Heuristic Analysis

Artem Larionov

3/19/2017

Statistics

problem	search	Expansions	Goal_Tests	New_Nodes	Plan_Length	Time
Problem 1	breadth_first_search	43	56	180	6	0.022
Problem 1	depth_first_graph_search	12	13	48	12	0.006
Problem 1	uniform_cost_search	55	57	224	6	0.027
Problem 1	A* h_1	55	57	224	6	0.028
Problem 1	A* h_ignore_preconditions	41	43	170	6	0.022
Problem 1	A* h_pg_levelsum	55	57	224	6	0.750
Problem 2	breadth_first_search	3346	4612	30534	9	10.000
Problem 2	depth_first_graph_search	1124	1125	10017	1085	6.860
Problem 2	uniform_cost_search	4852	4854	44030	9	5.980
Problem 2	A* h_1	4852	4854	44030	9	5.970
Problem 2	A* h_ignore_preconditions	1450	1452	13303	9	1.900
Problem 2	A* h_pg_levelsum	4852	4854	44030	9	66.000
Problem 3	breadth_first_search	14120	17673	124926	12	81.000
Problem 3	depth_first_graph_search	677	678	5608	660	2.600
Problem 3	uniform_cost_search	18235	18237	159716	12	24.860
Problem 3	A* h_1	18235	18237	159716	12	24.600
Problem 3	A* h_ignore_preconditions	5040	5042	44944	12	7.200
Problem 3	A* h_pg_levelsum	18235	18237	159716	12	303.000

Conclusions

- Depth-first Graph Search can be fast but usually it finds not optimal plan.
- Breadth-first Search finds an optimal plan, but explore a lot of nodes, which leads to slower execution time.
- The faster speed of DFS comparing to BFS is due to less number of expansions and nodes tested, it's also the reason why DFS can't guarantee the optimality of the solution, which BFS can.
- Uniform Cost Search, A* with h_1 and h_pg_levelsum heuristics have almost the same metrics.
- A* with h_pg_levelsum is the slowest algorithm among all variants though.
- A* with h_ignore_preconditions heuristic has the best metrics and finds an optimal plan.
- h_ignore_preconditions allows A* to minimize number of expansions and tests, hence to find an optimal solution much faster than other heuristics and uninform search algorithms.

The drawback of depth-first search is that it can get stuck going down the wrong path. Depth-first search will either get stuck in an infinite loop and never return a solution, or it may eventually find a solution path that is longer than the optimal solution.[1]

If there is a solution, breadth-first search is guaranteed to find it, and if there are several solutions, breadth-first search will always find the shallowest goal state first. In terms of the four criteria, breadth-first search is complete, and it is optimal provided the path cost is a nondecreasing function of the depth of the node. The memory requirements are a bigger problem for breadth-first search than the execution time.[2]

That A* search is complete, optimal, and optimally efficient among all such algorithms is rather satisfying.

[1] AIMA, Chapter 3. Solving Problems by Searching, Depth-first search
[2] AIMA, Chapter 3. Solving Problems by Searching, Breadth-first search

Unfortunately, it does not mean that A* is the answer to all our searching needs. It has been shown that exponential growth will occur unless the error in the heuristic function grows no faster than the logarithm of the actual path cost.[3]

Path Plans

Problem 1

Breadth First Search	Depth First Graph Search	Uniform Cost Search
Load(C2, JFK, P2)	Fly(P1, SFO, JFK)	Load(C1, SFO, P1)
Load(C1, SFO, P1)	Fly(P2, JFK, SFO)	Load(C2, JFK, P2)
Fly(P2, JFK, SFO)	Load(C1, SFO, P2)	Fly(P1, SFO, JFK)
Unload(C2, P2, SFO)	Fly(P2, SFO, JFK)	Fly(P2, JFK, SFO)
Fly(P1, SFO, JFK)	Fly(P1, JFK, SFO)	Unload(C1, P1, JFK)
Unload(C1, P1, JFK)	Unload(C1, P2, JFK)	Unload(C2, P2, SFO)
	Fly(P2, JFK, SFO)	
	Fly(P1, SFO, JFK)	
	Load(C2, JFK, P1)	
	Fly(P2, SFO, JFK)	
	Fly(P1, JFK, SFO)	
	Unload(C2, P1, SFO)	

A* h_1	A* h_ignore_preconditions	A* h_pg_levelsum
Load(C1, SFO, P1)	Load(C1, SFO, P1)	Load(C1, SFO, P1)
Load(C2, JFK, P2)	Fly(P1, SFO, JFK)	Load(C2, JFK, P2)
Fly(P1, SFO, JFK)	Unload(C1, P1, JFK)	Fly(P1, SFO, JFK)
Fly(P2, JFK, SFO)	Load(C2, JFK, P2)	Fly(P2, JFK, SFO)
Unload(C1, P1, JFK)	Fly(P2, JFK, SFO)	Unload(C1, P1, JFK)
Unload(C2, P2, SFO)	Unload(C2, P2, SFO)	Unload(C2, P2, SFO)

Problem 2

Breadth First Search	Depth First Graph Search	Uniform Cost Search
Load(C1, SFO, P1)	1085 steps	Load(C1, SFO, P1)
Load(C2, JFK, P2)		Load(C2, JFK, P2)
Load(C3, ATL, P3)		Load(C3, ATL, P3)
Fly(P1, SFO, JFK)		Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)		Fly(P2, JFK, SFO)
Fly(P2, JFK, SFO)		Fly(P3, ATL, SFO)
Unload(C2, P2, SFO)		Unload(C3, P3, SFO)
Fly(P3, ATL, SFO)		Unload(C2, P2, SFO)
Unload(C3, P3, SFO)		Unload(C1, P1, JFK)

A* h_1	A* h_ignore_preconditions	A* h_pg_levelsum
Load(C1, SFO, P1)	Load(C3, ATL, P3)	Load(C3, ATL, P3)
Load(C2, JFK, P2)	Fly(P3, ATL, SFO)	Fly(P3, ATL, SFO)
Load(C3, ATL, P3)	Unload(C3, P3, SFO)	Unload(C3, P3, SFO)

[3]AIMA, Chapter 4. Informed Search Methods, Proof of the optimality of A*

A* h_1	A* h_ignore_preconditions	A* h_pg_levelsum
Fly(P1, SFO, JFK)	Load(C2, JFK, P2)	Load(C1, SFO, P1)
Fly(P2, JFK, SFO)	Fly(P2, JFK, SFO)	Load(C2, JFK, P2)
Fly(P3, ATL, SFO)	Unload(C2, P2, SFO)	Fly(P2, JFK, SFO)
Unload(C3, P3, SFO)	Load(C1, SFO, P1)	Unload(C2, P2, SFO)
Unload(C2, P2, SFO)	Fly(P1, SFO, JFK)	Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)	Unload(C1, P1, JFK)	Unload(C1, P1, JFK)

Problem 3

Breadth First Search	Depth First Graph Search	Uniform Cost Search
Load(C1, SFO, P1)	660 steps	Load(C1, SFO, P1)
Load(C2, JFK, P2)		Load(C2, JFK, P2)
Fly(P1, SFO, ATL)		Fly(P1, SFO, ATL)
Load(C3, ATL, P1)		Load(C3, ATL, P1)
Fly(P2, JFK, ORD)		Fly(P2, JFK, ORD)
Load(C4, ORD, P2)		Load(C4, ORD, P2)
Fly(P1, ATL, JFK)		Fly(P2, ORD, SFO)
Unload(C1, P1, JFK)		Fly(P1, ATL, JFK)
Unload(C3, P1, JFK)		Unload(C4, P2, SFO)
Fly(P2, ORD, SFO)		Unload(C3, P1, JFK)
Unload(C2, P2, SFO)		Unload(C2, P2, SFO)
Unload(C4, P2, SFO)		Unload(C1, P1, JFK)

A* h_1	A* h_ignore_preconditions	A* h_pg_levelsum
Load(C1, SFO, P1)	Load(C2, JFK, P2)	Load(C1, SFO, P1)
Load(C2, JFK, P2)	Fly(P2, JFK, ORD)	Load(C2, JFK, P2)
Fly(P1, SFO, ATL)	Load(C4, ORD, P2)	Fly(P1, SFO, ATL)
Load(C3, ATL, P1)	Fly(P2, ORD, SFO)	Load(C3, ATL, P1)
Fly(P2, JFK, ORD)	Unload(C4, P2, SFO)	Fly(P2, JFK, ORD)
Load(C4, ORD, P2)	Load(C1, SFO, P1)	Load(C4, ORD, P2)
Fly(P2, ORD, SFO)	Fly(P1, SFO, ATL)	Fly(P2, ORD, SFO)
Fly(P1, ATL, JFK)	Load(C3, ATL, P1)	Unload(C4, P2, SFO)
Unload(C4, P2, SFO)	Fly(P1, ATL, JFK)	Fly(P1, ATL, JFK)
Unload(C3, P1, JFK)	Unload(C3, P1, JFK)	Unload(C3, P1, JFK)
Unload(C2, P2, SFO)	Unload(C2, P2, SFO)	Unload(C2, P2, SFO)
Unload(C1, P1, JFK)	Unload(C1, P1, JFK)	Unload(C1, P1, JFK)