Heuristic Analysis

Ilya Nikokoshev

February 25, 2018

Abstract

As part of the Project 3 in the Artificial Intelligence Nanodegree Program, we implement a planning search agent to solve deterministic logistics planning problems using an example of an Air Cargo transport system.

1 Problem Solutions

Problem 1 Solution

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

Problem 2 Solution

```
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 Solution

```
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)
```

2 Non-heuristic Searches

We present results for the runs of three uninformed planning strategies in the Table 1. The selected strategies are Breadth First Search (BFS), Uniform Cost Search (UCS) and Depth First Graph Search (DFGS). Those strategies do not use any additional information about the structure of the problem states.

The BFS algorithm works by traversing all possible solution paths from the path of length 0, then all of the paths of length 1, then all the paths of length 2 et cetera. When we find a path of length i that leads to a solution, it is guaranteed that the this solution is optimal, since we have already considered and discarded all paths of length $\leq i - 1$.

Regarding the UCS search method, the cost of a solution path that is used to define it is implemented simply as the number of actions (to see this, note the recursive definition of `Problem.path_cost` in `./aimacode/search.py` as the cost of the parent path +1 as well as the base case with cost 0). Therefore, the algorithm generally works similarly to BFS. However, UCS is defined to be able to

accept any admissible cost function. This means that for our particular case the implementation will continue to expand the nodes even once it finds a path of length i, until it goes through all paths of this length.

From this theoretical analysis follows that both BFS and UCS will find optimal solutions, and that UCS will always expand not less nodes and perform not less goal tests in comparison to BFS. On the other hand, there is no reason to think that DFGS will find solutions that are optimal. Our experimental results confirm all of these statements.

Figure 3.21 in Section 3.4.7 goes into more detail regarding the time and space complexity of selected methods. In particular, Depth First Graph Search has its own advantages as it uses less memory and is practically faster in our runs. However, we see that the solutions it finds are very far from optimal.

Table 1: Results for Non-heuristic planinng methods

	Search Method	Length	Optimal?	Time Elapsed	Expansions	Goal Tests
1	`breadth_first_search`	6	Yes	0.15	43	56
1	`depth_first_graph_search`	12	No	0.04	12	13
1	`uniform_cost_search`	6	Yes	0.19	55	57
2	`breadth_first_search`	9	Yes	45.23	3343	4609
2	`depth_first_graph_search`	575	No	10.36	582	583
2	`uniform_cost_search`	9	Yes	69.89	4853	4855
3	`breadth_first_search`	12	Yes	255.13	14663	18098
3	`depth_first_graph_search`	596	No	7.58	627	628
3	`uniform_cost_search`	12	Yes	184.44	18223	18225

3 Heuristic Searches

We present results for the runs of an A* search with two heuristics, Ignore Preconditions (`h_ignore_preconditions`) and Level Sum (`h_pg_levelsum`), in the Table 2.

As proven in Section 3.5.2, A* search always finds optional solution, provided that the heuristic is consistent. A quick proof of this statement can be obtained by noting that we only declare a solution of total cost T when we will examine and reject all paths of total cost $< T$. Since total cost for a solution is equal to path cost, and total cost for a non-solution is never less than path cost, the solution we find will be optimal.

This statement is confirmed by our results.

The results show that using A* search with Level Sum heuristics requires less Node Expansions and Goal Tests to find optional solutions than any other method. However, this heuristic itself is computationally expensive and using Ignore Preconditions allows us to perform the search much faster in practice, dominating all other optimal methods.

Table 2: Results for Heuristic planinng methods

	Heuristic	Length	Optimal?	Time Elapsed	Expansions	Goal Tests
1	h_ignore_preconditions	6	Yes	0.10	41	43
1	h_pg_levelsum	6	Yes	0.69	11	13
2	h_ignore_preconditions	9	Yes	12.85	1450	1452
2	h_pg_levelsum	9	Yes	67.36	86	88
3	h_ignore_preconditions	12	Yes	59.05	5040	5042
3	h_pg_levelsum	12	Yes	336.54	325	327

4 Conclusion

We have compared five different strategies of finding a solution to the Air Cargo problem. Using informed search we were able to achieve a good balance by finding an optional solution faster using A* search with domain-independent Ignore Preconditions heuristic.