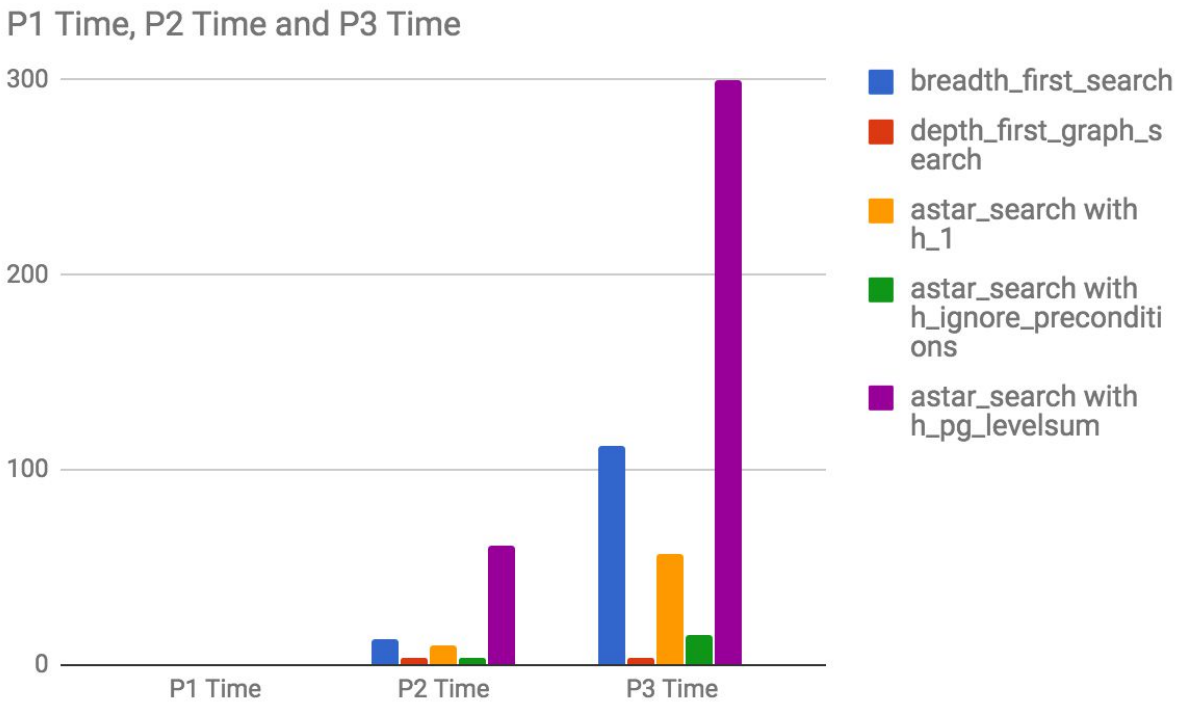
Heuristic Analysis

	Breadth first search	Depth first graph search	Astar search h_1	Astar search h_ignore_preconditions	Astar search h_pg_levelsum
P1	E 12 G 13 N 18 T 0.029	E 12 G 13 N 18 T 0.007	E 55 G 57 N 224 T 0.0317	E 41 G 43 N 170 T 0.0258	E 11 G 13 N 50 T 0.69307
P2	E 3343 G 4609 N 30509 T 13.476	E 582 G 583 N 5211 T 4.0286	E 4852 G 4855 N 44041 T 10.385	E 1450 G 1452 N 13303 T 3.6338	E 86 G 88 N 841 T 61.6285
P3	E 14663 G 18098 N 129631 T 112.192	E 627 G 628 N 5176 T 3.6914	E 18234 G 18236 N 159707 T 56.7113	E 5040 G 5042 N 44944 T 15.2904	E 325 G 327 N 3002 T 299.89

P1: Air Cargo Problem 1
P2: Air Cargo Problem 2
P3: Air Cargo Problem 3

E: Expansions
G: Goal Tests
N: New Nodes
T: Time Elapsed



WIth these results above for the different search strategies on planning search, the following observations were found:

1. PlanningGraph Level Sum heuristic shows much less node expansions. However, h_pg_level_sum being the heuristic constructing a PlanningGraph with mutex and adding up its level sum came with a cost of performance that in certain cases (particularly simpler cases) outweighs its benefits. Generally speaking complex heuristic came with a cost that may outweighs its controlled expansion of nodes; it's critical to use the right tool for the right job and use rational judgement for the trade offs.
2. All of the strategies grew proportionally with the complexity of the problem. They don't scale up too well with scripting language. Using pypy or any lower level optimization should significantly speed up the algorithms. However, this is still a computationally heavy problem.
3. Depth First Search seems to perform very well; however, the resulting plan can be very inefficient. For problem 2 and 3, it came up with very loopy plan to the goal state. That's due to the nature of the how it's searching as far deep as possible and the heuristic function in used. I don't think Depth First Search should be used for Planning Search for this very reason.
4. Breadth First Search performed reasonably well with accurate result. Not having the algorithm advantage of A Star Search and with no information on the A Star Search; it seems to be relatively efficient running h1 heuristic which isn't a real heuristic at all.

5. A Star Search, when given Ignored_Precondition heuristic, though it has a high volume of node expansions, outperformed other heuristic with accuracy. Ignored Precondition Heuristic can be a good automated heuristic applicable to general problems as a simple way to relax constraints of the actions to achieve fast performance with admissible heuristic.
6. If time and accuracy is our ways to concluding the winner, then A Star Search with Ignored Precondition provides the accurate plan with the least amount of time. However, A Star Search with PlanningGraph Level Sum Heuristic is worthy to be noted for more complex problem as its branching factor seems to be very low given a good heuristics.