Introduction

In this project I Implement methods and functions in **my_air_cargo_problems.py**
needed to solve deterministic logistics planning problems for an Air Cargo transport system using a planning search agent. Problem definitions based on initial states and goals provided were represented for the agent defined in the classical PDDL. I used a planning graph and domain-independent heuristics with A* search and compare their results/performance against several uninformed non-heuristic search methods like breadth-first, depth-first, etc.

Full Metric Table for Problem 1:

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time Elapsed (s)
BFS	43	56	180	6	0.02510137062743543
BFTS	1458	1459	5960	6	0.8740913515944279
DFGS	12	13	48	12	0.012284595731999137
DLS	101	271	414	50	0.08088325015499709
UCS	55	57	224	6	0.03518046385496154
RBFSH1	4229	4230	17029	6	2.287891061404732
GBFGSH1	7	9	28	6	0.005187226331616802
A*SH1	55	57	224	6	0.04219952689272377
A*SHIP	41	43	170	6	0.032063414469115425
A*SHPGLS	11	13	50	6	0.8114660281530379

Full Metric Table for Problem 2:

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time Elapsed (s)
BFS	3401	4672	31049	9	12.404242912418896
BFTS					Too Long
DFGS	350	351	3142	46	1.28835671817274
DLS					Too Long
UCS	4761	4763	43206	9	10.156481961217064
RBFSH1					Too Long
GBFGSH1	550	552	4950	9	1.1131120202927516
A*SH1	4761	4763	43206	9	9.82590708988088
A*SHIP	1450	1452	13303	9	4.145986627858136
A*SHPGLS	86	88	841	9	172.45329728706466

Metric Table for Problem 3:

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time Elapsed (s)
BFS	14491	17947	128184	12	92.26858930327485
BFTS					Too Long
DFGS	1948	1949	16253	1878	18.019126899261906
DLS					Too Long
UCS	17783	17785	155920	12	43.68629368763288
RBFSH1					Too Long
GBFGSH1	4031	4033	35794	22	10.172597866797837
A*SH1	17783	17785	155920	12	48.25741924276227
A*SHIP	5003	5005	44586	12	17.102076905210907
A*SHPGLS	--	--	--	--	Too Long

Optimal Plan

Problem 1:

Path Length : 6

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:

Path Length : 9

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

Problem 3:

Path Length : 12

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

Uninformed Search Strategies Analysis

Uninformed search strategies have no additional information about states beyond that provided in the problem definition. All they can do is generate successors and distinguish a goal state from a non-goal state. In this section, I compared the performance of three such strategies in terms of **speed** (execution time, measured in seconds), **memory usage** (measured in search node expansions) and **optimality** which is the plan length in nodes when optimal solution was found

Performance Metric for Problem 1

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time Elapsed (s)
BFS	43	56	180	6	0.02510137062743543
BFTS	1458	1459	5960	6	0.8740913515944279
DFGS	12	13	48	12	0.012284595731999137
DLS	101	271	414	50	0.08088325015499709
UCS	55	57	224	6	0.03518046385496154
RBFSH1	4229	4230	17029	6	2.287891061404732
GBFGSH1	7	9	28	6	0.005187226331616802

Metric Table for Problem 2:

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time Elapsed (s)
BFS	3401	4672	31049	9	12.404242912418896
BFTS					Too Long
DFGS	350	351	3142	46	1.28835671817274
DLS					Too Long
UCS	4761	4763	43206	9	10.156481961217064
RBFSH1					Too Long
GBFGSH1	550	552	4950	9	1.1131120202927516

Metric Table for Problem 3:

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time Elapsed (s)
BFS	14491	17947	128184	12	92.26858930327485
BFTS					Too Long
DFGS	1948	1949	16253	1878	18.019126899261906
DLS					Too Long
UCS	17783	17785	155920	12	43.68629368763288
RBFSH1					Too Long
GBFGSH1	4031	4033	35794	22	10.172597866797837

Analysis:

Given these three problems and running seven algorithms of uninformed search, I found that Breadth First Search and Uniform Cost Search are the only two uninformed search strategies that yield an optimal action plan.
under the 10 min time, limit. When it comes to execution speed and memory usage, Depth First Graph Search is the fastest and uses the least memory.
However, it does not generate an optimal action plan (problem 1: plan length of 12 instead of 6, problem 2: plan length of 46 instead of 9, problem 3: plan length of 1878 instead of 12).
If finding the optimal path length is critical, what strategy should we use? Because it performs faster and uses less memory than Uniform Cost Search,
Breadth First Search is the recommended search strategy. This is not much of a surprise, as BFS is complete and optimal. Its only downside is memory usage, if the problem's branching factor is high, as shown in [1] section 3.4.7:
Which search strategy should we use, if having an optimal path length is not the primary criteria? For problems 2 and 3, the Depth First Graph Search plan
lengths are so much longer than the optimal path length that it would not make sense to use this search strategy.
Greedy Best First Graph Search is the best alternative. In problems 1 and 2, it manages to find the optimal path. In problem 3, it does not find the optimal path but the path length it generates is 22

instead of 12, which is much better than Depth First Graph Search (1878 path length!). Moreover, it still provides execution time savings and uses less memory than the best search strategy for an optimal solution (Breadth First Search).

Informed (Heuristic) Search Strategies Analysis

A* Heuristic Metric for Problem 1

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time Elapsed (s)
A*SH1	55	57	224	6	0.04219952689272377
A*SHIP	41	43	170	6	0.032063414469115425
A*SHPGLS	11	13	50	6	0.8114660281530379

A* Heuristic Metric for Problem 2

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time Elapsed (s)
A*SH1	4761	4763	43206	9	9.82590708988088
A*SHIP	1450	1452	13303	9	4.145986627858136
A*SHPGLS	86	88	841	9	172.45329728706466

A* Heuristic Metric for Problem 3

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time Elapsed (s)
A*SH1	17783	17785	155920	12	48.25741924276227
A*SHIP	5003	5005	44586	12	17.102076905210907
A*SHPGLS	--	--	--	--	Too Long

Analysis

While all heuristics yield an optimal action plan, only the h1 and Ignore Preconditions heuristics return results within the 10 min max execution time. Which heuristic should we use? Of the two strategies mentioned above, **A* Search with Ignore Preconditions heuristic is the fastest**

Informed vs Uninformed Search Strategies

The search strategies that generate optimal plans are Greedy_best_first_graph_search h_1, Uniform Cost Search and A* Search with all three heuristics. When it comes to execution speed and memory usage of uninformed search strategies, Depth First Graph Search is faster and uses less memory than Uniform Cost Search.

For informed search strategies, A* Search with Ignore Preconditions heuristic is the fastest and uses the least memory. Therefore, really, the choice is between Depth First Graph, Greedy_best_first_graph_search h_1, Uniform Cost Search and A* Search with Ignore Preconditions heuristic.

Problem 1 Results:

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time Elapsed (s)
GBFGSH1	7	9	28	6	0.005187226331616802
A*SHIP	41	43	170	6	0.032063414469115425

Problem 2 Results:

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time Elapsed (s)
GBFGSH1	550	552	4950	9	1.1131120202927516
A*SHIP	1450	1452	13303	9	4.145986627858136

Problem 3 Results:

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time Elapsed (s)
UCS	17783	17785	155920	12	43.68629368763288
A*SHIP	5003	5005	44586	12	17.102076905210907

Best Heuristic

From the results above, because it is faster and uses less memory, **A* Search with Ignore Preconditions heuristic** would be the best choice overall for our Air Cargo problem.

Conclusion:

As per the experiments and results found, level sum was quite slow to perform due to the heavy cost of the calculation for the heuristic- but it was able to minimize the number of nodes that had to be Examined suggesting that the heuristic was very good. Preconditions was a good balance in that the cost of calculating the heuristic was quite low and it did manage to lead the search generally in a good direction, allowing it to perform even faster than DFS while providing an optimal solution.