Heuristic Analysis

•Problem 1

Search Strategy	Optimal	Path Length	Execution Time (s)	Node Expansions
Breadth First Search	**Yes**	**6**	0.065	43
Breadth First Tree Search	**Yes**	**6**	1.357	1458
Depth First Graph Search	No	12	**0.019**	12
Depth Limited Search	No	50	0.154	101
Uniform Cost Search	**Yes**	**6**	0.074	55
Recursive Best First Search	**Yes**	**6**	3.956	4229
Greedy Best First Graph Search	**Yes**	**6**	**0.013**	**7**
A* Search with h1 heuristic	**Yes**	**6**	0.077	55
A* Search with Ignore Preconditions heuristic	**Yes**	**6**	0.078	41
A* Search with h_pg_level heuristic	**Yes**	**6**	2.22	55

Optimal actions:

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

•Problem 2

Search Strategy	Optimal	Path Length	Execution Time (s)	Node Expansions
Breadth First Search	**Yes**	**9**	34.286	3346
Breadth First Tree Search	--	--	Longer than 10 min	--
Depth First Graph Search	No	1085	**19.336**	1124
Depth Limited Search	--	--	Longer than 10 min	--
Uniform Cost Search	**Yes**	**9**	78.178	4853

Search Strategy	Optimal	Path Length	Execution Time (s)	Node Expansions
Recursive Best First Search	--	--	Longer than 10 min	--
Greedy Best First Graph Search	No	21	3.848	998
A* Search with h1 heuristic	**Yes**	**9**	**29.946**	4853
A* Search with Ignore Preconditions heuristic	**Yes**	**9**	**8.223**	1450
A* Search with h_pg_level heuristic	--	--	Took longer than 10 min	--

Optimal actions:
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Load(C3, P3, ATL)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)
Fly(P2, JFK, SFO)
Unload(C2, P2, SFO)
Fly(P3, ATL, SFO)
Unload(C3, P3, SFO)

- Problem 3

Search Strategy	Optimal	Path Length	Execution Time (s)	Node Expansions
Breadth First Search	**Yes**	**12**	391.555	14120
Breadth First Tree Search	--	--	Longer than 10 min	--
Depth First Graph Search	No	2031	**119.541**	5591
Depth Limited Search	--	--	Longer than 10 min	--
Uniform Cost Search	**Yes**	**12**	155.174	18223
Recursive Best First Search	--	--	Longer than 10 min	--
Greedy Best First Graph Search	No	26	45.371	5655
A* Search with h1 heuristic	**Yes**	**12**	335.881	18223
A* Search with Ignore	**Yes**	**12**	**53.26**	5040

Search Strategy	Optimal	Path Length	Execution Time (s)	Node Expansions
Preconditions heuristic				
A* Search with h_pg_level heuristic	--	--	Took longer than 10 min	--

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

Uninformed Search Strategies:

From the heuristic analysis, Breadth First Search and Uniform Cost Search yield consistent and optimal search plans. Both these strategies require more memory usage though. Hence, if the path length needs to be optimal and is an important factor, then we must use **Breadth First Search**, as it takes lesser time and fewer node expansions on average, compared to the Uniform Cost Search.

If memory usage is a constraint, then the Depth First Graph Search relies on fewer node expansions than other strategies. However it does not yield an optimal search plan.

Explanation:

The reason Breadth First search takes shorter time is because the shallowest unexpanded node is chosen for expansion. Hence if the goal node is at a finite distance d, and if d is shallow, then breadth-first search eventually finds it. Since Breadth first search stores all expanded nodes in the explored set, it utilizes more memory, when compared to other uninformed search strategies.

Depth First Search takes the reverse approach to breadth first seatch. This search expands all the way down to the deepest level of the tree, before backing up to the next deepest node which still hasn't been explored yet. If in the process it encounters a goal node, then it will return that as a solution, even though that might not necessarily be the most optimum one. Hence, it does not yield an optimal plan all the time, and in worst case scenarios, it could perform poorly.

Informed Search Strategies:

Informed Search strategies take advantage of the fact that they know what a problem already is. Except for the A* Search with h1 heuristic and A* Search with preconditions ignored heuristic, all other search strategies take longer than 10 minutes. Based on the heuristic results from the

mentioned two, A* Search with preconditions ignored seems to be the best, yielding optimal results in shorter time compared to the other strategies.

Explanation:

The h_levelsum heuristic goes through more loops than the ignore preconditions heuristic, and that's the reason that it takes significantly longer time. The Ignore Preconditions heuristic on the other hand, tries to find the minimum number of actions that are needed to achieve the goal states.

Which one should be used:

Given a choice between the Breadth First Search, Uniform Cost Search and the A* Search with h1 heuristic, the A* Search with h1 heuristic seems to be the best search strategy for our given problems because it is faster and has less memory overhead.