# Introduction:

# Problem Setting:

We want to fly cargos from some airports to other airports. To fly a cargo, we need to first have a plane at the same airport of the cargo. To fly a plane we need to load the cargo into the plane. When the plane reach destination, the cargo is unloaded. Hence, to get a cargo from airport A to airport C, where A different from C, the sequence of actions are as follows:

1. Load cargo into a plane at airport A
2. Fly plane to airport C
3. Unload cargo at Airport C.

From this sequence of steps to getting a cargo from starting point to destination, one can assume that if 3 actions are required to move one cargo, than it will take at least 3\*n actions to move n cargo. This is used in calculating the constant heuristic in A\* search.

For all these problems, we first ran non-heuristic searches: Breadth First Search, Depth First Graph Search, and Uniform Cost Search, then we ran A\* Search with three heuristics: constant heuristic, ignore precondition heuristic, and Planning Graphs Levelsum heuristic. (We will come back to them later.)

## Problem 1

In problem 1, we want to move 2 cargos (C1, and C2) from SFO, and JFK, respectively, to JFK, and SFO using 2 planes. Both planes are located at each Cargo initial location

|  |  |
| --- | --- |
|  |  |

As we have one plane for each cargo, we can state that it will take at most 6 actions to move the 2 cargos.

## Problem 2

In problem2, we want to move 3 cargos using 3 planes located in the same airports as the cargos to 3 distinct airports. The graph in the left shows the initial conditions where C1 and P1 are in SFO, C2 and P2 are in JFK, and C3 and P3 are in ATL. The graph in the right shows the final destination of the cargos: C1 ends up in JFK, C2 and C3 ends up in SFO.

|  |  |
| --- | --- |
|  |  |

## Problem 3:

Problem 3 has more cargos to move but uses only 2 planes to move them, and some cargos have the same destination: C1 and C2 destination is JFK, while C2 and C4 destination is SFO.

|  |  |
| --- | --- |
|  |  |

# Results

## Non Heuristic Searches:

For all three problems, we ran Breadth First Search, BFS, Depth First Graph Search, DFGS, and Uniform Cost Search (UCS). For all these searches we measured the number of expansions, number of goal test, number of new nodes created, the plan length, and time it took to reach the goal.

From Table 1, DFGS produced the least number of expansions, goal tests, number of new nodes, and was faster than both BFS, and UCS for problems 2 and 3—see orange numbers under DFGS. However, it had a big flaw, its plan length was extremely long compared to BFS and UCS in problem 2 and 3. In problem 2, DFGS produced a plan of length of 575 versus the 9 produced by BFS and UCS; in problem 3, DFGS produced a plan of length of 596 versus 12 produced by BFS and UCS. While DFGS was faster than BFS and UCS in problem 2 and problem 3, its outcome was undesirable. This outcome is expected, as DFGS tends to revisits paths and hence is suboptimal in graph search.

One has to understand, the reason of UCS underperforming BFS in time, has to do with the fact that while BFS will stop when it finds the goal (or destination), UCS will continue looking for other paths that may be shorter than the path already found. In here, it is up to the practitioner to choose the search given the desired outcome. In these three problems, BFS would have been adequate.

### Table 1. Non-Heuristic Searches Performances

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Description | Problems | BFS | DFGS | UCS |
| Expansions | **Problem 1** | **43** | 1,458 | 55 |
|  | **Problem 2** | 3,343 | **582** | 4,852 |
|  | **Problem 3** | 14,663 | **627** | 18,223 |
| Goal Tests | **Problem 1** | **56** | 1,459 | 57 |
|  | **Problem 2** | 4,609 | **583** | 4,854 |
|  | **Problem 3** | 18,098 | **628** | 18,225 |
| New Nodes | **Problem 1** | **180** | 5,960 | 224 |
|  | **Problem 2** | 30,509 | **5,211** | 44,030 |
|  | **Problem 3** | 129,631 | **5,176** | 159,618 |
| Plan Length | **Problem 1** | 6 | 6 | 6 |
|  | **Problem 2** | 9 | 575 | 9 |
|  | **Problem 3** | 12 | 596 | 12 |
| Time | **Problem 1** | **0.03** | 0.97 | 0.044 |
|  | **Problem 2** | 18.92 | **4.66** | 56.3 |
|  | **Problem 3** | 134.9 | **4.37** | 489.96 |

## Heuristic Searches:

We decided solve our three problems using A\* Search under various heuristics: constant heuristic, A\* h\_1, ignore preconditions heuristics, A\* h\_IP, and planning graphs level sum heuristic, A\*h\_PG\_L. As expected, all the heuristics resulted in the same plan length for all three problems. However, A\* h\_IP was the fastest: 0.034 versus (0.52 and 4.26), 15.59 versus (56.74 and 2394.74), and 104.13 versus (480.49 and 8102.65.) Still, while in problem 1 and 2 it produced the least of expansions or performed few goal tests, and created least number of nodes than A\*h\_1 and A\*h\_PG\_L exp, in problem 3, it produced more expansions, performed more goal tests, and created more new nodes than A\*h\_PG\_L. On the other hand, A\*h\_1, on all three problems, produced the highest number of expansions and new nodes, and passed through goal tests.

In short, the heuristic search for all our 3 problems is A\* h\_IP. It got the optimal plan length at a reasonable time.

Table . A\* Heuristic Searches Performances

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Descriptions | Problems | A\* h\_1 | A\* h\_IP | A\* h\_PG\_L |
| Expansions | **Problem 1** | 55 | **41** | 47 |
|  | **Problem 2** | 4,852 | **1,506** | 2,061 |
|  | **Problem 3** | 18,223 | 5,118 | **4,293** |
| Goal Tests | **Problem 1** | 57 | **43** | 49 |
|  | **Problem 2** | 4,854 | **1,508** | 2,063 |
|  | **Problem 3** | 18,225 | 5,120 | **4,295** |
| New Nodes | **Problem 1** | 224 | **170** | 196 |
|  | **Problem 2** | 44,030 | **13,820** | 19,112 |
|  | **Problem 3** | 159,618 | 45,650 | **40,036** |
| Plan Length | **Problem 1** | 6 | 6 | 6 |
|  | **Problem 2** | 9 | 9 | 9 |
|  | **Problem 3** | 12 | 12 | 12 |
| Time | **Problem 1** | 0.53 | **0.034** | 4.26 |
|  | **Problem 2** | 56.74 | **15.59** | 2,394.74 |
|  | **Problem 3** | 480.49 | **104.13** | 8,102.65 |

## Comparing Heuristic and Non Heuristic Searches:

A\*h\_IP over performed all the non-heurist searches—excluding DFGS. It had the lowest expansions, goal tests, new nodes created, and time lower than BFS. A\*h\_1 performance was equivalent to UCS, which is expected as h\_1 is a constant heuristic, and uniform cost search assume constant, or equal distance between 2 nodes. This shows, that using right heuristic one can speed up search and still find the shortest plan length.

# Conclusion:

The purpose of this report is to evaluate heuristic and non-heuristic searchs for our air cargo-planning problem. We created 3 problems with different degrees of difficulties going from the trivial problem--where one has 2 cargos, 2 planes at the same location as the cargos, but each cargo is at different airplane—to the most complicated one—where we have 4 cargos but only 2 planes, 4 starting airports but only 2 destination airports.

For the non-heuristic searches, we chose Breath First Seach, Depth First Graph Search, and Uniform Cost Search. For the heuristic searches, we chose A\* h1, A\*h1 Ignore Precondition, and A\* Planning Graph Level Sum. For the non-heuristic, it came apparent DFS failed at coming with a desirable path, and was inadequate as a search for our planning problem.

BSF gave the best performance in time and resulted in the reached one possible short path, but not the optimal path. UCS was able to find the optimal path but with more time. When compared to the heuristic searches, A\* h\_IP was able to find the optimal path but at the fraction of time of UCS. Hence, A\*h\_IP is the search to use in air cargo planning.