Heuristic Analysis

Non-Heuristic Planning Solution Searches

& Metrics of A* Searches

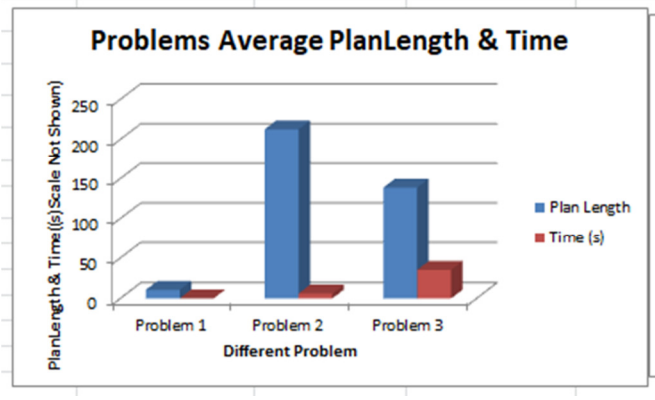
by Andrew Quaife

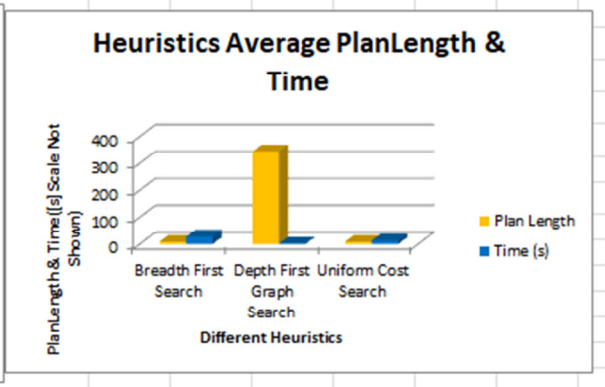
NOTE on Graphs: I could not get the Y Scale to be relative for each Series specifically, but uses on of the Metric Series to scale.

Non-Heuristic Planning Solution Searches:

Average Across Problems			
	Problem 1	**Problem 2**	**Problem 3**
Plan Length	11	212	139
Time (s)	0.0159	5.8452	35.1184
Expansions	40	2940	11102
Goal Tests	45	3363	12248
New Nodes	163	26714	97570

Average Across Heuristics			
	Breadth First Search	**Depth First Graph Search**	**Uniform Cost Search**
Plan Length	9	344	9
Time (s)	27.2539	1.1570	12.5686
Expansions	6016	351	7714
Goal Tests	7588	352	7716
New Nodes	53440	3017	67990





The Non-Heuristic Planning Searches showed some interesting results, and averages are used to show how well each heuristic performs. Additionally, averaging each metric over the three problems for each heuristic allows one to easily discern between them with little loss from decomposition of averaging (See Table Above). To begin, using average time across heuristics to gauge Problem difficulty, Problem1 was the easiest, then Problem2 was almost 370 times in difficulty, and Problem3 was 2,200 times more difficult than Problem1.

Using an Average for Metrics of Problem 1, 2 & 3 for each Heuristic:

Breath First Search had the minimum average Plan Length of 9 consistently, but the worst time of 27.2539(s), and 53,440 New Nodes all on average. Depth First Graph Search had an average Plan Length of 344 which is very clearly worse than the others, but had the quickest time of 1.1570(s), and the minimum 3017 New Nodes (Average). Uniform Cost Search had the minimum average Plan Length of 9, a time of 12.5686(s), and the maximum New Nodes of 67,990 (Average).

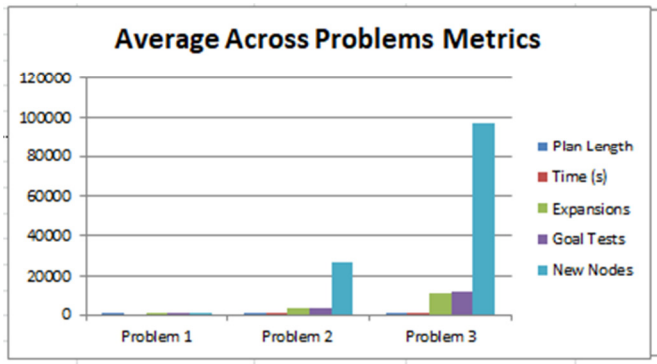
Resolution:

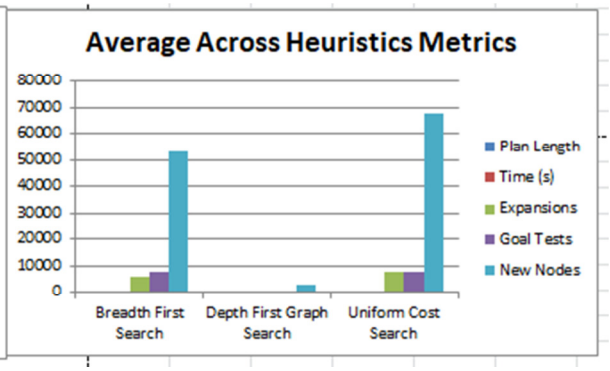
This practically makes Uniform Cost Search the most efficient algorithm of them all since it got the correct answer consistently, and searched the max number of nodes in the least amount of time 5,410 Nodes/sec., compared to Breadth First Search 1,961 Nodes/sec. Note I am removing Depth First searches time since it was not the smartest algorithm since it searched only 5% of the nodes compared to the others, and the solutions where by far the worst. Breath First Search is second to Uniform Search although it had the worst time, but it searched a good amount of nodes (a little less than Uniform Search), and came to the proper solution on each problem like Uniform Search. Last, of course is Depth First Search that continually had the worst solution, even though it had the best time.

Breadth First Search				
	Problem 1	Problem 2	Problem 3	Average
Plan Length	6	9	12	9
Time (s)	0.01864	8.4878	73.2552	27.25388
Expansions	43	3343	14663	6016.3333
Goal Tests	56	4609	18098	7587.6667
New Nodes	180	30509	129631	53440

Depth First Graph Search				
	Problem 1	Problem 2	Problem 3	Average
Plan Length	20	619	392	343.66667
Time (s)	0.0091	2.3436	1.1184	1.1570333
Expansions	21	624	408	351
Goal Tests	22	625	409	352
New Nodes	84	5602	3364	3016.6667

Uniform Cost Search				
	Problem 1	Problem 2	Problem 3	Average
Plan Length	6	9	12	9
Time (s)	0.02	6.7043	30.9816	12.568633
Expansions	55	4852	18235	7714
Goal Tests	57	4854	18237	7716
New Nodes	224	44030	159716	67990



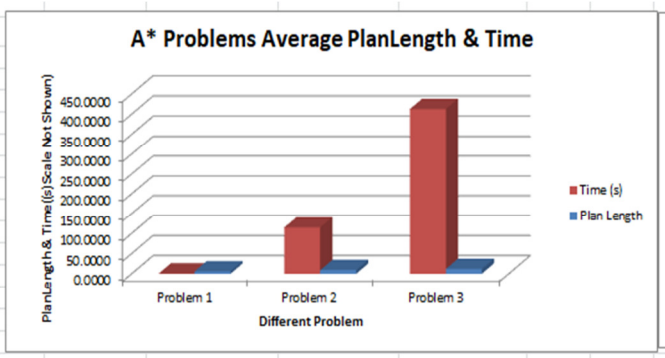


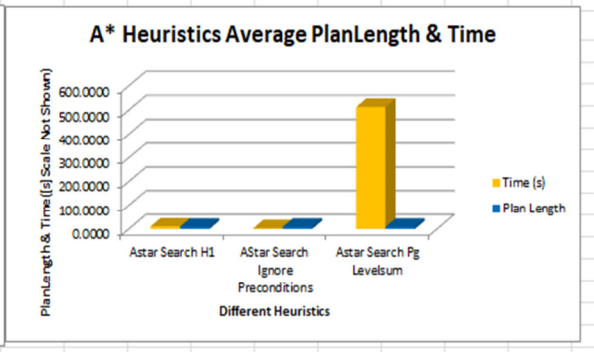
Optimal Plans		
Problem 1	**Problem 2**	**Problem 3**
Load(C1, P1, SFO) Load(C2, P2, JFK) Fly(P1, SFO, JFK) Fly(P2, JFK, SFO) Unload(C1, P1, JFK) Unload(C2, P2, SFO)	Load(C1, P1, SFO) Load(C2, P2, JFK) Load(C3, P3, ATL) Fly(P1, SFO, JFK) Fly(P2, JFK, SFO) Fly(P3, ATL, SFO) Unload(C3, P3, SFO) Unload(C2, P2, SFO) Unload(C1, P1, JFK)	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)

Metrics of A* Searches:

Average Across Problems			
	Problem 1	Problem 2	Problem 3
Plan Length	6	9	12
Time (s)	0.2985	114.8012	412.8634
Expansions	45	2477	8434
Goal Tests	47	2479	8436
New Nodes	184	22522	74198

Average Across Heuristics			
	Astar Search H1	AStar Search Ignore Preconditions	Astar Search Pg Levelsum
Plan Length	9	9	9
Time (s)	11.0898	3.4951	513.3783
Expansions	7714	2177	1065
Goal Tests	7716	2179	1067
New Nodes	67990	19472	9441





The Metrics of A* Searches showed some intriguing results, and showed the performance penalty of having an elaborate function over a simpler one. On another note, the average time and Plan Length would be the most effective way to explain each heuristic without using all metrics (Graphs). To start, using average time across heuristics to gauge problem difficulty, Problem1 was the easiest, then Problem2 was almost 380 times in difficulty, and Problem3 was about 1,400 times more difficult than Problem1. All heuristics for all problems came to the correct solution 6,9,12 for problems 1,2,3 respectively.

Using an Average for Metrics of Problem 1, 2 & 3 for each Heuristic:

Note below, Averages are used to show how well each heuristic performs. Additionally, averaging each metric over the three problems for each heuristic allows one to easily discern between them with little loss from decomposition of averaging (See Table Above). First, H1 had an average Plan Length of 9, a time of 11.0898(s), and the maximum of 67,990 Nodes Expansions (on Average). Ignore Preconditions had an average Plan Length of 9, a minimum time of 3.4951(s), and 19,472 Node Expansions (on Average). PG Levelsum had an average Plan Length of 9, a maximum time of 513.3783(s),

and the minimum of 9,441 Node Expansions (on Average). To sum up, one of the ideas to take from this is the scheme of this data seems to show the cost of intelligence to execution time.

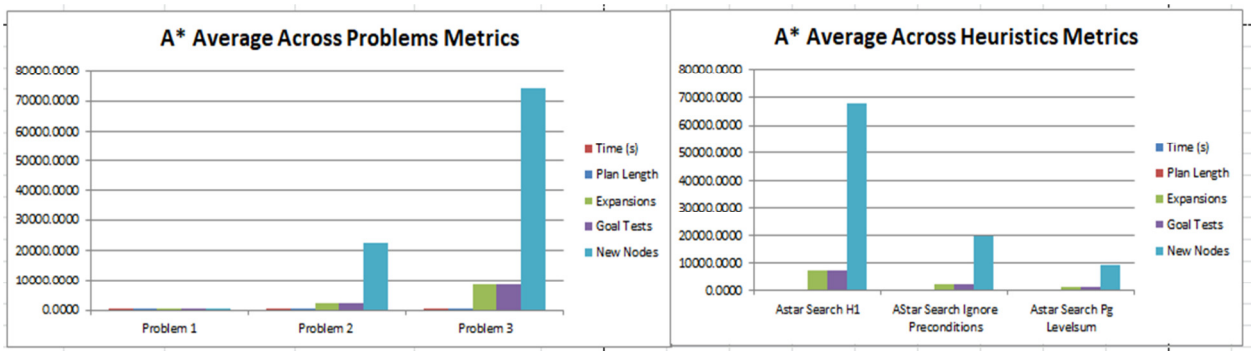
Resolution:

On another note, Time out of all the metrics seems to be the best to gauge, and new nodes a second. However, it begs the question how well would these heuristics do on difficult problems since they all came to the correct solution? The quickest time heuristic may be the one to solve this, but the heuristic may be unintelligent on those difficult problems. On the other side, intelligent heuristics like PG Levelsum may be better, but one has to balance the maximum computation time of the heuristic. So these two reasons, two sides of the coin, are why I fall somewhere in the middle and rank H1, Ignore Preconditions, and PG levelsum from best to worst. Even though, that rank is not in respect to time.

Observed Results Justification from Udacity's Material:

- **Udacity's Teachers quote "*do the stupid thing first …*":** A justification of why more intelligent PG Levelsum performed the worst in terms of computation time.
- **"Planning & Execution":** A Justification Topic of why the lesser algorithms performed better than the intelligent ones in terms of computation time.
- (**Note: Be sure to read the Paragraph below**)

The Paragraph I explained above where I was using the current metrics to figure out how the heuristics would perform on difficult problems looks to have a better solution from one of Udacity's teacher's quotes in the Simulated Annealing Section. Another reason for this issue is "Planning and Execution" concept from Udacity's Planning Section. The quote explains the observed (more intelligent heuristics [i.e. PG levelsum] have a greater cost in execution time) results of this experiment. He said, "*You've heard me say 'do the stupid thing first, and only and intelligence when necessary*." Which the quote is a great explanation on how to attack a greater problem that needs more intelligence, but to not get caught in the hang up of the cost of execution time of adding too much intelligence shown in this experiment. Finally, in accordance to finding a good generic solution, the quote 'explains the reason for the observed results' of why the lesser algorithms performed better than the intelligent ones in terms of computation time. The Test and the quote also explain the importance of the method to solve highly difficult problems especially with Python.



Astar Search H1				
	Problem 1	Problem 2	Problem 3	Average
Plan Length	6	9	12	**9**
Time (s)	0.0178	5.7877	27.4638	**11.0898**
Expansions	55	4852	18235	**7714**
Goal Tests	57	4854	18237	**7716**
New Nodes	224	44030	159716	**67990**

AStar Search Ignore Preconditions				
	Problem 1	Problem 2	Problem 3	Average
Plan Length	6	9	12	**9**
Time (s)	0.0141	1.8659	8.6054	**3.4951**
Expansions	41	1450	5040	**2177**
Goal Tests	43	1452	5042	**2179**
New Nodes	170	13303	44944	**19472**

Astar Search Pg Levelsum				
	Problem 1	Problem 2	Problem 3	Average
Plan Length	6	9	12	**9**
Time (s)	0.8637	336.7501	1202.5211	**513.3783**
Expansions	39	1129	2026	**1065**
Goal Tests	41	1131	2028	**1067**
New Nodes	158	10232	17933	**9441**