Analysis of search techniques in an air cargo planning problem

Nicholas Chen

Introduction

In this report, we discuss the various methods of planning search applied to deterministic logistics planning problems for an Air Cargo transport system. We first ran uninformed (non-heuristic) planning searches, and compare them to A* planning searches with domain-independent heuristics, using the metrics of number of node expansions (a proxy for memory required), number of goal tests, time elapsed and optimality.

Optimal plans

Note, there may be more than 1 optimal solution for each of the problems.

Problem 1

```
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

```
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)
```

Problem 3

```
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)
```

Comparison of non-heuristic search result metrics

Problem	Algo	Expansions	Goal Tests	Time (s)	Optimality	Plan length
1	breadth_first_search	43	56	0.036924517	**Yes**	**6**
1	depth_first_graph_search	**12**	**13**	**0.007663818**	No	12
1	uniform_cost_search	55	57	0.039505066	**Yes**	**6**
2	breadth_first_search	3343	4609	13.27345397	**Yes**	**9**
2	depth_first_graph_search	**1669**	**1670**	12.99159979	No	1444
2	uniform_cost_search	4852	4854	**12.4427912**	**Yes**	**9**
3	breadth_first_search	14663	18098	101.6007613	**Yes**	**12**
3	depth_first_graph_search	**592**	**593**	**3.231227321**	No	4927
3	uniform_cost_search	18235	18237	51.37885191	**Yes**	**12**

Figure 1: Comparison of metrics for non-heuristic search

We see that depth first search typically gives the least number of node expansions (and thus requires the least amount of memory) and finds the solution in the shortest time compared to the other two algorithms, but fails to find the optimal solution.

On the other hand, breadth first search and uniform cost search finds the optimal solution, but typically requires a larger amount of node expansions and takes a much longer time to arrive at a solution. While the time taken to find the solution for problem 1 and 2 is about the same for these two algorithms, in problem 3 we see that uniform cost search runs twice as fast.

These results agree with theory: Depth first search is not guaranteed to find the best solution, but breadth first search and uniform cost search are, as mentioned in the video lectures. The main benefit of depth first search is the smaller memory requirement.

Comparison of heuristic search result metrics

Problem	Algo	Expansions	Goal Tests	Time (s)	Optimality	Plan length
1	astar_search with h_ignore_preconditions	41	43	**0.038059927**	**Yes**	**6**
1	astar_search with h_pg_levelsum	**11**	**13**	1.148074169	**Yes**	**6**
2	astar_search with h_ignore_preconditions	1450	1452	**4.463247137**	**Yes**	**9**
2	astar_search with h_pg_levelsum	**86**	**88**	185.9902293	**Yes**	**9**
3	astar_search with h_ignore_preconditions	5040	5042	**17.41547025**	**Yes**	**12**
3	astar_search with h_pg_levelsum	**318**	**320**	1195.438052	**Yes**	**12**

Figure 2: Comparison of metrics for A* search with different heuristics

A* search found the optimal solution for both ignore_preconditions and levelsum heuristics. We see another tradeoff between the two heuristics; ignore_preconditions typically requires more node expansions and consumes more memory, but takes a shorter time to find the solution whereas levelsum consumes significantly less memory but requires a lot more time to find the solution.

Using both time and optimality as a measure, the ignore_preconditions heuristic is the best one. Under this criteria, A* search with ignore_preconditions heuristic is also better than non-heuristic seach planning methods for problems 2 and 3, and on par with the non-heuristic methods for problem 1. The effectiveness of A* with heuristics can only be seen when the problem gets harder.

These results support the fact that A* search is guaranteed to find the optimal path(s) if the heuristic is admissible (does not overestimate the cost), as mentioned in the video lectures. Furthermore, it is not surprising that A* with levelsum takes a longer time to find the optimal solution: The levelsum heuristic takes a longer time to compute than the ignore_preconditions heuristic. Furthermore, we expect heuristic search to be better than non-heuristic search in general because the latter only sees nodes as 'goal' and 'non-goal' ones and will blindly expand nodes till it reaches the goal state.