Analysis of Planning Search Problem

For each of the 3 problems, the following searches were run. Optimal solutions and test statistics can be found in tables on pages 2 and 3 respectively.

Breadth-first Search

With this search method we find that it excels at finding solutions quickly with limited state space, such as problem 1. This is because it has less distance to travel within a short tree, of which it is searching layer by layer. As the state space increases, we see the number of expansions grow in addition to the runtime, placing it behind others with a more specific search method.

Depth-first Graph Search

Per the video lectures, we find that Depth-first search is unable to find the optimal solution to a problem because of the way node expansion is performed. By continually expanding the left-most node, it fails to compare other nodes that may have better initial performance, thus finding a solution that must include the initially expanded node.

Uniform Cost Search

This search utilizes step cost as a means of branching and outperforms Breadth-first in terms of speed as the state-space increases in size by calculating the step-cost before moving to expand a node. This causes a deeper expansion than breadth-first, making it the best correct search of the uninformed methods.

A* Search with Ignore Preconditions Heuristic

By removing preconditions from the problem, it allows the search to start with a greater breadth of options in initial selection. Combining this with A* causes the search to quickly calculate the lowest cost for each node, with the cost being the distance to the goal, which should be the number of goals achieved in a state. The heuristic will never be overestimated as a result, thus making it admissible for A* search, as well as the best performing search in terms of runtime.

A* Search with Level-sum Heuristic

This search significantly outperformed others in terms of the amount of expansion required to find the solution, because of the specificity of the heuristic to the goal. While this may seem optimal in terms of memory usage, its runtime performance suffers from the complexity of the calculations, thus making it sub-optimal in this case. It is worth noting that if the amount of goals increase without large changes in state-space, the performance would improve over others.

Conclusion

The correlation between runtime and expansions(Page 4) seems clear that expansions would cause greater runtime, but it is worth noting that the depth of the expansions is not recorded by the tests, as is the case with searches like breadth-first and uniform cost. However, these two are outperformed by A* Ignore Preconditions search which requires less depth, and has better node selection earlier in the search. This makes A* Ignore Preconditions the optimal solution for this problem, but is questionable as the goal complexity increases.

Figure 1: Optimal Solutions

Step	Problem 1	Problem 2	Problem 3
1	Load(C1, P1, SFO)	Load(C1, P1, SFO)	Load(C1, P1, SFO)
2	Load(C2, P2, JFK)	Load(C2, P2, JFK)	Load(C2, P2, JFK)
3	Fly(P1, SFO, JFK)	Load(C3, P3, ATL)	Fly(P1, SFO, ATL)
4	Fly(P2, JFK, SFO)	Fly(P1, SFO, JFK)	Load(C3, P1, ATL)
5	Unload(C1, P1, JFK)	Fly(P2, JFK, SFO)	Fly(P2, JFK, ORD)
6	Unload(C2, P2, SFO)	Fly(P3, ATL, SFO)	Load(C4, P2, ORD)
7		Unload(C1, P1, JFK)	Fly(P2, ORD, SFO)
8		Unload(C2, P2, SFO)	Fly(P1, ATL, JFK)
9		Unload(C3, P3, SFO)	Unload(C1, P1, JFK)
10			Unload(C2, P2, SFO)
11			Unload(C3, P1, JFK)
12			Unload(C4, P2, SFO)

Figure 2: Test Results

problem #	search type	expansions	goal tests	new nodes	plan length	runtime
1	depth_first_graph	21	22	84	20	0.020
1	**breadth_first**	43	56	180	6	**0.030**
1	h_1	55	57	224	6	0.039
1	uniform_cost	55	57	224	6	0.040
1	h_ignore_preconditions	41	43	170	6	0.040
1	h_pg_levelsum	11	13	50	6	0.594
2	depth_first_graph	624	625	5602	619	3.670
2	**h_ignore_preconditions**	1450	1452	13303	9	**4.670**
2	uniform_cost	4853	4855	44041	9	12.910
2	h_1	4853	4855	44041	9	13.005
2	breadth_first	3343	4609	30509	9	14.840
2	h_pg_levelsum	86	88	841	9	51.410
3	depth_first_graph	408	409	3364	392	1.890
3	**h_ignore_preconditions**	5038	5040	44926	12	**18.104**
3	h_1	18151	18153	159038	12	56.390
3	uniform_cost	18151	18153	159038	12	57.440
3	breadth_first	14663	18098	129631	12	106.190
3	h_pg_levelsum	314	316	2894	12	251.612

Figure 3

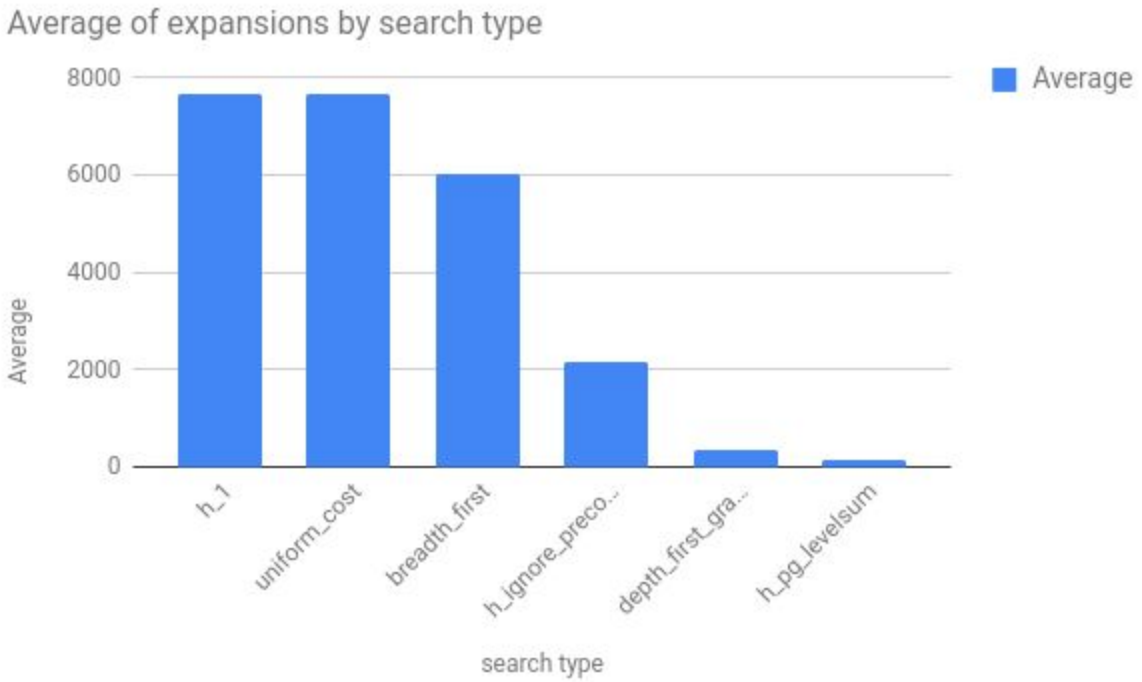

Figure 4