Udacity AI Nanodegree

Project2

Young Mo Kang

Searches

Search Algorithm Index	Name
1	breadth_first_search
2	depth_first_graph_search
3	uniform_cost_search
4	greedy_best_first_graph_search, 'h_unmet_goals'
5	greedy_best_first_graph_search, 'h_pg_levelsum'
6	greedy_best_first_graph_search, 'h_pg_levelmax'
7	greedy_best_first_graph_search, 'h_pg_setlevel'
8	astar_search, 'h_unmet_goals'
9	astar_search, 'h_pg_levelsum'
10	astar_search, 'h_pg_levelmax'
11	astar_search, 'h_pg_setlevel'

problem	1	1	1	1	1	1	1	1	1	1	1
search	1	2	3	4	5	6	7	8	9	10	11
action	20	20	20	20	20	20	20	20	20	20	20
expansion	43	21	60	7	6	6	6	50	28	43	33
plength	6	20	6	6	6	6	6	6	6	6	6
time	0.018645	0.010011	0.014196	0.001727	0.5065	0.157889	0.525699	0.013002	0.216243	0.185976	0.465905

problem	2	2	2	2	2	2	2	2	2	2	2
search	1	2	3	4	5	6	7	8	9	10	11
action	72	72	72	72	72	72	72	72	72	72	72
expansion	3343	624	5154	17	9	27	9	2467	357	2887	1037

plength	9	619	9	9	9	9	9	9	9	9	9
time	0.315513	0.460976	0.542123	0.010742	1.625311	1.429987	2.366873	0.605089	17.40377	99.68187	143.9131
problem	3	3	3	3	3	4	4	4	4	4	
search	1	4	6	8	9	1	4	6	8	9	
action	88	88	88	88	88	104	104	104	104	104	
expansion	14663	25	21	7388	369	99736	29	56	34330	1208	
plength	12	15	13	12	12	14	18	17	14	15	
time	0.839089	0.028849	3.016939	1.094969	30.63941	4.709518	0.032146	7.129627	3.478529	167.7043	

Q. Analyze the number of nodes expanded against number of actions in the domain

A. For each problem, the number of actions is the same (20 for P1, 72 for P2, 88 for P3, and 104 for P4). The number of node expansions depend on the search algorithm. For example, greedy_best_first_search algorithms (searches 4 through 7) consistently scores the least expansions throughout the problems. In general, # expansion is proportional to # actions.

Q. Analyze the search time against the number of actions in the domain

A. For each problem, the number of actions is the same (20 for P1, 72 for P2, 88 for P3, and 104 for P4). The search time increases from P1 to P4, so one can assume that in general #actions is proportional to search time.

Q. Analyze the length of the plans returned by each algorithm on all search problems

A. The lengths of the plans are all similar for each problem by each algorithm, except for algorithm 2 (depth_first_graph_search). For example, the plan length is all 9 for P2, except for algorithm 2, which yields 619!

Q. 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. P1 has only 20 actions, so we can assume this problem is a good representation of a restricted domain. Looking at the search time, we find algorithm 4 (greedy_best_first_graph_search, 'h_unmet_goals') finds the solution fastest.

Q. 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. P4 has the largest number of actions (104), so based on P4, algorithm 4 again takes the least amount of time.

Q. Which algorithm or algorithms would be most appropriate for planning problems where it is important to find only optimal plans?

A. We can assume optimal plans have the shorted plan lengths. Throughout the problems, A* searches mostly found shorted plan lengths solutions.