Heuristic Analysis - Planning

Frederick Chyan

Aug 17, 2017

Problem Description

Classical PDDL problems given below

Air Cargo Action Schema:

```
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))
```

Problem 1:

```
Init(At(C1, SFO) ∧ At(C2, JFK)
    ∧ At(P1, SFO) ∧ At(P2, JFK)
    ∧ Cargo(C1) ∧ Cargo(C2)
    ∧ Plane(P1) ∧ Plane(P2)
    ∧ Airport(JFK) ∧ Airport(SFO))
Goal(At(C1, JFK) ∧ At(C2, SFO))
```

Problem 2:

```
Init(At(C1, SFO) ∧ At(C2, JFK) ∧ At(C3, ATL)
    ∧ At(P1, SFO) ∧ At(P2, JFK) ∧ At(P3, ATL)
    ∧ Cargo(C1) ∧ Cargo(C2) ∧ Cargo(C3)
    ∧ Plane(P1) ∧ Plane(P2) ∧ Plane(P3)
    ∧ Airport(JFK) ∧ Airport(SFO) ∧ Airport(ATL))
Goal(At(C1, JFK) ∧ At(C2, SFO) ∧ At(C3, SFO))
```

Problem 3:

```
Init(At(C1, SFO) ∧ At(C2, JFK) ∧ At(C3, ATL) ∧ At(C4, ORD)
    ∧ At(P1, SFO) ∧ At(P2, JFK)
    ∧ Cargo(C1) ∧ Cargo(C2) ∧ Cargo(C3) ∧ Cargo(C4)
    ∧ Plane(P1) ∧ Plane(P2)
    ∧ Airport(JFK) ∧ Airport(SFO) ∧ Airport(ATL) ∧ Airport(ORD))
Goal(At(C1, JFK) ∧ At(C3, JFK) ∧ At(C2, SFO) ∧ At(C4, SFO))
```

Result

Problem 1

Algorithm	Expansions	Goal Tests	New Nodes	Elapsed Time (s)	Path Length	Optimal
Breadth First Search	43	56	180	0.030490892	6	YES
Breadth First Tree Search	1458	1459	5960	0.947382754	6	YES
Depth First Graph Search	21	22	84	0.012808286	20	NO
Greedy Best First Graph Search	7	9	28	0.006106681	6	YES
A Star Search Heuristic: Ignore Preconditions	41	43	170	0.048111511	6	YES
A Star Search Heuristic: Planning Graph Level Sum	11	13	50	0.796748964	6	YES

Problem 2

Algorithm	Expansions	Goal Tests	New Nodes	Elapsed Time (s)	Path Length	Optimal
Breadth First Search	3343	4609	30509	13.43590876	9	YES
Breadth First Tree Search				Timeout		
Depth First Graph Search	624	625	5602	3.450186569	619	NO
Greedy Best First Graph Search	998	1000	8982	2.335751873	17	NO
A Star Search Heuristic: Ignore Preconditions	1450	1452	13303	4.173740015	9	YES
A Star Search Heuristic: Planning Graph Level Sum	86	88	841	68.66236802	9	YES

Problem 3

Algorithm	Expansions	Goal Tests	New Nodes	Elapsed Time (s)	Path Length	Optimal
Breadth First Search	14663	18098	129631	110.2354232	12	YES
Breadth First Tree Search				Timeout		
Depth First Graph Search	408	409	3364	1.73957705	392	NO
Greedy Best First Graph Search	5398	5400	47665	14.53414782	26	NO
A Star Search Heuristic: Ignore Preconditions	5038	5040	44926	14.85381065	12	YES
A Star Search Heuristic: Planning Graph Level Sum	314	316	2894	329.5419026	12	YES

Optimal Plan

Below are the optimal plans for the given problems. They are obtained by breadth first search or A* search.

Problem 1:

```
Path length: 6
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P2, JFK, SFO)
Unload(C2, P2, SFO)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)
```

Problem 2:

```
Path length: 9
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:

```
Path length: 12
Load(C1, P1, SFO)
Load(C2, P2, JFK)
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)
```

Non-heuristic Search

As demonstrated in the result, **breadth first search** always return optimal result. This is because BFS will check all nodes at each depth incrementally. Therefore, the first node that meets the goal is necessarily optimal. However, this is exhaustive and brute force. **Depth first search**, on the other hand, terminates much faster, but produces non-optimal result (e.g. path length of 392 vs. optimal path length of 12). This is understandable, as it simply keep trying various action until the goal is met. While fast, the resulting plan is almost useless. For completeness, **tree search** result is included here. Tree search is not effective in these problems as there might be loops in the search path so it might never terminate, also, duplicated states will be searched as visited nodes are not recorded. **Greedy best first graph search** looks to be a pretty good compromise, it achieves optimal result in problem 1 and problem 2 and achieves a reasonable result in problem 3 (path length: 26) which is slightly worse than optimal (12) but a lot better than DFS (path length: 392).

Heuristic Search using A*

Both yield optimal result. Also, it is worth pointing out that `ignore preconditions` is faster than BFS, therefore it would be good search strategy when optimal result is necessary. Comparing `ignore preconditions` with `level-sum` showed some interesting facts. The `level-sum` heuristic expands a lot less nodes but takes much longer time. This is because every time the `level-sum` heuristic function is evaluated, it needs to create a new Planning Graph and compute the heuristic value. This means that, while the `level-sum` heuristic explores fewer nodes, there are overheads such as having to recreate objects every time at different states.

Conclusion

The best overall search strategy should be used is **A* with the "ignore preconditions" heuristic**. It is guaranteed to provide optimal plan[1], and terminates faster than BFS. This is more evident when the search space is larger like in problem 3, where the heuristic search is 10 times faster than non-heuristic search. Simple heuristic search expands less nodes than BFS, and there isn't too much computational overhead unlike planning graph level sum heuristic.

---

1. Russell, S. J., & Norvig, P. (2010). Artificial intelligence: A modern approach. Pearson Education, Inc. p. 376 ↩