Heuristic analysis

Part 1. Planning problems

After experimenting with 3 given PDDL problems (their states and goals), metrics were documented for non-heuristic planning solution searches.

		Node expansions	Goal tests	Time elapsed	New nodes	Plan length
Problem 1	**breadth_first_search**	43	56	0.04	180	6
	breadth_first_tree_search	1,458	1,459	0.98	5,960	6
	depth_first_graph_search	21	22	0.02	84	20
	depth_limited_search	101	271	0.13	414	50
	uniform_cost_search	55	57	0.05	224	6
Problem 2	**breadth_first_search**	3,343	4,609	17.02	30,509	9
	breadth_first_tree_search	N/A	N/A	N/A	N/A	N/A
	depth_first_graph_search	624	625	4.26	5,602	619
	depth_limited_search	222,719	2,053,741	1,263.85	2,054,119	50
	uniform_cost_search	4,853	4,855	14.91	44,041	9
Problem 3	**breadth_first_search**	5,621	7,281	31.82	39,000	12
	breadth_first_tree_search	N/A	N/A	N/A	N/A	N/A
	depth_first_graph_search	1,292	1,293	4.11	5,744	875
	depth_limited_search	N/A	N/A	N/A	N/A	N/A
	uniform_cost_search	7,302	7,304	23.26	50,692	12

Table 1

Breadth first tree search was aborted in problems 2 and 3 because it exceeded 10 minutes, similarly depth limited search was aborted in problem 3 as it exceeded the same time limit as well; those are marked as *N/A* in the *Table 1*.

After running tests following optimal plans were computed (*Table 2*).

Problem 1	Problem 2	Problem 3

`Load(C1, P1, SFO)`
`Load(C2, P2, JFK)`
`Fly(P2, JFK, SFO)`
`Unload(C2, P2, SFO)`
`Fly(P1, SFO, JFK)`
`Unload(C1, P1, JFK)`
	`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)`
		`Load(C1, P1, SFO)`
		`Fly(P1, SFO, ATL)`
		`Load(C3, P1, ATL)`
		`Fly(P1, ATL, JFK)`
		`Load(C2, P1, JFK)`
		`Unload(C3, P1, JFK)`
		`Unload(C1, P1, JFK)`
		`Fly(P1, JFK, ORD)`
		`Load(C4, P1, ORD)`
		`Fly(P1, ORD, SFO)`
		`Unload(C4, P1, SFO)`
		`Unload(C2, P1, SFO)`

Table 2

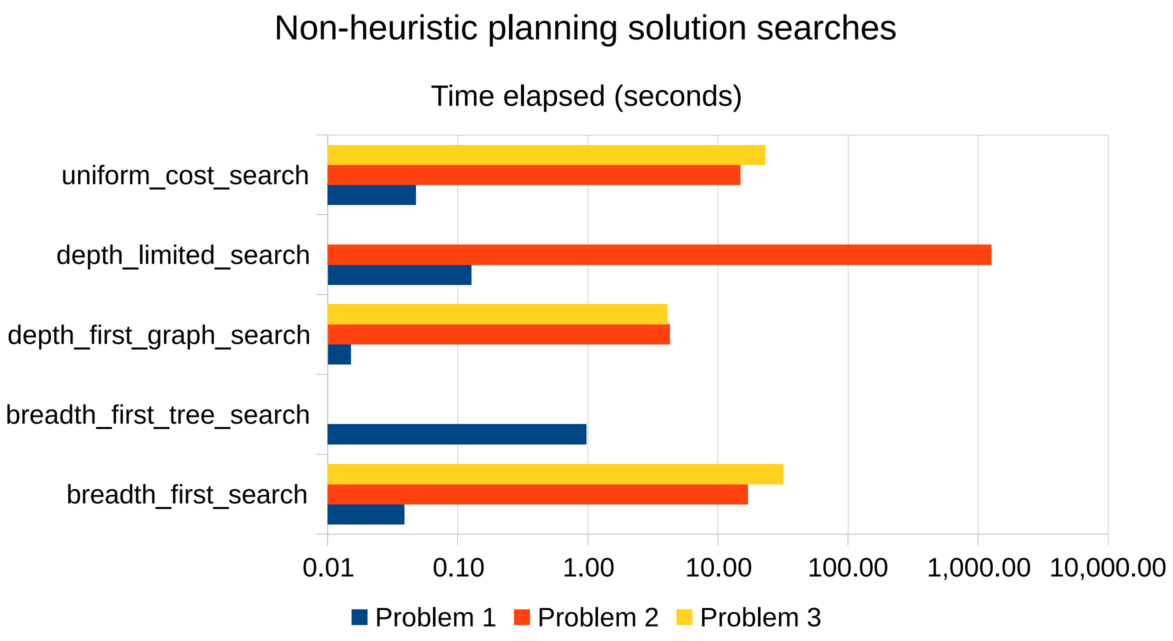

Figure 1

If algorithm is chosen based solely on time it takes to find the solution then depth first graph search is the fastest one from non-heuristic solutions (*Figure 1*), but it does not find the optimal solution, just the first matching one, e.g. in Problem 3 its plan is 875 steps (*Table 1*), where uniform cost search finds a solution with just 12 steps. Instead I would recommend using uniform cost search as it outperforms breadth first search and finds shortest search plan.

Part 2. Domain-independent heuristics

Same problems that were mentioned above were tested with graph search algorithms that used heuristics (*Table 3*).

		Node expansions	Goal tests	Time elapsed	New nodes	Plan length
Problem 1	**A* ignore preconditions**	41	43	0.05	170	6
	A* level-sum	11	13	0.73	50	6
Problem 2	**A* ignore preconditions**	1450	1452	5.25	13303	9
	A* level-sum	86	88	63.32	841	9
Problem 3	**A* ignore preconditions**	2829	2831	10.84	20879	13
	A* level-sum	167	169	78.38	1180	12

Table 3

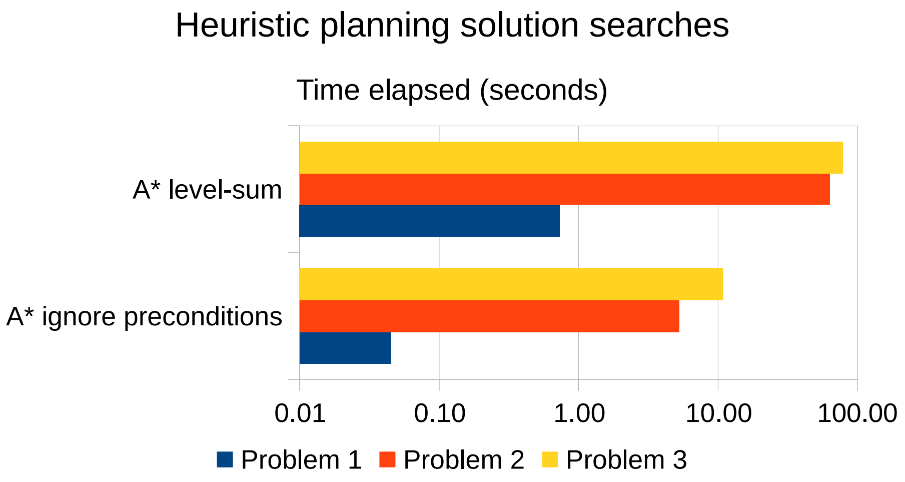

Figure 2

Although A* ignore preconditions search is faster (*Figure 2*) it expands far more nodes that A* level-sum (*Table 3*) because when algorithm ignores preconditions there are more actions allowed (edges created). Furthermore it does not calculate optimal plan length in Problem 3, this might be because A* level sum is more accurate that ignore preconditions heuristic.

Summary

Both non-heuristic and heuristic searches find optimal plans but there is no clear winner. Best non-heuristic is uniform cost search which finds optimal plan (12 steps) for Problem 3 by expanding 7302 nodes in 23.26 seconds, best heuristic A* ignore preconditions search finds worse plan (13 steps) but expands less nodes (2829) and does it under 11 seconds.

In current environment I would pick A* ignore preconditions search as being the optimal one. But to my mind it would fail if goal complexity increases and its accuracy diminishes.