Part 1: Heuristic Analysis

ToDo Introduction

Air cargo 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

```
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))
```

The 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)
```

	Plan length	Time elapsed	Expansions	Goal Tests	New Nodes
breath first	6	0,0267	43	56	180
depth first	20	0,0127	21	22	84
uniform cost	6	0,0343	55	57	224
greedy best first h_1	6	0,0047	7	9	28
A* h_1	6	0,0409	55	57	224
A* ignore pre	6	0,0262	41	43	170
A* levelsum	6	0,9721	39	41	158

Despite that, depth first was the fastest among the non heuristic searches, it did not find the optimal solution. Uniform cost and breath first came up with the correct solution, where the latter performed slightly better.

Among the heuristic searches there is the greedy best first that crushed all the others by far. It came up with the correct solution in a fraction of the time. All the A* searches did perform equaly well and in the range o the non heuristic searches.

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))
```

The 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(C1, P1, JFK)
Unload(C2, P2, SFO)
Unload(C3, P3, SFO)
```

	Plan length	Time elapsed	Expansions	Goal Tests	New Nodes
breath first	9	14,72	3343	4609	30509
depth first	619	3,58	624	625	5602
uniform cost	9	11,91	4780	4782	43381
greedy best first h_1	21	1,57	598	600	5382
A* h_1	9	12,98	55	57	224
A* ignore pre	9	4,16	1450	1452	13303
A* levelsum	9	347,84	1129	1131	10232

Depth first was again the fastest method, but it came up with an incredibly long plan of 619 steps. Among the other non heuristic uniform cost seems to work better with complex plans, however with the cost of having more expanded nodes.

Regarding the heuristic the greedy algorithm was still the fastest, but as depth first it did not find the optimal plan. Among the A* the ignore precondition was the fastest again with the cost of an increased number of expansions.

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))
```

The optimal solution is

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

	Plan length	Time elapsed	Expansions	Goal Tests	New Nodes
breath first	12	126,45	14663	18098	129631
depth first	392	2,09	408	409	3364
uniform cost	12	65,13	17882	17884	156769
greedy best first h_1	26	17,62	4498	4500	39970
A* h_1	12	80,88	17882	17884	156769
A* ignore pre	12	20,11	5034	5036	44886
A* levelsum	12	1537,87	2025	2027	17924

This sample just improved the confidence in the observations made in the previous sample. Depth first is again really fast, but the solution is not optimal. uniform cost again performed a bit better than breath first, with the cost of more expansions made. Among the heuristic functions the greedy algorithm was again the fastest, but it came up with a solution that has about double the steps of the best one. And finally as before the ignore precondition was almost as fast as the greedy algorithm.

Summarising

The non-heuristic based algorithms could only perform well for small problem sets. The bigger they became the less well those algorithms performed. The ignore precondition algorithm on A* was able to perform best in all problems, because the algorithm can use most of its time for search without having to consider preconditions at all.