Udacity planning agent project

Problem description

This project development combines symbolic logic and classical search to implement an agent that performs progression search to solve planning problems.

Experiments results

To run the experiments was used the interact command to execute all search for all problems:

- python run_search.py -p 4 -s 1 2 3 4 5 6 7 8 9 10 11

Detailed results were generated on the console and copied to a text file to extract the data.

Air Cargo Problem 1 - 20 Actions					
	Expansions	**Goal Tests**	**New Nodes**	**Time**	**Path length**
breadth_first_search	43	56	178	0.0064	6
depth_first_graph_search	21	22	84	0.0045	20
uniform_cost_search	60	62	240	0.0108	6
greedy_best_first_graph_search					
h_unmet_goals	7	9	29	0.0020	6
h_pg_levelsum	6	8	28	0.3074	6
h_pg_maxlevel	6	8	24	0.2326	6
h_pg_setlevel	6	8	28	0.4161	6
astar_search					
h_unmet_goals	50	52	206	0.0101	6
h_pg_levelsum	28	30	122	0.7575	6
h_pg_maxlevel	43	45	180	0.8023	6
h_pg_setlevel	33	35	138	0.9881	6

Table 1: Problem 1 with 20 actions

For the problem 1 with 20 actions, we can observe that time execution is fast with less than 1 second. The depth first graph search takes a long path length and do not achieved the optimal solution.

Air Cargo Problem 2 - 72 actions					
	Expansions	Goal Tests	New Nodes	Time	Path length
breadth_first_search	3343	4609	30503	1.8974	9
depth_first_graph_search	624	625	5602	3.1013	619
uniform_cost_search	5154	5156	46618	3.2909	9
greedy_best_first_graph_search					
h_unmet_goals	17	19	170	0.0199	9
h_pg_levelsum	9	11	86	7.1646	9
h_pg_maxlevel	27	29	249	14.2936	9
h_pg_setlevel	9	11	84	10.7485	9
astar_search					
h_unmet_goals	2467	2469	22522	2.1963	9
h_pg_levelsum	357	359	3426	182.2547	9
h_pg_maxlevel	2887	2889	26594	1053.5323	9
h_pg_setlevel	1037	1039	9605	988.3511	9

Table 2: Problem 2 with 72 actions

For the problem 2 with 72 actions, we can observe that time execution increase because the execution needs more steps and as the problem 1 the depth first graph search takes a long path length (619) and do not achieved the optimal solution. For the case of A* searches time increases significantly.

Air Cargo Problem 3 - 88 actions					
	Expansions	Goal Tests	New Nodes	Time	Path length
breadth_first_search	14663	18098	129625	10.2827	12
depth_first_graph_search	408	409	3364	1.1722	392
uniform_cost_search	18510	18512	161936	14.2625	12
greedy_best_first_graph_search					
h_unmet_goals	25	27	230	0.0371	15
h_pg_levelsum	14	16	126	16.7290	14
h_pg_maxlevel	21	23	195	20.0952	13
h_pg_setlevel	35	37	345	61.9499	17
astar_search					
h_unmet_goals	7388	7390	65711	8.1939	12
h_pg_levelsum	369	371	3403	297.7101	12
h_pg_maxlevel	N/A	N/A	N/A	Too long	N/A
h_pg_setlevel	N/A	N/A	N/A	Too long	N/A

Table 3: Problem 1 with 88 actions

For the problem 3 with 88 actions, we can observe that time execution increase because the execution needs more steps. For the A* search the max_level and set_level took long time without results. The depth first graph search does not achieve the optimal solution.

Air Cargo Problem 4 - 104 actions					
	Expansions	**Goal Tests**	**New Nodes**	**Time**	**Path length**
breadth_first_search	99736	114953	944130	95.18790743	14
depth_first_graph_search	N/A	N/A	N/A	Too long	N/A
uniform_cost_search	113339	113341	1066413.0000	117.12752	14
greedy_best_first_graph_search					
h_unmet_goals	29	31	280.0000	0.059282207	18
h_pg_levelsum	17	19	165	28.0772	17
h_pg_maxlevel	56	58	580	68.1010	17
h_pg_setlevel	107	109	1164	247.0403	23
astar_search					
h_unmet_goals	34330	34332	328509	55.9976	14
h_pg_levelsum	1208	1210	12210	1603.9967	15
h_pg_maxlevel	N/A	N/A	N/A	Too long	N/A
h_pg_setlevel	N/A	N/A	N/A	Too long	N/A

Table 4: Problem 1 with 104 actions

For the problem 3 with 104 actions, we can observe that time execution increase because the execution needs more steps. For the A* search the max_level and set_level took long time without results and depth first graph happens the same.

Results analysis

Is clear in the results that time increases with more actions for all searches. In all the cases A* search take more time than greedy, that had a better performance in general. Breadth first search achieve optimal result and good execution time in all the problems.

Questions

a) Which algorithm or algorithms would be most appropriate for planning in a very restricted domain?

In this case, time execution with optimal result in an important factor to consider, for very restricted domains is like problem 1. Observing the results obtained on the experiment we can see:

Air Cargo Problem 1 - 20 Actions					
	Expansions	**Goal Tests**	**New Nodes**	**Time**	**Path length**
breadth_first_search	43	56	178	0.0064	6
depth_first_graph_search	21	22	84	0.0045	20
uniform_cost_search	60	62	240	0.0108	6
h_unmet_goals	7	9	29	0.0020	6
h_pg_levelsum	6	8	28	0.3074	6
h_pg_maxlevel	6	8	24	0.2326	6
h_pg_setlevel	6	8	28	0.4161	6
h_unmet_goals	50	52	206	0.0101	6
h_pg_levelsum	28	30	122	0.7575	6
h_pg_maxlevel	43	45	180	0.8023	6
h_pg_setlevel	33	35	138	0.9881	6

Result shows that breadth first search, uniform cost search and greedy best first fix better taking less time with less memory required.

b) Which algorithm or algorithms would be most appropriate for planning in large domains?

Large domains situation is like problem 4. Observing the results obtained on the experiment we can see:

Air Cargo Problem 4 - 104 actions					
	Expansions	**Goal Tests**	**New Nodes**	**Time**	**Path length**
breadth_first_search	99736	114953	944130	95.18790743	14
depth_first_graph_search	N/A	N/A	N/A	Too long	N/A
uniform_cost_search	113339	113341	1066413.0000	117.12752	14
greedy_best_first_graph_search					
h_unmet_goals	29	31	280.0000	0.059282207	18
h_pg_levelsum	17	19	165	28.0772	17
h_pg_maxlevel	56	58	580	68.1010	17
h_pg_setlevel	107	109	1164	247.0403	23
astar_search					
h_unmet_goals	34330	34332	328509	55.9976	14
h_pg_levelsum	1208	1210	12210	1603.9967	15
h_pg_maxlevel	N/A	N/A	N/A	Too long	N/A
h_pg_setlevel	N/A	N/A	N/A	Too long	N/A

In this case, only greedy best first graph search – unmet goals achieve a really good performance with low memory and low time to execute. Is important note that the optimal path is reached by A* ummet goals and breath first search, uniform cost search but takes long time (x100) to finish.

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

Observing the results obtained on the experiment we can see:

	Air Cargo Problem 1	Air Cargo Problem 2	Air Cargo Problem 3	Air Cargo Problem 4
breadth_first_search	6	9	12	14
depth_first_graph_search	20	619	392	N/A
uniform_cost_search	6	9	12	14
greedy_best_first_graph_search				
h_unmet_goals	6	9	15	18
h_pg_levelsum	6	9	14	17
h_pg_maxlevel	6	9	13	17
h_pg_setlevel	6	9	17	23
astar_search				
h_unmet_goals	6	9	12	14
h_pg_levelsum	6	9	12	15
h_pg_maxlevel	6	9	N/A	N/A
h_pg_setlevel	6	9	N/A	N/A

Comparing results breath first_search, uniform cost search and A* - unmet_goals achieve the optimal path for all problems experiment. But from these three, breath first_search, and A* - unmet_goals had better time and memory used performance.