AIND
Alexandru Oprisan

# Planning Project
# Heuristic Analisys

## I. Metrics

| Problem # | search method | expansions | tests | new nodes | seconds | plan lengt | is optimal |
|---|---|---|---|---|---|---|---|
| Air Cargo 1 | A* no heuristic | 55 | 57 | 224 | 0.03 | 6 | yes |
| Air Cargo 1 | A* ignore_preconditions | 41 | 43 | 170 | 0.03 | 6 | yes |
| Air Cargo 1 | A* pg_levelsum | 11 | 13 | 50 | 0.62 | 6 | yes |
| Air Cargo 1 | breadth_first_search | 43 | 56 | 180 | 0.03 | 6 | yes |
| Air Cargo 1 | depth_first_graph_search | 12 | 13 | 48 | 0.01 | 12 | NO |
| Air Cargo 1 | uniform_cost_search | 55 | 57 | 224 | 0.03 | 6 | yes |
| Air Cargo 2 | A* no heuristic | 4853 | 4855 | 44041 | 10.33 | 9 | yes |
| Air Cargo 2 | A* ignore_preconditions | 1450 | 1452 | 13303 | 3.74 | 9 | yes |
| Air Cargo 2 | A* pg_levelsum | 86 | 88 | 841 | 56.85 | 9 | yes |
| Air Cargo 2 | breadth_first_search | 3343 | 4609 | 30509 | 12.11 | 9 | yes |
| Air Cargo 2 | depth_first_graph_search | 1669 | 1670 | 14863 | 12.26 | 1444 | NO |
| Air Cargo 2 | uniform_cost_search | 4853 | 4855 | 44041 | 10.39 | 9 | yes |
| Air Cargo 3 | A* no heuristic | 18223 | 18225 | 159618 | 47.76 | 12 | yes |
| Air Cargo 3 | A* ignore_preconditions | 5040 | 5042 | 44944 | 14.60 | 12 | yes |
| Air Cargo 3 | A* pg_levelsum | 316 | 318 | 2912 | 282.00 | 12 | yes |
| Air Cargo 3 | breadth_first_search | 14663 | 18098 | 129631 | 92.88 | 12 | yes |
| Air Cargo 3 | depth_first_graph_search | 592 | 593 | 4927 | 2.69 | 571 | NO |
| Air Cargo 3 | uniform_cost_search | 18223 | 18225 | 159618 | 45.10 | 12 | yes |

## II. Optimal Solutions

| Air Cargo Problem 1 | Air Cargo Problem 2 | Air Cargo Problem 3 |
|---|---|---|
| Steps: 6 | Steps: 9 | Steps: 12 |
| Load(C1, P1, SFO)<br>Fly(P1, SFO, JFK)<br>Load(C2, P2, JFK)<br>Fly(P2, JFK, SFO)<br>Unload(C1, P1, JFK)<br>Unload(C2, P2, SFO) | Load(C1, P1, SFO)<br>Fly(P1, SFO, JFK)<br>Load(C2, P2, JFK)<br>Fly(P2, JFK, SFO)<br>Load(C3, P3, ATL)<br>Fly(P3, ATL, SFO)<br>Unload(C3, P3, SFO)<br>Unload(C2, P2, SFO)<br>Unload(C1, P1, JFK) | Load(C2, P2, JFK)<br>Fly(P2, JFK, ORD)<br>Load(C4, P2, ORD)<br>Fly(P2, ORD, SFO)<br>Load(C1, P1, SFO)<br>Fly(P1, SFO, ATL)<br>Load(C3, P1, ATL)<br>Fly(P1, ATL, JFK)<br>Unload(C4, P2, SFO)<br>Unload(C3, P1, JFK)<br>Unload(C2, P2, SFO)<br>Unload(C1, P1, JFK) |

## III. Conclusions and Considerations:

1. deph_first_graph_search does not find an optimal solution.

This is no surprise, since DFS is non optimal in general for planning problems. DFS expands the longest path first and returns the first solution found. Most of the time the first solution found this way is not the shortest one. DFS “favors” solutions situated to the left of the search tree over shorter, but more “to the right” solutions.

2. A* with no real heuristic ( using A1) is just uniform cost search

3. Heuristics: pg_levelsum vs ignore_preconditions

Pg_levelsum performs better in terms of expansions, test, new nodes, but is significantly slower that ignore_preconditions.
Ignore_preconditions performs better with respect to time although is expands more nodes.

I think this is because ignore_preconditions does not need to build a planning graph, but levelsum needs this planning graph. Generating a planning graph and using it for heuristics too “expensive” for the current problems ( compared to using a faster but less “precise” heuristic that expands more nodes).

4. Uninformed vs informed search strategies

A* search with ignore_preconditions and A* with pg_levelsum are informed search strategies. They use heuristics to determine what nodes to expand first.
BFS, DFS and UCS are uninformed search strategies. DFS is not optimal for planning problems ( AIMA 3.4.3), BFS and UCS are optimal ( AIMA 3.4.1, AIMA 3.4.2.). UCS outperforms BFS over any metrics ( for our problems).

We can see in the results that A* with ignore_preconditions outperforms UCS over any metrics. This shows the advantages of using heuristics for search strategies.

5. Performance tradeoffs

Using h_ignore_preconditions we can reduce the computing cost of solving the three problems and improve the time required to reach an optimal solution.
We can further reduce the number of nodes and expansions by using h_levelsum, but this seems to come at a big computing cost and the time to find a solution grows significantly and using heuristics is not beneficial.