Itsuo Okamoto
July 4, 2017

HEURISTIC ANALYSIS

Search Functions/Results

The following search functions are evaluated with run_search.py and the table shows the results.

Non-heuristic Searches:

- breadth-first search
- depth-first graph search
- uniform cost search

Heuristic Searches:

- A* search h_1
- A* search h_ignore_preconditions
- A* search h_pg_levelsum

Search	Problem 1				Problem 2				Problem3			
	Optimality	Plan Length	Elapse Time	Expansion	Optimality	Plan Length	Elapse Time	Expansion	Optimality	Plan Length	Elapse Time	Expansion
breadth-first search	Y	6	0.034	43	Y	9	16.26	3343	Y	12	112.9	14663
depth-first graph search	N	12	0.009	12	N	318	4.23	1288	N	596	3.47	627
uniform cost search	Y	6	0.036	55	Y	9	12.35	4853	Y	12	55.73	18223
A* search h_1	Y	6	0.38	55	Y	9	12.42	4853	Y	12	56.39	18223
A* search h_ignore_preconditions	Y	6	0.039	41	Y	9	4.58	1450	Y	12	18.17	5040
A* search h_pg_levelsum	Y	6	1.029	11	Y	9	202.2	100	Y	12	931.2	318

Non-heuristic Search Comparison

Depth-first graph search is the fastest in terms of elapse time and also it uses the least memory among the above three non-heuristic searches. Depth-first graph search, however, failed to find the optimal path for all three problems while all other searches found it. Since finding the optimal path is important for search functions, depth-first graph search is not recommended.

While breadth-first search and uniform cost search use more memory than depth-first graph search, both of them successfully found the optimal path for all three problems. From performance point of view, uniform cost search is faster than breadth-first search but uniform cost search uses more memory than breadth-first search. If memory usage is estimated by comparing the number of expansions, uniform cost search uses 25% more memory than breadth-first search. While it's significant, uniform cost search gets twice faster than breadth-first search when the problem becomes as complex as problem 3. Therefor, if machine resource is enough, uniform cost search is recommended among non-heuristic searches.

Heuristic Search Comparison

A* search h_ignore_preconditions is the fastest among the above three heuristic searches. A* search h_ignore_preconditions uses more memory than A* search h_pg_levelsum. A* search h_pg_levelsum is extremely slow. While A* search h_ignore_preconditions took 18.17 secs for problem 3, A* search h_pg_levelsum took 931.2 secs. Therefore, unless the machine resource is limited, A* search h_ignore_preconditions is recommended among heuristic search.

Summary

Generally speaking, heuristic searches had less expansions than non-heuristic search so heuristic searches are more memory efficient than non-heuristic searches. In addition to that, A*search h_ignore_preconditions is the fastest among all searches, which have optimality. So, A* search h_ignore_preconditions is most recommended as long as there is enough machine resource to resolve the problem.

If the problem gets more complex and machine resource is limited, A* search h_pg_levelsum may make sense. Though A* search h_pg_levelsum is slow, it is highly memory efficient. So, it would be worth for consideration for such case.