Analysis of Search Algorithm Efficiency in the Classical Planning Graph

Daniel Szemerey

Udacity

Author Note

This project is an obligatory delivery for Project II in

Udacity's Artificial Intelligence Nanodegree

Abstract

This report summarizes the analysis done for Project II, Classical Planning of Udacity's Artificial Intelligence Nanodegree. Analysis was done both on uninformed searches: Depth First Search, Breath First Search, Uniform Cost Search and on informed searches: Greedy Best First Search and A* Search; both with four combination of heuristics: setlevel, maxlevel, levelsum, unmet goals. All search strategies were executed on Air Cargo Problem 1-2, and further selected strategies were executed on Air Cargo Problem 3-4.

Keywords: Artificial Intelligence, Classical Planning Problem, A*, Heuristic, Plangraph

Table of Contents

Charting the Data

Deciding on which algorithms to perform full analysis:

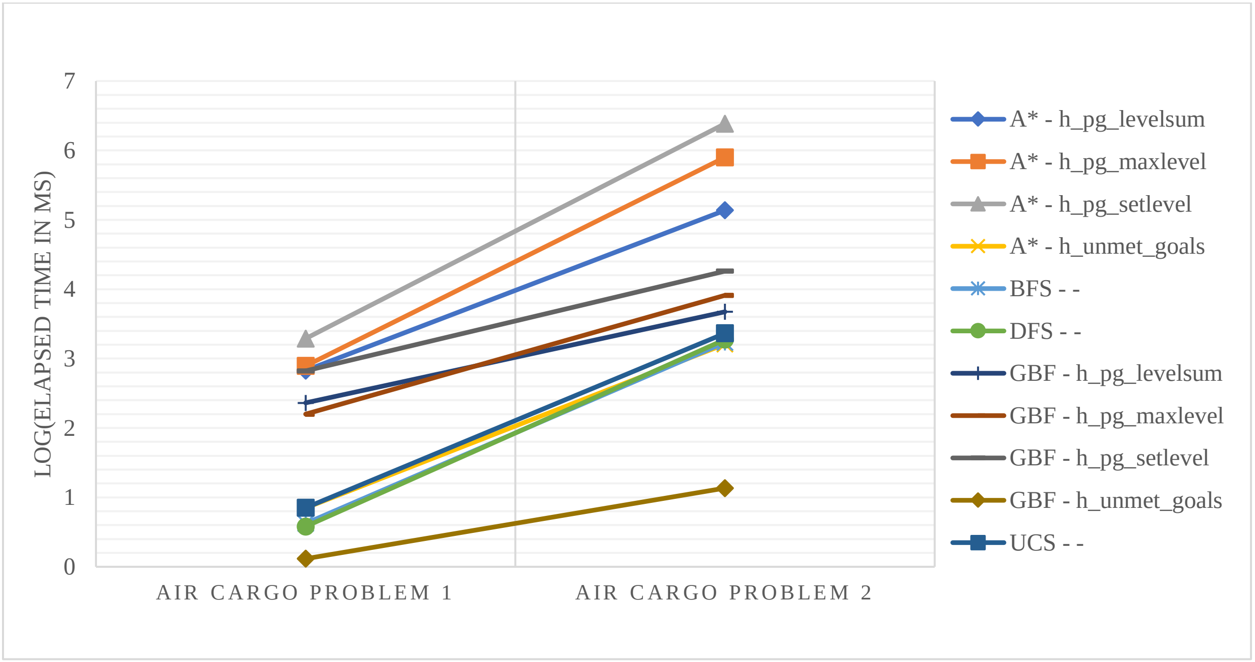

Figure 1 Increase of execution time (ms in log) for each algorithm as complexity of problem increases

Search Strategy	Heuristic	Air Cargo Problem 1 Time Elapsed *(ms in log)*	Air Cargo Problem 2 Time Elapsed *(ms in log)*	Ratio of Increase *(smaller is better)*
GBF	h_pg_setlevel	3.076805539	4.51993388	1.469034628
GBF	h_pg_levelsum	2.531396265	3.939101245	1.556098229
A*	h_pg_levelsum	3.175175786	5.374074233	1.692528097
GBF	h_pg_maxlevel	2.464796754	4.224988853	1.714132756
A*	h_pg_setlevel	3.617011177	6.385116761	1.765301916
A*	h_pg_maxlevel	3.137100393	6.146609808	1.959328373
A*	h_unmet_goals	0.863441829	3.177491077	3.680029125
UCS	-	0.819767605	3.379936699	4.123042527
BFS	-	0.625158053	3.31034697	5.295216069
DFS	-	0.371030899	3.312653592	8.928241816
GBF	h_unmet_goals	0.036708722	1.133944274	30.8903232

Figure 2 Sorted table according to relative increase in elapsed times. Green indicates the best uninformed search strategy; whose time increases least with the size of the problem

As Figure 2 shows, out of the uninformed search, the Uniform Cost Search algorithm increases the least in terms of time to find solution as the complexity of problem increases. Therefore, I executed **Uniform Cost Search** for problem 3 and 4, but not Depth First Search or Breath First Search; along with the informed search strategies' following heuristics:

- Greedy Best First: LevelSum, SetLevel
- A*: Unmet Goals, LevelSum

Charting full analysis:

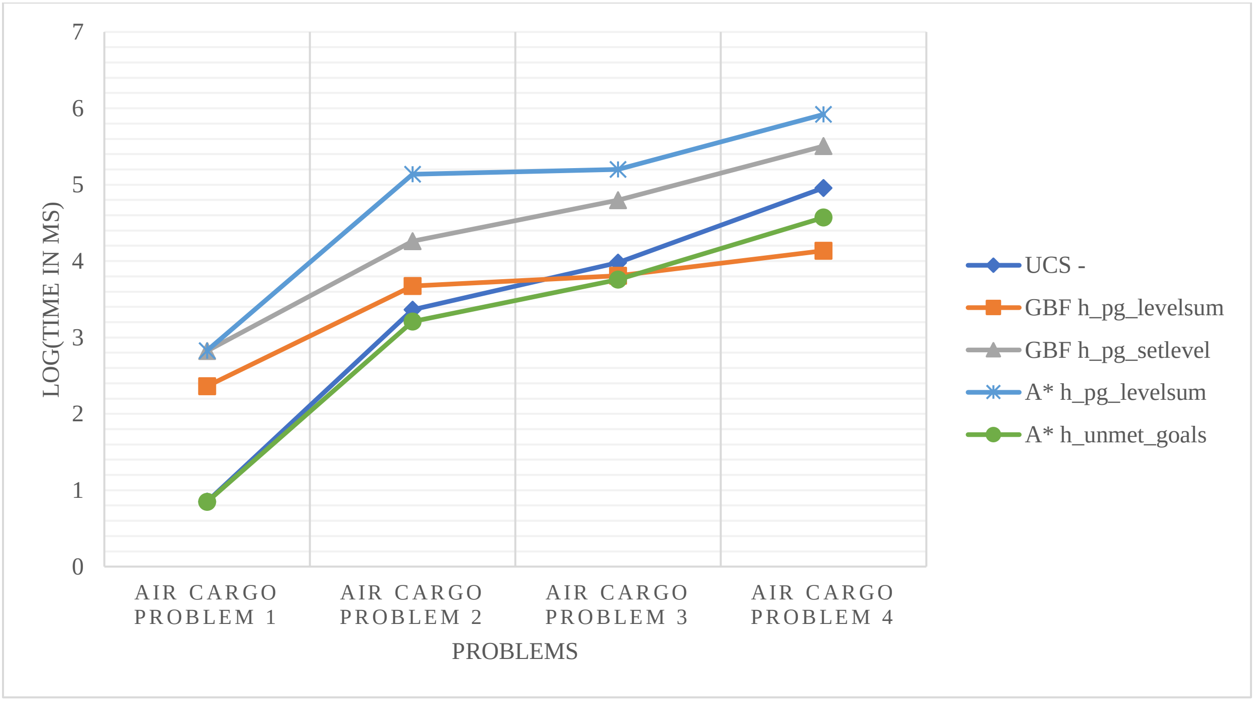

Figure 3 Log(Elapsed time in ms) for selected search strategy - problem combination

Average (taking multiple measurements)
of Time Elapsed *(LOG in ms)*

		Air Cargo Problem 1	Air Cargo Problem 2	Air Cargo Problem 3	Air Cargo Problem 4
Search Strategy	**Heuristic**	**# of Actions: 22**	**72**	**88**	**104**
UCS	-	0.850353448	3.365092814	3.978762634	4.958484921
GBF	h_pg_levelsum	2.362314573	3.674983929	3.806212524	4.135663264
	h_pg_setlevel	2.823392173	4.260397646	4.798214534	5.505344441
A*	h_pg_levelsum	2.828624237	5.137072541	5.198165439	5.920578658
	h_unmet_goals	0.847457432	3.2092329	3.757364002	4.570192398

“Use a table or chart to analyze the number of nodes expanded against number of actions in the domain”

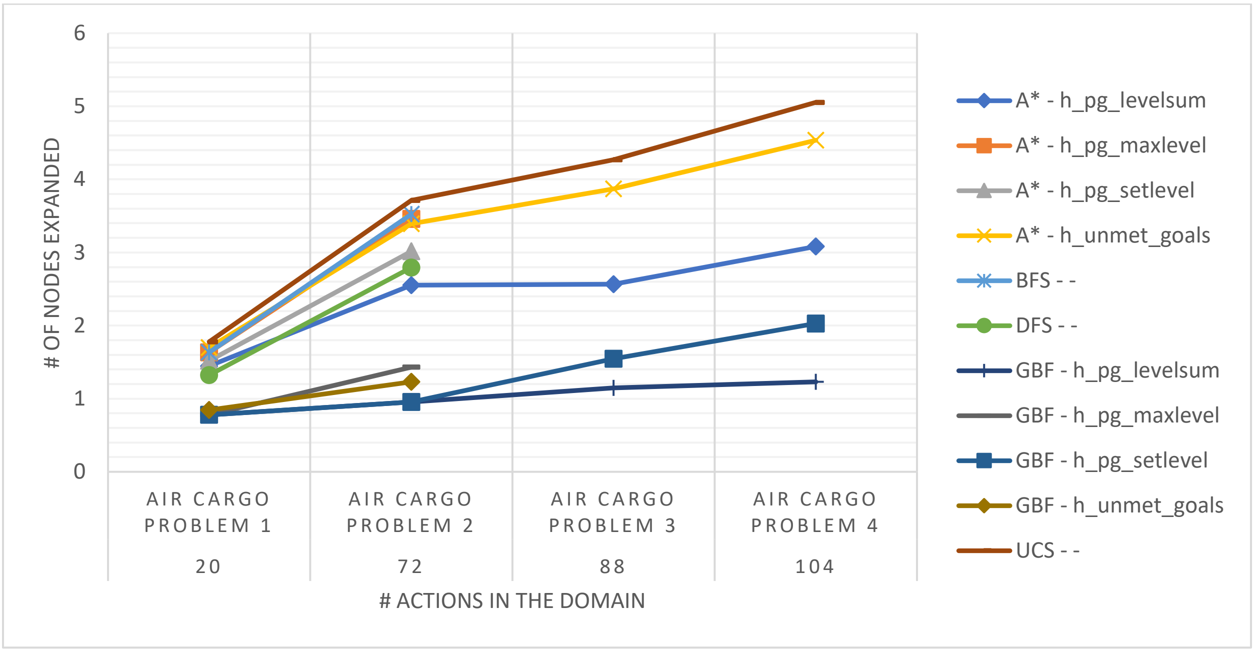

Figure 4 Number of possible actions in the domain in relationship to how many nodes each search strategy expanded. Only selected algorithms were analysed with Air Cargo Problem 3-4

Uniform Cost Search performs badly in terms of how many nodes it visited. This means that the computer needs more memory to store values; this might have been a bottleneck in past research.

"Use a table or chart to analyze the search time against the number of actions in the domain."

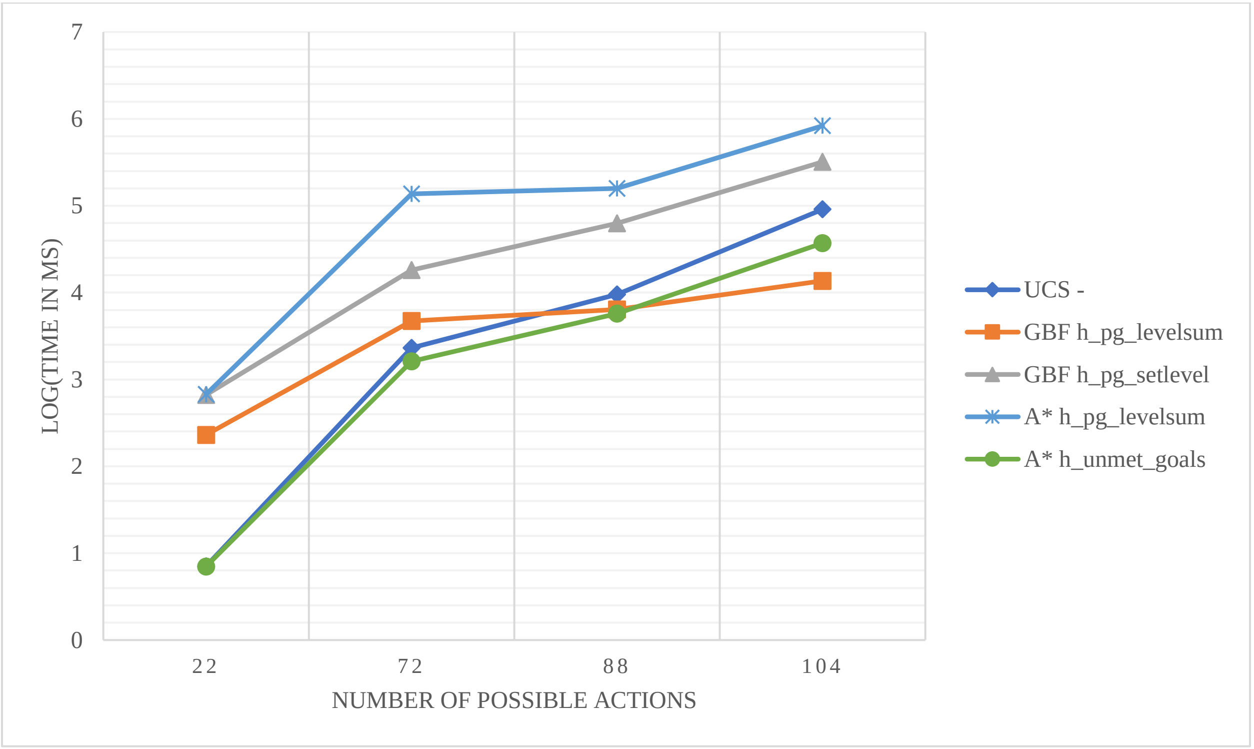

Figure 5 Elapsed time (log in ms) in relation to number of actions for each search strategies and heuristics. (X- axis is not scaled to actual values!)

Average of Time Elapsed *(LOG in ms)*

Search Strategy	Heuristic	Air Cargo Problem 1	Air Cargo Problem 2	Air Cargo Problem 3	Air Cargo Problem 4
UCS	-	0.850353448	3.365092814	3.978762634	4.958484921
GBF	h_pg_levelsum	2.362314573	3.674983929	3.806212524	4.135663264
	h_pg_setlevel	2.823392173	4.260397646	4.798214534	5.505344441
A*	h_pg_levelsum	2.828624237	5.137072541	5.198165439	5.920578658
	h_unmet_goals	0.847457432	3.2092329	3.757364002	4.570192398

Greedy Best Search with levelsum heuristic shows promising efficiency as complexity increases.

“Use a table or chart to analyze the length of the plans returned by each algorithm on all search problems”

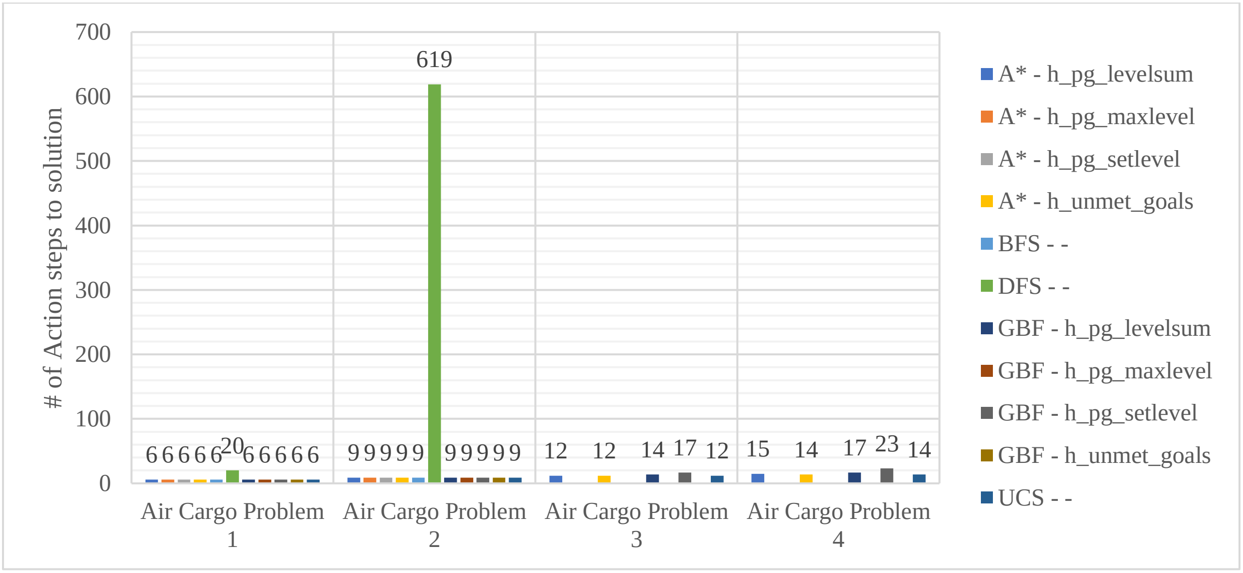

Figure 6 Depth First Search can lead to extreme solutions. Number of Action steps for each search for each problem.

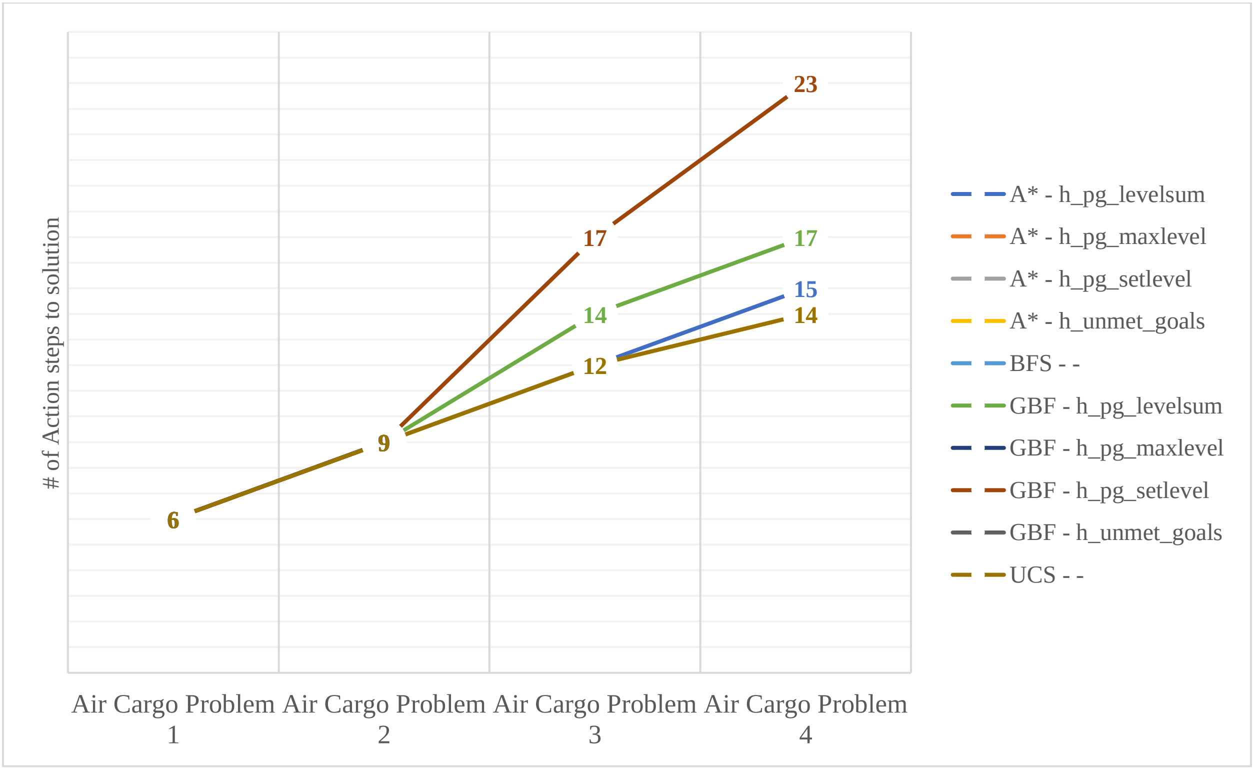

Figure 7 Number of Action steps for each algorithm for each problem (Depth First Search omitted)

Evaluating the Data

“Which algorithm or algorithms would be most appropriate for planning in a very restricted domain (i.e., one that has only a few actions) and needs to operate in real time?”

I would choose Greedy Best First Search with an unmet need heuristic, as this performed the best on both Air Cargo Problem 1 and Air Cargo Problem 2 (which were restricted problems) – it had both the lowest execution time, while keeping the action step count low.

If the problem is very small (Air Cargo Problem 1), then alternatively one could choose Breath First Search and Uniform Cost Search. Depth First Search is unsuitable as it didn’t find the optimal number of steps, therefore time sensitive activities would suffer.

“Which algorithm or algorithms would be most appropriate for planning in very large domains (e.g., planning delivery routes for all UPS drivers in the U.S. on a given day)”

For large domains informed search outperforms uninformed search in most cases. Given that Greedy Best First with either levelsum or unmet goals heuristic both performed well in terms of number of action needed and time need for each problem; as well as it’s ‘flat’ increase in resources as complexity grows, it seems the best suited for large domains.

“Which algorithm or algorithms would be most appropriate for planning problems where it is important to find only optimal plans?”

My analysis showed that the following algorithms find the optimal plans:

- A* with maxlevel, setlevel and unmet goals heuristics
- Greedy Best First search with maxlevel and unmet goals heuristics
- Breath First Search
- Uniform Cost Search

Depth First Search is not optimal and can give very skewed results (like in Air Cargo Problem 2 Figure 6).

Setlevel algorithm and LevelSum seem like they are not admissible heuristics, as the algorithms don't find the optimal solutions with them. Both A* and Greedy Best First search has found an optimal solution with UnmetGoals and MaxLevel heursitics.

I would therefore choose A*, Greedy Best First with either unmet goals or maxlevel heuristics or Breath First Search or Uniform Cost Search from the uninformed search strategies.

References

AIMA Pseudocode, https://github.com/aimacode/aima-seudocode/blob/master/md/GraphPlan.md, accessed: 07/01/2021

Artificial Intelligence a Modern Approach (AIMA) 4th edition, Stuart Russell and Peter Norvig, 2020

Dataset

Entire Project Folder with code can be found here (folder): ai-udacity/Classical Planning at main · szemyd/ai-udacity (github.com)

Raw Dataset of tests can be found here (csv): ai-udacity/search_diagnostic.csv at main · szemyd/ai-udacity (github.com)