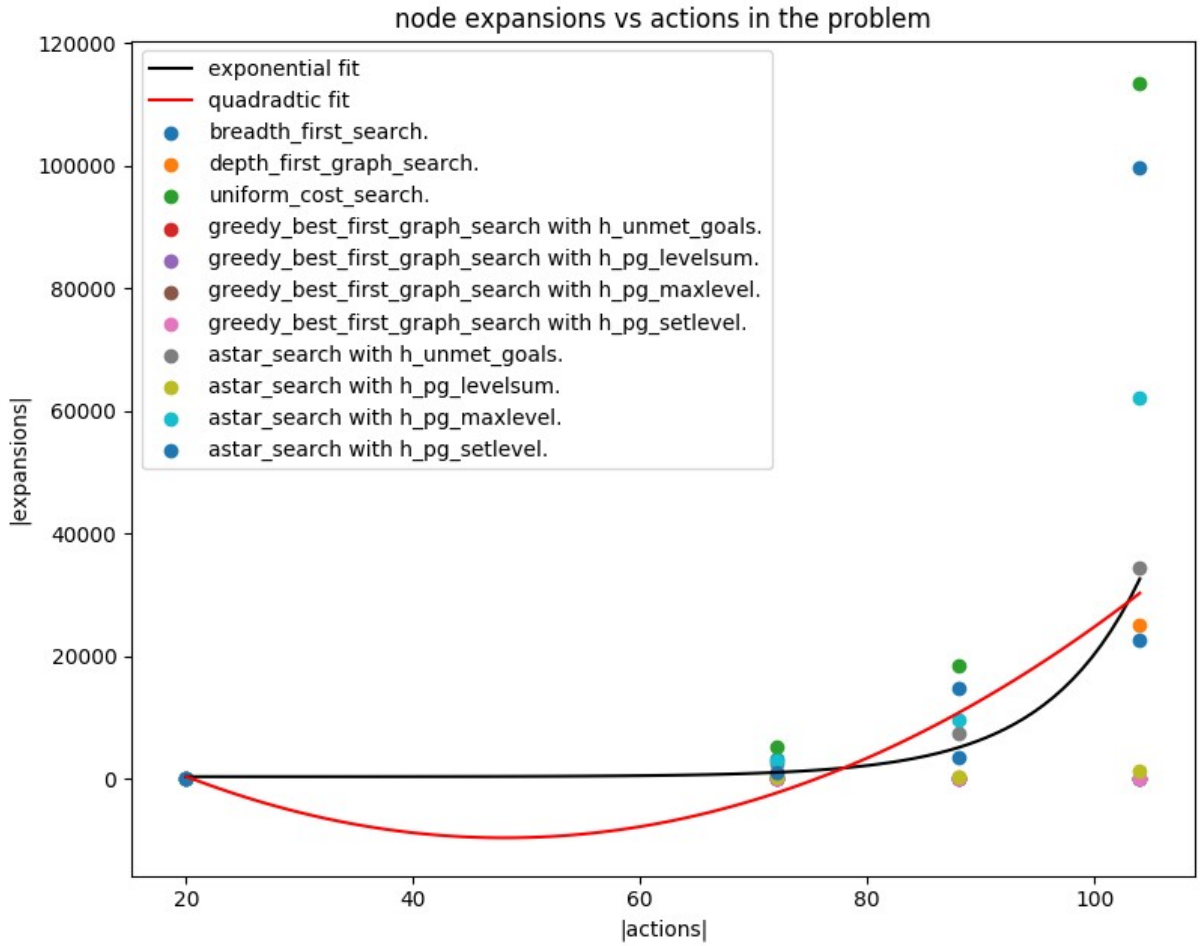
The Charts Include Everything

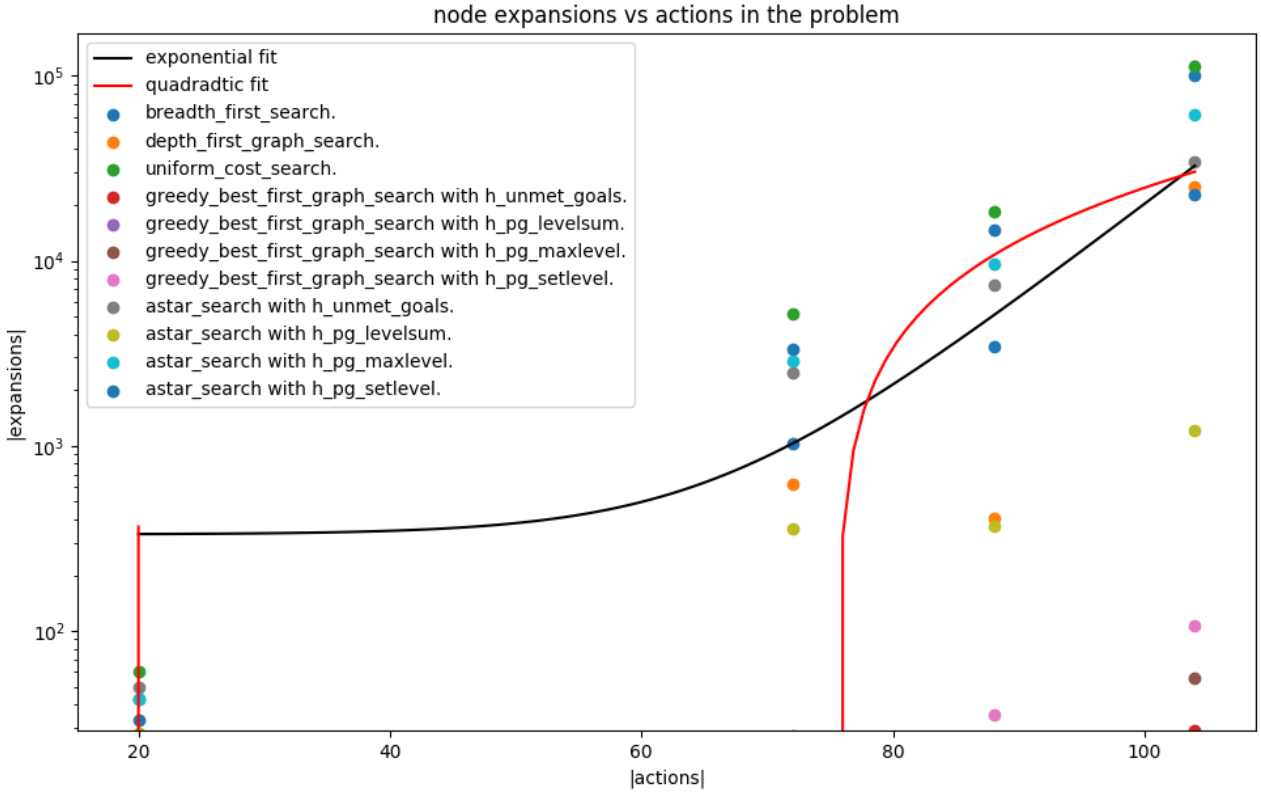
I ran all algorithms on all problems overnight (actually took like three hours) and used `> out.txt` to direct all the output to a file. Then I read that file in with a Python script to generate a .csv that looks like this:

problem	algorithm	|actions|	|expansions|	|goal tests|	|new nodes|	plan length	time
1	breadth_first_search.	20	43	56	178	6	0.026887484986219
1	depth_first_graph_search.	20	21	22	84	20	0.006128831999376
1	uniform_cost_search.	20	60	62	240	6	0.015531684999587
1	greedy_best_first_graph_search with h_unmet_goals.	20	7	9	29	6	0.001766043002135
1	greedy_best_first_graph_search with h_pg_levelsum.	20	6	8	28	6	0.262220106000314
1	greedy_best_first_graph_search with h_pg_maxlevel.	20	6	8	24	6	0.082493590001832
1	greedy_best_first_graph_search with h_pg_setlevel.	20	6	8	28	6	0.517192218991113
1	astar_search with h_unmet_goals.	20	50	52	206	6	0.011172624013852
1	astar_search with h_pg_levelsum.	20	28	30	122	6	0.238229433001834
1	astar_search with h_pg_maxlevel.	20	43	45	180	6	0.106828834002954
1	astar_search with h_pg_setlevel.	20	33	35	138	6	0.317087642994011
2	breadth_first_search.	72	3343	4609	30503	9	0.379143211990595
2	depth_first_graph_search.	72	624	625	5602	619	0.613408571996843
2	uniform_cost_search.	72	5154	5156	46618	9	0.684181618009461
2	greedy_best_first_graph_search with h_unmet_goals.	72	17	19	170	9	0.019964142993558
2	greedy_best_first_graph_search with h_pg_levelsum.	72	9	11	86	9	0.307732144006877
2	greedy_best_first_graph_search with h_pg_maxlevel.	72	27	29	249	9	0.676083297003061
2	greedy_best_first_graph_search with h_pg_setlevel.	72	9	11	84	9	1.02380514399556
2	astar_search with h_unmet_goals.	72	2467	2469	22522	9	0.590164686000207
2	astar_search with h_pg_levelsum.	72	357	359	3426	9	7.26005451899255
2	astar_search with h_pg_maxlevel.	72	2887	2889	26594	9	41.051321624007
2	astar_search with h_pg_setlevel.	72	1037	1039	9605	9	81.2527059419954
3	breadth_first_search.	88	14663	18098	129625	12	0.894120686003589
3	depth_first_graph_search.	88	408	409	3364	392	0.216567565003061
3	uniform_cost_search.	88	18510	18512	161936	12	1.53997297699971
3	greedy_best_first_graph_search with h_unmet_goals.	88	25	27	230	15	0.013089401007164
3	greedy_best_first_graph_search with h_pg_levelsum.	88	14	16	126	14	1.25079125299817
3	greedy_best_first_graph_search with h_pg_maxlevel.	88	21	23	195	13	0.711904764000792
3	greedy_best_first_graph_search with h_pg_setlevel.	88	35	37	345	17	6.36577354399196
3	astar_search with h_unmet_goals.	88	7388	7390	65711	12	1.12290695300908
3	astar_search with h_pg_levelsum.	88	369	371	3403	12	15.0431990990037
3	astar_search with h_pg_maxlevel.	88	9580	9582	86312	12	230.680279823006
3	astar_search with h_pg_setlevel.	88	3423	3425	31596	12	418.768557762
4	breadth_first_search.	104	99736	114953	944130	14	5.15611103898846
4	depth_first_graph_search.	104	25174	25175	228849	24132	1017.763237123
4	uniform_cost_search.	104	113339	113341	1066413	14	7.68130260299949
4	greedy_best_first_graph_search with h_unmet_goals.	104	29	31	280	18	0.015134433007916
4	greedy_best_first_graph_search with h_pg_levelsum.	104	17	19	165	17	0.995969434996368
4	greedy_best_first_graph_search with h_pg_maxlevel.	104	56	58	580	17	1.8216014750069
4	greedy_best_first_graph_search with h_pg_setlevel.	104	107	109	1164	23	20.4449730060005
4	astar_search with h_unmet_goals.	104	34330	34332	328509	14	3.74191658101336
4	astar_search with h_pg_levelsum.	104	1208	1210	12210	15	63.9328331969882
4	astar_search with h_pg_maxlevel.	104	62077	62079	599376	14	2184.39295000001
4	astar_search with h_pg_setlevel.	104	22606	22608	224229	14	4283.132790287

I read that back in to a different script as a pandas Dataframe to make plots. So all my plots contain data from all possible experiments.

Number of Expansions vs Number of Actions



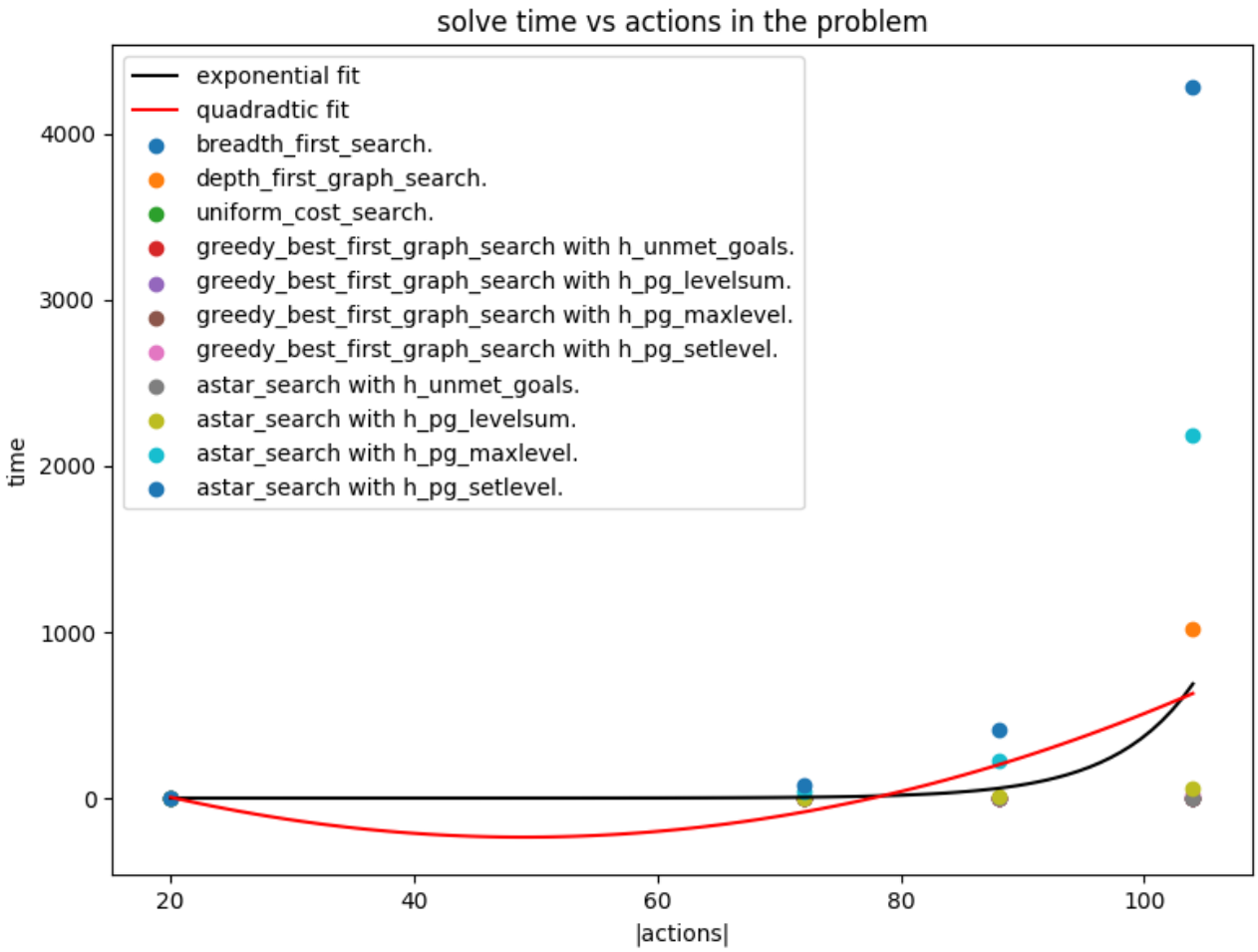


(linear and then logspace. Sorry if you're colorblind.)

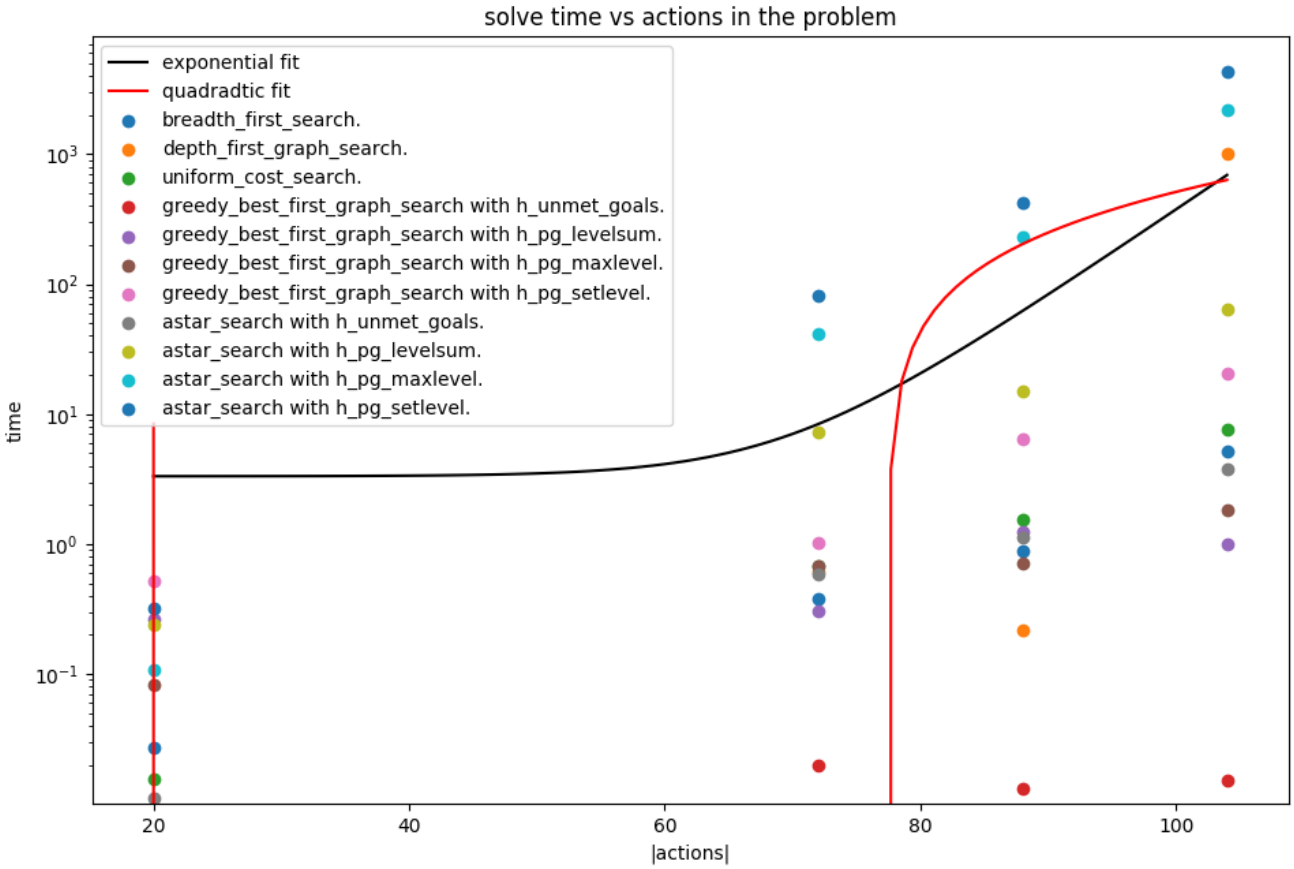
I've tried to fit the data to a polynomial (quadratic shown), but no polynomial of any order captures the data well, as is especially apparent in logspace. An exponential curve is much better, which matches the fact the search spaces grow exponentially with problem complexity.

Breadth-first search and Uniform-cost search seem to make the most expansions.

Search Time vs Number of Actions



Again it grows exponentially.

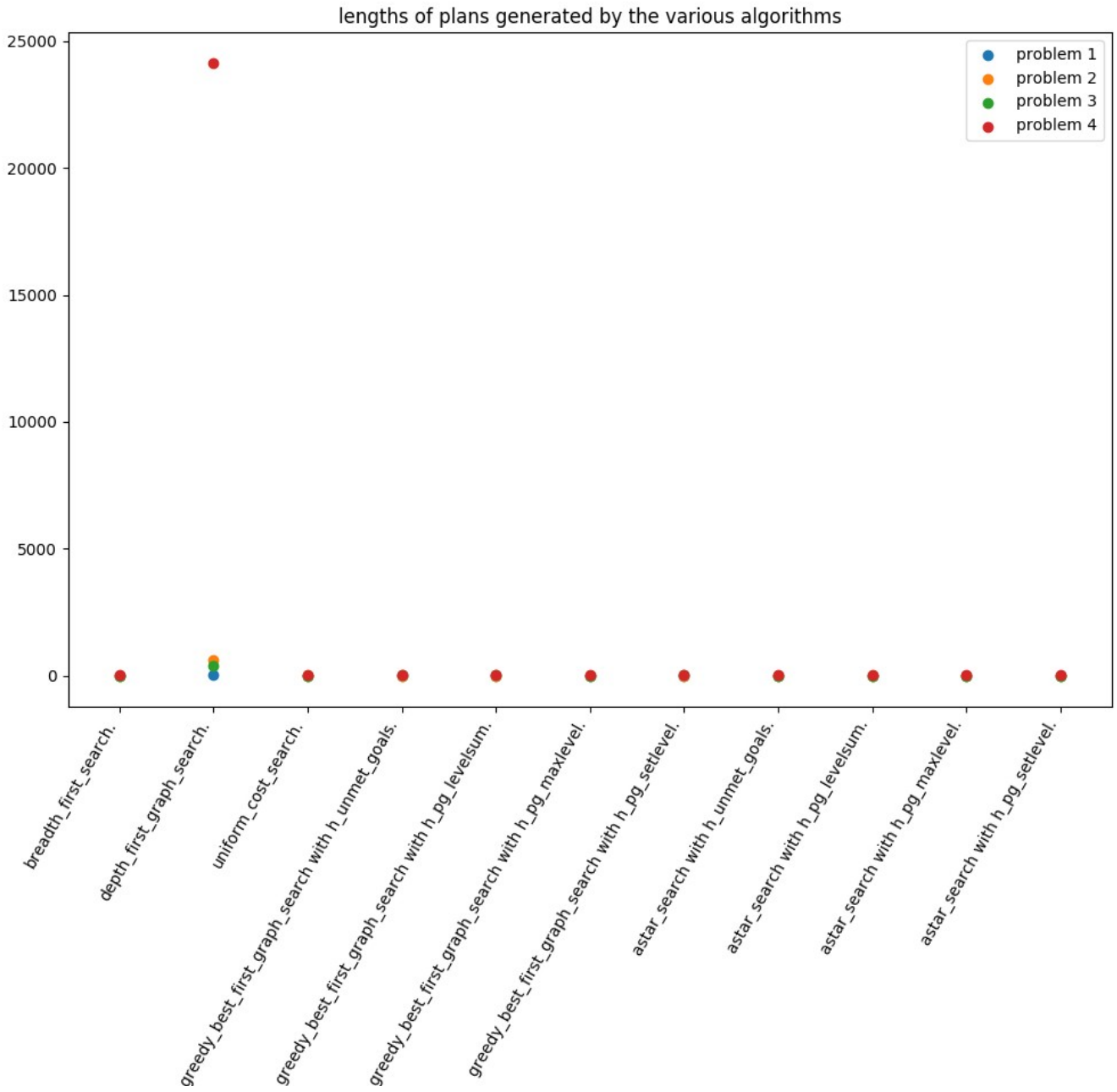


And here it is in logspace.

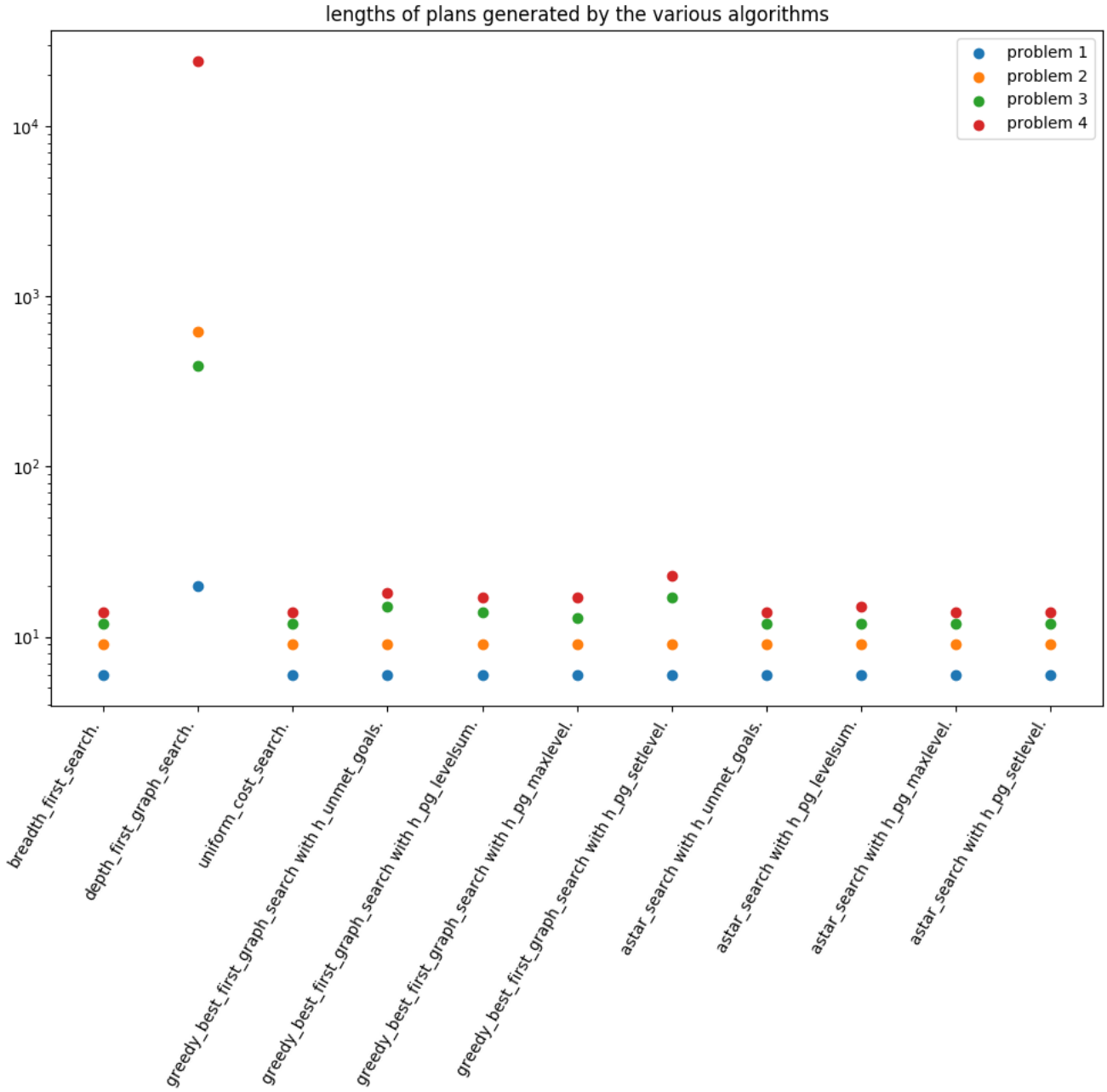
Greedy best-first with the unmet goals heuristic seems to be reliably fastest. That doesn't mean the solution it finds is optimal, just that it finds a candidate without doing too much work.

Again, time spent tends to grow exponentially, because the search space itself grows exponentially. This doesn't fit a polynomial.

Solution Lengths



Linear space



Logspace

So clearly depth-first search is the loser. No surprises there. It doesn't back up when it does something wrong or unproductive, so it has to stumble its way to the goal, all the while getting in its own way.

Also no surprise, breadth-first search always finds the shortest plan, just like it always finds the shortest path through a graph.

Questions

1. Real time operation in a small search space

I'd favor Greedy Best-First with the Unmet Goals heuristic, because it dominates the others on timeliness and space-efficiency and seems to return optimal plans (as good as any of the other approaches) so long as the search space is small. (And it doesn't do much worse than optimal even for slightly larger spaces.)

2. Large domains

In incredibly large domains, time and space are going to matter most. Greedy Best-First with Unmet Goals heuristic is fast and small. So is Greedy Best-First with Level Sum heuristic and Greedy Best-First with Max Level heuristic. A* with Unmet Goals heuristic is also very fast and might give a slightly more optimal answer, if you can spare the memory for all the expansions.

3. When optimality matters most

If actions are pretty expensive, like actually flying a plane from one place to another, then you really want to make sure the plan is optimal. From Robotics: Thinking is cheaper than communicating; communicating is cheaper than moving. BFS, Uniform Cost Search, and any of the A* with an admissible heuristic (not levelsum) will achieve shortest plan. In these experiments BFS seems to dominate time-wise—at least for this implementation of A* and heuristics—, so if you're in a hurry and don't care about expanding more nodes and all actions happen to have the same cost, BFS. If you care about minimizing the number of nodes explored, which in an optimized implementation probably corresponds better with time, use A* with something like the set-level heuristic. If the cost of achieving a literal is actually more complicated than just its level in a planning graph, then definitely A* with a heuristic that encodes that complication.