Heuristic Analysis

This document presents a detailed analysis of the heuristics developed during the project. Each heuristic was executed 10 times and the mean values of each execution will be presented and analyzed, due to the great discrepancy between the values found the data will be presented in logarithmic scale. The results will be analyzed under the metrics listed below.

- Time (s)
- Expansions
- Goal Test
- New Nodes
- Plan Length

The following algorithms were analyzed:

- breadth_first_search
- depth_first_graph_search
- greedy_best_first_graph_search
- astar_search with h_ignore_preconditions
- astar_search with h_pg_levelsum

The problem domain use the same action schema as [1], This allows the comparison of the results.

```
Air Cargo Action Schema:
‘‘‘
Action(Load(c, p, a),
PRECOND: At(c, a)  At(p, a)  Cargo(c)  Plane(p)  Airport(a)
EFFECT:  At(c, a)  In(c, p))
Action(Unload(c, p, a),
PRECOND: In(c, p)  At(p, a)  Cargo(c)  Plane(p)  Airport(a)
EFFECT: At(c, a)   In(c, p))
Action(Fly(p, from, to),
PRECOND: At(p, from)  Plane(p)  Airport(from)  Airport(to)
EFFECT:  At(p, from)  At(p, to))
‘‘‘
```

1 Problems

Three problems were solved in this study:

```
- Problem 1 initial state and goal:
‘‘‘
Init(At(C1, SFO)  At(C2, JFK)
 At(P1, SFO)  At(P2, JFK)
 Cargo(C1)  Cargo(C2)
 Plane(P1)  Plane(P2)
 Airport(JFK)  Airport(SFO))
Goal(At(C1, JFK)  At(C2, SFO))
‘‘‘

- Problem 2 initial state and goal:
‘‘‘
Init(At(C1, SFO)  At(C2, JFK)  At(C3, ATL)
 At(P1, SFO)  At(P2, JFK)  At(P3, ATL)
 Cargo(C1)  Cargo(C2)  Cargo(C3)
 Plane(P1)  Plane(P2)  Plane(P3)
 Airport(JFK)  Airport(SFO)  Airport(ATL))
Goal(At(C1, JFK)  At(C2, SFO)  At(C3, SFO))
‘‘‘

- Problem 3 initial state and goal:
‘‘‘
Init(At(C1, SFO)  At(C2, JFK)  At(C3, ATL)  At(C4, ORD)
 At(P1, SFO)  At(P2, JFK)
 Cargo(C1)  Cargo(C2)  Cargo(C3)  Cargo(C4)
 Plane(P1)  Plane(P2)
 Airport(JFK)  Airport(SFO)  Airport(ATL)  Airport(ORD))
Goal(At(C1, JFK)  At(C3, JFK)  At(C2, SFO)  At(C4, SFO))
‘‘‘
```

The problems vary in their order of complexity and have optimal solutions of 6, 9 and 12 respectively. Just as the state space size is 2^{12}, 2^{27} and 2^{32}.

2 Results

The graphical analysis allowed the visualization of the results of the execution of the algorithms from the metric of the analysis of the execution time, the Figure 1 contain theses results. From this analysis the exponential performance of the A * algorithm becomes clear as the complexity of the problem increases. However, the possibility of adding heuristics during the execution of A * makes possible a great improvement in performance as can be seen by h_ignore_preconditions

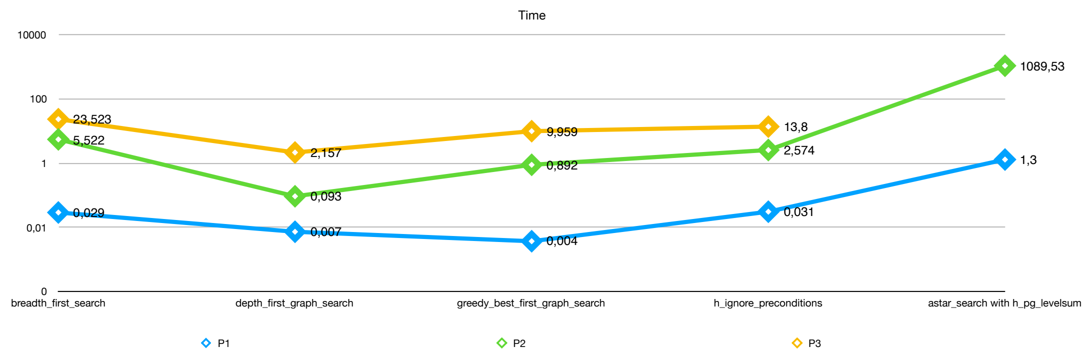

Figure 1: Time Results

However, the simple analysis of the execution time does not allow a real evaluation of the performance of the algorithm. Therefore, other metrics should be taken into consideration during such analysis. Two important metrics are the number of expansions (Figure 2) and new nodes (Figure 3, these metrics are directly related to the memory consumption and the execution time.

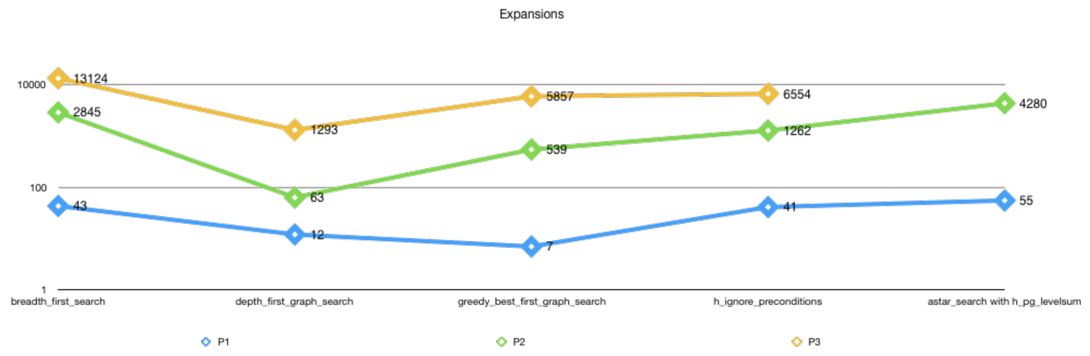

Figure 2: Expansion Results

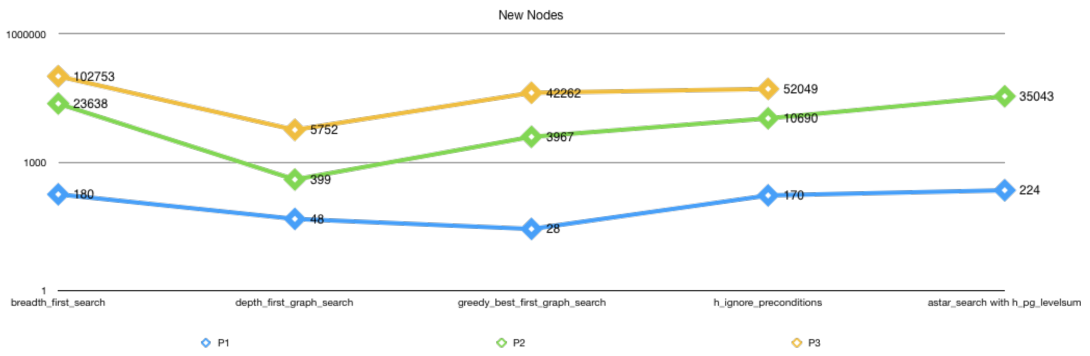

Figure 3: New Nodes Results

Based on the values found in the two metrics evaluated so far, the depth_first_graph_search algorithm has presented as the best solution for the problems used as an evaluation parameter. This is due to the fact that it has an extremely fast processing time for all problems and a low memory consumption because it requires few expansions to find the solution.

We also analyzed the test number to find the solution of the three problems analyzed in this review. Again the depth_first_graph_search presented the smallest test values as seen in Figure 4

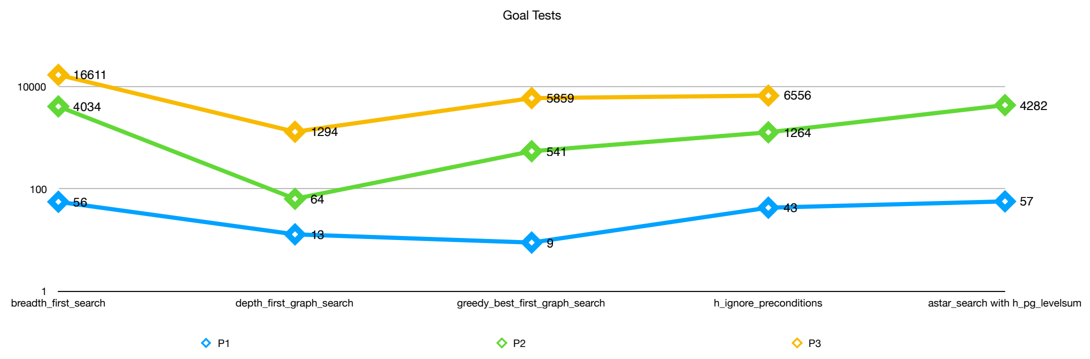

Figure 4: Goal Tests Results

Thus, among breadwinners analyzed breadth_first_search was the best choice in a scenario where the complexity of the problem is unknown. Otherwise, situations where there is previous knowledge of complexity and knowing that it is low greedy_best_first_graph_search algorithm is recommended because it used a reduced number of tests and expansions to find the optimal solution in a similar time.

Among the heuristics the results of h_ignore_preconditions preconditions make it the best choice for any scenario with a lower memory consumption and execution time.

References

[1] Richard Waldinger. Achieving several goals simultaneously**the research reported herein was sponsored by the national science foundation under grant gj-36146. In Bonnie Lynn Webber and Nils J. Nilsson, editors, *Readings in Artificial Intelligence*, pages 250 – 271. Morgan Kaufmann, 1981.