Heuristic Analysis for AIND's Planning Search Project

This is an analysis for the Planning Search Project where 3 different cargo problems are tested by uninformed and heuristic search algorithms. Here are the problems' initial state and goal.

Problem 1: Get Cargo, C1, to JFK and Cargo, C2 to SFO

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

Goal(At(C1, JFK) ^ At(C2, SFO))
```

Problem 2: Get C1 to JFK, C2 and C3 to SFO

```
Init(Cargo(C1) ^ Cargo(C2) ^ Cargo(C3)
     ^ Plane(P1) ^ Plane(P2) ^ Plane(P3)
     ^ Airport(SFO) ^ Airport(JFK) ^ Airport(ATL)
     ^ At(C1, SFO) ^ At(C2, JFK) ^ At(C3, ATL)
     ^ At(P1, SFO) ^ At(P2, JFK) ^ At(P3, ATL))

Goal(At(C1, JFK) ^ At(C2, SFO) ^ At(C3, SFO))
```

Problem 3: Get C1 and C3 to JFK, get C2 to SFO and get C4 to SFO

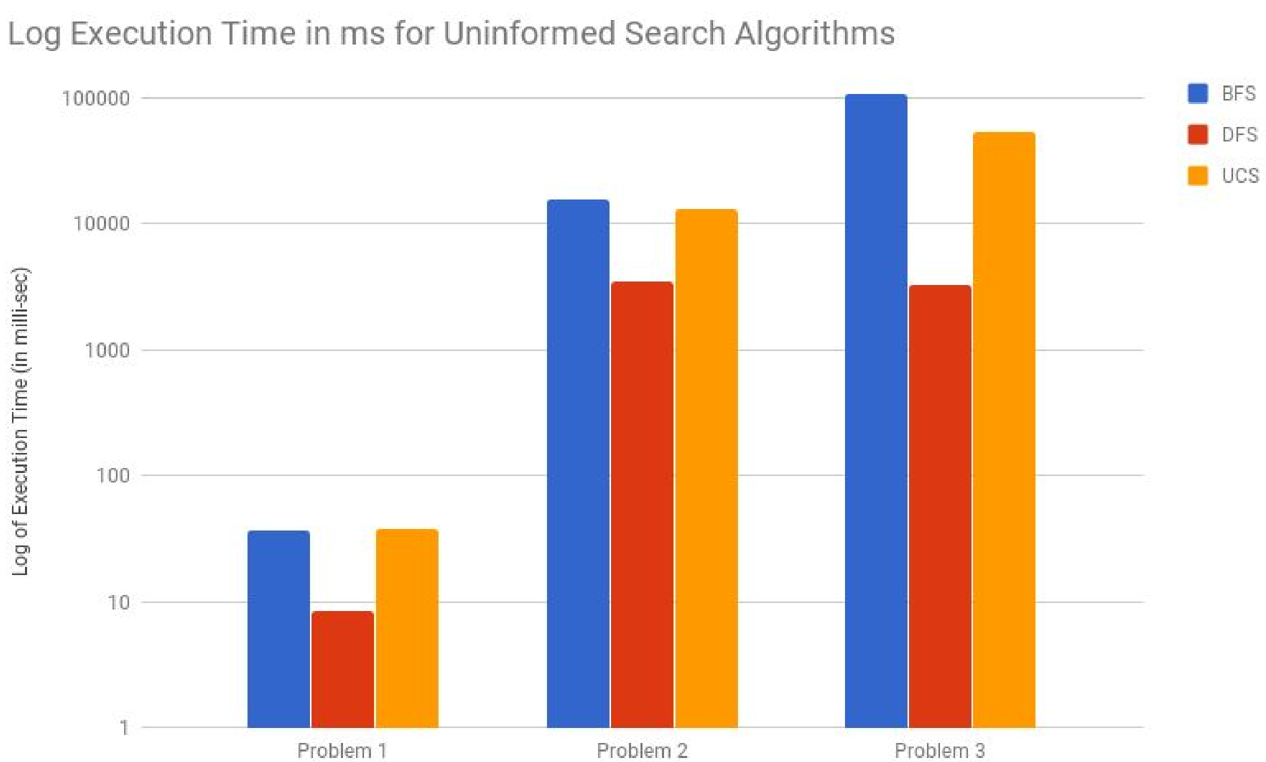
```
Init(Cargo(C1) ^ Cargo(C2) ^ Cargo(C3) ^ Cargo(C4)
     ^ Plane(P1) ^ Plane(P2)
     ^ Airport(SFO) ^ Airport(JFK) ^ Airport(ATL) ^ Airport(ORD)
     ^ At(C1, SFO) ^ At(C2, JFK) ^ At(C3, ATL) ^ At(C4, ORD)
     ^ At(P1, SFO) ^ At(P2, JFK))

Goal(At(C1, JFK) ^ At(C3, JFK) ^ At(C2, SFO) ^ AT(C4, SFO))
```

1. Compare and contrast non-heuristic search result metrics

The is the summary of the uninformed non-heuristic planning searches which include Breadth First Search (BFS), Depth First Search (DFS) and Uniform Cost Search (UCS).

Problem	Search Type	Nodes Expansion	Goals Tested	Plan Length	Time Elapsed (seconds)	Optimal
Problem 1	BFS	43	56	6	0.0364	Y
Problem 1	DFS	12	13	12	0.0084	N
Problem 1	UCS	55	57	6	0.0376	Y
Problem 2	BFS	3343	4609	9	15.33	Y
Problem 2	DFS	582	583	575	3.40	N
Problem 2	UCS	4852	4854	9	12.92	Y
Problem 3	BFS	14663	18098	12	107.77	Y
Problem 3	DFS	627	628	596	3.26	N
Problem 3	UCS	18235	18237	12	53.29	Y

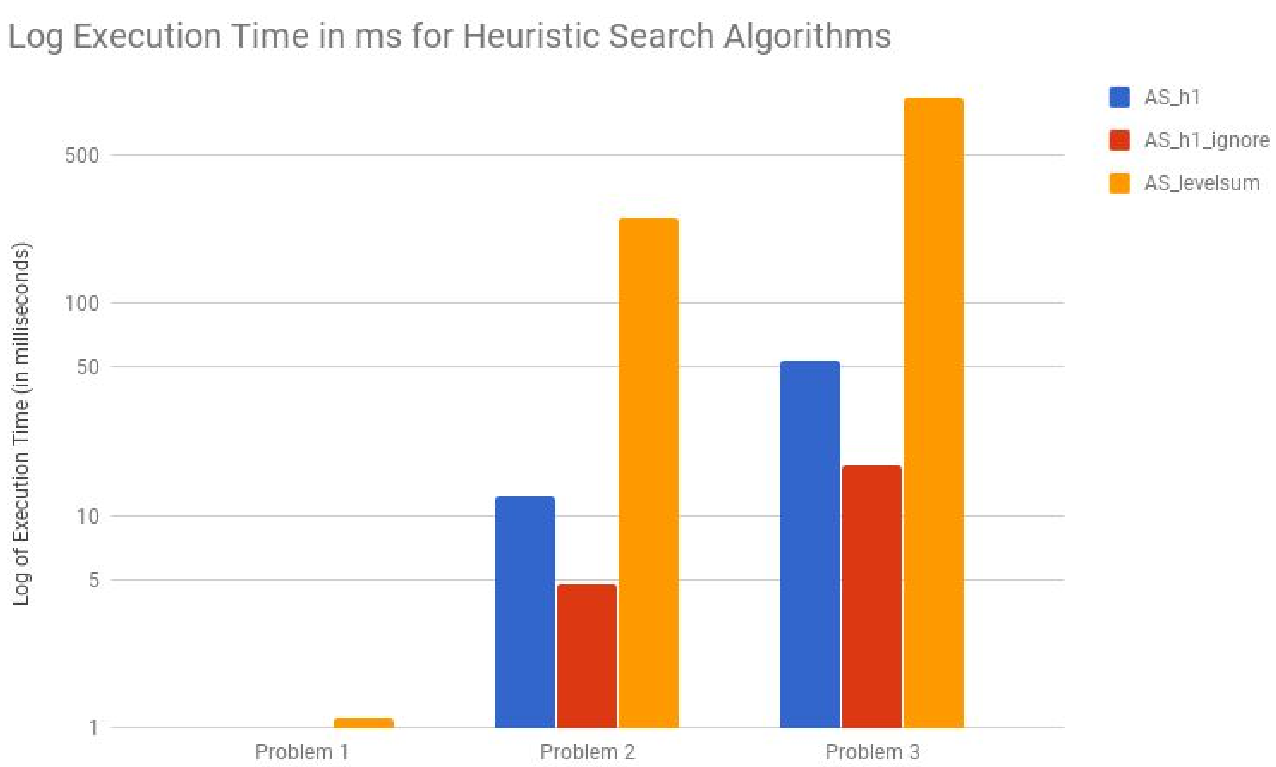


In the uninformed search algorithms, Breadth First Search and Uniform Cost Search have the shortest plan length for all three problems.

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

The is the summary of the heuristic planning searches which A* Search with h1 (ASH1), h_ignore_preconditions (ASI) and level_sum (ASLS).

Problem	Search Type	Nodes Expansion	Goals Tested	Plan Length	Time Elapsed (seconds)	Optimal
Problem 1	ASH1	55	57	6	0.0445	Y
Problem 1	ASI	41	43	6	0.0394	Y
Problem 1	ASLS	11	13	6	1.1807	Y
Problem 2	ASH1	4852	4854	9	12.2557	N
Problem 2	ASI	1450	1452	9	4.7383	Y
Problem 2	ASLS	86	88	9	250.955	N
Problem 3	ASH1	18235	18237	12	53.2215	N
Problem 3	ASI	5040	5042	12	17.1656	Y
Problem 3	ASLS	325	327	12	927.75	N



Broadly speaking, the Depth First Search algorithm perform very well for all three problems in terms of execution time. However, if you examined the nodes expanded by Depth First Search for problem 3, the plan length is 596. For a real life cargo flying problem, each action would incur an execution cost for labor and fuel. My thought is to find an optimal plan by choosing the lowest plan length and then comparing execution time if plan lengths are the same between two solutions.

In the uninformed category, Breadth First Search and Uniform Cost Search tie for the optimal solution because both returned the same plan length and their execution time is within the same magnitude in duration of each other.

In the heuristics search planning, for a simple problem such as problem 1, there are no differences in plan length or execution time between h_1, h_ignore_preconditions and h_pg_levelsum. In fact, all the variations of the A* Search return the same plan length. So, in a real life cargo flying problem, it will be the same to execute the resulting plan with incurring the same labor and fuel costs.

However, for more complex problems such as problem 2 and 3, A* Search with Ignore Preconditions performs the best in execution time and thus is the optimal solution. This is due to the heuristics obtained by relaxed problem as defined in Russell and Norvig's "Artificial Intelligence, A Modern Approach" 3rd edition book, which makes it easier to solve.

Optimal Solution

Problem 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)`**
Problem 2 **`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)`**

Problem 3

```
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)
```

In summary, since you will not know whether the problem is simple or complex in advance, it is best to choose A* Search with Ignore Preconditions to get the shortest plan length which also has a quick execution time.