Analysis

I looked at the non-heuristic algorithms (Breadth First Search, Depth First Graph Search, Uniform Cost Search and Greedy Best First Search) in comparison to heuristic algorithms (A* Search with Constant Heuristic, A* Search with Ignore Preconditions and A* Search with Levelsum Heuristic). The results for the Air Cargo planning problem are documented below.

Results

Air Cargo Problem 1

Non-heuristic algorithms

In the below results we can see that BFS and UCS algorithms both achieve the optimal solution with an equivalent number of expansions, goal tests and new nodes. They perform about the same in terms of execution time. DFGS has a much faster execution time and expands the fewest nodes of the non-heuristic algorithms but does not find the optimal solution.

Heuristic algorithms

A* Search h_1 and A* Search h_ignore_preconditions have similar execution time to BFS and UCS and use a similar level of expansions in order to arrive at the optimal solution. A* Search with Levelsum uses the fewest number of expansions but takes considerably longer to arrive at the correct solution. This could save on RAM usage at the expense of execution time.

Optimal Plan

Given the small search space for problem 1, we do not see any major gains in execution time by using the heuristic algorithms. The recommended algorithm to use here would be BFS if we are not using a heuristic (since it is the fastest and is optimal). The best heuristic algorithm would be A* Search with Ignore Preconditions (since it uses the fewest expansions and arrives at the optimal solution).

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time (seconds)
Breadth First Search	55	57	224	6	0.06
Depth First Graph Search	12	13	48	12	0.01
Uniform Cost Search	55	57	224	6	0.07
A* Search h_1	55	57	224	6	0.06
A* Search h_ignore_preconditions	41	43	170	6	0.07
A* Search with Levelsum	11	13	50	6	0.7

The optimal plan length is of length 6.

The plan is as follows:

LOAD(C1, P1, SFO)
FLY(P1, SFO, JFK)
UNLOAD(C1, P1, JFK)
LOAD(C2, P2, JFK)
FLY(P2, JFK, SFO)
UNLOAD(C2, P2, SFO)

Air Cargo Problem 2

Non-heuristic algorithms

In the below results we see that BFS and UCS are the slowest of the non-heuristic algorithms. BFS requires quite considerably less expansions than UCS and is faster to find the optimal solution. DFGS is faster than the other former algorithms but overestimates the plan length by a significant margin.

Heuristic algorithms

The search space has increased considerably for this problem and we start to see significant gains for the heuristic algorithms. A* Search h_1 and A* Search h_ignore_preconditions outperform the non-heuristic algorithms in terms of execution time. Notice that the A* Search h_1 uses considerably more expansions than A* Search h_1 however. Further A* Search h_1 is three times slower than A* Search h_ignore_preconditions.

A* Search with Levelsum uses the least number of expansions but takes considerably longer than all algorithms.

Optimal Plan

The optimal plan is to use A* Search h_ignore_preconditions since there are large gains in execution time and it uses the least expansions.

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time (seconds)
Breadth First Search	3343	4609	30509	9	22.55
Depth First Graph Search	1669	1670	14863	619	20.49
Uniform Cost Search	4852	4854	414	9	32
A* Search h_1	4852	4854	44030	9	18
A* Search h_ignore_preconditions	1450	1452	13303	9	6.53

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time (seconds)
A* Search with Levelsum	86	88	841	9	70

The optimal plan length is of length 9.

The plan is as follows:

LOAD(C1, P1, SFO)
FLY(P1, SFO, JFK)
UNLOAD(C1, P1, JFK)
LOAD(C2, P2, JFK)
FLY(P2, JFK, SFO)
UNLOAD(C2, P2, SFO)
LOAD(C3, P3, ATL)
FLY(P3, ATL, SFO)
UNLOAD(C3, P3, SFO)

Air Cargo Problem 3

Non-heuristic algorithms

In the below results we see that BFS and UCS are the slowest of the non-heuristic algorithms. BFS requires quite considerably less expansions but it is considerably slower than UCS. Due to this, when there is a large search space it seems it would be advisable to use UCS instead of BFS. DFGS is faster than the other former algorithms but overestimates the plan length by a significant margin.

Heuristic algorithms

A* Search h_1 and A* Search h_ignore_preconditions outperform the non-heuristic algorithms in terms of execution time. A* Search h_1 again uses considerably more expansions than A* Search h_1 however. Further A* Search h_1 is a lot slower than A* Search h_ignore_preconditions.

A* Search with Levelsum uses the least number of expansions but takes considerably longer than all algorithms.

Optimal Plan

Since there are large gains in execution time and uses fewer expansions than all other algorithms, A* Search h_ignore_preconditions is the best algorithm to use.

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time (seconds)
Breadth First Search	14663	18098	129631	12	167.11
Depth First Graph Search	592	593	4927	392	5.31

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time (seconds)
Uniform Cost Search	18235	18237	159716	12	83.53
A* Search h_1	18235	18237	159716	12	80
A* Search h_ignore_preconditions	5040	5042	44944	12	26
A* Search with Levelsum	325	327	3002	12	369.87

The optimal plan length is of length 12.

The plan is as follows:

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)

Conclusion

The A* Search h_ignore_preconditions and A* Search h_1 heuristic algorithms are faster than the non-heuristic algorithms as the search space increases. The reason for this is that we have essentially relaxed the problem by removing preconditions, reducing the complexity of the search (as noted in the number of nodes expanded and execution time). Peter Norvig notes that this is one of the simplest heuristics we can use to improve algorithmic efficiency for planning problems.[1] Since it uses the fewest expansions, is the fastest and is optimal the recommendation is to use the A* Search h_ignore_preconditions algorithm. A* Search with Levelsum might be advantageous if we are optimising for the fewest expansions (and therefore less RAM usage) instead of execution time.

[1] Peter Norvig 'Artificial Intelligence: A Modern Approach 2nd Edition', chapter 11 p. 386 @ http://aima.cs.berkeley.edu/2nd-ed/newchap11.pdf