Self Training

September 26, 2017

Isolation Heuristic Analysis

Ngoc Minh VU

Version 0.1

– Draft –

Abstract

We summarize obtained results for **Air-Cargo Planning Problem**.

Contents

1 Problem Context - Air Cargo Planning

In this project, we want to plan a list of actions in order to arrive at goal-state from a given initial state. We will work on Air-Cargo with the following **Action-Schema**

$$
\begin{aligned}
\texttt{Action}(\quad & \texttt{Load}(c, p, a), \\
& \texttt{PRECOND}: \texttt{At}(c, a) \wedge \texttt{At}(p, a) \wedge \texttt{Cargo}(c) \wedge \texttt{Plane}(p) \wedge \texttt{Airport}(a) \\
& \texttt{EFFECT}: \neg\texttt{At}(c, a) \wedge \texttt{In}(c, p)) \\
\texttt{Action}(\quad & \texttt{Unload}(c, p, a), \\
& \texttt{PRECOND}: \texttt{In}(c, p) \wedge \texttt{At}(p, a) \wedge \texttt{Cargo}(c) \wedge \texttt{Plane}(p) \wedge \texttt{Airport}(a) \\
& \texttt{EFFECT}: \texttt{At}(c, a) \wedge \neg\texttt{In}(c, p)) \\
\texttt{Action}(\quad & \texttt{Fly}(p, from, to), \\
& \texttt{PRECOND}: \texttt{At}(p, from) \wedge \texttt{Plane}(p) \wedge \texttt{Airport}(from) \wedge \texttt{Airport}(to) \\
& \texttt{EFFECT}: \neg\texttt{At}(p, from) \wedge \texttt{At}(p, to))
\end{aligned}
$$

We are given three following problems' state and goal

- **Problem 1**

$$
\begin{aligned}
\texttt{Init}\quad & (\texttt{At}(C1, SFO) \wedge \texttt{At}(C2, JFK) \\
& \wedge\texttt{At}(P1, SFO) \wedge \texttt{At}(P2, JFK) \\
& \wedge\texttt{Cargo}(C1) \wedge \texttt{Cargo}(C2) \\
& \wedge\texttt{Plane}(P1) \wedge \texttt{Plane}(P2) \\
& \wedge\texttt{Airport}(JFK) \wedge \texttt{Airport}(SFO)) \\
\texttt{Goal}\quad & (\texttt{At}(C1, JFK) \wedge \texttt{At}(C2, SFO))
\end{aligned}
$$

- **Problem 2**

$$
\begin{aligned}
\texttt{Init}\quad & (\texttt{At}(C1, SFO) \wedge \texttt{At}(C2, JFK) \wedge \texttt{At}(C3, ATL) \\
& \wedge\texttt{At}(P1, SFO) \wedge \texttt{At}(P2, JFK) \wedge \texttt{At}(P3, ATL) \\
& \wedge\texttt{Cargo}(C1) \wedge \texttt{Cargo}(C2) \wedge \texttt{Cargo}(C3) \\
& \wedge\texttt{Plane}(P1) \wedge \texttt{Plane}(P2) \wedge \texttt{Plane}(P3) \\
& \wedge\texttt{Airport}(JFK) \wedge \texttt{Airport}(SFO) \wedge \texttt{Airport}(ATL)) \\
\texttt{Goal}\quad & (\texttt{At}(C1, JFK) \wedge \texttt{At}(C2, SFO) \wedge \texttt{At}(C3, SFO))
\end{aligned}
$$

- **Problem 3**

$$
\begin{aligned}
\texttt{Init}\quad & (\texttt{At}(C1, SFO) \wedge \texttt{At}(C2, JFK) \wedge \texttt{At}(C3, ATL) \wedge \texttt{At}(C4, ORD) \\
& \wedge\texttt{At}(P1, SFO) \wedge \texttt{At}(P2, JFK) \\
& \wedge\texttt{Cargo}(C1) \wedge \texttt{Cargo}(C2) \wedge \texttt{Cargo}(C3) \wedge \texttt{Cargo}(C4) \\
& \wedge\texttt{Plane}(P1) \wedge \texttt{Plane}(P2) \\
& \wedge\texttt{Airport}(JFK) \wedge \texttt{Airport}(SFO) \wedge \texttt{Airport}(ATL) \wedge \texttt{Airport}(ORD)) \\
\texttt{Goal}\quad & (\texttt{At}(C1, JFK) \wedge \texttt{At}(C3, JFK) \wedge \texttt{At}(C2, SFO) \wedge \texttt{At}(C4, SFO))
\end{aligned}
$$

2 Search Result Metrics

Now we solve the three planning problems using the following uninformed search strategies

- **Breadth First Search**: with flag `-s 1`
- **Depth First Search**: with flag `-s 3`
- **Uniform Cost Search**: with flag `-s 5`

And the following informed search strategies

- **Greedy best-first search**: with flag `-s 7`
- **A* Search with heuristic `h_ignore_predonditions`**: with flag `-s 9`
- **A* Search with heuristic `h_pg_levelsum`**: with flag `-s 10`

We recall that informed search strategies employ a **heuristic function** $h(n)$ that estimates best cost from the state at node n to a goal state and

- **Greedy best-first search**: tries to expand the node that is closest to the goal i.e

$$\arg\min_n h(n)$$

- **A* search**: tries to expand to the cheapest estimated solution i.e

$$\arg\min_n f(n) + h(n)$$

  where $f(n)$ is the cost from initial state to the state at node n.

We obtain the following results (cell Yes in green means found solution is ensured to be optimal)

Table 1: Problem 1 - Metrics

Search Type	Expansions	Goal Tests	New Nodes	Plan length	Time (s)	Optimal
Breadth First Search	43	56	180	6	0.0585	Yes
Death First Search	21	22	84	20	0.0292	No
Uniform Cost Search	55	57	227	6	0.0646	Yes
Greedy best-first Search	7	9	28	6	0.0111	Yes
A* h_ignore_precond	41	43	170	6	0.0620	Yes
A* h_pg_levelsum	11	13	50	6	0.7587	Yes

Table 2: Problem 2 - Metrics

Search Type	Expansions	Goal Tests	New Nodes	Plan length	Time (s)	Optimal
Breadth First Search	3343	4609	30509	9	14.3175	Yes
Death First Search	624	625	5602	619	3.5660	No
Uniform Cost Search	4853	4855	44041	9	12.1856	Yes
Greedy best-first Search	895	897	8013	21	2.2762	No
A* h_ignore_precond	1450	1452	13303	9	4.6142	Yes
A* h_pg_levelsum	86	88	841	9	65.0560	Yes

Table 3: Problem 3 - Metrics

Search Type	Expansions	Goal Tests	New Nodes	Plan length	Time (s)	Optimal
Breadth First Search	14663	18098	129631	12	103.6890	Yes
Death First Search	408	409	3364	392	1.7997	No
Uniform Cost Search	18233	18235	159697	12	53.2599	Yes
Greedy best-first Search	5185	5187	45704	17	15.2546	Yes
A* h_ignore_precond	4951	4953	44051	12	17.0359	Yes
A* h_pg_levelsum	306	308	2825	12	311.5720	Yes

Looking at above metrics, we notice that

- BFS, UCS and A* h_ignore_precond search always found optimal solution.
- Informed search strategies perform better than un-informed ones when problem's complexity increase. As we can see A* h_ignore_precond is 3/4 times faster than BFS/UCS for problem 2 and 3.
- DFS uses less memory than BFS/UCS but the solution is very far from optimal
- Greedy best-first search is better than DFS but its solution is still not optimal
- A* h_pg_levelsum seems having the best heuristic (very few node-expansion/goal tests/new nodes), however due to its complexity of the heuristic, it runs slowest comparing to the others strategies

Let's explain above result

- From [1] p. 85-86, we know that DFS always expands the deepest node in the current frontier of the search tree which is the reason that its solution is often not optimal (e.g in our Air-Cargo it will explore all available action from current state and try to apply an action while that action might not be helpful to arrive at the goal).
- In constrat, BFS, UCS are optimal because it always expands the shallowest unexpanded node [1] p. 83. Regarding A*, to ensure that it find an optimal solution, we need to ensure that its heuristic function is admissible and consistent.

  It's clear that `h_ignore_precond` is admissible and consistent since it verifies

  - It always estimates an lower bound of number of the actions to achieve the goals: since for each Cargo that is not at the required destination, we need at least an Unload action.
  - For any node n', we need to prove

    $$h(n) \leq c(n, a, n') + h(n')$$

    If $h(n') \leq h(n)$, then we don't need to prove anything, in the case when $h(n') < h(n)$, similar as above for each goal that not satisfied in n but satisfied in n' we need at least one action Unload so we must have

    $$c(n, a, n') \geq h(n) - h(n')$$

  That concludes the proof of `h_ignore_precond` is both admissible and consistent which implies that A* `h_ignore_precond`'s solution will be optimal.

- From [1] p. 382, we know that `h_pg_levelsum` can be inadmissible so we can NOT ensure that it always return optimal solution. However in practice, it works well (as we seen in above metrics).
- From [1] p. 92-93 we know that Greedy best-first search at each step it tries to get as close to the goal as it can, however its solution might not be optimal since it ignores the cost to arrive at current state.

Combining time/precision A* with h_ignore_precond is the best strategies for this Air-Cargo Planning problem. Here is the optimal plan for the three problems

Table 4: Optimal Plan

Problem	Search Type	Optimal Plan
Problem 1	A* h_ignore_precond	Load(C1, P1, SFO) Fly(P1, SFO, JFK) Unload(C1, P1, JFK) Load(C2, P2, JFK) Fly(P2, JFK, SFO) Unload(C2, P2, SFO)
Problem 2	A* h_ignore_precond	Load(C3, P3, ATL) Fly(P3, ATL, SFO) Unload(C3, P3, SFO) Load(C2, P2, JFK) Fly(P2, JFK, SFO) Unload(C2, P2, SFO) Load(C1, P1, SFO) Fly(P1, SFO, JFK) Unload(C1, P1, JFK)
Problem 3	A* h_ignore_precond	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(C2, P2, SFO) Unload(C1, P1, JFK)

3 Conclusition

Go through this project we have learnt how to solve a planning problem by using PDDL and general search strategies (BFS/DFS/A*). We also experiment the performance of heuristic/informed search v.s uninformed search.

References

[1] S. Russell and P. Norvig (2009) *Artificial Intelligence: A Modern Approach* (3rd Edition) 2