Heuristic Analysis

by Jaudat Syed

Performance Comparison:

Problem	Search Method	Expansion	Goal Tests	New Nodes	Plan Length	Time Elapsed
Problem 1	1. breadth_first_search	43	56	180	6	0.0286
Problem 1	3. depth_first_graph_search	21	22	84	20	0.0138
Problem 1	5. uniform_cost_search	55	57	224	6	0.0366
Problem 1	8. astar_search h1	55	57	224	6	0.03557
Problem 1	9. astar_search h_ignore_precon.	41	43	170	6	0.02903
Problem 1	10. astar_search h_pg_levelsum	11	13	50	6	1.885
Problem 2	1. breadth_first_search	3343	4609	30509	9	14.74
Problem 2	3. depth_first_graph_search	624	625	5602	619	4.199
Problem 2	5. uniform_cost_search	4852	4854	44030	9	49.48
Problem 2	8. astar_search h1	4852	4854	44030	9	40.96
Problem 2	9. astar_search h_ignore_precon.	1506	1508	13820	9	11.63
Problem 2	10. astar_search h_pg_levelsum	86	88	841	9	216.88
Problem 3	1. breadth_first_search	14663	18098	129631	12	117.13
Problem 3	3. depth_first_graph_search	408	409	3364	392	1.873
Problem 3	5. uniform_cost_search	18235	18237	159716	12	418.869
Problem 3	8. astar_search h1	18235	18237	159716	12	385.35
Problem	9. astar_search h_ignore_precon.	5118	5120	45650	12	75.025

3						
Problem 3	10. astar_search h_pg_levelsum	408	410	3758	12	1473.82

Analysis:

The worst performing by a long margin was **depth_first_graph_search**, which even though always found a solution the fastest, it was never the optimal one. Of the three non-heuristic search, **breadth_first_search** performed the best, even beating out **uniform_cost_search**. Although both **breadth_first_search** and **uniform_cost_search** both found optimal solutions, **breadth_first_search** was usually faster.
Of the searches that used heuristics, the best was **astar_search h_ignore_preconditions**, which found an optimal solution in less time then any other method of search. This makes sense since **astar_search h_ignore_preconditions** uses a planning graph and ignore preconditions is a good way to make a heuristic function.