Non-heuristic planning searches

air_cargo_p1

	breadth_first_search	**depth_first_graph_search**	**uniform_cost_search**
Node expansions	43	12	55
Number of goal tests	56	13	57
Number of new nodes	180	48	224
Time elapsed	0.09597467813346049	0.03175109856746723	0.12417614847362035
Optimality	6 (yes)	12 (no)	6 (yes)

air_cargo_p2

	breadth_first_search	**depth_first_graph_search**	**uniform_cost_search**
Node expansions	3343	1669	4852
Number of goal tests	4609	1670	4854
Number of new nodes	30509	14863	44030
Time elapsed	39.84646643102561	38.27383534860826	27.098313837703174
Optimality	9 (yes)	1444 (no)	9 (yes)

air_cargo_p3

	breadth_first_search	**depth_first_graph_search**	**uniform_cost_search**
Node expansions	14663	592	18235
Number of goal tests	18098	593	18237
Number of new nodes	129631	4927	159716
Time elapsed	367.5679160225517	10.030126605382083	145.11125012888883
Optimality	12 (yes)	571 (no)	12 (yes)

A* planning searches

air_cargo_p1

	astar_search h1	astar_search h_ignore_preconditions	astar_search h_pg_levelsum
Node expansions	55	41	11
Number of goal tests	57	43	13
Number of new nodes	224	170	50
Time elapsed	0.1179017805820126	0.12336153151149373	1.3945349552454498
Optimality	6 (yes)	6 (yes)	6 (yes)

air_cargo_p2

	astar_search h1	astar_search h_ignore_preconditions	astar_search h_pg_levelsum
Node expansions	4852	1450	86
Number of goal tests	4854	1452	88
Number of new nodes	44030	13303	841
Time elapsed	37.17039677218893	11.854786032677124	114.4765078611327
Optimality	9 (yes)	9 (yes)	9 (yes)

air_cargo_p3

	astar_search h1	astar_search h_ignore_preconditions	astar_search h_pg_levelsum
Node expansions	18235	5040	325
Number of goal tests	18237	5042	327
Number of new nodes	159716	44944	3002
Time elapsed	148.07765567302494	42.123683506692934	576.972911418109
Optimality	12 (yes)	12 (yes)	12 (yes)

Provide an optimal plan for Problems 1, 2, and 3.

Problem 1	Problem 2	Problem 3
- Load(C1, P1, SFO) - Fly(P1, SFO, JFK) - Load(C2, P2, JFK) - Fly(P2, JFK, SFO) - Unload(C1, P1, JFK) - Unload(C2, P2, SFO)	- Load(C1, P1, SFO) - Fly(P1, SFO, JFK) - Load(C2, P2, JFK) - Fly(P2, JFK, SFO) - Load(C3, P3, ATL) - Fly(P3, ATL, SFO) - Unload(C3, P3, SFO) - Unload(C2, P2, SFO) - Unload(C1, P1, JFK)	- Load(C2, P2, JFK) - Fly(P2, JFK, ORD) - Load(C4, P2, ORD) - Fly(P2, ORD, SFO) - Load(C1, P1, SFO) - Fly(P1, SFO, ATL) - Load(C3, P1, ATL) - Fly(P1, ATL, JFK) - Unload(C4, P2, SFO) - Unload(C3, P1, JFK) - Unload(C2, P2, SFO) - Unload(C1, P1, JFK)

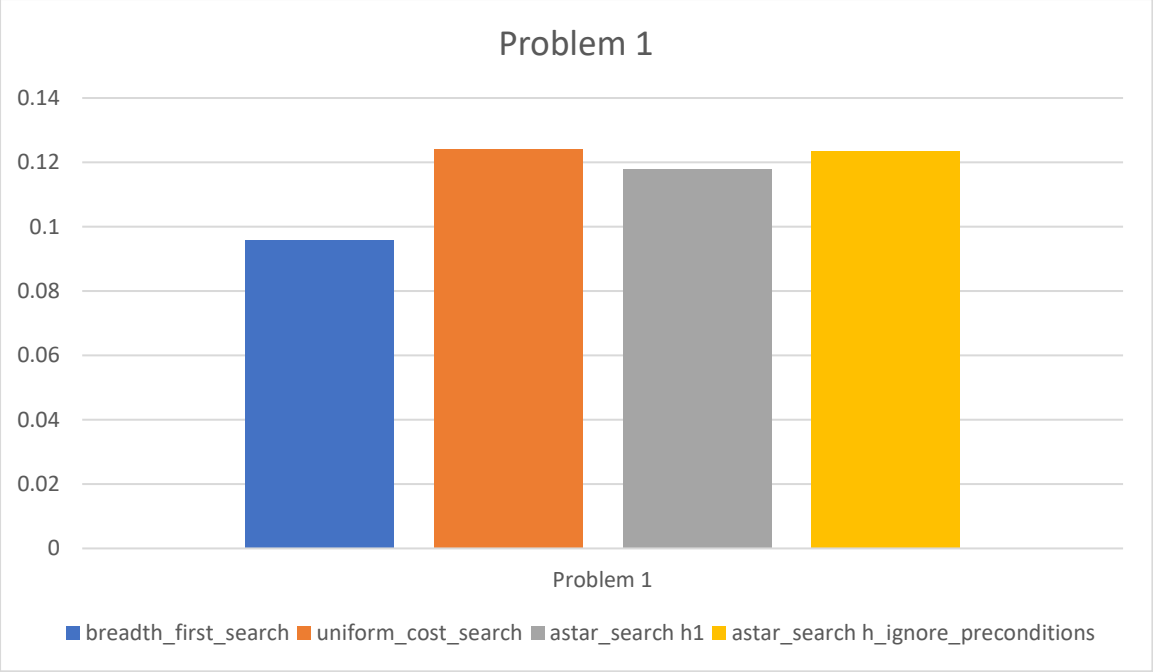
What was the best heuristic used in these problems? Was it better than non-heuristic search planning methods for all problems? Why or why not?

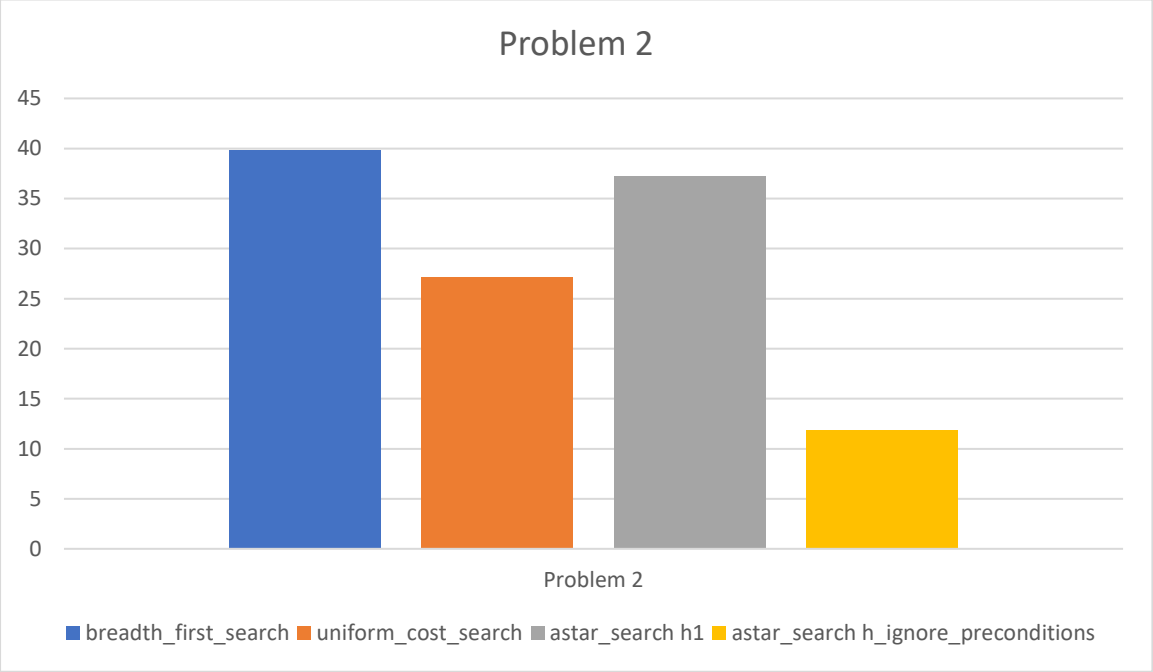
The best heuristic used was the "ignore preconditions" heuristic because it found the optimal solution at a shorter time. It was better than non-heuristic search planning methods for all problems except breadth_first_search and depth_first_graph_search. As the problem got more complex in Problems 2 and 3, the "ignore preconditions" heuristic was able to find the optimal solution and at a much faster time than the non-heuristics search planning methods.

Depth first search does not provide an optimal plan for the current problem set because there are multiple optimal solutions and the number of steps in the optimal solution is not too far from the start state. Therefore, DFS finds a much more complicated solution by expanding deeper into the search tree instead of considering alternative successors at each level of the search tree.

The fastest uninformed algorithm is DFS while the shortest is BFS. DFS is the fastest because it expands the least number of nodes and has the least goal tests. However, the trade-off is a solution that is not optimal. BFS is the slowest because it expands a lot more nodes than DFS and UCS. Due to this, memory requirements and time is of increasing concern for BFS as the depth of the solution increases (AIMA 3rd Edition, pg. 83)

The fastest heuristic search is "ignore preconditions". This is likely due to the fact that the value can be easily calculated with reference to the current state. On the other hand, the slowest is "level sum" despite expanding the least nodes. This is probably because the level sum requires more time and space to calculate.


Problem 1
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
Problem 1
breadth_first_search
uniform_cost_search
astar_search h1
astar_search h_ignore_preconditions


Problem 2
45
40
35
30
25
20
15
10
5
0
Problem 2
breadth_first_search
uniform_cost_search
astar_search h1
astar_search h_ignore_preconditions

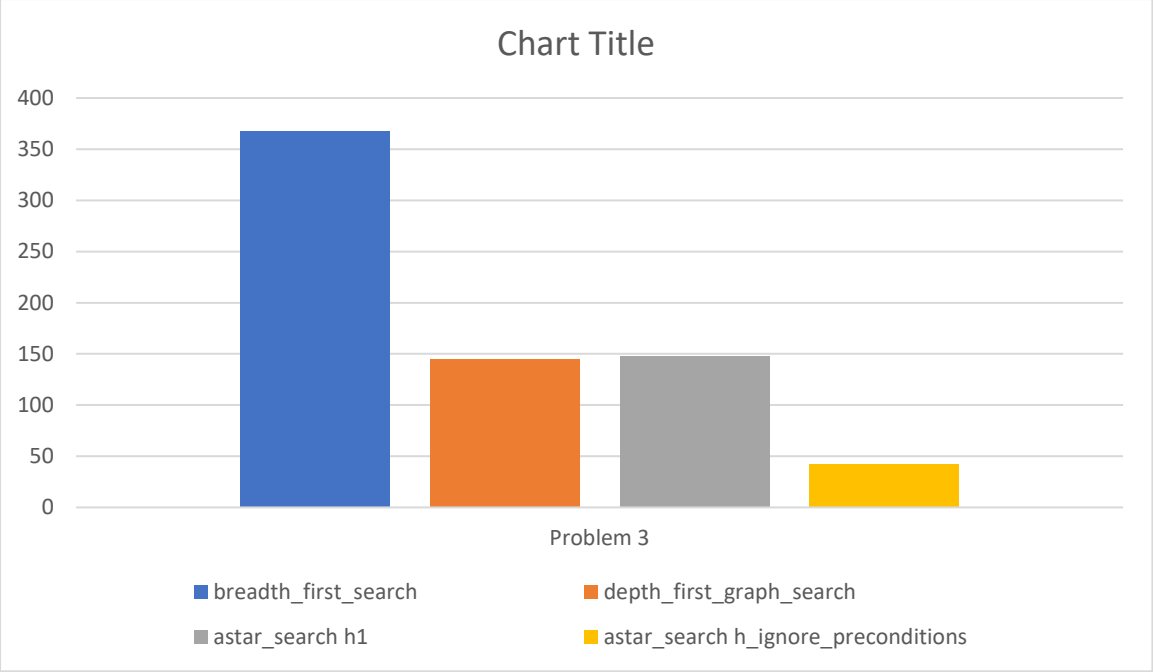


Chart Title
400
350
300
250
200
150
100
50
0
Problem 3
breadth_first_search
depth_first_graph_search
astar_search h1
astar_search h_ignore_preconditions