Search Algoritms Used

Uninformed Searches

Breadth-first Tree Search

Searches first through all possible solutions of length n, then length $n + 1$, and so on. As a tree search, it does not notice re-entering a previous state and hence expands to loops. If it converges, it is guaranteed (in our problem) to return an optimal solution

Breadth-first Search

Graph-version of the previous breadth-first search. Does not re-enter state, will be more efficient than BfTS

Depth-first Graph Search

Traces out a single path until it reaches a solution. Hence it expands many Nodes, but only goal tests one. Does most likely not return an optimal solution

Depth-limited Search

Traces one branch of a search tree until the depth limit, then backtraces to the last branch, branches out, and so on.

Uniform Cost Search

In this case (since we don't define a cost) practically the same as breadth-first search

Informed Searches

A*

As discussed in the lecture

Recursive Best-first Search

To my understanding, a recursive approach that follows the same steps as an A*, but as a tree search, hence re-entering previous states. Less efficient, but should always result in the optimal solution if it converges

Greedy Best-first Graph Search

A best-first search which orders not (like a*) according to the minimum expected total cost, but according to the minimum expected cost of the remaining unexplored path. Should converge faster, but will not guarantee an optimal solution

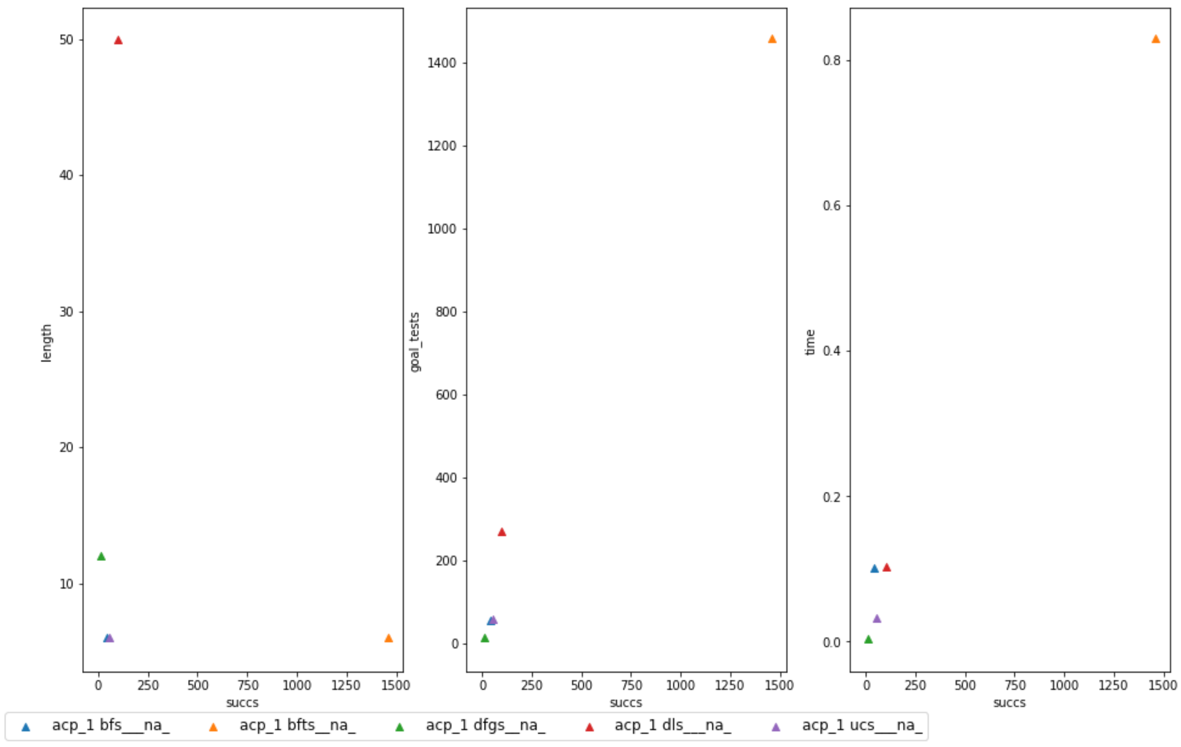
Problem 1

All informed searches and most of the uniformed search provide a solution of size 6, which is indeed optimal since two cargos need to be loaded, unloaded, and flown between different endpoints. The solution is shown below. All searches find solutions.

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

In [56]:

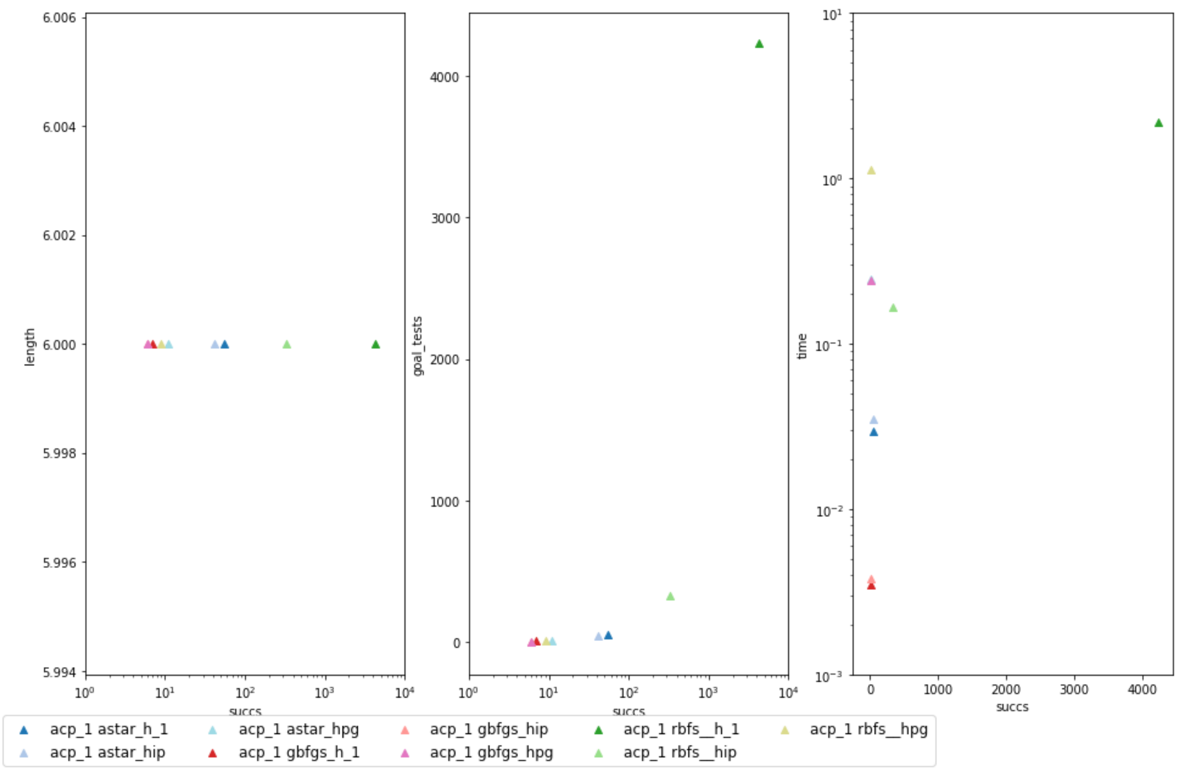
```
plot_subset((('succs','length', False, False),
             ('succs','goal_tests', False, False),
             ('succs','time', False, False)), problems=['Air Cargo Problem 1'],
             searches=simple_searches)
```



All naive searches converge. The three breadth-oriented approaches find the optimal solution. Both depth-first searches find an solution within short time - depth-first graph search's solution is only about twice the length of the optimal solution.

In [58]:

```
plot_subset((('succs','length', True, False),
             ('succs','goal_tests', True, False),
             ('succs','time', False, True)), problems=['Air Cargo Problem 1'],
             searches=complex_searches)
```



All informed searches (as well as the pseudo-informed h1-searches) return the same (optimal) result. However, the greedy search is of orders of magnitude faster than a*, which again is orders of magnitude faster than the uninformed RBF search.

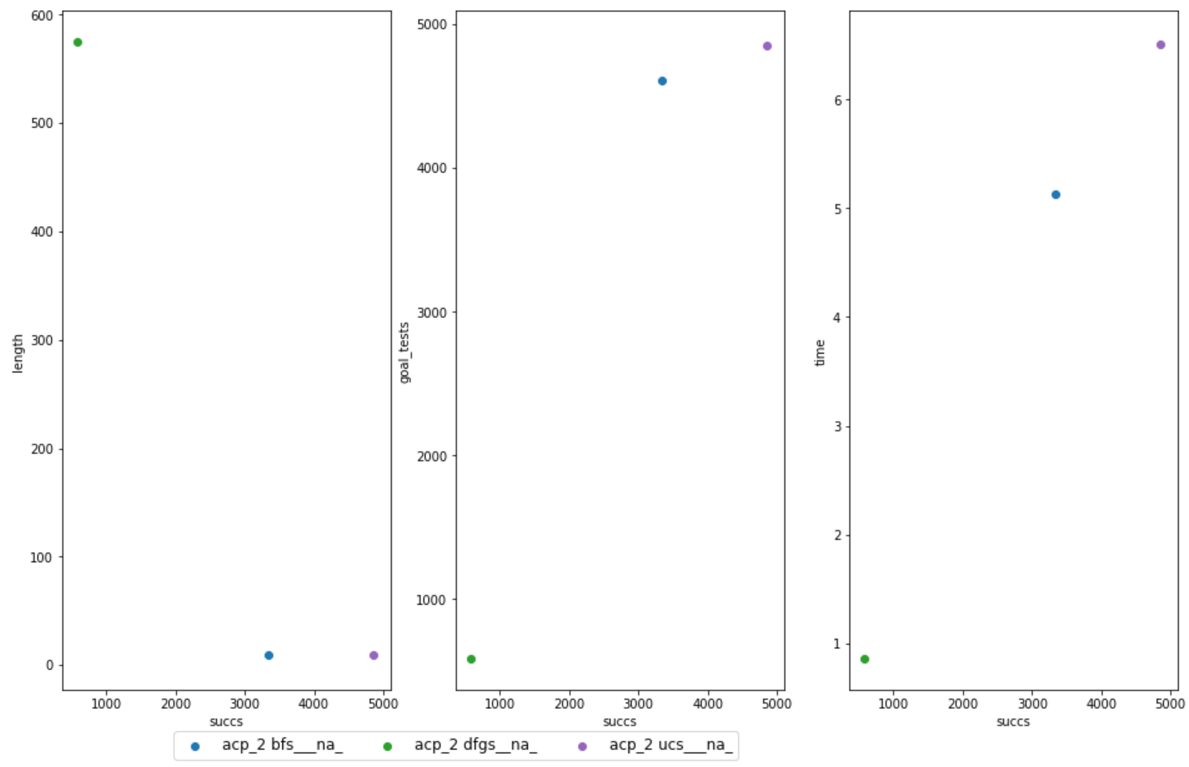
Problem 2

One found optimal solution (length 9) is

- Load(C2, P2, JFK)
- Load(C1, P1, SFO)
- Load(C3, P3, ATL)
- Fly(P2, JFK, SFO)
- Unload(C2, P2, SFO)
- Fly(P1, SFO, JFK)
- Unload(C1, P1, JFK)
- Fly(P3, ATL, SFO)
- Unload(C3, P3, SFO) Again, this is certainly optimal, since we have 3 cargos * (load + unload + fly), as none of origin and destination overlap

In [59]:

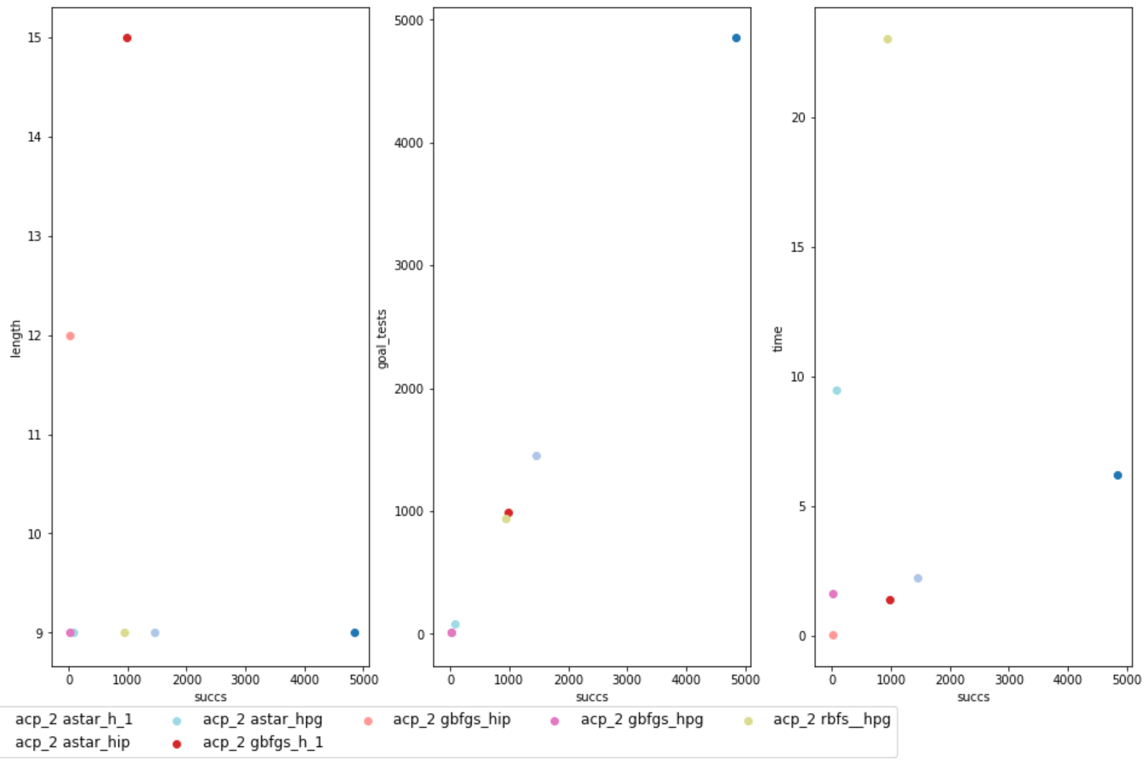
```
plot_subset((('succs','length', False, False),
             ('succs','goal_tests', False, False),
             ('succs','time', False, False)), problems=['Air Cargo Problem 2'],
             searches=simple_searches)
```



Depth-limited seach and breadth-first tree search do no converge anymore within the limit of 100,000 expansions. Depth-limited search does not converge, and depth-first graph search finds a solution rougly 70 times longer than optimal

In [60]:

```
plot_subset((('succs','length', False, False),
             ('succs','goal_tests', False, False),
             ('succs','time', False, False)), problems=['Air Cargo Problem 2'],
             searches=complex_searches)
```



Recursive best-first graph-search only converges for the `pg_levelsum` heuristic. For this heuristic, GBFGS still finds the optimal solution.

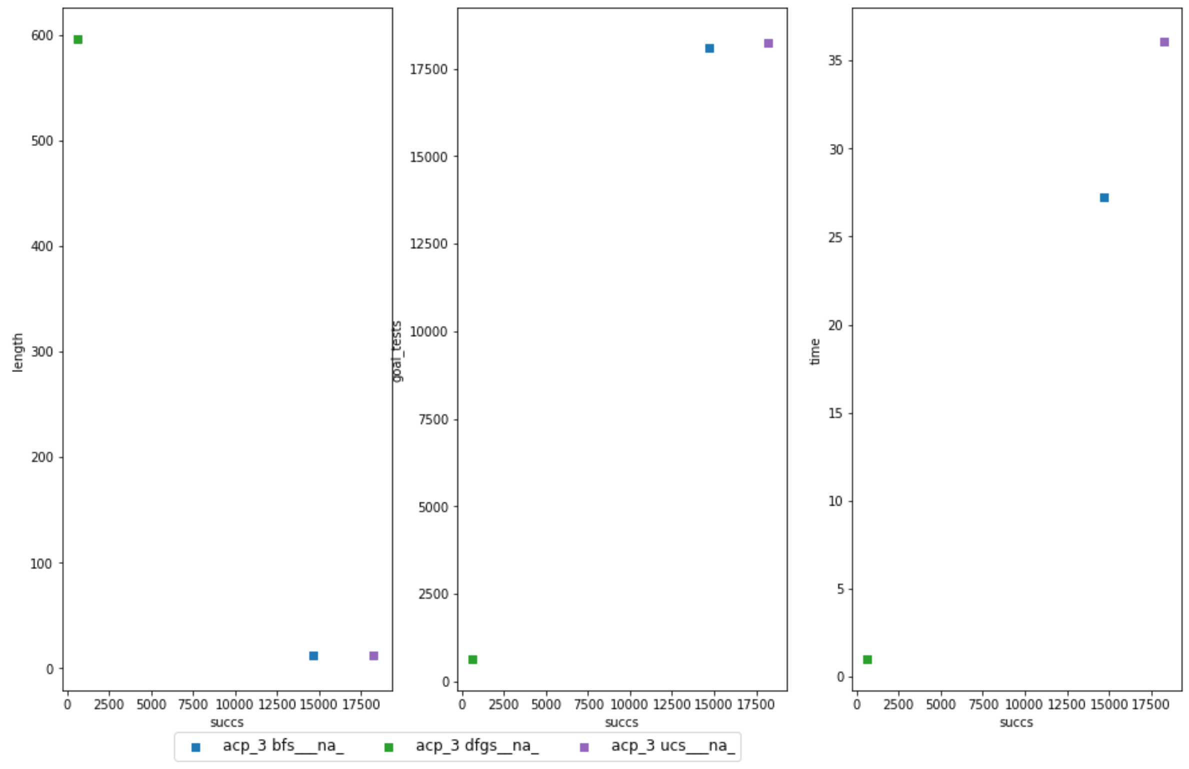
Problem 3

The optimal solution of length 12 is

- 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)

In [65]:

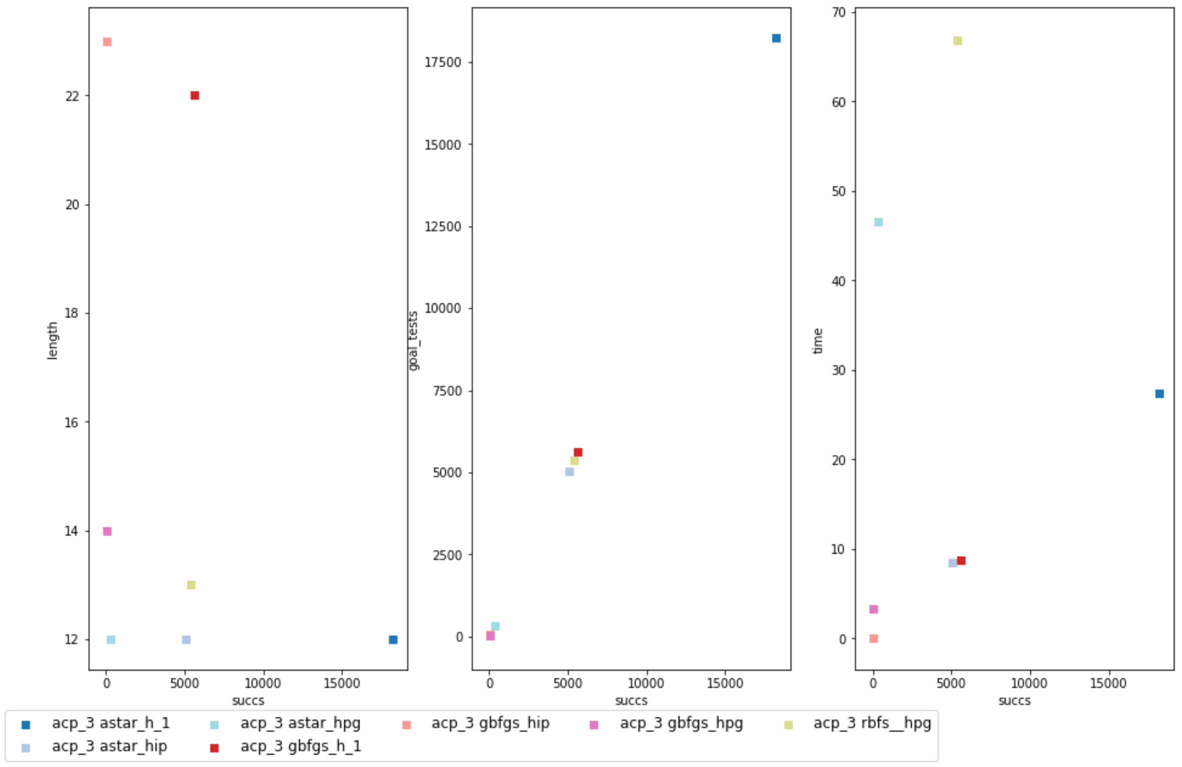
```
plot_subset((('succs','length', False, False),
             ('succs','goal_tests', False, False),
             ('succs','time', False, False)), problems=['Air Cargo Problem 3'],
             searches=simple_searches)
```



As for problem 2, depth-limited search and breadth-first tree search don't converge anymore. By somewhat lucky chance (pop-ing from a set results in a random item), the solution of depth-first search is nigh identical to the one in problem 2, even through the problems complexity has increased

In [66]:

```
plot_subset((('succs','length', False, False),
             ('succs','goal_tests', False, False),
             ('succs','time', False, False)), problems=['Air Cargo Problem 3'],
             searches=complex_searches)
```



As before, RBF search only converges for the `pg_levelsum` heuristic. Only a* search finds the optimal solution. a* with the `ignore_preconditions` heuristic is significantly faster than the same search with `pg_levelsum`, despite the latter requiring fewer goal tests and expansions. Interestingly, the RBF search that finds a solution does not find the optimal one

Problem 4

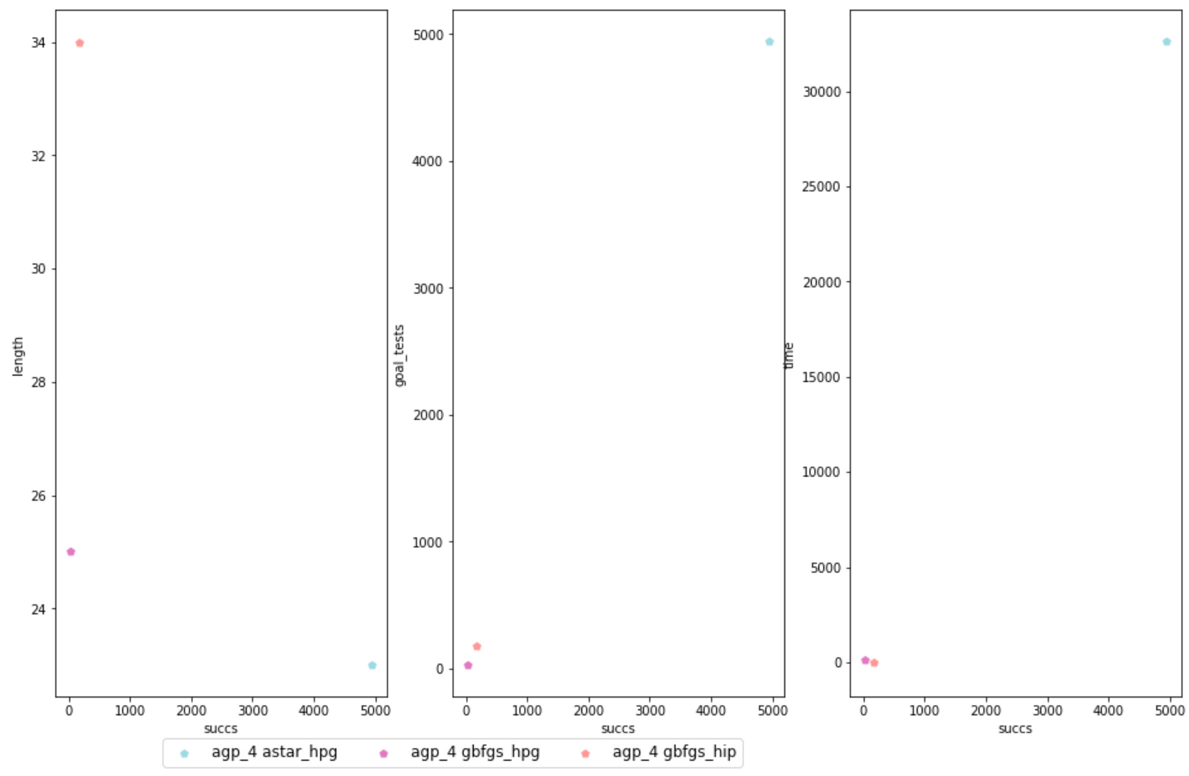
Just for fun - a significantly more complex problem. Solution:

- Fly(P2, YYZ, ORD)
- Load(C2, P2, JFK)
- Fly(P2, JFK, SFO)
- Load(C4, P2, ORD)
- Fly(P1, SFO, HKG)
- Unload(C4, P2, SFO)
- Unload(C2, P2, SFO)
- Load(C1, P2, SFO)
- Fly(P2, SFO, ATL)
- Load(C6, P2, ORD)
- Unload(C6, P2, ATL)
- Load(C3, P2, ATL)
- Fly(P2, ATL, JFK)
- Load(C5, P2, ORD)
- Fly(P2, ORD, SIN)
- Load(C8, P2, SIN)

- Unload(C8, P2, JFK)
- Unload(C3, P2, JFK)
- Unload(C1, P2, JFK)
- Fly(P2, JFK, HKG)
- Load(C7, P2, HKG)
- Unload(C7, P2, SIN)
- Unload(C5, P2, HKG) The optimal solution is only of length 23, illustrating that the algorithm is capable of finding paths that contain planes carrying multiple cargos

In [75]:

```
plot_subset((('succs','length', False, False),
             ('succs','goal_tests', False, False),
             ('succs','time', False, False)), problems=['Air Cargo Problem 4'],
             searches=complex_searches)
```


length
goal_tests
time
succs
succs
succs
agp_4 astar_hpg
agp_4 gbfgs_hpg
agp_4 gbfgs_hip