Planning Search Heuristics Analysis

Xiaoyang Yao

non-heuristic search result

Non-heuristic search doesn't use any information about the "goodness" of a searching state. They just simply check whether the current state is the goal state. Thus, one advantage of non-heuristic search is they don't investigate extra resource to estimate the goodness of states. The tables below compared 7 non-heuristic search strategies in terms of optimality (plan length and optimality), time complexity (execution time) and space complexity (node expanded). For each property, best option is labeled by bold. Some search strategies are skipped in problem 2 and 3 because the execution time is longer than 20 minutes.

1. Problem 1

Search Strategy	Plan length	Execution time (s)	Node expanded	Optimality
Breadth First Search	**6**	0.036	43	**Yes**
Breadth First Tree Search	**6**	1.02	1458	**Yes**
Depth First Graph Search	20	0.014	21	No
Depth Limited Search	50	0.09	101	No
Uniform Cost Search	**6**	0.043	55	**Yes**
Recursive Best First Search	**6**	2.84	4229	**Yes**
Greedy Best First Graph Search	**6**	**0.005**	**7**	**Yes**

2. Problem 2

Search Strategy	Plan length	Execution time (s)	Node expanded	Optimality
Breadth First Search	**9**	13.6	3343	**Yes**
Breadth First Tree Search	-	-	-	-
Depth First Graph Search	619	3.44	**624**	No
Depth Limited Search	50	929.4	222719	No
Uniform Cost Search	**9**	12.56	4852	**Yes**
Recursive Best First Search	-	-	-	-
Greedy Best First Graph Search	17	**2.53**	990	No

3. Problem 3

Search Strategy	Plan length	Execution time (s)	Node expanded	Optimality
Breadth First Search	**12**	104.88	14663	**Yes**
Breadth First Tree Search	-	-	-	-
Depth First Graph Search	392	**1.79**	**408**	No
Depth Limited Search	-	-	-	-
Uniform Cost Search	**12**	56.54	18235	**Yes**
Recursive Best First Search	-	-	-	-
Greedy Best First Graph Search	22	16.96	5614	No

Among the 7 non-heuristic strategies, Breadth First Search and Uniform Cost Search are the only two that generate optimal plans within 20 minutes. These two algorithms are guaranteed to find the shortest path in searching (Figure 3.21). [1] In problem 3, Depth First Graph Search uses both the least execution time and expanded nodes. In problem 1 and 2, its execution time and expanded nodes are either least or close to least. This indicates that Depth First Graph Search is the fastest and uses the least memory. However, it doesn't generate an optimal plan for any of the 3 problems, which is generally more crucial than speed and memory.

Criterion	Breadth-First	Uniform-Cost	Depth-First	Depth-Limited	Iterative Deepening	Bidirectional (if applicable)
Complete?	Yes[a]	Yes[a,b]	No	No	Yes[a]	Yes[a,d]
Time	$O(b^d)$	$O(b^{1+\lfloor C^*/\epsilon \rfloor})$	$O(b^m)$	$O(b^\ell)$	$O(b^d)$	$O(b^{d/2})$
Space	$O(b^d)$	$O(b^{1+\lfloor C^*/\epsilon \rfloor})$	$O(bm)$	$O(b\ell)$	$O(bd)$	$O(b^{d/2})$
Optimal?	Yes[c]	Yes	No	No	Yes[c]	Yes[c,d]

Figure 3.21 Evaluation of tree-search strategies. b is the branching factor; d is the depth of the shallowest solution; m is the maximum depth of the search tree; l is the depth limit. Superscript caveats are as follows: [a] complete if b is finite; [b] complete if step costs $\geq \epsilon$ for positive ϵ; [c] optimal if step costs are all identical; [d] if both directions use breadth-first search.

heuristic search result

Heuristic search strategies take into consideration the goodness of a state when choosing the next step of searching. A* Search always generates an optimal path if the heuristic is admissible, since it's a Breadth First Search-based search strategy (AIMA 3.5.2). [1] The tables below compared 3 heuristic search strategies.

1. Problem 1

Search Strategy	Plan length	Execution time (s)	Node expanded	Optimality
A* Search with h1 heuristic	**6**	0.039	55	**Yes**
A* Search with Ignore Preconditions heuristic	**6**	**0.029**	41	**Yes**
A* Search with Level Sum heuristic	**6**	0.526	**11**	**Yes**

2. Problem 2

Search Strategy	Plan length	Execution time (s)	Node expanded	Optimality
A* Search with h1 heuristic	**9**	12.92	4852	**Yes**
A* Search with Ignore Preconditions heuristic	**9**	**3.95**	1450	**Yes**
A* Search with Level Sum heuristic	**9**	42.54	**86**	**Yes**

3. Problem 3

Search Strategy	Plan length	Execution time (s)	Node expanded	Optimality
A* Search with h1 heuristic	**12**	55.19	18235	**Yes**
A* Search with Ignore Preconditions heuristic	**12**	**15.84**	5040	**Yes**
A* Search with Level Sum heuristic	**12**	216.32	**325**	**Yes**

As expected, all 3 strategies generate optimal plan for all 3 problems. Among the 3 different heuristics, Ignore Preconditions heuristic uses the least execution time, while Level Sum heuristic uses the least memory.

Optimal plan for each problem

The table below shows the optimal search strategy, optimal plan, and reason for choosing this strategy for each problem.

Problem	Problem 1	Problem 2	Problem 3
Optimal strategy	Greedy Best First Graph Search	A* Search with Ignore Preconditions heuristic	A* Search with Ignore Preconditions heuristic
Optimal plan	Load(C1, P1, SFO) Load(C2, P2, JFK) Fly(P1, SFO, JFK) Fly(P2, JFK, SFO) Unload(C1, P1, JFK) Unload(C2, P2, SFO)	Load(C3, P3, ATL) Fly(P3, ATL, SFO) Unload(C3, P3, SFO) Load(C1, P1, SFO) Fly(P1, SFO, JFK) Unload(C1, P1, JFK) Load(C2, P2, JFK) Fly(P2, JFK, SFO) Unload(C2, P2, SFO)	Load(C2, P2, JFK) Fly(P2, JFK, ORD) Load(C4, P2, ORD) Fly(P2, ORD, SFO) Unload(C4, P2, SFO) Load(C1, P1, SFO) Fly(P1, SFO, ATL) Load(C3, P1, ATL) Fly(P1, ATL, JFK) Unload(C3, P1, JFK) Unload(C2, P2, SFO) Unload(C1, P1, JFK)
Reason	Generates an optimal plan with the least execution time and least memory space.	Generates an optimal plan with least execution time and relatively small memory space. In this problem as well as problem 3, there is a trade-off between time complexity and space complexity. Generally speaking, time complexity weighs more than space complexity. In these two cases, Ignore Preconditions heuristic is $\sim 10^1$ times faster than Level Sum heuristic. Level Sum heuristic uses $\sim 10^1$ times less spaces. Thus, Ignore Preconditions heuristic is preferred.	Same as Problem 2.

The experiment results indicate that heuristic search is **not always better** than non-heuristic search. For problems with small searching space (small branching factor and/or shallow depth) like Problem 1, it might not worth investigating extra resources to improve the efficiency of searching. However, for more complex problems like Problem 2 and 3, searching efficiency becomes more important.

The best strategy overall for the cargo problem is **A* Search with Ignore Preconditions heuristic.**

Reference

[1]. Stuart J. Russell, Peter Norvig (2010), Artificial Intelligence: A Modern Approach (3rd Edition).