Plane Cargo Planning Search Heuristic Analysis

Optimal Plan

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)

Problem 2

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

Problem 3

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

Summary

Search	Problem	Expansions	Goal Tests	New Nodes	Plan Length	Elapsed Time (s)
1	1	43	56	180	6	0.03
	2	769	979	5,041	9	1.72
	3	14,663	18,098	129,631	12	119.82
2	1	1,458	1,459	5,960	6	0.97
	2	DNF	DNF	DNF	DNF	DNF
	3	DNF	DNF	DNF	DNF	DNF
3	1	21	22	84	20	0.02
	2	81	82	502	76	0.17
	3	408	409	3,364	392	2.35
4	1	101	271	414	50	0.09
	2	DNF	DNF	DNF	DNF	DNF
	3	DNF	DNF	DNF	DNF	DNF
5	1	55	57	224	6	0.04
	2	992	994	6,413	9	1.95
	3	18,223	18,225	159,618	12	55.34

Search	Problem	Expansions	Goal Tests	New Nodes	Plan Length	Elapsed Time (s)
8	1	55	57	224	6	0.05
	2	992	994	6,413	9	1.99
	3	18,223	18,225	159,618	12	53.60
9	1	41	43	170	6	0.04
	2	443	445	2,921	9	1.32
	3	5,040	5,042	44,944	12	18.29
10	1	55	57	224	6	1.37
	2	581	583	3,790	9	292.24
	3	DNF	DNF	DNF	DNF	DNF

Detailed Runs

```
python run_search.py -p 1 -s 1
Solving Air Cargo Problem 1 using breadth_first_search...
Expansions   Goal Tests   New Nodes
    43          56         180
Plan length: 6  Time elapsed in seconds: 0.033023833440357496
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P2, JFK, SFO)
Unload(C2, P2, SFO)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)
```

```
python run_search.py -p 2 -s 1
Solving Air Cargo Problem 2 using breadth_first_search...
Expansions   Goal Tests   New Nodes
   769         979         5041
Plan length: 9  Time elapsed in seconds: 1.7234296342540387
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P2, JFK, ATL)
Load(C3, P2, ATL)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)
Fly(P2, ATL, SFO)
Unload(C2, P2, SFO)
Unload(C3, P2, SFO)
```

```
python run_search.py -p 3 -s 1
Solving Air Cargo Problem 3 using breadth_first_search...
Expansions   Goal Tests   New Nodes
  14663       18098       129631
Plan length: 12  Time elapsed in seconds: 119.81605781892941
Load(C1, P1, SFO)
Load(C2, P2, JFK)
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)
```

```
python run_search.py -p 1 -s 2
Solving Air Cargo Problem 1 using breadth_first_tree_search...
Expansions   Goal Tests   New Nodes
   1458        1459        5960
Plan length: 6  Time elapsed in seconds: 0.969300651025223
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P2, JFK, SFO)
Unload(C2, P2, SFO)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)
```

```
python run_search.py -p 2 -s 2
Solving Air Cargo Problem 2 using breadth_first_tree_search...
DNR
```

```
python run_search.py -p 3 -s 2
Solving Air Cargo Problem 3 using breadth_first_tree_search...
DNR
```

```
python run_search.py -p 1 -s 3
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: 0.01671942013282896
Fly(P1, SFO, JFK)
Fly(P2, JFK, SFO)
Load(C2, P1, JFK)
Fly(P1, JFK, SFO)
Fly(P2, SFO, JFK)
Unload(C2, P1, SFO)
Fly(P1, SFO, JFK)
Fly(P2, JFK, SFO)
Load(C2, P2, SFO)
Fly(P1, JFK, SFO)
Load(C1, P2, SFO)
Fly(P2, SFO, JFK)
Fly(P1, SFO, JFK)
Unload(C2, P2, JFK)
Unload(C1, P2, JFK)
Fly(P2, JFK, SFO)
Load(C2, P1, JFK)
Fly(P1, JFK, SFO)
Fly(P2, SFO, JFK)
Unload(C2, P1, SFO)
```

```
python run_search.py -p 2 -s 3
Solving Air Cargo Problem 2 using depth_first_graph_search...
Expansions   Goal Tests   New Nodes
   81         82        502
Plan length: 76  Time elapsed in seconds: 0.16632848816583495
Fly(P1, SFO, ATL)
Fly(P2, JFK, ATL)
Fly(P1, ATL, JFK)
Fly(P2, ATL, SFO)
Load(C2, P1, JFK)
…
Unload(C2, P2, SFO)
Fly(P2, SFO, ATL)
Fly(P1, SFO, ATL)
Fly(P2, ATL, JFK)
Fly(P1, ATL, JFK)
Unload(C1, P2, JFK)
```

```
python run_search.py -p 3 -s 3
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: 2.3463289968838787
Fly(P1, SFO, ORD)
Fly(P2, JFK, ORD)
Fly(P1, ORD, ATL)
Fly(P2, ORD, ATL)
Fly(P1, ATL, JFK)
Fly(P2, ATL, SFO)
Load(C2, P1, JFK)
…
Unload(C2, P2, SFO)
Fly(P2, SFO, ORD)
Fly(P1, SFO, ORD)
Fly(P2, ORD, ATL)
Fly(P1, ORD, ATL)
Fly(P2, ATL, JFK)
Fly(P1, ATL, JFK)
Unload(C3, P1, JFK)
```

```
python run_search.py -p 1 -s 4
Solving Air Cargo Problem 1 using depth_limited_search...
Expansions   Goal Tests   New Nodes
   101         271         414
Plan length: 50  Time elapsed in seconds: 0.09122653259119999
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Unload(C1, P1, SFO)
…
Unload(C2, P2, SFO)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)
```

```
python run_search.py -p 2 -s 4
Solving Air Cargo Problem 2 using depth_limited_search...
DNF
```

```
python run_search.py -p 3 -s 4
Solving Air Cargo Problem 3 using depth_limited_search...
DNF
```

```
python run_search.py -p 1 -s 5
Solving Air Cargo Problem 1 using uniform_cost_search...
Expansions   Goal Tests   New Nodes
    55          57         224
Plan length: 6  Time elapsed in seconds: 0.038812063043552954
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P1, SFO, JFK)
Fly(P2, JFK, SFO)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)
```

python run_search.py -p 2 -s 5

```
Solving Air Cargo Problem 2 using uniform_cost_search...
Expansions   Goal Tests   New Nodes
   992         994         6413
Plan length: 9  Time elapsed in seconds: 1.9511916273934848
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P2, JFK, ATL)
Load(C3, P2, ATL)
Fly(P1, SFO, JFK)
Fly(P2, ATL, SFO)
Unload(C3, P2, SFO)
Unload(C2, P2, SFO)
Unload(C1, P1, JFK)
```

python run_search.py -p 3 -s 5

```
Solving Air Cargo Problem 3 using uniform_cost_search...
Expansions   Goal Tests   New Nodes
  18223       18225       159618
Plan length: 12  Time elapsed in seconds: 55.34345340120405
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(P2, ORD, SFO)
Fly(P1, ATL, JFK)
Unload(C4, P2, SFO)
Unload(C3, P1, JFK)
Unload(C2, P2, SFO)
Unload(C1, P1, JFK)
```

python run_search.py -p 1 -s 8

```
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: 0.04660080404225572
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P1, SFO, JFK)
Fly(P2, JFK, SFO)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)
```

```
python run_search.py -p 2 -s 8
Solving Air Cargo Problem 2 using astar_search with h_1...
Expansions   Goal Tests   New Nodes
   992        994        6413
Plan length: 9  Time elapsed in seconds: 1.991041653744187
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P2, JFK, ATL)
Load(C3, P2, ATL)
Fly(P1, SFO, JFK)
Fly(P2, ATL, SFO)
Unload(C3, P2, SFO)
Unload(C2, P2, SFO)
Unload(C1, P1, JFK)
```

```
python run_search.py -p 3 -s 8
Solving Air Cargo Problem 3 using astar_search with h_1...
Expansions   Goal Tests   New Nodes
  18223      18225      159618
Plan length: 12  Time elapsed in seconds: 53.5967329591727
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(P2, ORD, SFO)
Fly(P1, ATL, JFK)
Unload(C4, P2, SFO)
Unload(C3, P1, JFK)
Unload(C2, P2, SFO)
Unload(C1, P1, JFK)
```

```
python run_search.py -p 1 -s 9
Solving Air Cargo Problem 1 using astar_search with h_ignore_preconditions...
Expansions   Goal Tests   New Nodes
    41         43        170
Plan length: 6  Time elapsed in seconds: 0.041061578534240514
Load(C1, P1, SFO)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)
Load(C2, P2, JFK)
Fly(P2, JFK, SFO)
Unload(C2, P2, SFO)
```

python run_search.py -p 2 -s 9

```
Solving Air Cargo Problem 2 using astar_search with h_ignore_preconditions...
Expansions   Goal Tests   New Nodes
   443         445         2921
Plan length: 9  Time elapsed in seconds: 1.1371106689329906
Load(C2, P2, JFK)
Fly(P2, JFK, ATL)
Load(C3, P2, ATL)
Fly(P2, ATL, SFO)
Unload(C3, P2, SFO)
Unload(C2, P2, SFO)
Load(C1, P1, SFO)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)
```

python run_search.py -p 3 -s 9

```
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: 18.28552424956279
Load(C2, P2, JFK)
Fly(P2, JFK, ORD)
Load(C4, P2, ORD)
Fly(P2, ORD, SFO)
Unload(C4, P2, SFO)
Load(C1, P1, SFO)
Fly(P1, SFO, ATL)
Load(C3, P1, ATL)
Fly(P1, ATL, JFK)
Unload(C3, P1, JFK)
Unload(C2, P2, SFO)
Unload(C1, P1, JFK)
```

python run_search.py -p 1 -s 10

```
Solving Air Cargo Problem 1 using astar_search with h_pg_levelsum...
Expansions   Goal Tests   New Nodes
    55          57         224
Plan length: 6  Time elapsed in seconds: 1.373673484232195
Load(C2, P2, JFK)
Fly(P2, JFK, SFO)
Unload(C2, P2, SFO)
Load(C1, P1, SFO)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)
```

```
python run_search.py -p 2 -s 10
Solving Air Cargo Problem 2 using astar_search with h_pg_levelsum...
Expansions   Goal Tests   New Nodes
   581         583         3790
Plan length: 9  Time elapsed in seconds: 292.24293160821395
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P2, JFK, ATL)
Load(C3, P2, ATL)
Fly(P2, ATL, SFO)
Unload(C2, P2, SFO)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)
Unload(C3, P2, SFO)
```

```
python run_search.py -p 3 -s 10
Solving Air Cargo Problem 3 using astar_search with h_pg_levelsum...
DNF
```

Comparison

Compare and contrast non-heuristic search result metrics (optimality, time elapsed, number of node expansions) for Problems 1,2, and 3. Include breadth-first, depth-first, and at least one other uninformed non-heuristic search in your comparison; Your third choice of non-heuristic search may be skipped for Problem 3 if it takes longer than 10 minutes to run, but a note in this case should be included.

Breadth-first search provided an optimal solution across all three problems. As we learned, due to the path of the path of traversal in breadth-first search, it is guaranteed to produce the optimal solution given one exists. On the other hand, depth-first search will find a solution; but, the solution is not guaranteed to be an optimal solution. Alike, the solutions to the problems proposed by depth-first search have plan length many times that of the optimal solution. Lastly, uniform-cost search, much like breadth-first search, produced an optimal solution for all three problems. Though, uniform-cost search solved problem 3 in less time than breadth-first search but the number of node expansions required was higher. In all cases, depth-first search executed the fastest and required the least node expansions; however, never produced an optimal solution to any of the three problems.

Compare and contrast heuristic search result metrics using A* with the "ignore preconditions" and "level-sum" heuristics for Problems 1, 2, and 3.

Both ignore-preconditions and level-sum heuristics produced optimal solutions for both problem 1 and 2. Ignore-preconditions additionally produced an optimal solution for problem 3; but, level-sum had not terminated after 2.5 hours of execution. Of the heuristic searches, ignore-preconditions performed the best across the board. With both heuristics having the same plan length, ignore-preconditions performed better both in terms of execution time and number of node expansions.

What was the best heuristic used in these problems? Was it better than non-heuristic search planning methods for all problems? Why or why not?

The best heuristic overall was ignore-preconditions. For each problem, ignore-preconditions proposed an optimal solution. It also performed better than all non-heuristic search algorithms tested. In all cases, ignore-preconditions made the least number of node expansions for each problem; and, it executed faster than all the algorithms for the respective problems except for a 0.1 second difference with breadth-first search for problem 1. Heuristic searches like A* take shortcuts to reduce the search space in finding an optimal solution. And, given the heuristic is executes performantly and relatively quickly while still effectively estimating the cost of reaching the goal, heuristic searches can outperform their non-heuristic counterparts.