods compared? In depth-first search the idea is to travel as deep as possible from neighbour to neighbour before backtracking (see [1]). This obviously is fast, but not optimal. For finding optimal paths, Breadth First Search does a better job compared to Uniform Cost Search when comparing elapsed time as well as the number of nodes expanded. Depth First Search

Using	Expansions	Goal Test	Plan Length	Elapsed time(s)	Optimality
Air Cargo Problem 1					
breadth_first_search	43	56	6	0.026	Y
breadth_first_tree_search	1'458	1'459	6	0.81	Y
depth_first_graph_search	21	22	20	0.012	N
depth_limited_search	101	271	50	0.079	N
uniform_cost_search	55	57	6	0.035	Y
recursive_best_first_search with h_1	4'229	4'230	6	2.36	Y
greedy_best_first_graph_search with h_1	7	9	6	0.004	Y
astar_search with h_1	55	57	6	0.03	Y
astar_search with h_ignore_preconditions	41	43	6	0.04	Y
astar_search h_pg_levelsum	11	13	6	1.32	Y
Air Cargo Problem 2					
breadth_first_search	3'343	4'609	9	12.07	Y
breadth_first_tree_search	na	na	na	na	na
depth_first_graph_search	624	625	619	2.97	N
depth_limited_search	222'719	2'053'741	50	801.24	N
uniform_cost_search	4'852	4'854	9	39.42	Y
recursive_best_first_search with h_1	na	na	na	na	na
greedy_best_first_graph_search with h_1	990	992	17	6.27	N
astar_search with h_1	4'852	4'854	9	38.56	Y
astar_search with h_ignore_preconditions	1'506	1'508	9	12.34	Y
astar_search h_pg_levelsum	86	88	9	131.73	Y
Air Cargo Problem 3					
breadth_first_search	7'320	111'132	10	43.07	Y
breadth_first_tree_search	na	na	na	na	na
depth_first_graph_search	273	274	261	0.92	N
depth_limited_search	na	na	na	na	na
uniform_cost_search	12'558	12'560	10	242.77	Y
recursive_best_first_search with h_1	na	na	na	na	na
greedy_best_first_graph_search with h_1	15'455	15'457	26	245.88	N
astar_search with h_1	12'558	12'560	10	247.08	Y
astar_search with h_ignore_preconditions	2'036	2'038	10	18.55	Y
astar_search h_pg_levelsum	189	191	10	406.36	Y

Table 2: Summary table for non-heuristic and heuristic searches run on Air Cargo Problems 1, 2 and 3

expands fewer nodes and simply returns the first best solution. Finding any solutions would be an application for this method.
Hence the best choice for this experiment with non-heuristic search given the constraints is Breadth First Search.

Comparison of heuristic search results: The first heuristic which ignores the preconditions (A*_search with h_ignore_preconditions) is faster than the level sum heuristic (A*_search h_pg_levelsum). The computation for level sum is expensive. Nevertheless, comparing the number of nodes expanded, level sum does find optimal paths with less expansions.
On comparing the heuristic search with non-heuristic ones, both methods are better than the breadth first search, which was the best non-heuristic search. The comparison is based on the number of nodes expanded parameter. Considering the elapsed time metric, heuristic search performs better at least the first considered method. Heuristic search provides a solution by expanding fewer nodes as it computes which nodes may lead to the solution and propagating the estimated right path in the planning graph.

References

[1] Russell S. Norvig P. *Artificial Intelligence, A Modern Approach.* Pearson, 2014.