Udacity Artificial Intelligence Nanodegree

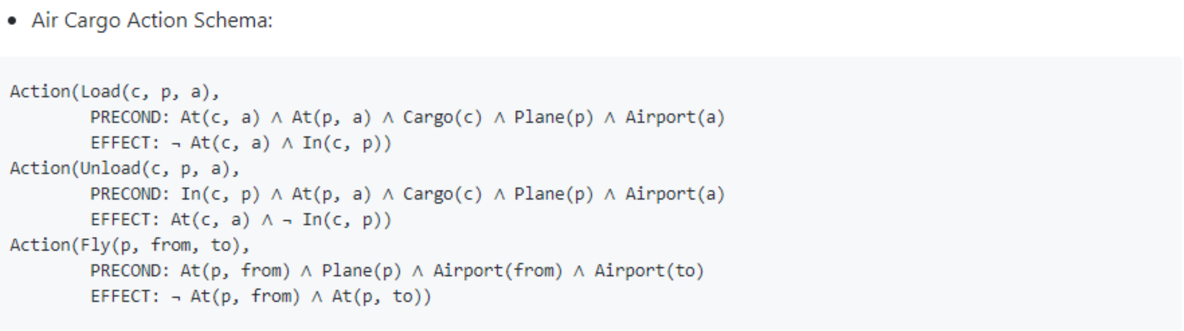
Project 3 – Planning Search

Submission By: Lim Si Jie

As part of the Planning Search project, our task is to solve three deterministic logistics planning problems for an Air Cargo transport system using a planning search agent. In this task, domain-independent heuristics will be adopted.

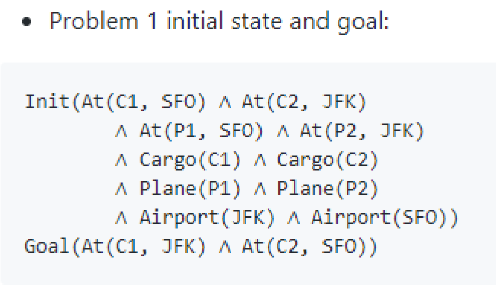
Background Of Planning Search Problems

Three classical Planning Domain Definition Language (PDDL) problems were given to us. Each problem has the same action schema defined, but with different initial states and goals.



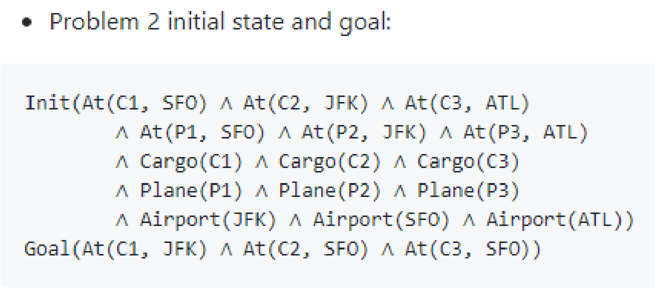
- Air Cargo Action Schema:

```
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))
```



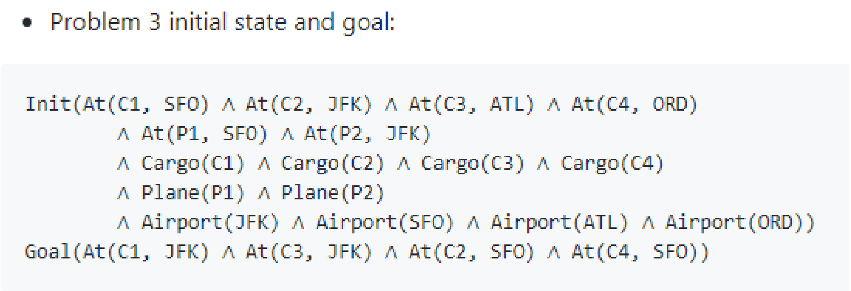
- Problem 1 initial state and goal:

```
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))
```

Criteria For Evaluating Performance Of Planning Search Strategies

There are two types of planning search strategies that were used in this project:

1. **Uninformed non-heuristics search strategy** and
2. **Automatic domain-independent heuristics with A* search**

There were three criteria used to objectively evaluate the performance of each planning search strategy.

The three criteria are:

1. **Solution optimality** – whether the search strategy is able to find the shortest number of actions to achieve the goal.
2. **Speed/Execution time** – Time taken to find the solution
3. **Memory usage** – Number of node expansions during search for the solution

Problem 1

For problem 1, the optimal plan length is 6. One such optimal solution is: Load(C1, P1, SFO), Load(C2, P2, JFK), Fly(P1, SFO, JFK), Fly(P2, JFK, SFO), Unload(C1, P1, JFK), Unload(C2, P2, SFO).

Among the search strategies, depth first graph strategy had the shortest execution time. However, this strategy was unable to obtain the optimal solution. It returned the best solution of 12 actions instead of the optimal solution with only 6 actions. The Greedy Best First Graph Search offered a much better search strategy with the **least time taken** as well as **least memory usage**.

Interestingly, Greedy Best First Graph Search also performed better than A* search strategies (with heuristics) with **lesser time taken** and **lower memory usage**.

	Search Strategy	**Expansions**	**Goal Tests**	**New Nodes**	**Plan Length**	**Optimal?**	**Time Taken (Seconds)**
Uninformed	Breadth First Search	43	56	180	6	Yes	0.23233023
	Breadth First Search Tree Search	1458	1459	5960	6	Yes	3.57346881
	Depth First Graph Search	12	13	48	12	No	0.016257987
	Depth Limited Search	101	271	414	50	No	0.254626694
	Uniform Cost Search	55	57	224	6	Yes	0.089883068
	Recursive Best First Search	4229	4230	17023	6	Yes	9.563046281
	Greedy Best First Graph Search	7	9	28	6	Yes	0.018867404
Heuristics	A* Search with h1 heuristic	55	57	224	6	Yes	0.203518015
	A* Search with Ignore Precondition heuristic	41	43	170	6	Yes	0.25654647
	A* Search with Level Sum heuristic	11	13	50	6	Yes	6.063609035

Problem 2

Problem 2 offered a slightly more complicated problem for the search strategies. For problem 2, the optimal plan takes 9 actions for completion. One such optimal solution is: 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).

Unlike problem 1 where all search strategies could come up with a solution, only some search strategies could solve problem 2. Breadth First Tree Search, Depth Limited Search and Recursive Best First Search were unable to find a satisfactory solution within 10 minutes.

Among the search strategies that could offer an optimal solution, A* Search with Ignore Precondition heuristic took the **least time**. However, in terms of **memory usage**, A* Search with Level Sum heuristic would have been the preferred search strategy.

	Search Strategy	**Expansions**	**Goal Tests**	**New Nodes**	**Plan Length**	**Optimal?**	**Time Taken (Seconds)**
Uninformed	Breadth First Search	3343	4609	30509	9	Yes	57.93787932
	Breadth First Search Tree Search	-	-	-	-	-	-
	Depth First Graph Search	476	477	4253	466	No	9.13423469
	Depth Limited Search	-	-	-	-	-	-
	Uniform Cost Search	4852	4854	44030	9	Yes	172.0550782
	Recursive Best First Search	-	-	-	-	-	-
	Greedy Best First Graph Search	990	992	8910	21	No	12.77882513
Heuristics	A* Search with h1 heuristic	4852	4854	44030	9	Yes	146.2963658
	A* Search with Ignore Precondition heuristic	1506	1508	13820	9	Yes	52.51326836
	A* Search with Level Sum heuristic	86	88	841	9	Yes	496.3377861

Problem 3

Problem 3 had a more complex problem than problem 2. For problem 3, the optimal plan takes 12 actions for completion. (Note: As the problem becomes more complex, the **time taken** and **memory usage** increases exponentially, despite the optimal plan taking just 3 more actions than the previous problem.) One optimal solution is: 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).

Similar to problem 2, some search strategies couldn't find a satisfactory solution within 10 minutes. These search strategies include: Breadth First Search Tree Search, Depth Limited Search, Recursive Best First Search and A* Search with Level Sum heuristic.

Among the search strategies that could offer an optimal solution, A* Search with h1 heuristic took the **least time**. However, in terms of **memory usage**, A* Search with Ignore Precondition heuristic would have been the preferred search strategy. Overall, A* Search with Ignore Precondition heuristic should be the preferred search strategy given that it only took around 15 seconds more than A* Search with h1 heuristic.

	Search Strategy	**Expansions**	**Goal Tests**	**New Nodes**	**Plan Length**	**Optimal?**	**Time Taken (Seconds)**
Uninformed	Breadth First Search	14663	18098	129631	12	Yes	373.5412284
	Breadth First Search Tree Search	-	-	-	-	-	-
	Depth First Graph Search	1511	1512	12611	1442	No	49.59636543
	Depth Limited Search	-	-	-	-	-	-
	Uniform Cost Search	18234	18236	159707	12	Yes	1655.25834
	Recursive Best First Search	-	-	-	-	-	-
	Greedy Best First Graph Search	5605	5607	49360	22	No	329.3902482
Heuristics	A* Search with h1 heuristic	18234	18236	159707	12	Yes	266.6937077
	A* Search with Ignore Precondition heuristic	5118	5120	45650	12	Yes	281.8772537
	A* Search with Level Sum heuristic	-	-	-	-	-	-