Heuristic Analysis

Natalie Young

Optimal Sequence of Actions for Air Cargo Problems

Optimal Plan for problem 1:

Load(C1, P1, SFO)
Load(C2, P2, JFK)
Fly(P1, SFO, JFK)
Fly(P2, JFK, SFO)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)

Optimal Plan for problem 2:

Load(C1, P1, SFO)
Load(C2, P2, JFK)
Load(C3, P3, ATL)
Fly(P1, SFO, JFK)
Fly(P2, JFK, SFO)
Fly(P3, ATL, SFO)
Unload(C3, P3, SFO)
Unload(C1, P1, JFK)
Unload(C2, P2, SFO)

Optimal plan for problem 3:

Load(C2, P2, JFK)
Load(C1, P1, SFO)
Fly(P2, JFK, ORD)
Load(C4, P2, ORD)
Fly(P1, SFO, ATL)
Load(C3, P1, ATL)
Fly(P1, ATL, JFK)
Unload(C1, P1, JFK)

Unload(C3, P1, JFK)
Fly(P2, ORD, SFO)
Unload(C2, P2, SFO)
Unload(C4, P2, SFO)

Performance of algorithms on Air Cargo Problems

Below table is the comparison of plan length when choosing different non-heuristic searching methods. The ones highlighted in green are the optimal plans; while the ones in red are not efficient.

	breadth_first_search	depth_first_graph_search	uniform_cost_search
Problem 1	6	12	6
Problem 2	9	1444	9
Problem 3	12	571	12

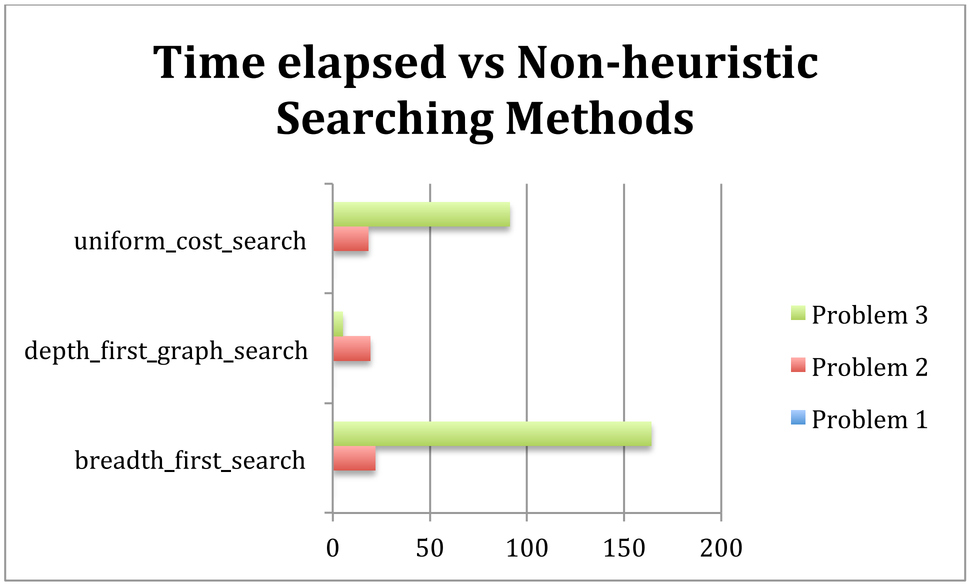

Figure 1

Time elapsed	breadth_first_search	depth_first_graph_search	uniform_cost_search
Problem 1	0.054627484	0.054627484	0.064863021
Problem 2	22.0286862	19.44165249	18.38693293
Problem 3	164.0640457	5.200526744	91.05116056

From the above Figure 1, we can see the differences of elapsed time when using breadth first search, depth first graph search and uniform cost search for problem 1, 2 and 3. The elapsed time is the shortest when using depth first graph search.

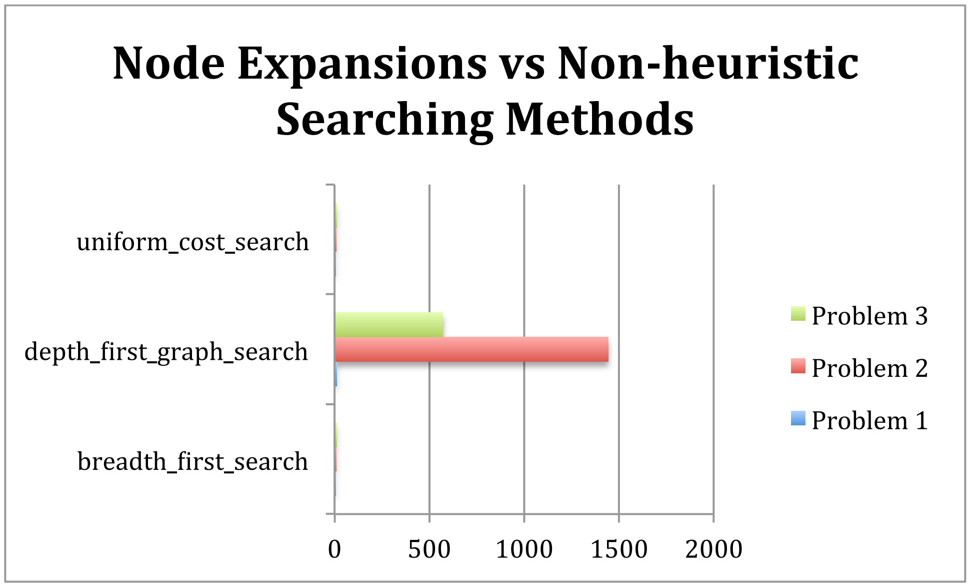

Figure 2

	breadth_first_search	depth_first_graph_search	uniform_cost_search
Problem 1	43	12	55
Problem 2	3343	1669	4853
Problem 3	14663	592	18233

The above Figure 2 shows us the differences of the number of node expansions when using breadth first search, depth first graph search and uniform cost search for problem 1, 2 and 3. The number of node expansions is the smallest when using depth first graph search.

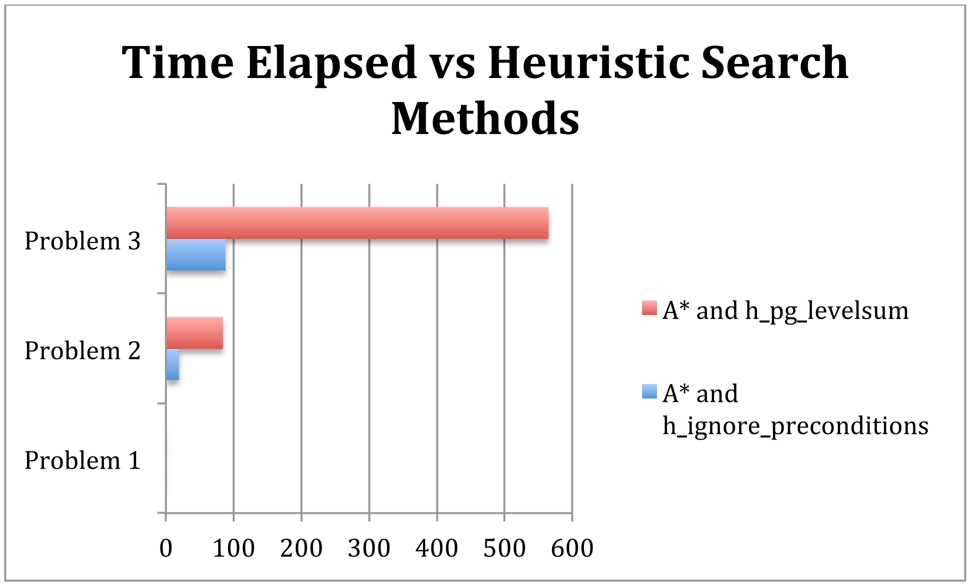

Figure 3

Time elapsed	A* and h_ignore_preconditions	A* and h_pg_levelsum
Problem 1	0.080607418	1.044020006
Problem 2	19.09419798	83.97237894
Problem 3	87.86272059	564.7240924

From the above Figure 3, we can see the differences of elapsed time when using A* with “ignore preconditions” and A* with “level-sum” heuristics for problem 1, 2 and 3. The elapsed time is the shortest when using A* with “ignore preconditions”.

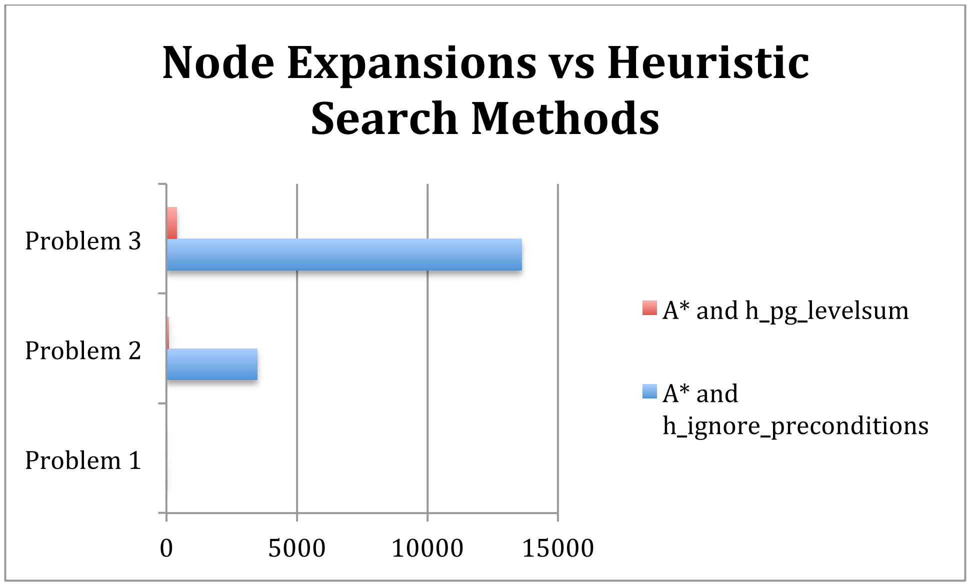

Figure 4

Expansions	A* and h_ignore_preconditions	A* and h_pg_levelsum
Problem 1	42	11
Problem 2	3488	86
Problem 3	13606	400

The above Figure 4 shows us the differences of the number of node expansions when using A* with "ignore preconditions" and A* with "level-sum" heuristics for problem 1, 2 and 3. The number of node expansions is the lowest when using A* with "level-sum".

After compared uninformed and heuristic algorithms, A* with "level-sum" has the lowest node expansions and goal tests in problem 1, 2, 3. And that's because for all uninformed search, it's doing blind search and generating successors and distinguish a goal state from a non-goal state. All search strategies are distinguished by the order in which nodes are expanded, which leads to more node expansion and goal tests.

While heuristic search strategy evaluate nodes by calculating the path cost, which is more efficient. Also in the book of "Artificial Intelligence A Modern Approach", it mentions that an informed search strategy can find solutions more efficiently than can an uniformed strategy.