AI Nanodegree Project 3: Planning

Written Analysis

Optimal plan

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

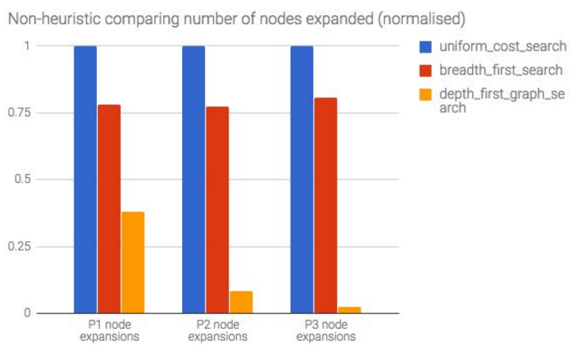
Problem 3

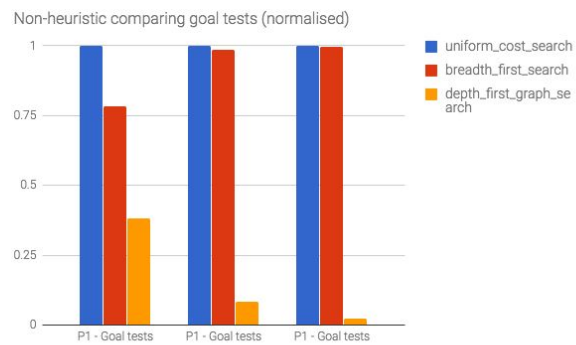
Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P1, SFO, ATL)
Load(C3, P1, ATL)
Fly(P2, JFK, ORD)
Load(C4, P2, ORD)
Fly(P2, ORD, SFO)

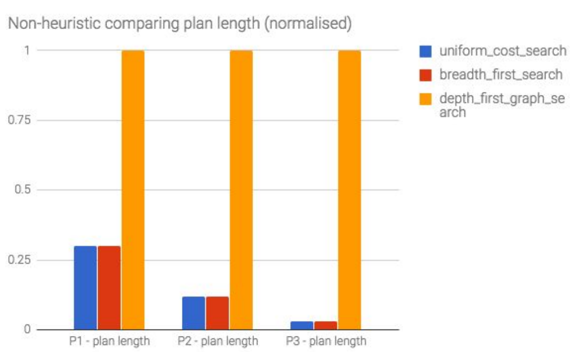
Fly(P1, ATL, JFK)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)
Unload(C3, P1, JFK)
Unload(C4, P2, SFO)

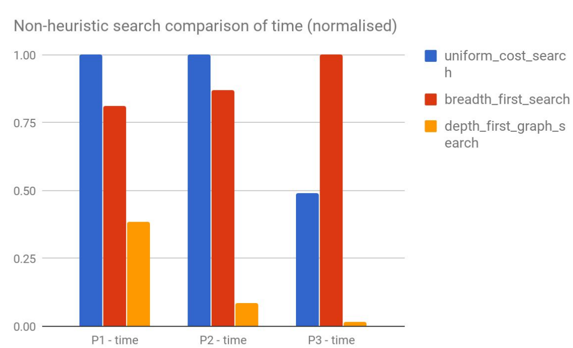
A comparison between non-heuristic search results

The following plots show a comparison between **uniform cost search**, **breadth first search**, and **depth first graph search** comparing **node expansion**, **goal tests**, **plan length** and **time** for each problem.







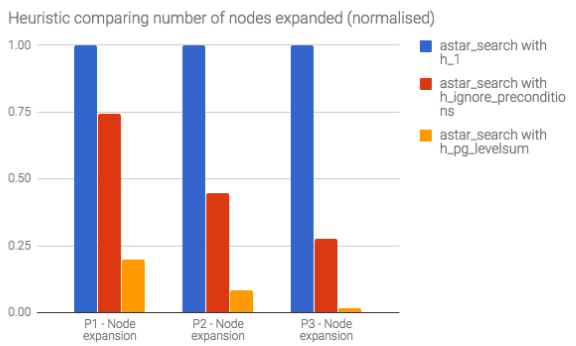


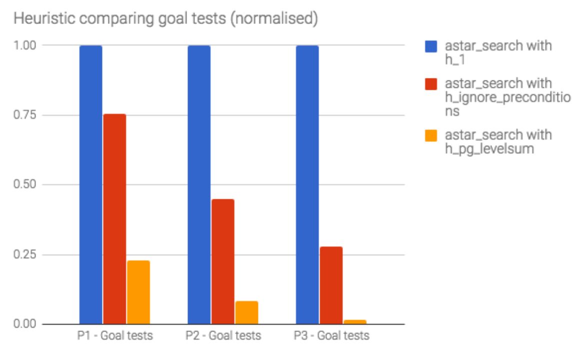
The table of the actual results can be found in the appendix.

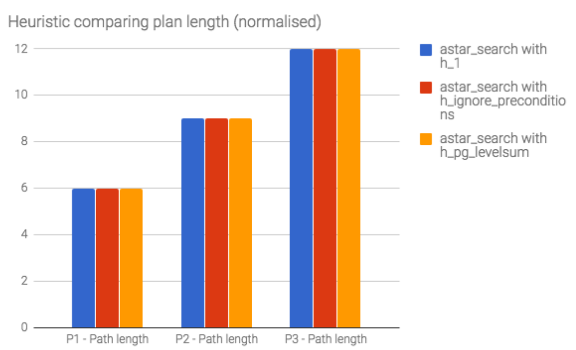
As highlighted in the plots above; **depth first** outperforms the other search approaches in terms of time but performs poorly finding the optimum path as highlighted in the **path length** plot, which makes sense as depth first will terminate its search as soon as the goal is reached while the other searches evaluate nodes per level/depth, therefore being able to discover the optimum path before terminating prematurely. The others, excluding depth first, find the optimum plan therefore the comparison is made on the time taken between **uniform cost** and **breadth first** search. Interestingly, **breadth first search** outperforms **uniform cost search** for problems 1 and 2 but this is switched for problem 3 which suggests **uniform cost** favors complexity more than a **breadth first search**.

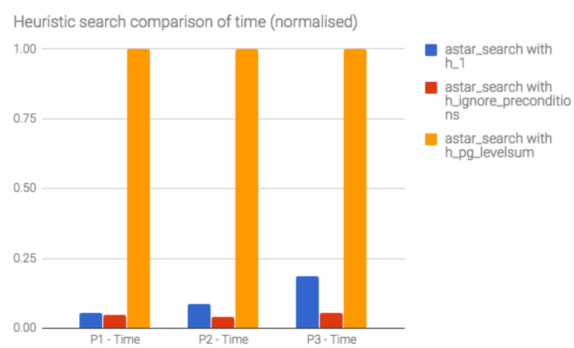
A comparison between heuristic search results

The following plots show a comparison between **astar_search with h_1**, **astar_search with h_ignore_preconditions**, and **astar_search with h_pg_levelsum** comparing **node expansion**, **goal tests**, **plan length** and **time** for each problem.







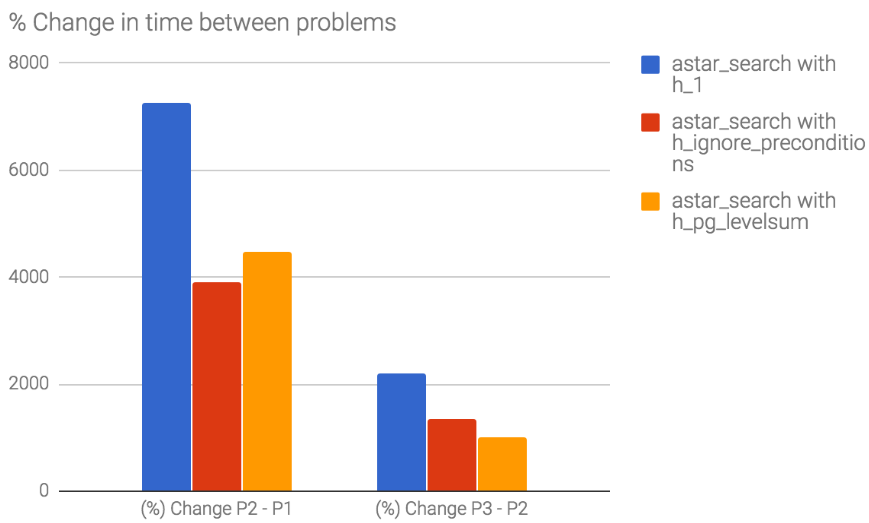


The table of the actual results can be found in the appendix.

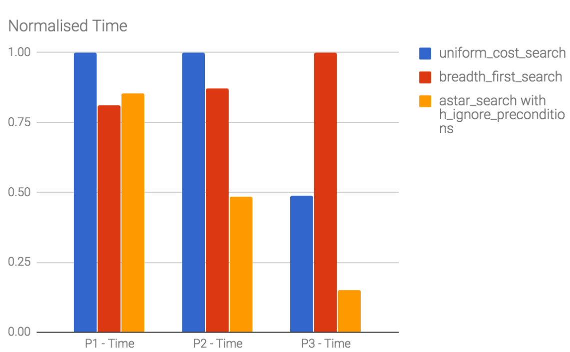
All heuristics achieved equal plan lengths; the differences of the three approaches were then based on the number of **nodes expanded**, **goal tests** performed and the **time** they took. The noticeable difference here was between **astar_search with h_pg_levelsum** and the other two**, h_1, astar_search ignore_preconditions.** While **h_pg_levelsum** had significantly expanded less nodes and performed less goal tests, it took a great more time to perform the search.

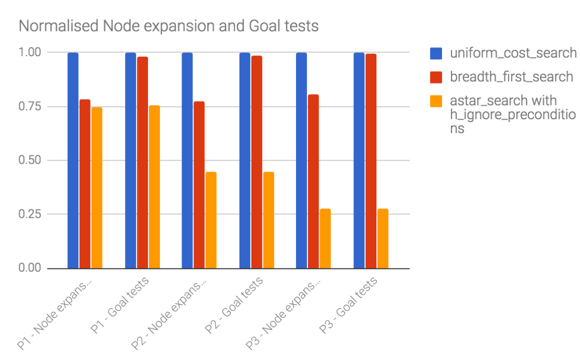
Another important comparison is how well the searches performed with increasing complexity of the search, the following plot shows the percentage change for each of the searches between Problem 2 and

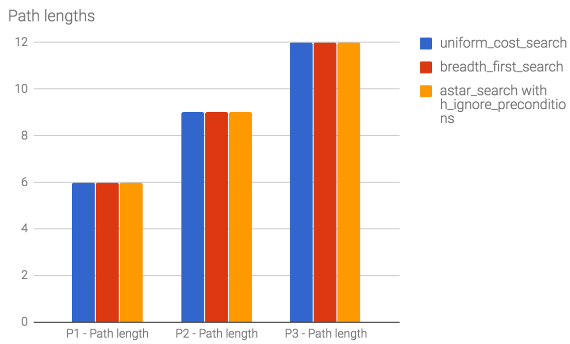
Problem 1, and Problem 3 and Problem 2. As illustrated below, an increase in complexity had the greatest impact on h_1 compared to the others, suggesting that h_1 doesn't scale well with increases in the size of the search space (complexity).



Therefore the best heuristic for these problems would be **h_ignore_preconditions**. So how does it compare with the non-heuristic approaches presented above; below we present the same plots comparing **uniform_cost_search** and **breadth_first_search** (the best 2) with **astar_h_ignore_preconditions**.







As seen above; all searches achieved the same path length but the ignore preconditions heuristic outperformed in time, node expansion, and goal tests. Therefore, for this problem, you would opt for the heuristic approach compared to the others for reasons of better performance and scalability.

References

1. Artificial Intelligence: Pearson New International Edition: A Modern Approach by Stuart Russell and Peter Norvig https://www.amazon.co.uk/Artificial-Intelligence-Pearson-International-Approach-ebook/dp/B00IZ0L90C/ref=tmm_kin_swatch_0?_encoding=UTF8&qid=&sr=
2. Udacity AI Nanodegree; Logic and Reasoning Module https://classroom.udacity.com/nanodegrees/nd889/parts/6be67fd1-9725-4d14-b36e-ae2b5b20804c/modules/72832d87-79e7-4fee-b9dd-c3e88993aa7d/lessons/0da3193e-e047-40d9-aa79-c211e5973221/concepts/7dfa5f13-f5cd-480f-92bb-2c7c62d41907

Appendix

Results from Problem 1

	time	number of node expansions	number of goal tests	plan length	loads	unloads	flys
uniform_cost_search	0.04	55	57	6	8	8	4
breadth_first_search	0.03	43	56	6	8	8	4
depth_first_graph_search	0.02	21	22	20	8	8	4
astar_search with h_1	0.04	55	57	6	8	8	4
astar_search with h_ignore_preconditions	0.03	41	43	6	8	8	4
astar_search with h_pg_levelsum	0.73	11	13	6	8	8	4

Results from Problem 2

	time	number of node expansions	number of goal tests	plan length	loads	unloads	flys
uniform_cost_search	2.79	992	994	9	27	27	18
breadth_first_search	2.43	769	979	9	27	27	18
depth_first_graph_search	0.24	81	82	76	27	27	18
astar_search with h_1	2.84	992	994	9	27	27	18
astar_search with h_ignore_preconditions	1.35	443	445	9	27	27	18
astar_search with h_pg_levelsum	32.62	82	84	9	27	27	18

Results from Problem 3

	time	number of node expansions	number of goal tests	plan length	loads	unloads	flys
uniform_cost_search	58.35	18151	18153	12	32	32	24

breadth_first_search	119.12	14663	18098	12	32	32	24
depth_first_graph_search	2.04	408	409	392	32	32	24
astar_search with h_1	62.19	18151	18153	12	32	32	24
astar_search with h_ignore_preconditions	18.08	5038	5040	12	32	32	24
astar_search with h_pg_levelsum	331.01	314	316	12	32	32	24