Heuristic analysis:

Plan search run stats:

1. ***Problem*: *Air Cargo Problem 1***

Problem: Air Cargo Problem 1					
Search	**Expansions**	**Goal_Tests**	**New Nodes**	**Plan length**	**Time**
breadth_first_search	43	56	180	6	0.0511
depth_first_graph_search	21	22	84	20	0.0175
uniform_cost_search	55	57	224	6	0.06
recursive_best_first_search h_1	4229	4230	17023	6	4.163
greedy_best_first_graph_search with h_1	7	9	28	6	0.006895
astar_search with h_1	55	57	224	6	0.133
astar_search h_ignore_preconditions.	41	43	170	6	0.06203
astar_search with h_pg_levelsum	11	13	50	6	11.8528

Optimal Plan Length: 6

Optimal Plan:

Load(C1, P1, SFO)
Fly(P1, SFO, JFK)
Load(C2, P2, JFK)
Fly(P2, JFK, SFO)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)

2. ***Problem: Air Cargo Problem 2***

Problem: Air Cargo Problem 2					
Search	**Expansions**	**Goal_Tests**	**New Nodes**	**Plan length**	**Time**
breadth_first_search	3343	4609	30509	9	45.1548
depth_first_graph_search	624	625	5602	619	11.66935
uniform_cost_search	4852	4854	44030	9	132.0314
recursive_best_first_search h_1	inf	inf	inf	inf	inf
greedy_best_first_graph_search with h_1	990	992	8910	17	27.7241
astar_search with h_1	4852	4854	44030	9	146.3674
astar_search h_ignore_preconditions.	1506	1508	13820	9	49.48725
astar_search with h_pg_levelsum	86	88	841	9	475.7886

*inf refer to too long

Optimal Plan Length: 9

Optimal Plan:

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(C1, P1, JFK)
Unload(C2, P2, SFO)

3. ***Problem: Air Cargo Problem 3***

Problem: Air Cargo Problem 3					
Search	**Expansions**	**Goal_Tests**	**New Nodes**	**Plan length**	**Time**
breadth_first_search	14663	18098	129631	12	240.1591
depth_first_graph_search	408	409	3364	392	3.8626
uniform_cost_search	18235	18237	159716	12	1091.995
recursive_best_first_search h_1	inf	inf	inf	inf	inf
greedy_best_first_graph_search h_1	5614	5616	49429	22	253.84279
astar_search with h_1	18235	18237	159716	12	1115.49438
astar_search h_ignore_preconditions.	5118	5120	45650	12	260.283
astar_search with h_pg_levelsum	408	410	3758	12	2374.5656

*inf refers to too long

Optimal Plan Length: 12

Optimal Plan:

Load(C2, P2, JFK)
Fly(P2, JFK, ORD)
Load(C4, P2, ORD)
Fly(P2, ORD, SFO)
Load(C1, P1, SFO)
Fly(P1, SFO, ATL)
Load(C3, P1, ATL)
Fly(P1, ATL, JFK)
Unload(C4, P2, SFO)
Unload(C3, P1, JFK)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)

Analysis:

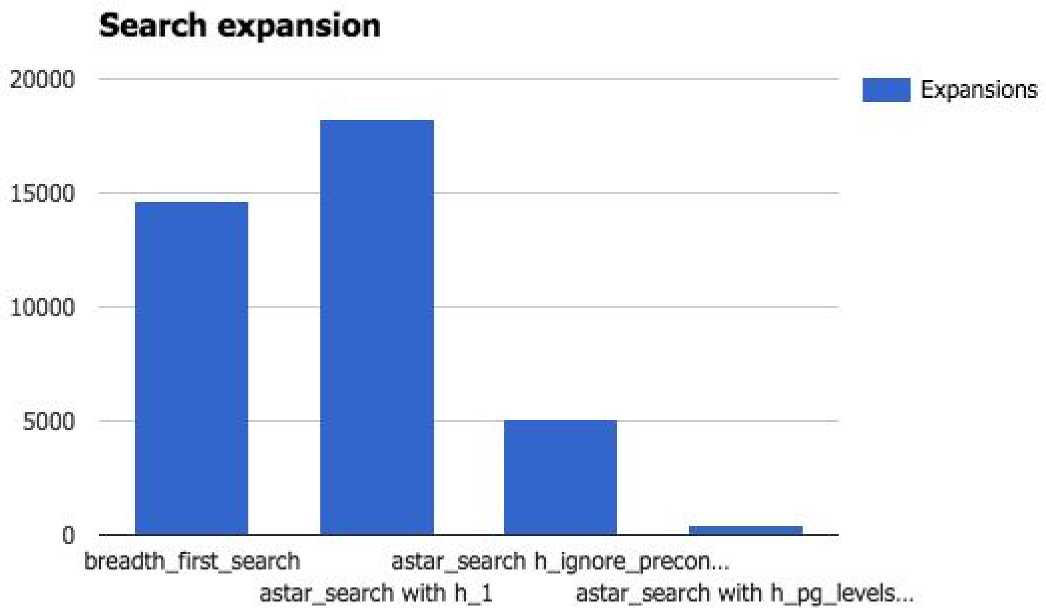
Interference from above search run results:

1. **Breadth first search:** It provides optimal plan. But it's consumes more more time and more expansion compatibility with other function. As the complexity of the problem increases it become unusable to use.(Getting slower)
2. **Depth first Search graph:** As we see from above result, its produces higher plan length that optimal, even though its produces results in less amount of time. It's not optimal at all. For example consider simple problem 1 - Its produces plan of length 20, when optimal plan is 6. Which reiterate the Peter's point **SEARCH AND OPTIMIZATION (Lesson 7 video)** it's not guaranteed to produce shortest path
3. **Uniform Cost search and A* Search with h1:** Both have same number of expanded nodes, goal test and nodes created and reasonably same amount to time to produce results. According AIMA code - Uniform cost search uses best_first_graph_search and A* search is best-first graph search with f(n) = g(n)+h(n). (h(1) in this case). Even though, both produces optimal plan, in terms to node expansion and time consumed, it performs poorly compared to Breadth first search. This is mainly because once goal is found is breadth first search it doesn't expand unlike uniform cost search plan - which look for cheap path. (With reference with **SEARCH AND OPTIMIZATION)**
4. **Recursive best first search h1:** This is really slow, to produce result. Even though it produces ideal plan its very time consuming, it because of recursion of expanding same node over again.
5. **Greedy best first search with h1:** It doesn't return optimal plan. Even though provides result with less expansion than any of the above searches in less time. For example for problem 2 it produces a plan with 17 length instead of optimal 9. Greedy best-first search is accomplished by specifying f(n) = h(n). (best_first_graph_search)
6. **A* with h_ignore precondition and h_pg_levelsum:**
h_ignore precondition, heuristic estimates the minimum number of actions that must be carried out from the current state in order to satisfy all of the goal conditions by ignoring the preconditions required for an action to be executed. h_pg_levelsum, refers to The sum of the level costs of the individual goals (admissible if goals independent). *Both A* with h_ignore precondition and h_pg_levelsum produces ideal plan.* It better than any above searches in terms of node expansion,goal test and new node. Comparing A* h_ignore precondition and h_pg_levelsum, h_ignore precondition produces plan optimal plan with less time, even though h_pg_sum expansion of node is less. This is mainly due to heuristic calculation duration produced by h_pg_sum.

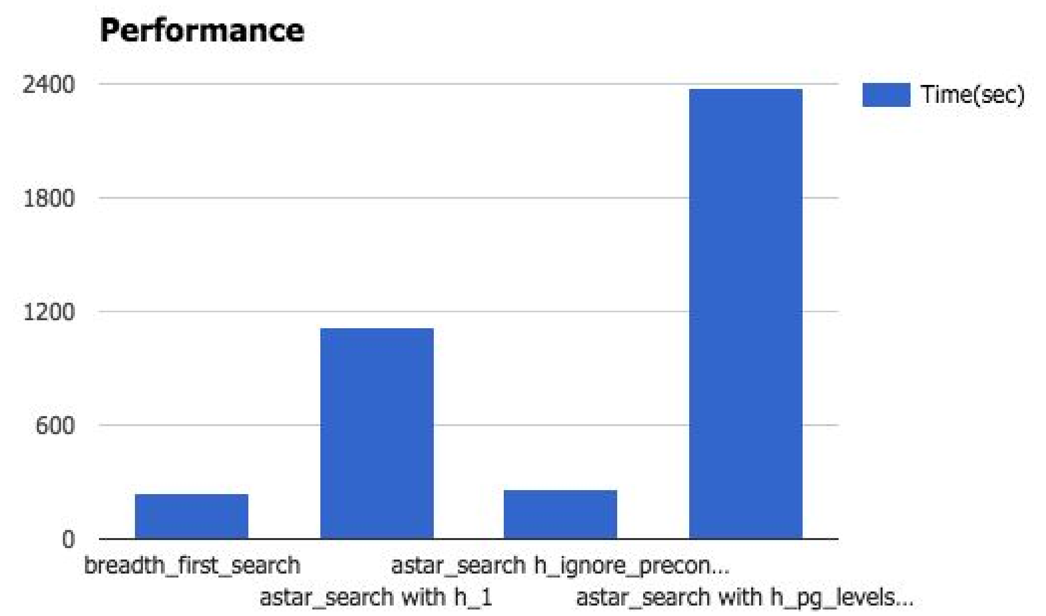
comparison graph - **Using Air cargo problem 3 data**

Breadth first search vs Uniform cost Search/A* Search with h1 vs A* with h_ignore precondition vs A* with   h_pg_levelsum:

<u>Node expansion:</u>



<u>Performance: (in sec)</u>



Conclusion:

From the above analysis and graph even though best optimal heuristic planning search for the Aircargo problem is A* with h_ignore precondition. As mentioned in analysis, h_ignore precondition produce significantly quicker than A* with h_pg_levelsum, even though node expansion of h_pg_levelsum is minimal, as calculation timing of h_pg_levelsum is expensive. Best non heuristic optimal planning search is breadth first search.