**Shriram Jaju**

AIND Project 3: Implement a Planning Search

**Air Cargo Planning Heuristic Analysis**

The following tables show the results gathered after solving the air cargo problems for this project with both uninformed and heuristic based search. The goal of this analysis is to document the results obtained from each search type and find an optimal solution for each air cargo problem, that is; a search algorithm that finds the lowest path among all possible paths from start to goal.

For each set of problems, the optimal and fastest solution has been highlighted with green color.

For each problem I ran 4 uninformed non-heuristic planning searches and 3 domain independent heuristic A\* searches. The information about No. of nodes expanded, goal tests, new nodes created and time elapsed (in sec.) for each search has been recorded in below tables.

**Problem 1:**



**Problem 2:**



**Problem 3:**



**Analysis:**

Non-heuristic search:

From the above table we can see that BFS is optimal and guaranteed to find shortest path but as complexity increases (Problem 2 and 3) it becomes less optimal since it has exponential space complexity. DFS finds solution in short time but it lacks optimality due to linear space complexity. As you can see in above table, its plan length is always longest. Uniform cost is very much similar to BFS except in UCS we expand node with smallest path cost, which means UCS is equivalent to BFS if all the costs are equal hence UCS does more work by expanding nodes at depth d unnecessarily.

Heuristic Search:

Heuristic based search did perform better as the problem complexity increased. This is more evident in the air cargo problem 2 and 3, where the “A\* Search with ‘h\_ignore\_preconditions’” performance was optimal and the fastest amongst those that were optimal. ‘levelsum’ heuristic expands less and gives optimal result but takes more time compared to ‘ignore precondition’ heuristic as it is inadmissible i.e. it could overestimate the optimal cost from the current state and the goal state but it largely works well for problems that are largely decomposable.

BFS gives optimal results with simple problem such as problem 1 but as complexity increases it becomes slow and that is where we require heuristics to optimize our search, and ‘h\_ignore\_preconditions’ is found to optimal in my test results.

**Optimal Sequence of Actions:**

The following table describes an optimal sequence of actions to solve each of the air cargo problems provided using the highlighted approaches from the tables above.

|  |  |  |
| --- | --- | --- |
| **Problem** | **Search Type** | **Optimal Sequence Action** |
| **Air Cargo Problem 1** | breadth\_first\_search | Load(C1, P1, SFO) Load(C2, P2, JFK) Fly(P2, JFK, SFO) Unload(C2, P2, SFO) Fly(P1, SFO, JFK) Unload(C1, P1, JFK) |
| **Air Cargo Problem 2** | astar\_search with h\_ignore | Load(C1, P1, SFO) Load(C2, P2, JFK) Load(C3, P3, ATL) Fly(P2, JFK, SFO) Unload(C2, P2, SFO) Fly(P1, SFO, JFK) Unload(C1, P1, JFK) Fly(P3, ATL, SFO) Unload(C3, P3, SFO) |
| **Air Cargo Problem 3** | astar\_search with h\_ignore | 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(C1, P1, JFK) Unload(C2, P2, SFO) |

**References:**

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