Heuristic Analysis

Fernando Maciel Motta

Optimal Choices

For each problem it is possible to identify an optimal choice of heuristics. In the case of the first problem the optimal choice is the greedy_best_first heuristic. It uncovered a minimal path in 0.003 seconds.

In the second and third problems, which were more resource intensive, the depth_first_tree, the depth_limited and the recursive_best_first heuristics timed out and could not be measured. In the second case, once again, the greedy_best_first search heuristic proved to be the optimal search heuristic, finding a minimal path in the least amount of time.

In the third case, the greedy_best_first heuristic was once more the fastest. However, this time it was unable to find a minimal path. This means that the best heuristic for this problem was the astar_search with the h_ignore_preconditions choice heuristic.

Discussion

While they may not always be the fastest, this study shows that some methods reliably find an optimal path in a reasonable amount of time. This dependability makes them very valuable, because one would usually not want to take the risk of obtaining a non-optimal method.

Having said that, the processing time scales quite quickly in this sort of problem. Even more important than that is the fact that the increase in time needed for the simulation varies a lot between the methods. This means

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time (s)
breadth_first_search	43	56	180	6	0.018
breadth_first_tree_search	1458	1459	5960	6	0.512
depth_first_graph_search	12	13	48	12	0.004
depth_limited_search	101	271	414	50	0.048
uniform_cost_search	55	57	224	6	0.020
recursive_best_first_search	4229	4230	17029	6	1.507
greedy_best_first_search	7	9	28	6	0.003
astar_search h1	55	57	224	6	0.019
astar_search h_ignore_pre	41	43	170	6	0.020
astar_search h_pg_levelsum	11	13	50	6	0.565

Tabela 1: Results for Problem 1

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time (s)
breadth_first_search	3401	4672	31049	9	4.386
breadth_first_tree_search	350	351	3142	346	0.779
depth_first_graph_search	–	–	–	–	TIMEOUT
depth_limited_search	–	–	–	–	TIMEOUT
uniform_cost_search	4761	4763	43206	9	6.002
recursive_best_first_search	–	–	–	–	TIMEOUT
greedy_best_first_search	550	552	4950	9	0.687
astar_search h1	4761	4763	43206	9	5.997
astar_search h_ignore_pre	1450	1452	13303	9	2.218
astar_search h_pg_levelsum	86	88	841	9	97.672

Tabela 2: Results for Problem 2

Algorithm	Expansions	Goal Tests	New Nodes	Plan Length	Time (s)
breadth_first_search	14491	17947	128184	12	21.805
breadth_first_tree_search	1948	1949	16253	1878	10.271
depth_first_graph_search	–	–	–	–	TIMEOUT
depth_limited_search	–	–	–	–	TIMEOUT
uniform_cost_search	17783	17785	155920	12	26.638
recursive_best_first_search	–	–	–	–	TIMEOUT
greedy_best_first_search	4031	4033	35794	22	6.042
astar_search h1	17783	17785	155920	12	26.745
astar_search h_ignore_pre	5003	5005	44586	12	8.770
astar_search h_pg_levelsum	311	313	2863	12	579.847

Tabela 3: Results for Problem 3

that methods such as breadth first search, while reliaby giving the optimal answer, may quickly become unfeasible given the CPU power available.

For this reason it is important to study the problem beforehand and evaluate the complexity, so as to choose the search method that fits best.

It is also important to try different approaches in different reduced datasets, similar to the one which is the target, so as to establish the performance of the particular methods in the specific kind of problem one wishes to solve.