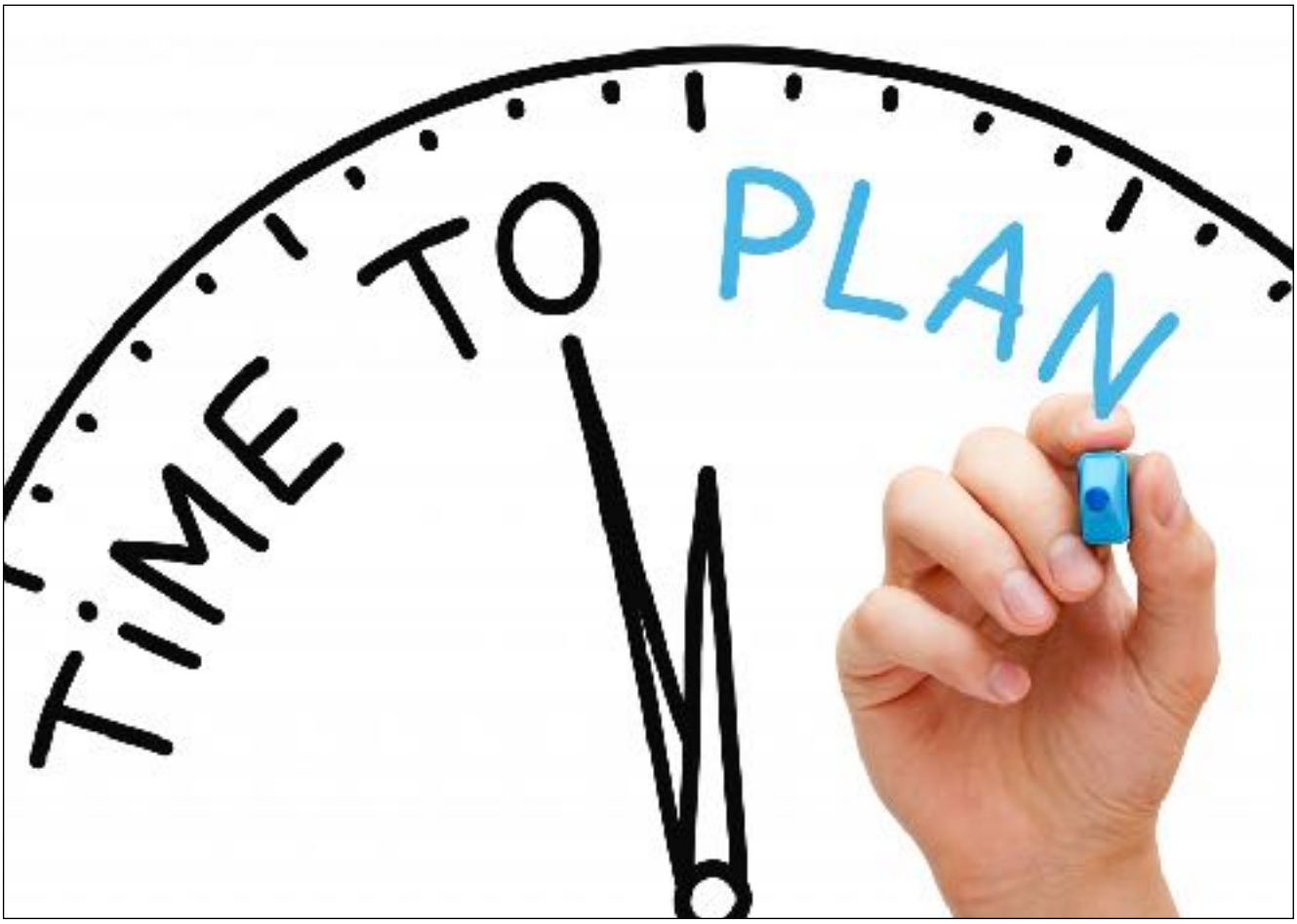
Heuristic Analysis

AI Nano-degree Planning Project

Zhuo Li - October 23, 2017



Introduction

This report analyzed the performance of four non-heuristic search algorithms and two heuristic search algorithms in three air cargo domain planning problems. The four non-heuristic search algorithms are bread-first search (BFS), depth-first graph search (DFGS), depth-limited search (DLS) and uniform cost search (UCS). The two heuristic search algorithms are A* search with the heuristic of ignoring preconditions (A*IP) and A* search with the heuristic of level sum (A*LS) . The algorithms are compared based on the metric of optimality, time elapsed and number of node expansions. Finally, the report also suggested the best algorithm to use.

Problems

The three problems are using the same action schema but with different initial states and goals. The schema is defined as following:

Action	Load(c, p, a)	Unload(c, p, a)	Fly(p, from, to)
PRECON	At(c, a) ∧ At(p, a) ∧ Cargo(c) ∧ Plane(p) ∧ Airport(a)	In(c, p) ∧ At(p, a) ∧ Cargo(c) ∧ Plane(p) ∧ Airport(a)	At(p, from) ∧ Plane(p) ∧ Airport(from) ∧ Airport(to)
EFFECT	¬ At(c, a) ∧ In(c, p)	At(c, a) ∧ ¬ In(c, p)	¬ At(p, from) ∧ At(p, to)

The three problems are defined as following:

Problems	Problem 1	Problem 2	Problem 3
Initial State	At(C1, SFO) ∧ At(C2, JFK) ∧ At(P1, SFO) ∧ At(P2, JFK) ∧ Cargo(C1) ∧ Cargo(C2) ∧ Plane(P1) ∧ Plane(P2) ∧ Airport(JFK) ∧ Airport(SFO)	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)	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 State	At(C1, JFK) ∧ At(C2, SFO)	At(C1, JFK) ∧ At(C2, SFO) ∧ At(C3, SFO)	At(C1, JFK) ∧ At(C3, JFK) ∧ At(C2, SFO) ∧ At(C4, SFO)

The goals can be achieved by many plans, however, the optimal plans are described as following:

Problem	Problem 1	Problem 2	Problem 3
Optimal Plan	Load(C1, P1, SFO) Load(C2, P2, JFK) Fly(P2, JFK, SFO) Unload(C2, P2, SFO) Fly(P1, SFO, JFK) Unload(C1, P1, JFK)	Load(C1, P1, SFO) Fly(P1, SFO, JFK) Load(C2, P2, JFK) Fly(P2, JFK, SFO) Load(C3, P3, ATL) Fly(P3, ATL, SFO) Unload(C3, P3, SFO) Unload(C2, P2, SFO) Unload(C1, P1, JFK)	Load(C1, P1, SFO) Fly(P1, SFO, ATL) Load(C3, P1, ATL) Fly(P1, ATL, JFK) Load(C2, P2, JFK) Fly(P2, JFK, ORD) Load(C4, P2, ORD) Fly(P2, ORD, SFO) Unload(C4, P2, SFO) Unload(C3, P1, JFK) Unload(C2, P2, SFO) Unload(C1, P1, JFK)

Non-Heuristic Search

The non-heuristic search algorithms do not have the additional knowledge in regards of every states like heuristic search algorithms, hence, the path towards reaching the goal states can not be guided or directed. These algorithms will explore all directions and all possibilities based on its searching priorities until a goal state is reached. These algorithms will be compared based on the metric of optimality, time elapsed and number of node expansions. Optimality indicates whether the search has found the optimal plan as we defined; Time Elapsed indicates how much time the search took to reach the goal state; Number of node expansions indicates how many nodes the search has explored in order to reach the goal state. The Comparison is as follows:

1. Result for Problem 1

Non-heuristic Search	Optimality (Length)	Time Elapsed (s)	Number of Node Expansion
Bread-first Search	Yes (6)	0.027	43
Depth-first Graph Search	No (12)	0.007	12
Depth-limited Search	No (50)	0.083	101
Uniform Cost Search	Yes (6)	0.031	55

2. Result for Problem 2

Non-heuristic Search	Optimality (Length)	Time Elapsed (s)	Number of Node Expansion
Bread-first Search	Yes (9)	11.97	3343
Depth-first Graph Search	No (575)	2.70	582

Non-heuristic Search	Optimality (Length)	Time Elapsed (s)	Number of Node Expansion
Depth-limited Search	-	> 10 mins	-
Uniform Cost Search	Yes (9)	9.93	4853

3. Result for Problem 3

Non-heuristic Search	Optimality (Length)	Time Elapsed (s)	Number of Node Expansion
Bread-first Search	Yes (12)	88.98	14663
Depth-first Graph Search	No (596)	2.79	627
Depth-limited Search	-	> 10 mins	-
Uniform Cost Search	Yes (12)	43.54	17783

From the results above, it is obvious that BFS and UCS are always providing the optimal plans for all the problems while DFGS and DLS are not once achieved optimal plans. From the time elapsed point of view, DFGS is the fastest and the execution does not increase as much as other algorithms. The more complicated the problem is, the bigger speed advantage we shall have from DFGS. UCS is the second fastest (Over 4 times slower than DFGS in problem 2 and 20 times in problem 3)and UCS is the third. Notice that the DLS has taken over 10 mins to execute in problem 2 and 3, hence, we did not have any records. From the memory usage point of view, DFGS uses the least among of memory while BFS and UCS use a lot more memories (Over 10 times more in problem 2 and 100 times more in problem 3). Between BFS and UCS, BFS uses slightly less memory than UCS.

BFS is finding the optimal plan because the path cost is a nondecreasing function, however, the large amount of memory needed for BFS is due to the large amount of frontier nodes. DFGS and DLS are not finding the optimal plan because they do not consider if a node is better, rather, they would simply go as deep as they needed before they switch to another branch. UCS is just a more general version of BFS hence it share the aspect of BFS.

To conclude, if finding the optimal plan is critical to the task, the best algorithms are BFS and UCS. Between these two algorithms, UCS will use less time while BFS will use less memory; if time and memory is critical to the task, the best algorithm will be DFGS.

Heuristic Search

Heuristic search algorithms can have additional information with every state and thus, can use that information to guide or direct the search in order to reach the goal efficiently. The heuristics can be generated automatically by relaxing the preconditions and are domain

independent. The two algorithms in this analysis are both based on A* search but with different heuristics: Ignore preconditions and Level sum. These algorithms will be compared based on the metric of optimality, time elapsed and number of node expansions. Optimality indicates whether the search has found the optimal plan as we defined; Time Elapsed indicates how much time the search took to reach the goal state; Number of node expansions indicates how many nodes the search has explored in order to reach the goal state. The Comparison is as follows:

1. Result for Problem 1

Heuristic Search	Optimality (Length)	Time Elapsed (s)	Number of Node Expansion
A* with Ignore Precondition	Yes (6)	0.035	41
A* with Level Sum	Yes (6)	0.903	11

2. Result for Problem 2

Heuristic Search	Optimality (Length)	Time Elapsed (s)	Number of Node Expansion
A* with Ignore Precondition	Yes (9)	3.53	1428
A* with Level Sum	Yes (9)	144.90	86

3. Result for Problem 3

Heuristic Search	Optimality (Length)	Time Elapsed (s)	Number of Node Expansion
A* with Ignore Precondition	Yes (12)	14.01	5003
A* with Level Sum	Yes (12)	834.08	311

From the above results, it is obvious that both A*IP and A*LS are providing the optimal plans. From the time elapsed point of view, A*IP uses significant less time than A*LS. A*IP uses only 0.035 s in problem 1, 3.53 s in problem 2 and 14.01 s in problem 3 while A*LS uses 0.903 s in problem 1, 144.90 s in problem 2 and 834.08 s in problem 3. From the memory usage point of view, A*LS uses significant less amount of memory than A*IP. A*LS expands only 11 nodes in problem 1, 86 nodes in problem 2 and 311 nodes in problem 3 while A*IP expands 41 nodes in problem 1, 1428 nodes in problem 2 and 5003 nodes in problem 3.

A*IP and A*LS are both giving the optimal plans since the heuristics are admissible. The A*IP is using more memory because more possible plans to reach the goal are there when

ignoring the precondition. A*LS is taking more time because it needs to calculate the level sum for every goals.

To conclude, while both algorithms provides the optimal solution, the choice would really be a trade-off between time and memory space. A*IP would be better if time is a concern and A*LS would be better if memory space is a concern.

Non-heuristic v.s. Heuristic

Considering all three aspect, optimality, time elapsed and memory usage, the best algorithms should be among BFS and A*IP.

1. Result for Problem 3

Heuristic Search	Optimality (Length)	Time Elapsed (s)	Number of Node Expansion
A* with Ignore Precondition	Yes (12)	14.01	5003
Bread-first Search	Yes (12)	88.98	14663

By comparing these two algorithms, we can easily conclude that A*IP would be the best algorithm for the air cargo domain planning problem.

Conclusion

This report has analyzed performance of both non-heuristic search and heuristic search algorithms in the air cargo domain problems. It is obvious that heuristic search algorithms have the advantage over non-heuristic algorithms in all aspects (optimality, time elapsed and number of node expansion).