Deterministic Logistics Planning Problems for an Air Cargo Transport
Heuristic Analysis

This is an analysis of three problems of air cargo transport using different search methods:

Problem 1 initial state and goal:

Init(At(C1, SFO) ∧ At(C2, JFK) ∧ At(P1, SFO) ∧ At(P2, JFK) ∧ Cargo(C1) ∧ Cargo(C2) ∧ Plane(P1) ∧ Plane(P2) ∧ Airport(JFK) ∧ Airport(SFO)) Goal(At(C1, JFK) ∧ At(C2, SFO))

Problem 2 initial state and goal:

Init(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)) Goal(At(C1, JFK) ∧ At(C2, SFO) ∧ At(C3, SFO))

Problem 3 initial state and goal:

Init(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(At(C1, JFK) ∧ At(C3, JFK) ∧ At(C2, SFO) ∧ At(C4, SFO))

Part 1. Optimal Plans
Optimal plans produced by A* Search ignoring preconditions heuristic:

I. For Problem 1 :
1. Load(C1, P1, SFO)
2. Fly(P1, SFO, JFK)
3. Unload(C1, P1, JFK)
4. Load(C2, P2, JFK)
5. Fly(P2, JFK, SFO)
6. Unload(C2, P2, SFO)

II. For Problem2:
1. Load(C3, P3, ATL)
2. Fly(P3, ATL, SFO)
3. Unload(C3, P3, SFO)
4. Load(C2, P2, JFK)
5. Fly(P2, JFK, SFO)
6. Unload(C2, P2, SFO)
7. Load(C1, P1, SFO)
8. Fly(P1, SFO, JFK)
9. Unload(C1, P1, JFK)

III. For Problem 3:
1. Load(C2, P2, JFK)
2. Fly(P2, JFK, ORD)
3. Load(C4, P2, ORD)
4. Fly(P2, ORD, SFO)
5. Unload(C4, P2, SFO)
6. Load(C1, P1, SFO)
7. Fly(P1, SFO, ATL)
8. Load(C3, P1, ATL)
9. Fly(P1, ATL, JFK)
10. Unload(C3, P1, JFK)
11. Unload(C1, P1, JFK)
12. Unload(C2, P2, SFO)

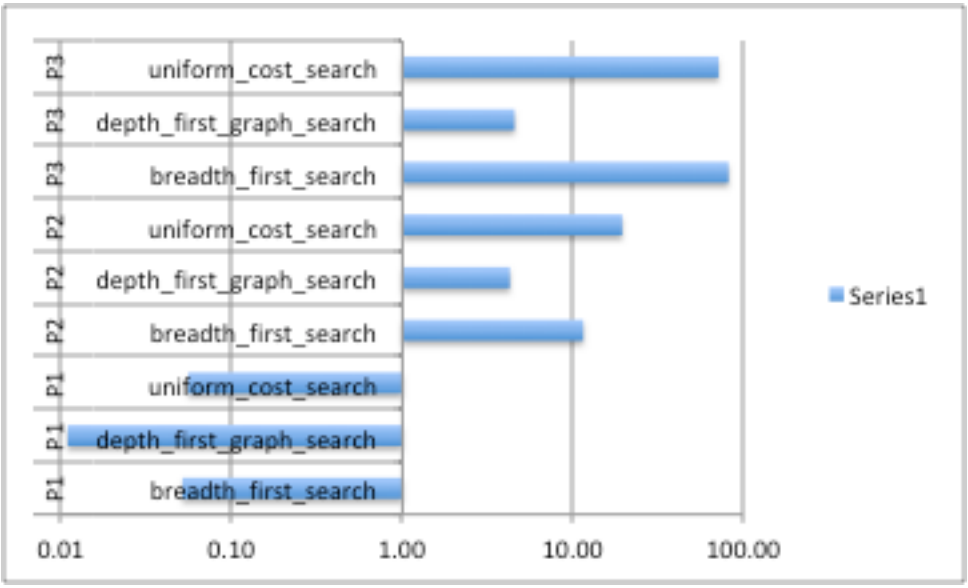
Part 2. Uninformed Non-heuristic search
In this part three uninformed non-heuristic search result metrics are compared i.e.
1. Breadth First Search (BFS)
2. Depth First Graph Search (DFS)
3. Uniform Cost Search (UCS)

The results are presented in Tab. 1:

Tab.1

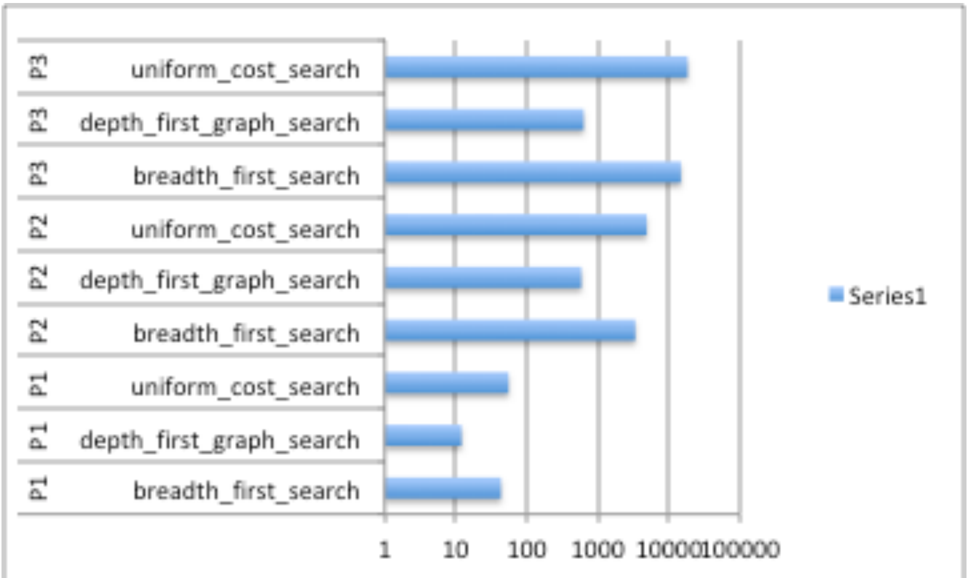
Problem	Search	Expansions	Goal Tests	New Nodes	Time Elapsed [s]	Plan length	Optimal
P1	breadth_first_search	43	56	180	0.05	6	Yes
P1	depth_first_graph_search	12	13	48	0.01	12	No

P1	uniform_cost_search	55	57	224	0.06	6	Yes
P2	breadth_first_search	3343	4609	30509	11.48	9	Yes
P2	depth_first_graph_search	582	583	5211	4.27	575	No
P2	uniform_cost_search	4852	4854	44030	19.42	9	Yes
P3	breadth_first_search	14663	18098	129631	81.56	12	Yes
P3	depth_first_graph_search	627	628	5176	4.54	596	No
P3	uniform_cost_search	18235	18237	159716	71.29	12	Yes



Time Elapsed vs. Uniformed Search for P1, P2, P3 (log scale)

Time needed to finish search (Time Elapsed) is the shortest for Depth First Graph Search. Uniform Cost Search and Breadth Firs Search took significantly longer especially for the most complicated Problem 3.



New Nodes vs. Uniformed Search for P1, P2, P3 (log scale)

Memory consumption can be analyzed through number of New Nodes expanded with the given search method. Depth First Graph Search expanded the fewest nodes whereas Uniform Cost Search required the most memory for its search.

Plan lengths produced by Breadth First Search and Uniform Cost Search are the optimal. Depth First Graph Search despite its better performance produced considerably longer paths therefore the results are not optimal.

Part 3. Heuristic Search

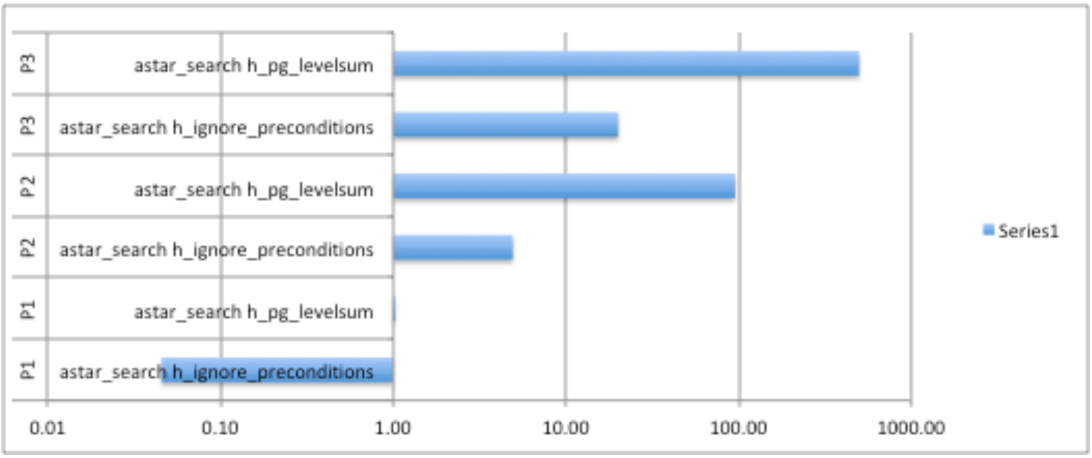
In this part 2 heurystic search result metrics are compared i.e.

1. A* Search with 'Ignore Preconditions'
2. A* Search with 'Level Sum'

The results are presented in Tab. 1:

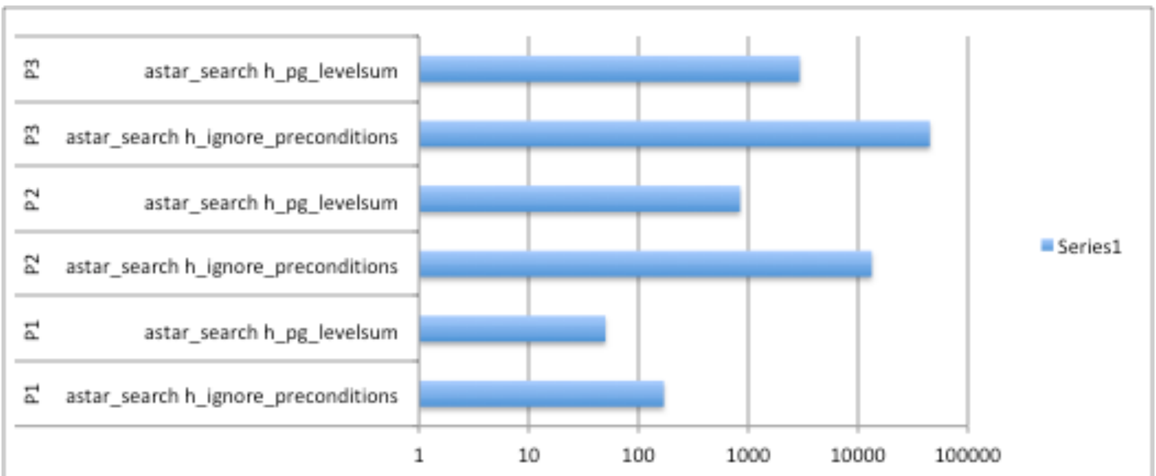
Tab. 2

Problem	Search	Expansions	Goal Tests	New Nodes	Time Elapsed [s]	Plan length	Optimal
P1	astar_search h_ignore_preconditions	41	43	170	0.05	6	Yes
P1	astar_search h_pg_levelsum	11	13	50	1.01	6	Yes
P2	astar_search h_ignore_preconditions	1450	1452	13303	4.92	9	Yes
P2	astar_search h_pg_levelsum	84	88	841	95.35	9	Yes
P3	astar_search h_ignore_preconditions	5040	5042	44944	20.03	12	Yes
P3	astar_search h_pg_levelsum	318	320	2934	498.00	12	Yes



Time Elapsed vs. A Search with h for P1, P2, P3 (log scale)*

Computation time is much shorter for "Ignore Preconditions' heuristic. That is probably due to complicated computation needed for Level Sum, in comparison to simple 'Ignore Precondition' heuristic.



New Nodes vs. A Search with h for P1, P2, P3 (log scale)*

On the other hand, if we look at the number of New Nodes we can see that the first search needed significantly more memory to find the goal path.

Finally, both searches provided the optimal solution.

Final Conclusion [1]

Uninformed searches provided results that reflect the theory. BFS and UCS uninformed have exponential time and space complexities, $O(b^d)$ and $O(b^{1+\lfloor C*/\varepsilon \rfloor})$, respectively whereas DFS has time complexity of $O(b^m)$ where m is max depth of search but linear space complexity O(bm), which is reflected in the New Nodes result (DFS result is significantly lower). Although low memory complexity of DFS could be advantageous, due to the nature of the search that explores a single branch till its max depth and stops when goal is achieved, the solution provided by DFS is not optimal therefore can't be a good recommendation.

In the problems that analyzed here optimal uniformed searches provided solutions in relatively acceptable timeframe (especially for problems 1 and 2). BFS and UCS explore every branch in the tree search, which means that those searches will always provide an optimal solution, however time complexity of those searches can be problematic for very complex problems. The literature suggests that generally exponential complexity problems can't be solved by uninformed searches and informed search strategies should be used. Informed search algorithms are better option for exponential problems, which is our case.

In the second part of the project A* search algorithm was used which estimates the cost of cheapest solution by minimizing the sum of path cost function from start to n node and heuristic function which estimates the cost from n to goal and is type of informed search algorithm.

One of the properties of A* is that "*the tree-search version of* A∗ *is optimal if heuristic function is admissible*".

Since both our heuristics ('Ignore Preconditions' and 'Level Sum') are admissible therefore the solution is always optimal.

A* Search with 'Ignore Preconditions' seem the best choice in the set of problems given. First of all, the solution should be optimal, second of all the computation time should be the shortest, in order to make quick planning decisions. A* Search can have the worst – case space complexity similar to DFS or UCS but can do better. Therefore informed search can only win with optimal uninformed search. Despite larger memory requirements from the A* Search with 'Ignore Preconditions' provided optimal solution in the fastest time among informed search examined that provided optimal solution.

References:

[1] Artificial Intelligence: A Modern Approach, Third Edition, 1994 P. Norvig