Air Cargo Problem Heuristics Analysis

By: Anamika Sharma, Udacity AI Nanodegree, Term 1

Introduction

We have implemented a planning search agent to solve deterministic logistics planning problems for an Air Cargo transport system. We will be experimenting with various automatically generated heuristics, including planning graph heuristics, to solve the set of problems, and then provide an analysis of the results.

Things to be implemented :

1. Defining problems is classical PDDL(Planning Domain Definition Language). They have the same action schema defined, but different initial states and goals.
2. Implement relaxed problem heuristic in `my_air_cargo_problems.py`.
3. Implement Planning Graph and automatic heuristic in `my_planning_graph.py`.

Problems	Initial State	Goals
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))
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))
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))

We have compared each problem with different search algorithms and it provided the optimal plan lengths alongwith time elapsed.

Results

Searches	Expansions	Goal Test	New Nodes	Length	Time in secs	Optimal
Breadth_first_search	43	56	180	6	0.0543	Yes
Breadth_first_tree_search	1458	1459	5960	6	1.0943	Yes
Depth_first_graph_search	21	22	84	20	0.0161	No
Depth_limited_search	101	271	415	50	0.0964	No
Uniform_cost_search	55	57	224	6	0.0338	Yes
Recursive_best_first_search with h_1	4229	4230	17023	6	2.9125	Yes
Greedy_best_first_graph_search with h_1	7	9	28	6	0.0052	Yes
Astar_search with h_1	55	57	224	6	0.0411	Yes
Astar_search with h_ignore_preconditions	41	43	170	6	0.0510	Yes
Astar_search with h_pg_levelsum	11	13	50	6	1.4796	Yes

Air cargo problem 1 Results

Searches	Expansions	Goal Test	New Nodes	Length	Time in secs	Optimal
Breadth_first_search	3343	4609	30509	9	14.621	Yes
Breadth_first_tree_search	-	-	-	-	-	
Depth_first_graph_search	624	625	5602	619	3.655	No
Depth_limited_search	-	-	-	-	-	
Uniform_cost_search	4853	4855	44041	9	12.030	Yes
Recursive_best_first_search with h_1	-	-	-	-	-	
Greedy_best_first_graph_search with h_1	998	1000	8982	15	2.438	No
Astar_search with h_1	4853	4855	44041	9	12.880	Yes
Astar_search with h_ignore_preconditions	1450	1452	13303	9	6.0262	Yes
Astar_search with h_pg_levelsum	86	88	841	9	327.03	Yes

Air cargo problem 2 Results

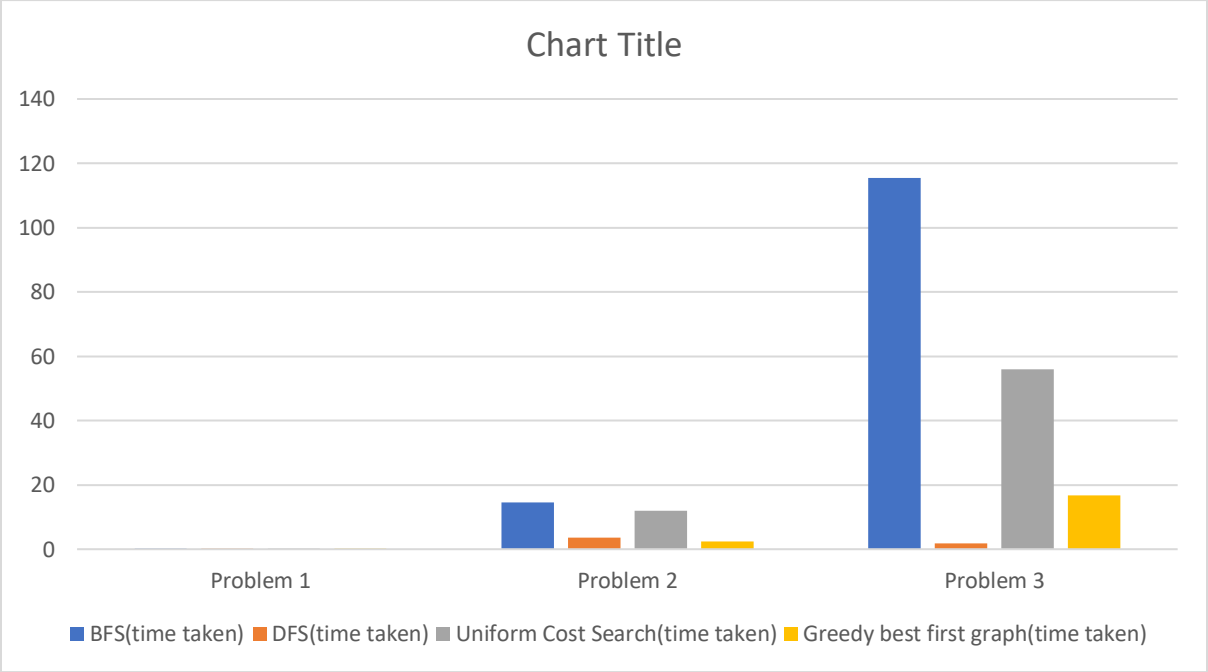
Searches	Expansions	Goal Test	New Nodes	Length	Time in secs	Optimal
Breadth_first_search	14663	18098	129631	12	115.441	Yes
Breadth_first_tree_search	-	-	-	-	-	
Depth_first_graph_search	408	409	3364	392	1.8957	No
Depth_limited_search	-	-	-	-	-	
Uniform_cost_search	18223	18225	159618	12	56.000	Yes
Recursive_best_first_search with h_1	-	-	-	-	-	
Greedy_best_first_graph_search with h_1	5578	5580	49150	22	16.787	No
Astar_search with h_1	18223	18225	159618	12	56.365	Yes
Astar_search with h_ignore_preconditions	5040	5042	44944	12	22.711	Yes
Astar_search with h_pg_levelsum	-	-	-	-	-	

Air cargo problem 3 Results

Analysis

Non-heuristic search result metrics:

They do not have additional information about states. They can just generate successors and distinguish a goal state from non-goal state. Below is the plot for time taken by BFS, DFGS, Uniform Cost Search and Greedy best first graph against each problem.



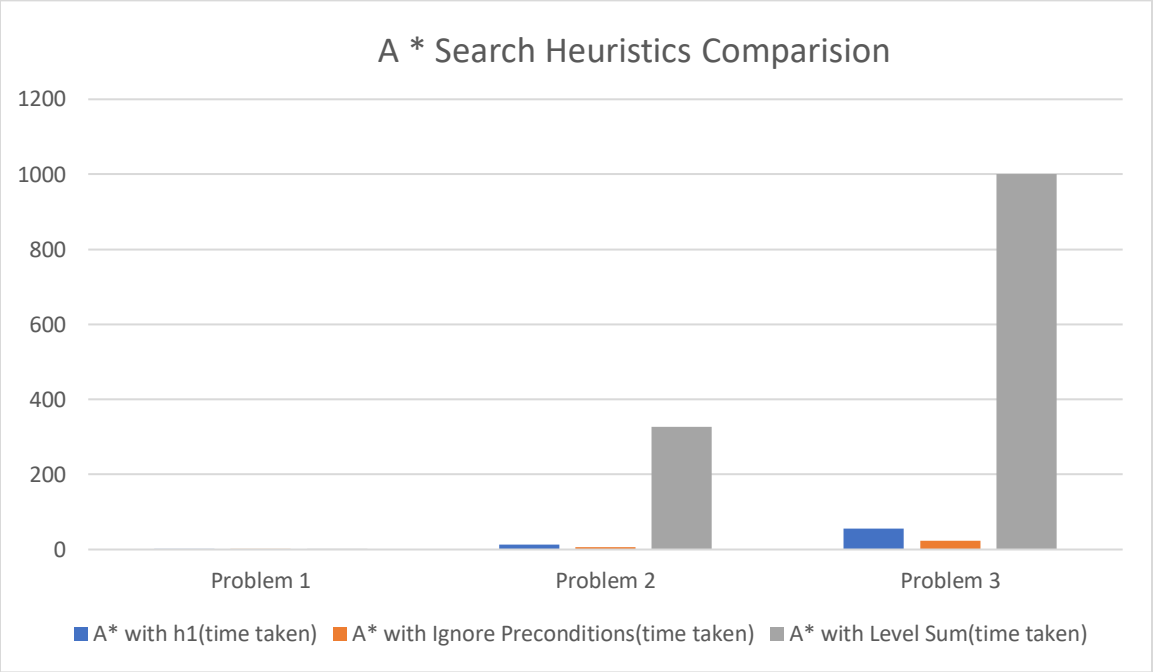
Inferences:

1. From above tables, we can see that **Breadth First Search** and **Uniform Cost Search** results better optimal plan length.
2. Although **Depth First Graph Search** takes less time but it does not provide optimal plan length and it searches as deep as possible in the graph even if goal is to its right.
3. **Greedy_best_first_graph_search with h_1** is the fastest and takes less memory but for problem 2 and 3 it gives optimal plan length as 15 and 22 respectively which is not relevant.

Heuristic search result metrics:

Search that uses problem-specific knowledge beyond the definition of the problem itself. It can find solutions more efficiently.

This is a plot for each problem against time taken by 3 different A* search heuristics.



Inferences:

1. Simple the problem, less is the time taken by A* search algorithms.
2. **A* search with ignore preconditions** have performed well as compared to other two. It took less time and provided optimal plan length.
3. For problem 3, A* search with Level Sum took a long time, so for plotting I have just assumed 1000.

Recommendations

1. If problem is **not complex**, then we can use **Breadth First Search** strategy as it can solve problem fast and optimally. It is not worth to use elaborated approaches like A* search with heuristics for such problems.
2. If problem is **complex**, then we can use **A* search with Ignore Preconditions** heuristics as it has given optimal results in less time.

The optimal plan lengths for problems 1,2, and 3 are 6, 9, and 12 actions, respectively. Below are sample plans with optimal length:

```
Solving Air Cargo Problem 1 using uniform_cost_search...

Expansions   Goal Tests   New Nodes
    55          57          224

Plan length: 6  Time elapsed in seconds: 0.037983710174625644
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P1, SFO, JFK)
Fly(P2, JFK, SFO)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)
```

Optimal Plan Length for Problem 1

```
Solving Air Cargo Problem 2 using uniform_cost_search...

Expansions   Goal Tests   New Nodes
   4853        4855       44041

Plan length: 9  Time elapsed in seconds: 12.030347073596953
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)
```

Optimal Plan Length for Problem 2

```
Solving Air Cargo Problem 3 using uniform_cost_search...

Expansions   Goal Tests   New Nodes
  18223       18225       159618

Plan length: 12  Time elapsed in seconds: 56.000154401697884
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)
```

Optimal Plan Length for Problem 3