AIND-Planning Module

Heuristic Analysis

Aaron Caroltin

June 05, 2017

Resubmission: July 23, 2017

Action Schema for Air Cargo Problems

```
Action(Load(c, p, a),
        PRECOND: At(c, a) ∧ At(p, a) ∧ Cargo(c) ∧ Plane(p) ∧ Airport(a)
        EFFECT: ¬ At(c, a) ∧ In(c, p))
Action(Unload(c, p, a),
        PRECOND: In(c, p) ∧ At(p, a) ∧ Cargo(c) ∧ Plane(p) ∧ Airport(a)
        EFFECT: At(c, a) ∧ ¬ In(c, p))
Action(Fly(p, from, to),
        PRECOND: At(p, from) ∧ Plane(p) ∧ Airport(from) ∧ Airport(to)
        EFFECT: ¬ At(p, from) ∧ At(p, to))
```

Optimal Sequence of Actions

Problem-1	Problem-2	Problem-3
Optimal plan length: 6 Load(C1, P1, SFO) Fly(P1, SFO, JFK) Unload(C1, P1, JFK) Load(C2, P2, JFK) Fly(P2, JFK, SFO) Unload(C2, P2, SFO)	Optimal plan length: 9 Load(C3, P3, ATL) Fly(P3, ATL, SFO) Unload(C3, P3, SFO) Load(C1, P1, SFO) Fly(P1, SFO, JFK) Unload(C1, P1, JFK) Load(C2, P2, JFK) Fly(P2, JFK, SFO) Unload(C2, P2, SFO)	Optimal plan length: 12 Load(C2, P2, JFK) Fly(P2, JFK, ORD) Load(C4, P2, ORD) Fly(P2, ORD, SFO) Unload(C4, P2, SFO) Load(C1, P1, SFO) Fly(P1, SFO, ATL) Load(C3, P1, ATL) Fly(P1, ATL, JFK) Unload(C3, P1, JFK) Unload(C1, P1, JFK) Unload(C2, P2, SFO)

Uninformed Search Analysis and Results

Uninformed search problems are solved mostly using brute force algorithms (of time & memory usage). Such algorithms are non-heuristic in nature where they have little information about

the domain or the problem state beyond necessary. They are limited to generate successor states using applicable actions and able to identify goal states from non-goal states.

We found optimal path of 6, 9, and 12 respectively for problems 1, 2, 3. ***breath_first*** and ***uniform_cost*** are the only ones that are optimal across all problems. ***breath_first_tree***, ***depth_limited*** and ***recursive_best_first*** took more than 10 minutes to complete (for problem 2, 3) and hence aborted and ignored for further comparison.

Usually in Air Cargo planning problems (real-world), the cost of executing an action is relatively time consuming and involves money ex: Fly (P2, JFK, SFO). Even though each step-cost may vary within a given plan, non-heuristic agents will never able to identify or take advantage of them. Hence it becomes imperative to judge an agent / algorithm first - with its ability to come with optimal path and only then consider its execution time or efficiency.

Algorithms	Problem-1					Problem-2					Problem-3				
	Expansions	Goal Tests	New Nodes	Path Length	Time (sec)	Expansions	Goal Tests	New Nodes	Path Length	Time (sec)	Expansions	Goal Tests	New Nodes	Path Length	Time (sec)
breadth_first	43	56	180	6	0.04	3343	4609	30509	9	14.05	14663	18098	129631	12	106.85
breadth_first_tree	1458	1459	5960	6	1.08	-	-	-	-	>10m	-	-	-	-	>10m
depth_first_graph	12	13	48	12	0.01	582	583	5211	575	3.16	627	628	5176	596	3.50
depth_limited	101	271	414	50	0.10	222719	2053741	2054119	50	1038	-	-	-	-	>10m
uniform_cost	55	57	224	6	0.04	4853	4855	44041	9	12.64	18223	18225	159618	12	55.55
recursive_best_first	4229	4230	17029	6	3.07	-	-	-	-	>10m	-	-	-	-	>10m
greedy_best_first_graph	7	9	28	6	0.005	998	1000	8982	21	2.62	5578	5580	49150	22	17.20

Informed Search Analysis and Results

Informed search problems are solved mostly using intelligent algorithms which are heuristic in nature where the agent is taught to infer state variables and take advantage of extensive domain knowledge for building subsequent steps for the plan. They are usually directed to come with optimal path if solution is complete.

We found optimal path of 6, 9, and 12 respectively for problems 1, 2, 3 using all three heuristics with ***A* search*** algorithm.

h_1 is the most unintelligent agent here and as complexity of problem increases, it falls out of favor with unacceptably high memory utilization (new node generation).

A* heuristics	Problem-1					Problem-2					Problem-3				
	Expansions	Goal Tests	New Nodes	Path Length	Time (sec)	Expansions	Goal Tests	New Nodes	Path Length	Time (sec)	Expansions	Goal Tests	New Nodes	Path Length	Time (sec)
h_1	55	57	224	6	0.04	4853	4855	44041	9	12.70	18223	18225	159618	12	55.42
h_ignore _preconditions	41	43	170	6	0.04	1450	1452	13303	9	4.66	5040	5042	44944	12	18.36
h_pg _levelsum	11	13	50	6	0.58	86	88	841	9	44.79	325	327	3002	12	227.34

Observation

For problems 1, 2, 3 with respective state space [Ref: 1] of 2^{12}, 2^{27},2^{32}

For uninformed searches: ***uniform_cost*** opens up more nodes compared to ***breath_first*** or ***depth_first_graph*** because even after reaching to goal state, it tries to find if there exist any cheaper path cost [Ref: 2, 3]. ***breath_first*** requires to find shortest routes in bi-directional ways requiring to revisit nodes (duplicates) but with just two cities to travel in problem 1, the issue is not magnified [Ref: 2, 3] but becomes apparent with problem 2, 3. ***depth_first_graph*** is not viable as it has to traverse from bottom left node to bottom right node and solution could be neither optimal nor complete.

For informed searches: ***A* h_pg_levelsum*** uses a planning graph [Ref: 4, 5] and estimates the sum of all actions to be carried from the current state to satisfy every goal conditions which may involve re-executing / duplicating certain actions whereas ***A* h_ignore_preconditions*** finds the goal faster by estimating the minimum number of actions required to execute from current state to satisfy all goal conditions and also by ignoring any preconditions set on those actions, making the problem easier to solve [Ref: 5, 6]

Conclusion

For uninformed / non-heuristic search, our recommendation is restricted to ***uniform_cost***, when execution time is critical or ***breath_first*** when execution efficiency is critical (least nodes/memory used).

For informed / heuristic search, we recommend A* search with ***h_ignore_preconditions*** when execution time is critical and A* search with ***h_pg_levelsum*** when memory/space efficiency is critical.

Reference

1. AIND – Search video Lesson 8 slide 35 – State Space
2. AIND – Search video Lesson 8 slide 23 – Search Comparison
3. AIND – Search video Lesson 8 slide 26 - Uniform Cost
4. AIND – Search video Lesson 8 slide 33 – Optimistic Heuristic
5. AIND – Planning video Lesson 12 slide 17, 18 – Plan Space Search
6. AIMA book 3rd edition – Lesson 10.2.3 Heuristics for planning p382-383
7. Github repository for AIMA pseudo code reference
   https://github.com/aimacode/aima-python