Heuristic Analysis

1. Introduction

In this project, Implement a Planning Search, three air cargo problems are given:

Problem	Number of cargos	Number of planes	Number of airports	Number of state fluent	Space size
1	2	2	2	12	2^{12}
2	3	3	3	27	2^{27}
3	4	4	4	32	2^{32}

Table 1. Air Cargo Problems

This project is to define the problems in classical Planning Domain Definition Language (PDDL), implement planning graph for automatically generated heuristics, and set up the problems for searching for the goals using different search algorithms.

2. Test Results

After the problems were set up and planning graph was implemented, different search algorithms were run and tested. Here, the results of five search algorithms are present. The search algorithms are:
- Breadth first search (breadth_first_search)
- Depth first graph search (depth_first_graph_search)
- Uniform cost search (uniform_cost_search)
- A* search with ignore preconditions heuristic (astar_search_with_hip)
- A* search with planning graph level sum heuristic (astar_search_with_hpg)

Breadth first search, depth first graph search and uniform cost search are uninformed search algorithms. A* search is an informed search algorithm, which uses problem-specific knowledge, heuristic, as the guidance to search the goal. Heuristic can be generated in different ways. It can be generated by ignoring preconditions in planning problems: each goal can be regarded achievable by one step. Heuristic can also be generator automatically by using planning graph, which is a polynomial-size approximation of search tree representing all possible path.

In the tests, several metrics are examined: number of expanded nodes, number of goal tests, number of new nodes, time consumption, and path length.

The test results for solving problem 1 are presented in the table below:

Search algorithm	Node expansion	Goal tests	New node	Time (second)	Path
breadth_first_search	43	56	189	0.03524	6
depth_first_graph_search	21	22	84	1.10439	20
uniform_cost_search	55	57	224	0.04382	6
astar_search_with_hip	41	43	170	0.03698	6
astar_search_with_hpg	11	13	50	0.76845	6

Table 2. Problem 1 Test Results

Problem 2 test results are display below:

Search algorithm	Node expansion	Goal tests	New node	Time (second)	Path
breadth_first_search	3346	4612	30534	9.03184	9
depth_first_graph_search	107	108	959	0.35133	105
uniform_cost_search	4853	4855	44041	13.16218	9
astar_search_with_hip	1450	1452	13303	3.81081	9
astar_search_with_hpg	86	88	841	74.33849	9

Table 3. Problem 2 Test Results

Problem 3 test results are:

Search algorithm	Node expansion	Goal tests	New node	Time (second)	Path
breadth_first_search	14120	17673	124926	43.31415	12
depth_first_graph_search	292	293	2388	1.27109	288
uniform_cost_search	18223	18225	159618	54.83142	12
astar_search_with_hip	5040	5042	44944	16.0117	12
astar_search_with_hpg	325	327	3002	361.4829	12

Table 4. Problem 3 Test Results

First, let's compare the uninformed search algorithms. From the test results, we can see that depth first search was always the first one to return results. It always expands the deepest node in the frontier and returns once the goal is found. However, the result of search, Path, is the worst. Depth first search is not optimal and it also can be incomplete if there is a loop.

Both breadth first search and uniform cost search produce the same path length in all tests. Breadth first search checked fewer nodes and returned the result quicker. Therefore, breadth first search was the best uninformed algorithm in these tests.

Now, let's compare the informed search algorithms. In the tests, both A* search algorithms returned the same path length. When using level-sum heuristic generated by planning graph, the search algorithm checked much fewer nodes. This shows that level-sum heuristic provides knowledge which better presents the problem and current state in the problem. However, it took a long time for the algorithm to return the result. This is due to the complexity of planning graph and the search algorithm spent a lot of time generating the graph so as to obtain the heuristic.

The search algorithm with ignore precondition heuristic checked far more nodes. However, as the heuristic was easy to compute, it returned result very quickly. Therefore, A* search with ignore precondition heuristic was the best infomed algorithm in the tests.

Comparing A* search with ignore precondition heuristic and breadth first search, we can see that, when facing simple problems, such as problem 1, both performed similarly in all metrics. However, as problems become more and more complicated, A* search with ignore precondition heuristic started to outperform breadth first search. In problems 2 and 3, A* search with ignore precondition heuristic visited less than half of nodes and returned results in less than half of time in comparison.

Finally, these test results show that informed algorithms with proper and balanced heuristic perform better in this type of planning problems.