Performance Analysis

A. Optimal Solutions

The optimal planning solution for each of the 3 problems are described below:

- ### Problem 1

  Plan length: 6

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

- ### Problem 2

  Plan length: 9

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

- ### Problem 3

  Plan length: 12

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

B. Non – Heuristic search methods comparison:

The Performance analysis of few search methods are displayed below in their respective tables.

Breadth First Search

Problem	Plan length	Expansions	Goal Tests	New Nodes	Time in seconds
1	6	43	56	180	0.029
2	9	3343	4609	30509	13.11
3	12	14663	18098	129631	97.73

Depth First Graph Search

Problem	Plan length	Expansions	Goal Tests	New Nodes	Time in seconds
1	20	21	22	84	0.013
2	619	624	625	5602	3.42
3	392	408	409	3364	1.71

Depth Limited Search

Problem	Plan length	Expansions	Goal Tests	New Nodes	Time in seconds
1	50	101	271	414	0.084
2	50	222719	2053741	2054119	884.45
3	timeout	Timeout	Timeout	Timeout	timeout

Uniform Cost Search

Problem	Plan length	Expansions	Goal Tests	New Nodes	Time in seconds
1	6	55	57	224	0.037
2	9	4780	4782	43381	40.88
3	12	17882	17884	156769	379.98

Conclusion:

Among the non-heuristic search functions, breadth first search and uniform cost search algorithms are the most optimal. Though Depth first graph search is faster, it doesn't give an optimal solution. Similarly, we can see that depth limited search is not optimal because many nodes are visited multiple times as it doesn't keep track of the visited paths/nodes.

C. Automatic Heuristics Comparison:

The Planning problem is solved with A* search using 2 automatic heuristics - **Ignore Preconditions and Level sum heuristics**. The comparison table is laid out below. It is very clear that though both give plan-length of equal size, Level Sum (planning graph) heuristic is better in terms of node expansions and goal tests.

Ignore Preconditions

Problem	Plan length	Expansions	Goal Tests	New Nodes	Time in seconds
1	6	41	43	170	0.043
2	9	1506	1508	13820	12.81
3	12	5114	5116	45610	81.23

Level sum with Planning Graph

Problem	Plan length	Expansions	Goal Tests	New Nodes	Time in seconds
1	6	11	13	50	3.280
2	9	86	88	841	902.39
3	12	404	406	3718	5230.89

Conclusion:

The most optimal solution in terms of node expansions is found using A-star search with planning graph and Level Sum heuristic. As we can see it is taking more time than the ignore preconditions heuristics, which can be rectified by using a better implementation for planning graph using non-dynamic languages like C++ or even Cython.

Overall Summary:

It is evident that; heuristic approach gives optimal solution with minimum node expansions when compared to non-heuristic search algorithms. In larger problems decreasing the node expansions will be much more beneficial.