heuristic analysis

Zhenyuan Liu

January 2018

1 Discussion

The search result metrics (optimality, time elapsed, number of node expansions) are summarized in Table 1-3. The optimal plan for all the problems is A* with the "ignore preconditions" heuristic, if the problem is time-limited. In contrast, if the problem is space-limited, then A* with the "level-sum" heuristic is the optimal solution. All the plans found by various methods are summarized in section 2.

Method	**#expansions**	**# tests**	**time**	**optimality**
breadth first	43	56	0.033	6
depth first graph	12	13	0.009	12
uniform cost	55	57	0.035	6
A∗ (ignore precondition)	41	43	0.0357	6
A∗ (level sum)	11	13	0.4903	6

Table 1: Results of various methods in solving problem 1

Method	**#expansions**	**# tests**	**time**	**optimality**
breadth first	3343	4609	7.579	9
depth first graph	582	583	3.247	575
uniform cost	4853	4855	10.571	9
A∗ (ignore precondition)	1450	1452	4.1679	9
A∗ (level sum)	86	88	44.605	9

Table 2: Results of various methods in solving problem 2

Method	**#expansions**	**# tests**	**time**	**optimality**
breadth first	14663	18098	43.060	12
depth first graph	627	628	3.043	596
uniform cost	18151	18153	52.439	12
A∗ (ignore precondition)	5038	5040	15.049	12
A∗ (level sum)	314	316	205.706	12

Table 3: Results of various methods in solving problem 3

We first discuss the three non-heuristic search methods, namely breadth-first search, depth-first-graph search, and uniform cost search. Depth-first-graph search takes the shortest time at the cost of its optimality. The plan found by depth-first-graph has unrealistically too many steps, because depth-first algorithm can search very deep along a given branch, as explained in the video. Therefore, it's not practical to use depth-first-graph search. Uniform cost search takes longer time and expands more nodes than breadth-first search.In fact, breadth-first would never search paths deeper than the goal path it finds when it returns. Whereas, it's possible that uniform cost searches paths deeper than the goal path. This explains why Uniform cost search takes longer time and expands more nodes than breadth-first search. These facts are all explained in the video. Both of them are able to find the plan as good as the plan found by A*.
Next, we discuss the 2 search methods with heuristics. As shown in the table, the "ignore preconditions" heuristic is much faster than the "level-sum" heuristic, because calculating the "level-sum" is expensive, details are provided in the Norvig and Russel's textbook. However, A* with the "level-sum" heuristic expands much less nodes and goal tests, because the search space is polynomial instead of exponential, as guaranteed by the plan graph data structure, also explained in Norvig and Russel's textbook.
To conclude, none of the heuristic outperforms the other in every aspect. The "level-sum" heuristic requires less space, whereas the "ignore preconditions" heuristic requires less computation time. A* with the "ignore preconditions" heuristic is better than all non-heuristic search planning methods. A* with the "level-sum" heuristic is NOT better than the non-heuristic search planning methods. However, we expect A* with the "level-sum" heuristic to be much better in solving large-scale problems. Again the reason is that the plan graph data structure is polynomial.

2 Plans found by different methods

2.1 Problem 1

Breadth-first
Load(C2, P2, JFK)
Load(C1, P1, SFO)

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

Depth-first graph
Fly(P1, SFO, JFK)
Fly(P2, JFK, SFO)
Load(C1, P2, SFO)
Fly(P2, SFO, JFK)
Fly(P1, JFK, SFO)
Unload(C1, P2, JFK)
Fly(P2, JFK, SFO)
Fly(P1, SFO, JFK)
Load(C2, P1, JFK)
Fly(P2, SFO, JFK)
Fly(P1, JFK, SFO)
Unload(C2, P1, SFO)

Uniform cost
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P1, SFO, JFK)
Fly(P2, JFK, SFO)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)

A* with "ignore preconditions" heuristic
Load(C1, P1, SFO)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)
Load(C2, P2, JFK)
Fly(P2, JFK, SFO)
Unload(C2, P2, SFO)

A* with "level-sum" heuristic
Load(C1, P1, SFO)
Fly(P1, SFO, JFK)
Load(C2, P2, JFK)
Fly(P2, JFK, SFO)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)

2.2 Problem 2

Breadth-first
Load(C2, P2, JFK)
Load(C1, P1, SFO)
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)
Depth-first graph, in total 575 steps
Fly(P3, ATL, SFO)
Fly(P1, SFO, ATL)
Fly(P3, SFO, JFK)
Fly(P1, ATL, JFK)
....
Unload(C3, P1, SFO)
Uniform cost
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(C1, P1, JFK)
Unload(C2, P2, SFO)
Unload(C3, P3, SFO)
A* with "ignore preconditions" heuristic
Load(C1, P1, SFO)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)
Load(C2, P2, JFK)
Fly(P2, JFK, SFO)
Unload(C2, P2, SFO)
Load(C3, P3, ATL)
Fly(P3, ATL, SFO)
Unload(C3, P3, SFO)
A* with "level-sum" heuristic
Load(C1, P1, SFO)
Fly(P1, SFO, JFK)
Load(C2, P2, JFK)
Fly(P2, JFK, SFO)
Load(C3, P3, ATL)
Fly(P3, ATL, SFO)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)
Unload(C3, P3, SFO)

2.3 Problem 3

Breadth-first
Load(C2, P2, JFK)
Load(C1, P1, SFO)
Fly(P2, JFK, ORD)
Load(C4, P2, ORD)
Fly(P1, SFO, ATL)
Load(C3, P1, ATL)
Fly(P1, ATL, JFK)
Unload(C1, P1, JFK)
Unload(C3, P1, JFK)
Fly(P2, ORD, SFO)
Unload(C2, P2, SFO)
Unload(C4, P2, SFO)
Depth-first graph, in total 596 steps
Fly(P1, SFO, ORD)
Fly(P2, JFK, ORD)
Fly(P1, ORD, ATL)
Fly(P2, ORD, ATL)
....
Fly(P1, ORD, ATL)
Unload(C4, P2, SFO)
Uniform cost
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P1, SFO, ATL)
Load(C3, P1, ATL)
Fly(P2, JFK, ORD)
Load(C4, P2, ORD)
Fly(P2, ORD, SFO)
Fly(P1, ATL, JFK)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)
Unload(C3, P1, JFK)
Unload(C4, P2, SFO)
A* with "ignore preconditions" heuristic
Load(C1, P1, SFO)
Fly(P1, SFO, ATL)
Load(C3, P1, ATL)
Fly(P1, ATL, JFK)
Unload(C1, P1, JFK)
Load(C2, P2, JFK)
Fly(P2, JFK, ORD)
Load(C4, P2, ORD)
Fly(P2, ORD, SFO)

Unload(C2, P2, SFO)
Unload(C3, P1, JFK)
Unload(C4, P2, SFO)
A* with "level-sum" heuristic
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(C1, P1, JFK)
Unload(C2, P2, SFO)
Unload(C3, P1, JFK)
Unload(C4, P2, SFO)