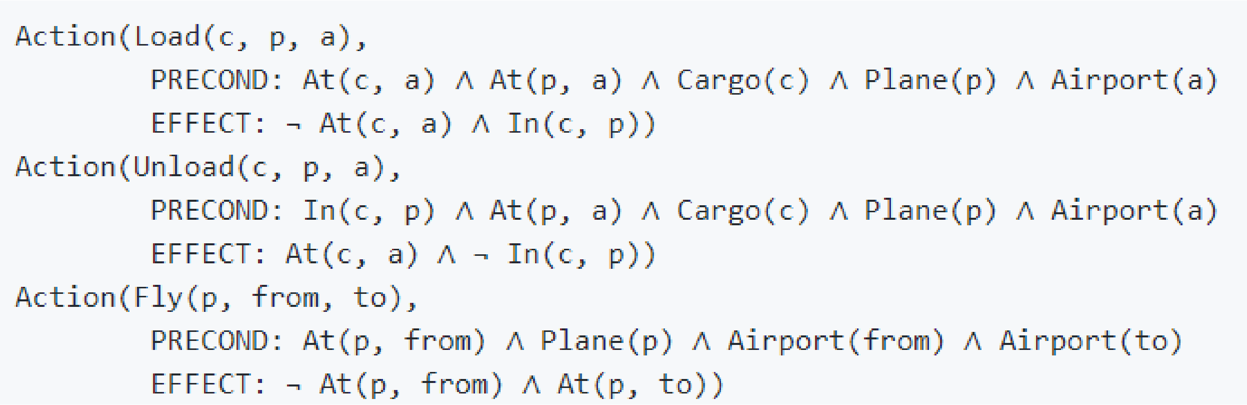
Heuristic Review

By Soeun

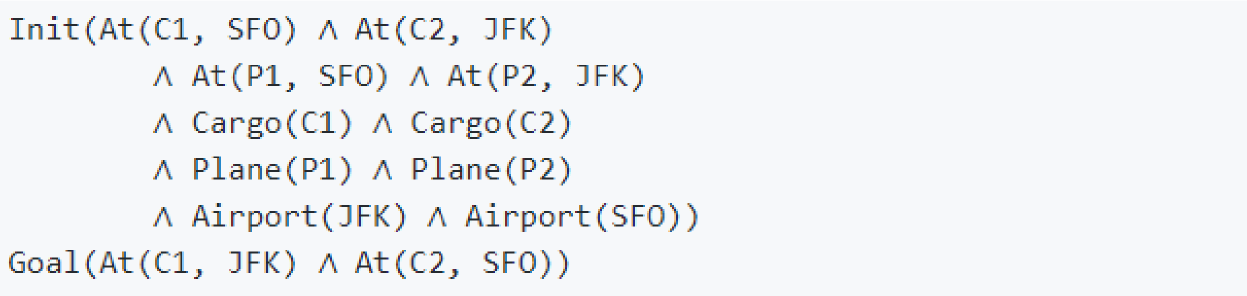
All problems are in the Air Cargo domain. They have the same action schema defined, but different initial states and goals.

Air Cargo Action Schema:



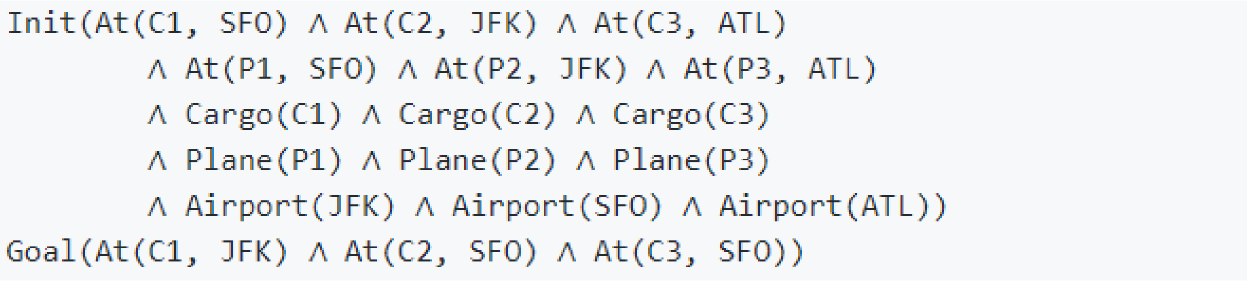
```
Action(Load(c, p, a),
        PRECOND: At(c, a) ∧ At(p, a) ∧ Cargo(c) ∧ Plane(p) ∧ Airport(a)
        EFFECT: ¬ At(c, a) ∧ In(c, p))
Action(Unload(c, p, a),
        PRECOND: In(c, p) ∧ At(p, a) ∧ Cargo(c) ∧ Plane(p) ∧ Airport(a)
        EFFECT: At(c, a) ∧ ¬ In(c, p))
Action(Fly(p, from, to),
        PRECOND: At(p, from) ∧ Plane(p) ∧ Airport(from) ∧ Airport(to)
        EFFECT: ¬ At(p, from) ∧ At(p, to))
```

Problem 1 initial state and goal:



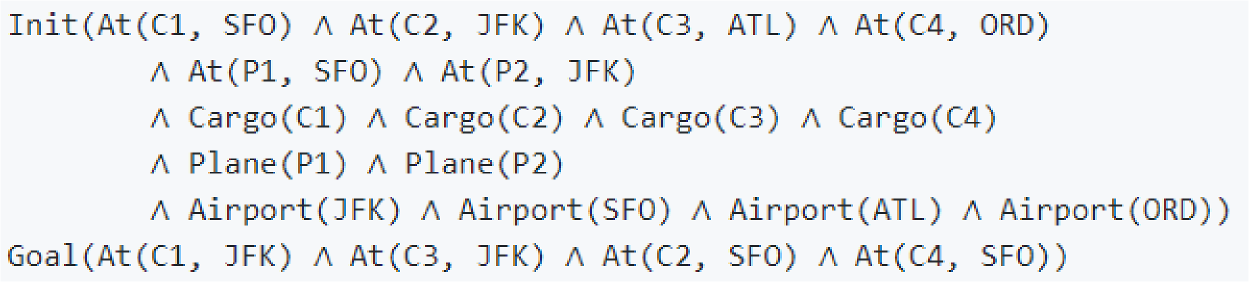
```
Init(At(C1, SFO) ∧ At(C2, JFK)
        ∧ At(P1, SFO) ∧ At(P2, JFK)
        ∧ Cargo(C1) ∧ Cargo(C2)
        ∧ Plane(P1) ∧ Plane(P2)
        ∧ Airport(JFK) ∧ Airport(SFO))
Goal(At(C1, JFK) ∧ At(C2, SFO))
```

Problem 2 initial state and goal:



```
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 initial state and goal:



```
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))
```

Problem 1 result:

Search	Plan Length	Time	Expansions	Optimal
Breadth First Search	6	0.033	43	O
Breath First Tree Search	6	1.030	1458	O
Depth First Graph Search	20	0.015	21	X
Depth Limited Search	50	0.103	101	X
Uniform Cost Search	6	0.042	55	O
Recursive Best First Search with h_1	6	3.029	4229	O
Greedy Best First Graph Search with h_1	6	0.005	7	O
A* Search with h_1	6	0.0433	55	O
A* Search with h_ignore_preconditions	6	0.0423	41	O
A* Search with h_pg_levelsum	6	1.295	11	O

Problem 2 result:

Search	Plan Length	Time	Expansions	Optimal
Breadth First Search	9	12.027	3401	O
Breath First Tree Search				
Depth First Graph Search	1138	11.858	1192	X
Depth Limited Search				
Uniform Cost Search	9	17.970	4761	O
Recursive Best First Search with h_1				
Greedy Best First Graph Search with h_1	9	1.804	550	O
A* Search with h_1	9	15.469	4761	O
A* Search with h_ignore_preconditions	9	5.953	1450	O
A* Search with h_pg_levelsum	9	185.982	86	O

Problem 3 result:

Search	Plan Length	Time	Expansions	Optimal
Breadth First Search	12	53.555	14491	O

Breath First Tree Search				
Depth First Graph Search	2014	27.258	2099	X
Depth Limited Search				
Uniform Cost Search	12	74.737	17783	O
Recursive Best First Search with h_1				
Greedy Best First Graph Search with h_1	22	18.786	4031	X
A* Search with h_1	12	92.557	17783	O
A* Search with h_ignore_preconditions	12	31.558	5003	O
A* Search with h_pg_levelsum				

Analysis:

Non-heuristic search strategies such as BFS, Uniform Cost Search, and DFS find a solution to all problems but doesn’t generate an optimal solution. BFS and Uniform Cost Search generate an optimal action plan to all problems but slower and use more memory than DFS. On the other hand, DFS doesn’t generate an optimal action plan but faster and use less memory than others.

Most of heuristic search strategies find an optimal action plan to all problems. A* Search with h_ignore_preconditions generate an optimal solution to all problems and the fastest among others. A* Search with h_pg_levelsum doesn’t generate an optimal solution to 3 problem and its performance is bad because it is too complex.

According to the results, BFS which is non-heuristic search strategy can solve most planning problems fast and optimally. If the complexity of the problem increases, heuristic search strategies could be considered and perform better than non-heuristic.

In addition, according to video, BFS always finds the shortest path and find a solution. And, DFS find a fast solution and uses less memory but it cannot find optimal solution because it explore nodes as deeply as possible. I can check these facts and these are proved by above result.

Problem 1 :

Search	An Optimal Sequence of actions

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

Breath First Tree Search	Load(C1, P1, SFO)
	Load(C2, P2, JFK)
	Fly(P2, JFK, SFO)
	Unload(C2, P2, SFO)
	Fly(P1, SFO, JFK)
	Unload(C1, P1, JFK)

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

Greedy Best First Graph Search with h_1	Load(C1, P1, SFO)
	Load(C2, P2, JFK)
	Fly(P1, SFO, JFK)
	Fly(P2, JFK, SFO)
	Unload(C1, P1, JFK)
	Unload(C2, P2, SFO)

Recursive Best First Search with h_1	Load(C2, P2, JFK)
	Load(C1, P1, SFO)
	Fly(P2, JFK, SFO)
	Unload(C2, P2, SFO)
	Fly(P1, SFO, JFK)
	Unload(C1, P1, JFK)

A* Search with h_1	Load(C1, P1, SFO)
	Load(C2, P2, JFK)
	Fly(P1, SFO, JFK)
	Fly(P2, JFK, SFO)
	Unload(C1, P1, JFK)
	Unload(C2, P2, SFO)

A* Search with h_ignore_preconditions	Load(C1, P1, SFO)
	Fly(P1, SFO, JFK)
	Unload(C1, P1, JFK)
	Load(C2, P2, JFK)
	Fly(P2, JFK, SFO)
	Unload(C2, P2, SFO)

A* Search with h_pg_levelsum	Load(C1, P1, SFO)
	Fly(P1, SFO, JFK)
	Load(C2, P2, JFK)

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

Problem 2:

Search	An Optimal Sequence of actions

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

Uniform Cost Search	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(C3, P3, SFO)
	Unload(C2, P2, SFO)
	Unload(C1, P1, JFK)

Greedy Best First Graph Search with h_1	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(C3, P3, SFO)
	Unload(C2, P2, SFO)
	Unload(C1, P1, JFK)

A* Search with h_1	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(C3, P3, SFO)
	Unload(C2, P2, SFO)
	Unload(C1, P1, JFK)

A* Search with h_ignore_preconditions	Load(C3, P3, ATL)
	Fly(P3, ATL, SFO)
	Unload(C3, P3, SFO)
	Load(C2, P2, JFK)

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

A* Search with h_pg_levelsum	Load(C1, P1, SFO)
	Fly(P1, SFO, JFK)
	Load(C2, P2, JFK)
	Fly(P2, JFK, SFO)
	Load(C3, P3, ATL)
	Fly(P3, ATL, SFO)
	Unload(C3, P3, SFO)
	Unload(C2, P2, SFO)
	Unload(C1, P1, JFK)

Problem 3:

Search	An Optimal Sequence of actions

Breadth First Search	Load(C1, P1, SFO)
	Load(C2, P2, JFK)
	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)

Uniform Cost Search	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)
	Fly(P2, ORD, SFO)
	Unload(C4, P2, SFO)
	Unload(C3, P1, JFK)
	Unload(C2, P2, SFO)
	Unload(C1, P1, JFK)

A* Search with h_1	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)
	Fly(P2, ORD, SFO)
	Unload(C4, P2, SFO)
	Unload(C3, P1, JFK)
	Unload(C2, P2, SFO)
	Unload(C1, P1, JFK)

A* Search with h_ignore_preconditions	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)