Planning Heuristics Analysis

Problem Definition as Provided in the Readme

Problem 1:

Init(At(C1, SFO) ¨ At(C2, JFK)
¨ At(P1, SFO) ¨ At(P2, JFK)
¨ Cargo(C1) ¨ Cargo(C2)
¨ Plane(P1) ¨ Plane(P2)
¨ Airport(JFK) ¨ Airport(SFO))
Goal(At(C1, JFK) ¨ At(C2, SFO))

Problem 2:

Init(At(C1, SFO) ¨ At(C2, JFK) ¨ At(C3, ATL)
¨ At(P1, SFO) ¨ At(P2, JFK) ¨ At(P3, ATL)
¨ Cargo(C1) ¨ Cargo(C2) ¨ Cargo(C3)
¨ Plane(P1) ¨ Plane(P2) ¨ Plane(P3)
¨ Airport(JFK) ¨ Airport(SFO) ¨ Airport(ATL))
Goal(At(C1, JFK) ¨ At(C2, SFO) ¨ At(C3, SFO))

Problem 3:

Init(At(C1, SFO) ¨ At(C2, JFK) ¨ At(C3, ATL) ¨ At(C4, ORD)
¨ At(P1, SFO) ¨ At(P2, JFK)
¨ Cargo(C1) ¨ Cargo(C2) ¨ Cargo(C3) ¨ Cargo(C4)
¨ Plane(P1) ¨ Plane(P2)
¨ Airport(JFK) ¨ Airport(SFO) ¨ Airport(ATL) ¨ Airport(ORD))
Goal(At(C1, JFK) ¨ At(C3, JFK) ¨ At(C2, SFO) ¨ At(C4, SFO))

Optimal Sequence of Actions:

Problem 1: Optimal Plan length is 6 as below:

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

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

Problem 2: Optimal Plan length is 9 as below:

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

Problem 3: Optimal Plan length is 12 as below:

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

Comparison of non-heuristic Search Result Metrics

Problem 1: (See 'problem1_non_heuristic.txt')

Search Strategy	**Optimality**	**Time Elapsed (seconds)**	**No. Node Expansion**	**Path Length**
Breadth-first Search	Yes	0.027	43	6
Depth-first Graph Search	No	0.007	12	6

Uniform Cost Search	Yes	0.035	55	6

Problem 2: (See 'problem2_non_heuristic.txt')

Search Strategy	**Optimality**	**Time Elapsed (Seconds)**	**No. Node Expansion**	**Path Length**
Breadth-first Search	Yes	12.34	3401	9
Depth-first Graph Search	No	1.35	350	346
Uniform Cost Search	Yes	10.22	4761	9

Problem 3: (See 'problem3_non_heuristic.txt')

Search Strategy	**Optimality**	**Time Elapsed (Seconds)**	**No. Node Expansion**	**Path Length**
Breadth-first Search	Yes	91	14491	12
Depth-first Graph Search	No	18.49	1948	1878
Uniform Cost Search	Yes	44.71	17783	12

Analysis:

Optimality:

- Breadth-first Search and Uniform Cost search are Optimal and gives us the optimal path lengths 6, 9 and 12 for all three problems.

Time Elapsed:

- Depth-first Graph search is significantly faster for all three problems but is sub-optimal giving us plan lengths significantly higher than Breadth-first Search and Uniform Cost search.

Memory Usage: (No. Node expansion):

- Depth-first Graph search expands the least number of nodes when compared to optimal search strategies like Breadth-first Search and Uniform Cost search.

Justification: (3.7 AIMA 3rd Edition Solving Problems by Searching)

- **Breadth-first search** expands the shallowest nodes first; it is complete, optimal for unit step costs, but has exponential space complexity (therefore the large number of node expansion)
- **Depth-first search** expands the deepest unexpanded node first. It is neither complete nor optimal, but has linear space complexity (therefor the small number of Node expansions and the large path length)
- **Uniform-Cost search** expands the node with the lowest path cost, g(n) and it is optimal for general step costs (therefore it finds the optimal path length same as BFS)

Comparison of heuristic Search Result Metrics

Problem 1:(See 'problem1_heuristic.txt')

Search Strategy	**Optimality**	**Time Elapsed (Seconds)**	**No. Node Expansion**	**Path Length**
A* h1 Heuristic	Yes	0.030	55	6
A* Ignore Preconditions	Yes	0.033	41	6
A* Level-sum	Yes	0.917	11	6

Problem 2: (See 'problem2_heuristic.txt'**)**

Search Strategy	**Optimality**	**Time Elapsed (Seconds)**	**No. Node Expansion**	**Path Length**
A* h1 Heuristic	Yes	9.948	4761	9
A* Ignore	Yes	3.929	1450	9

Preconditions				
A* Level-sum	Yes	166.76	86	9

Problem 3: (See 'problem3_heuristic.txt')

Search Strategy	**Optimality**	**Time Elapsed (Seconds)**	**No. Node Expansion**	**Path Length**
A* h1 Heuristic	Yes	42.941	17783	12
A* Ignore Preconditions	Yes	13.685	5003	12
A* Level-sum	Yes	825.14	311	12

Analysis:

Optimality:
- A * h1 Heuristic, A* Ignore Preconditions, A* level-Sum are all optimal search strategies and gives us the optimal path length of 12.

Time Elapsed:
- A * with Ignore Pre-conditions heuristic is faster on an average for 3 problems.

Memory Usage: (No. Node expansion):
- A * with Level-sum Heuristic has the least memory usage amongst the 3 search strategies under consideration with the least number of goal expansions.

Note: A* Level- Sum search strategy takes more than 10 minutes.

Comparison of non-heuristic and heuristic search strategies:

- Optimal search strategies are A* with heuristics and Breadth-first search and Uniform cost search.
- Comparing 2 optimal search strategies below, Uniform cost search and A* with Ignore pre-conditions

Problem 1:

Search Strategy	**Optimality**	**Time Elapsed (Seconds)**	**No. Node Expansion**	**Path Length**

Uniform Cost Search	Yes	0.035	55	6
A* Ignore Preconditions	Yes	0.033	41	6

Problem 2:

Search Strategy	**Optimality**	**Time Elapsed (Seconds)**	**No. Node Expansion**	**Path Length**
Uniform Cost Search	Yes	10.22	4761	9
A* Ignore Preconditions	Yes	3.929	1450	9

Problem 3:

Search Strategy	**Optimality**	**Time Elapsed (Seconds)**	**No. Node Expansion**	**Path Length**
Uniform Cost Search	Yes	44.71	17783	12
A* Ignore Preconditions	Yes	13.685	5003	12

Justification: (Figure 3.29 AIMA 3rd Edition Solving Problems by Searching)

	Search Cost (nodes generated)			Effective Branching Factor		
d	IDS	$A^*(h_1)$	$A^*(h_2)$	IDS	$A^*(h_1)$	$A^*(h_2)$
2	10	6	6	2.45	1.79	1.79
4	112	13	12	2.87	1.48	1.45
6	680	20	18	2.73	1.34	1.30
8	6384	39	25	2.80	1.33	1.24
10	47127	93	39	2.79	1.38	1.22
12	3644035	227	73	2.78	1.42	1.24
14	–	539	113	–	1.44	1.23
16	–	1301	211	–	1.45	1.25
18	–	3056	363	–	1.46	1.26
20	–	7276	676	–	1.47	1.27
22	–	18094	1219	–	1.48	1.28
24	–	39135	1641	–	1.48	1.26

Figure 3.29 Comparison of the search costs and effective branching factors for the ITERATIVE-DEEPENING-SEARCH and A^* algorithms with h_1, h_2. Data are averaged over 100 instances of the 8-puzzle for each of various solution lengths d.

Above Table depicts the number of nodes, Branching factor expanded for Iterative Deepening Search (Uninformed) vs A* with Heuristics for 1200 random 8-Puzzle problem and solution lengths from 2 to 24. As can be seen informed search strategies always have a lower branching factor and the number of nodes expanded

Summary:

- Heuristic Search strategies are much more memory and time effective over uninformed search strategies as expected.
- Also, in real world applications, human domain experts provide effective heuristic that help reduce the search space so an AI Search combined with Heuristics (which does not need to be clearly stated, can be learned through supervised learning) is effective than uninformed search strategies. Eg of this is the AlphaGo using Monte Carlo Tree search along with training by supervised learning.