Report for Project 2: Classical Planning

Yudai Furukawa

Analysis

1. The number of nodes expanded against number of actions in the domain
    a. According to table 1 and chart 1, as the number of actions increases, the number of nodes expanded increases in general. For example, for breadth first search, the number of nodes expanded exponentially increased from 43 to 99736 as the number of actions increased from 20 to 104.
2. The search time against the number of actions in the domain
    a. According to table 1 and chart 2, as the number of actions increases, the search time increases in general although the rate of increase depends on algorithms. According to chart 2, the search time of astar_search with h_pg_maxlevel increased the fastest among the algorithms as the number of actions increased.
3. The length of the plans returned by each algorithm on all search problems
    a. According to table 1 and chart 3, as the number of actions increases, the length of the plans increases although the rate of increase depends on algorithms. The rate of increase is similar among algorithms besides that depth_first_graph_search has higher rate of increase compared to other algorithms.

Questions

1. 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?
    a. The most appropriate algorithms for planning in a very restricted domain (i.e., one that has only a few actions) and needs to operate in real time would be blind search algorithms (depth_first_graph_search, depth_first_graph_search, and breadth_first_search) as there is no need for a heuristic given the domain is very restricted.
2. 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)
    a. When domain is big and there is modest time constraint, greedy_best_first_graph_search with h_unmet_goals seems to be the most appropriate among the algorithms. The algorithm is not guaranteed to find an optimal solution. However, the algorithm with a good heuristic can find a solution to a given problem without exponentially increase search time in very large domains.
3. Which algorithm or algorithms would be most appropriate for planning problems where it is important to find only optimal plans?
    a. When it is important to find only optimal plans, breadth_first_search and uniform_cost_search would be the most appropriate algorithms as the algorithms are guaranteed to find optimal plans. A* search would be a good candidate as well

since it finds optimal solution to problems as long as the heuristic is admissible which means it never overestimates the cost of the path to the from any given node.

Algorithms excluded from the test on Problem 3 and Problem 4

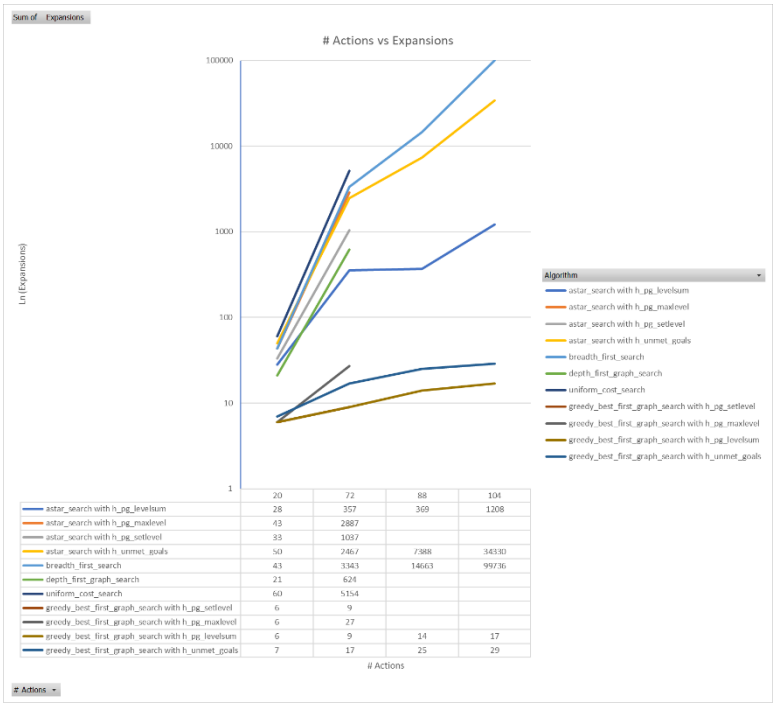
For Problem 3 and Problem 4, I excluded the following as Time elapsed in seconds exponentially increases as the number of actions increases.

- uniform_cost_search
- depth_first_graph_search
- greedy_best_first_graph_search with h_pg_setlevel
- greedy_best_first_graph_search with h_pg_maxlevel
- astar_search with h_pg_maxlevel
- astar_search with h_pg_setlevel

[Table1: Results]

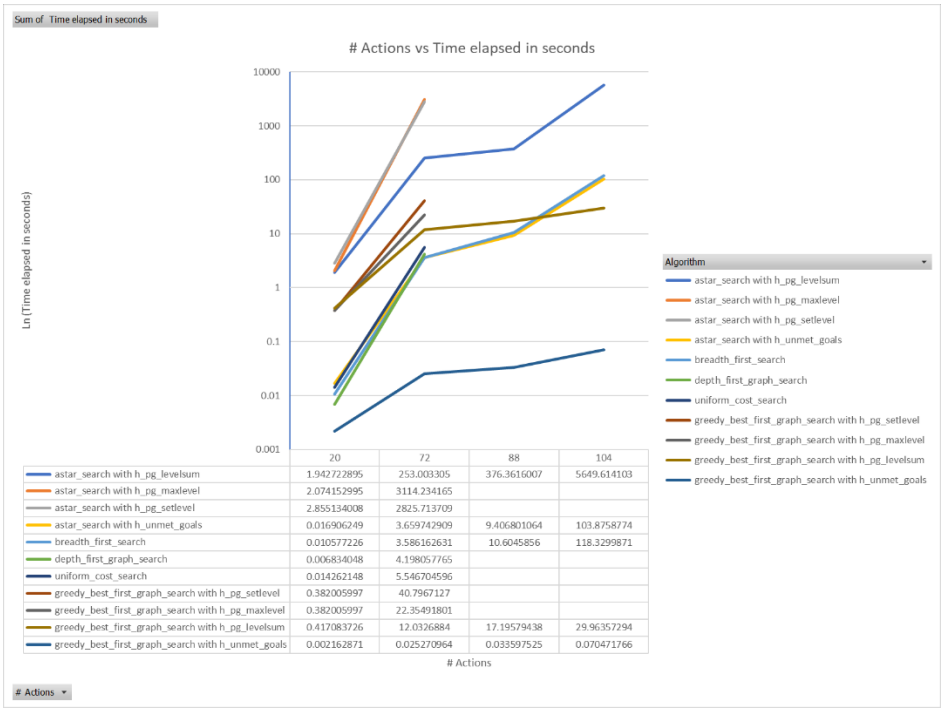
Problem: Algorithms	# Actions	Expansions	Goal Tests	New Nodes	Plan length	Time elapsed in seconds
Problem 1: astar_search with h_pg_levelsum	20	28	30	122	6	1.942722895
Problem 2: astar_search with h_pg_levelsum	72	357	359	3426	9	253.003305
Problem 3: astar_search with h_pg_levelsum	88	369	371	4303	12	376.3616007
Problem 4: astar_search with h_pg_levelsum	104	1208	1210	12210	15	5649.614103
Problem 1: astar_search with h_pg_maxlevel	20	43	45	180	6	2.074152995
Problem 2: astar_search with h_pg_maxlevel	72	2887	2889	26594	9	3114.234165
Problem 1: astar_search with h_pg_setlevel	20	33	35	138	6	2.855134008
Problem 2: astar_search with h_pg_setlevel	72	1037	1039	9605	9	2825.713709
Problem 1: astar_search with h_unmet_goals	20	50	52	206	6	0.016906249
Problem 2: astar_search with h_unmet_goals	72	2467	2469	22522	9	3.659742909
Problem 3: astar_search with h_unmet_goals	88	7388	7390	65711	12	9.406801064
Problem 4: astar_search with h_unmet_goals	104	34330	34332	328509	14	103.8758774
Problem 1: breadth_first_search	20	43	56	178	6	0.010577226
Problem 2: breadth_first_search	72	3343	4609	30503	9	3.586162631
Problem 3: breadth_first_search	88	14663	18098	12965	12	10.6045856
Problem 4: breadth_first_search	104	99736	114953	944130	14	118.3299871
Problem 1: depth_first_graph_search	20	21	22	84	20	0.006834048
Problem 2: depth_first_graph_search	72	624	625	5602	619	4.198057765
Problem 1: greedy_best_first_graph_search with h_pg_levelsum	20	6	8	28	6	0.417083726
Problem 2: greedy_best_first_graph_search with h_pg_levelsum	72	9	11	86	9	12.0326884
Problem 3: greedy_best_first_graph_search with h_pg_levelsum	88	14	16	126	14	17.19579438
Problem 4: greedy_best_first_graph_search with h_pg_levelsum	104	17	19	165	17	29.96357294
Problem 1: greedy_best_first_graph_search with h_pg_maxlevel	20	6	8	24	6	0.382005997
Problem 2: greedy_best_first_graph_search with h_pg_maxlevel	72	27	29	249	9	22.35491801
Problem 1: greedy_best_first_graph_search with h_pg_setlevel	20	6	8	28	6	0.382005997
Problem 2: greedy_best_first_graph_search with h_pg_setlevel	72	9	11	84	9	40.7967127
Problem 1: greedy_best_first_graph_search with h_unmet_goals	20	7	9	29	6	0.002162871
Problem 2: greedy_best_first_graph_search with h_unmet_goals	72	17	19	170	9	0.025270964
Problem 3: greedy_best_first_graph_search with h_unmet_goals	88	25	27	230	15	0.033597525
Problem 4: greedy_best_first_graph_search with h_unmet_goals	104	29	31	280	18	0.070471766
Problem 1: uniform_cost_search	20	60	62	240	6	0.014262148
Problem 2: uniform_cost_search	72	5154	5156	46618	9	5.546704596

[Chart 1: # Actions vs Expansions]



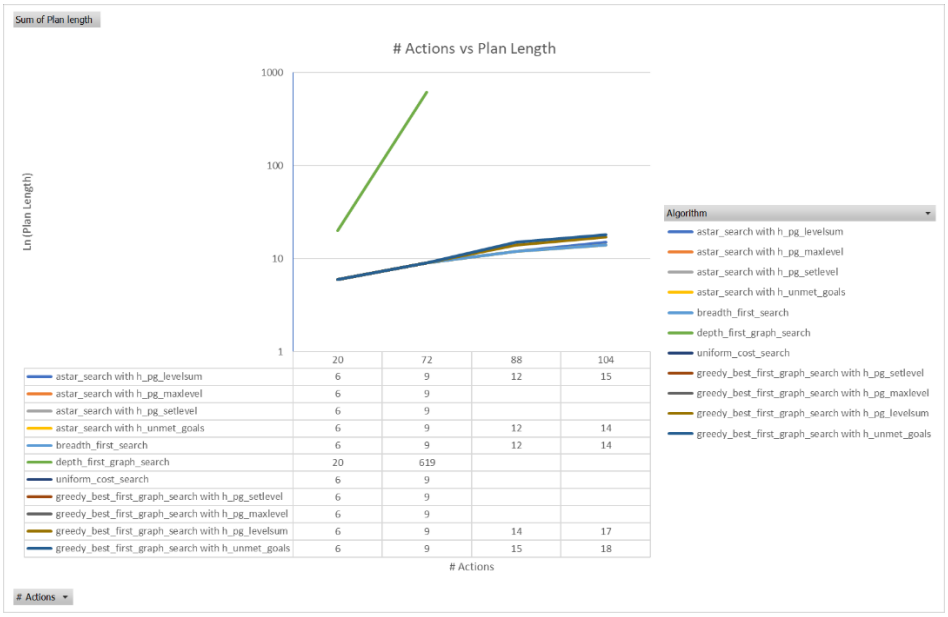
	20	72	88	104
astar_search with h_pg_levelsum	28	357	369	1208
astar_search with h_pg_maxlevel	43	2887		
astar_search with h_pg_setlevel	33	1037		
astar_search with h_unmet_goals	50	2467	7388	34330
breadth_first_search	43	3343	14663	99736
depth_first_graph_search	21	624		
uniform_cost_search	60	5154		
greedy_best_first_graph_search with h_pg_setlevel	6	9		
greedy_best_first_graph_search with h_pg_maxlevel	6	27		
greedy_best_first_graph_search with h_pg_levelsum	6	9	14	17
greedy_best_first_graph_search with h_unmet_goals	7	17	25	29

[Chart 2: # Actions vsTime elapsed in seconds]



	20	72	88	104
astar_search with h_pg_levelsum	1.942722895	253.003305	376.3616007	5649.614103
astar_search with h_pg_maxlevel	2.074152995	3114.234165		
astar_search with h_pg_setlevel	2.855134008	2825.713709		
astar_search with h_unmet_goals	0.016906249	3.659742909	9.406801064	103.8758774
breadth_first_search	0.010577226	3.586162631	10.6045856	118.3299871
depth_first_graph_search	0.006834048	4.198057765		
uniform_cost_search	0.014262148	5.546704596		
greedy_best_first_graph_search with h_pg_setlevel	0.382005997	40.7967127		
greedy_best_first_graph_search with h_pg_maxlevel	0.382005997	22.35491801		
greedy_best_first_graph_search with h_pg_levelsum	0.417083726	12.0326884	17.19579438	29.96357294
greedy_best_first_graph_search with h_unmet_goals	0.002162871	0.025270964	0.033597525	0.070471766

[Chart 3:Plan Length]



	20	72	88	104
astar_search with h_pg_levelsum	6	9	12	15
astar_search with h_pg_maxlevel	6	9		
astar_search with h_pg_setlevel	6	9		
astar_search with h_unmet_goals	6	9	12	14
breadth_first_search	6	9	12	14
depth_first_graph_search	20	619		
uniform_cost_search	6	9		
greedy_best_first_graph_search with h_pg_setlevel	6	9		
greedy_best_first_graph_search with h_pg_maxlevel	6	9		
greedy_best_first_graph_search with h_pg_levelsum	6	9	14	17
greedy_best_first_graph_search with h_unmet_goals	6	9	15	18