**Raw data from run\_search.py shows:**

==============================================

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*first problem\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

==============================================

Solving Air Cargo Problem 1 using breadth\_first\_search...

Expansions Goal Tests New Nodes

43 56 180

Plan length: 6 Time elapsed in seconds: 0.06741007995344729

----------------------------------------------------------------------

Solving Air Cargo Problem 1 using depth\_first\_graph\_search...

Expansions Goal Tests New Nodes

21 22 84

Plan length: 20 Time elapsed in seconds: 0.023054672229865264

--------------------------------------------------------------------------------

Solving Air Cargo Problem 1 using uniform\_cost\_search...

Expansions Goal Tests New Nodes

55 57 224

Plan length: 6 Time elapsed in seconds: 0.057928661265964645

------------------------------------------------------------------------

Solving Air Cargo Problem 1 using greedy\_best\_first\_graph\_search with h\_1...

Expansions Goal Tests New Nodes

7 9 28

Plan length: 6 Time elapsed in seconds: 0.008321193128550015

--------------------------------------------------------------------

Solving Air Cargo Problem 1 using astar\_search with h\_1...

Expansions Goal Tests New Nodes

55 57 224

Plan length: 6 Time elapsed in seconds: 0.05750736750927128

--------------------------------------------------------------------

Solving Air Cargo Problem 1 using astar\_search with h\_ignore\_preconditions...

Expansions Goal Tests New Nodes

55 57 224

Plan length: 6 Time elapsed in seconds: 0.05828529848996208

-------------------------------------------------------------------

Solving Air Cargo Problem 1 using astar\_search with h\_pg\_levelsum...

Expansions Goal Tests New Nodes

39 41 158

Plan length: 6 Time elapsed in seconds: 1.1064087452578795

==============================================

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*second problem\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

==============================================

Solving Air Cargo Problem 2 using breadth\_first\_search...

Expansions Goal Tests New Nodes

3343 4609 30509

Plan length: 9 Time elapsed in seconds: 13.254542506282203

------------------------------------------------------------

Solving Air Cargo Problem 2 using depth\_first\_graph\_search...

Expansions Goal Tests New Nodes

624 625 5602

Plan length: 619 Time elapsed in seconds: 7.286895916837331

----------------------------------------------------------------

Solving Air Cargo Problem 3 using uniform\_cost\_search...

Expansions Goal Tests New Nodes

18234 18236 159707

Plan length: 12 Time elapsed in seconds: 118.27142042705127

--------------------------------------------------------------

Solving Air Cargo Problem 2 using greedy\_best\_first\_graph\_search with h\_1...

Expansions Goal Tests New Nodes

990 992 8910

Plan length: 17 Time elapsed in seconds: 3.857492383614393

---------------------------------------------------------------------------------

Solving Air Cargo Problem 2 using astar\_search with h\_1...

Expansions Goal Tests New Nodes

4852 4854 44030

Plan length: 9 Time elapsed in seconds: 19.17898416376781

----------------------------------------------------------------------------

Solving Air Cargo Problem 2 using astar\_search with h\_ignore\_preconditions...

Expansions Goal Tests New Nodes

1450 1452 13303

Plan length: 9 Time elapsed in seconds: 5.897049866094269

---------------------------------------------------------------------

Solving Air Cargo Problem 2 using astar\_search with h\_pg\_levelsum...

Expansions Goal Tests New Nodes

1129 1131 10232

Plan length: 9 Time elapsed in seconds: 432.1446043353745

==============================================

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*third problem\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

==============================================

Solving Air Cargo Problem 3 using breadth\_first\_search...

Expansions Goal Tests New Nodes

14663 18098 129631

Plan length: 12 Time elapsed in seconds: 67.86497047608248

-------------------------------------------------------------------------------

Solving Air Cargo Problem 3 using depth\_first\_graph\_search...

Expansions Goal Tests New Nodes

408 409 3364

Plan length: 392 Time elapsed in seconds: 4.802690840683738

-------------------------------------------------------------------------

Solving Air Cargo Problem 3 using uniform\_cost\_search...

Expansions Goal Tests New Nodes

18234 18236 159707

Plan length: 12 Time elapsed in seconds: 118.27142042705127

------------------------------------------------------------

Solving Air Cargo Problem 3 using greedy\_best\_first\_graph\_search with h\_1...

Expansions Goal Tests New Nodes

5605 5607 49360

Plan length: 22 Time elapsed in seconds: 25.651536182770673

-----------------------------------------------------------------

solving Air Cargo Problem 3 using astar\_search with h\_1...

Expansions Goal Tests New Nodes

18234 18236 159707

Plan length: 12 Time elapsed in seconds: 85.09576402212895

----------------------------------------------------------------------

Solving Air Cargo Problem 3 using astar\_search with h\_ignore\_preconditions...

Expansions Goal Tests New Nodes

5040 5042 44944

Plan length: 12 Time elapsed in seconds: 24.690193092116573

-------------------------------------------------------------------------

Solving Air Cargo Problem 3 using astar\_search with h\_pg\_levelsum...

Expansions Goal Tests New Nodes

2026 2028 17933

Plan length: 12 Time elapsed in seconds: 1781.885962297031

**Summary:**



**Analysis**

First let’s talk about each problem optimal plan starting from 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)

Plan length: 6 steps

Problem 2 optimal plan:

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

Plan length: 9 steps

Problem 3 optimal plan:

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

Plan length: 12 steps

The goal is to get the number of minimum actions to reach the goal, so in my opinion the optimal search algorithm must give us the least plan length.

**For problem 1 “which had the least number of goals and preconditions”:**

greedy\_best\_first\_graph\_search with h\_1 is the optimal best search as it had the right plan length “minimum” and also the minimum time taken for execution, also the least number of goal tests and expansions which makes sense as this type of search starts with the node that is the nearest to the goal

**For problem 2 and problem 3 “both have more conditions and goals than problem 1” :**

greedy\_best\_first\_graph\_search with h\_1 is still the algorithm recording the least time, depth\_first\_graph\_search records the least number of goal tests and expansions which means less memory consumption, but both aren’t optimal “plan length of both is bigger than 9” which leads us to choose another search algorithm that is optimal with the least possible time for execution leading us to astar\_search with h\_ignore\_preconditions

**Comparing algorithms:**

|  |
| --- |
| **breadth\_first\_search:** This is a good un-informed algorithm that takes suitable time to reach an optimal plan  and I consider it as one of the best un-informed algorithm as it represent a balanced approach from the perspective of time, plan length and complexity “expansion and new nodes”, as the BFS heuristic searches the each level for the needed target so when we have a lot of nodes and expansions this algorithm will prove useful, but when compared to informed algorithms specially to astar\_search with h\_ignore\_preconditions, it loses. |
| **depth\_first\_graph\_search:** This is one of the fastest algorithms :second fastest algorithm, as it searches in depth until reaching a goal, so it doesn’t lead to un-needed expansions in each level, which leads to less complexity, but un-fortunately it doesn’t give us the an optimal plan for our cargo transfer, so we can’t use it. |
| **uniform\_cost\_search:** This an un-informed search algorithm and it isn’t an optimal as it has more complexity than BFS and takes more time but reaches an optimal plan for cargo transfer |
| **greedy\_best\_first\_graph\_search with h\_1:** This is the fastest algorithm and that is expected as it choses the node that is the nearest to the goal state using H\_1 heurestic, so it takes less time to get to the goal and also it’s the second algorithms in minimum number of nodes and minimum complexity but it doesn’t provide an optimal cargo transfer plan, so we can’t use it. |
| **astar\_search with h\_1:** This is also a good search algorithm that gives an optimal cargo transfer plan but BFS is better in all aspects “time, complexity and memory consumption” |
| **astar\_search with h\_ignore\_preconditions:** This informed heuristic is superior to BFS in all aspects which is expected as it’s an informed algorithm giving the optimal cargo transfer plan, so it’s our best choice here |
| **astar\_search with h\_pg\_levelsum**: This algorithm takes a lot of time and sometimes it timeouts so it can’t be considered |

Note that the best search algorithm is an informed search type and this is expected as according to Norvig and Russell's textbook at page 92:

***‘Informed search strategy—one that uses problem-specific knowledge beyond the definition of the problem itself—can find solutions more efficiently than can an uninformed strategy’ [1]***

***SO it was expected that Astar search informed with h\_ignore heuristic will be the best algorithm to choose***

**Summary:**

DFS is the best algorithm from memory consumption point of view, greedy\_best\_first\_graph\_search is the least time consuming but both don’t give us an optimized plan so the best algorithm here is the astar\_search with h\_ignore\_preconditions as it’s optimal and takes the least time and also less complex.

**References**:

 Video lessons from AIND

 Norvig and Russell's textbook ‘Artificial Intelligence A Modern Approach (3rd Edition)’[1]