Mark Merling – Project 2

Problem	Search	number of actions	number of new node expansions	time to complete the search plan	ratio nodes expanded / actions	time / actions
1	breadth_first_search	20	43	0.007	2.15	0.00035
1	depth_first_search	20	21	0.004	1.05	0.0002
1	uniform_cost_search	20	60	0.011	3	0.00055
1	greedy_best_first_graph_search with h_unmet_goals	20	7	0.002	0.35	0.0001
1	greedy_best_first_graph_search with h_pg_levelsum	20	6	0.361	0.3	0.01805
1	greedy_best_first_graph_search with h_pg_maxlevel	20	6	0.273	0.3	0.01365
1	greedy_best_first_graph_search with h_pg_setlevel	20	6	0.479	0.3	0.02395
1	astar_search with h_unmet_goals	20	50	0.01	2.5	0.0005
1	astar_search with h_pg_levelsum	20	28	0.918	1.4	0.0459
1	astar_search with h_pg_maxlevel	20	43	0.943	2.15	0.04715
1	astar_search with h_pg_setlevel	20	33	1.114	1.65	0.0557
2	breadth_first_search	72	3343	2.392	46.43055556	0.033222222
2	depth_first_search	72	624	3.278	8.666666667	0.045527778
2	uniform_cost_search	72	5154	3.967	71.58333333	0.055097222
2	greedy_best_first_graph_search with h_unmet_goals	72	17	0.02	0.236111111	0.000277778
2	greedy_best_first_graph_search with h_pg_levelsum	72	9	8.561	0.125	0.118902778
2	greedy_best_first_graph_search with h_pg_maxlevel	72	27	17.211	0.375	0.239041667
2	greedy_best_first_graph_search with h_pg_setlevel	72	9	11.313	0.125	0.157125
2	astar_search with h_unmet_goals	72	2467	2.564	34.26388889	0.035611111
2	astar_search with h_pg_levelsum	72	357	220.636	4.958333333	3.064388889
2	astar_search with h_pg_maxlevel	72	2887	1273.953	40.09722222	17.69379167
2	astar_search with h_pg_setlevel	72	1037	1041.77	14.40277778	14.46902778
3	breadth_first_search	88	14663	12.419	166.625	0.141125
3	depth_first_search	88	408	1.258	4.636363636	0.014295455
3	uniform_cost_search	88	18510	16.667	210.3409091	0.189397727
3	greedy_best_first_graph_search with h_unmet_goals	88	25	0.041	0.284090909	0.000465909
3	greedy_best_first_graph_search with h_pg_levelsum	88	14	19.49	0.159090909	0.221477273
3	greedy_best_first_graph_search with h_pg_maxlevel	88	21	23.479	0.238636364	0.266806818
3	greedy_best_first_graph_search with h_pg_setlevel	88	35	63.585	0.397727273	0.722556818
3	astar_search with h_unmet_goals	88	7388	9.707	83.95454545	0.110306818
3	astar_search with h_pg_levelsum	88	369	356.814	4.193181818	4.054704545
3	astar_search with h_pg_maxlevel	88	9580	6240.497	108.8636364	70.91473864
3	astar_search with h_pg_setlevel	88	3423	5532.933	38.89772727	62.87423864
4	breadth_first_search	104	99736	108.83	959	1.046442308
4	depth_first_search	104	25174	4154.921	242.0576923	39.95116346
4	uniform_cost_search	104	113339	130.481	1089.798077	1.254625
4	greedy_best_first_graph_search with h_unmet_goals	104	29	0.0677	0.278846154	0.000650962
4	greedy_best_first_graph_search with h_pg_levelsum	104	17	34.719	0.163461538	0.333836538
4	greedy_best_first_graph_search with h_pg_maxlevel	104	56	83.177	0.538461538	0.799778846
4	greedy_best_first_graph_search with h_pg_setlevel	104	107	287.12	1.028846154	2.760769231
4	astar_search with h_unmet_goals	104	34330	62.086	330.0961538	0.596980769
4	astar_search with h_pg_levelsum	104	1208	1975.552	11.61538462	18.99569231
4	astar_search with h_pg_maxlevel	104				
4	astar_search with h_pg_setlevel	104				

Above are the results from tall the tests being run except for A* search with the max level and set level heuristic in problem 4.

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: If we want to use an algorithm that has a few actions and needs to operate in real time, the best option is a greedy best first search that uses the unmet goals heuristic as it consistently has the lowest ratio or time/actions for each problem. See below.

Problem	Search	number of actions	number of new node expansions	time to complete the search plan	ratio nodes expanded / actions	time / actions
1	**greedy_best_first_graph_search with h_unmet_goals**	**20**	**7**	**0.002**	**0.35**	**0.000**
1	depth_first_search	20	21	0.004	1.05	0.000
2	**greedy_best_first_graph_search with h_unmet_goals**	**72**	**17**	**0.02**	**0.236111111**	**0.0002777**
1	breadth_first_search	20	43	0.007	2.15	0.000
3	**greedy_best_first_graph_search with h_unmet_goals**	**88**	**25**	**0.041**	**0.284090909**	**0.0004659**
1	astar_search with h_unmet_goals	20	50	0.01	2.5	0.000
1	uniform_cost_search	20	60	0.011	3	0.000
4	**greedy_best_first_graph_search with h_unmet_goals**	**104**	**29**	**0.0677**	**0.278846154**	**0.0006509**
1	greedy_best_first_graph_search with h_pg_maxlevel	20	6	0.273	0.3	0.013
3	depth_first_search	88	408	1.258	4.636363636	0.014295
1	greedy_best_first_graph_search with h_pg_levelsum	20	6	0.361	0.3	0.018
1	greedy_best_first_graph_search with h_pg_setlevel	20	6	0.479	0.3	0.023

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: If we look at the ratio of nodes expanded per a given action, a greedy best search first approach with a level sum heuristic seems to be the best option. As can been below for a problem with a low number of actions, a greedy best first search with a level sum heuristic has similar efficiency when compared to a greedy best first search with a set level heuristic, but for more actions, we get better efficiency using a level sum heuristic.

Problem	Search	number of actions	number of new node expansions	time to complete the search plan	ratio nodes expanded / actions	time / actions
2	**greedy_best_first_graph_search with h_pg_levelsum**	**72**	**9**	**8.561**	**0.125**	**0.11890277**
2	greedy_best_first_graph_search with h_pg_setlevel	72	9	11.313	0.125	0.15712
3	**greedy_best_first_graph_search with h_pg_levelsum**	**88**	**14**	**19.49**	**0.159090909**	**0.22147727**
4	**greedy_best_first_graph_search with h_pg_levelsum**	**104**	**17**	**34.719**	**0.163461538**	**0.33383653**
2	greedy_best_first_graph_search with h_unmet_goals	72	17	0.02	0.236111111	0.00027777
3	greedy_best_first_graph_search with h_pg_maxlevel	88	21	23.479	0.238636364	0.26680681
4	greedy_best_first_graph_search with h_unmet_goals	104	29	0.0677	0.278846154	0.00065096
3	greedy_best_first_graph_search with h_unmet_goals	88	25	0.041	0.284090909	0.00046590
1	greedy_best_first_graph_search with h_pg_maxlevel	20	6	0.273	0.3	0.0136
1	**greedy_best_first_graph_search with h_pg_levelsum**	**20**	**6**	**0.361**	**0.3**	**0.0180**
1	greedy_best_first_graph_search with h_pg_setlevel	20	6	0.479	0.3	0.0239

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

A: If we are only looking for optimal plans, A* searches are better than greedy best first searches given how it will try to lower the overall cost. Based on the results form the different heuristics with A* search, it seems like A* search with the level sum heuristic seems to find the best balance between ratio of nodes expanded given an action and the time it takes per action.

Problem	Search	number of actions	number of new node expansions	time to complete the search plan	ratio nodes expanded / actions	time / actions
1	astar_search with h_pg_levelsum	20	28	0.918	1.4	0.045
3	astar_search with h_pg_levelsum	88	369	356.814	4.193181818	4.05470454
2	astar_search with h_pg_levelsum	72	357	220.636	4.958333333	3.06438888
4	astar_search with h_pg_levelsum	104	1208	1975.552	11.61538462	18.9956923