# Heuristic Analysis

## Optimal Plans

I worked out following planning solutions

|  |  |
| --- | --- |
| Problem # | Plan |
| 1 | Load(C1, P1, SFO)  Fly(P1, SFO, JFK)  Load(C2, P2, JFK)  Fly(P2, JFK, SFO)  Unload(C1, P1, JFK)  Unload(C2, P2, SFO) |
| 2 | 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) |
| 3 | 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) |

## Analysis

The performance of the following searches/heuristics were analysed:

|  |  |  |
| --- | --- | --- |
| Name | Search | Heuristic |
| breadth\_first | Breadth First | None |
| depth\_first\_graph | Depth First Graph | None |
| uniform\_cost\_search | Uniform Cost Search | None |
| h\_ignore\_preconditions | A\*-Search | Ignore-Preconditions |
| h\_pg\_levelsum | A\*-Search | Levelsum |

Following tables compare the performance of these searches once by calculating the “optimality” using “Nr Node Expansions” and once by calculating the optimality using “Time Elapsed”.

The optimality is calculated by comparing each individual search with the hypothetical “best search”. The best search combines the best result of each individual search into once hypothetical best search.

The Optimality is defined by

whereas N is “Nr node expansion” or “Time Elapsed”, and P is “Plan Length”. P is to the power of 2 to give a small plan length a higher weight.

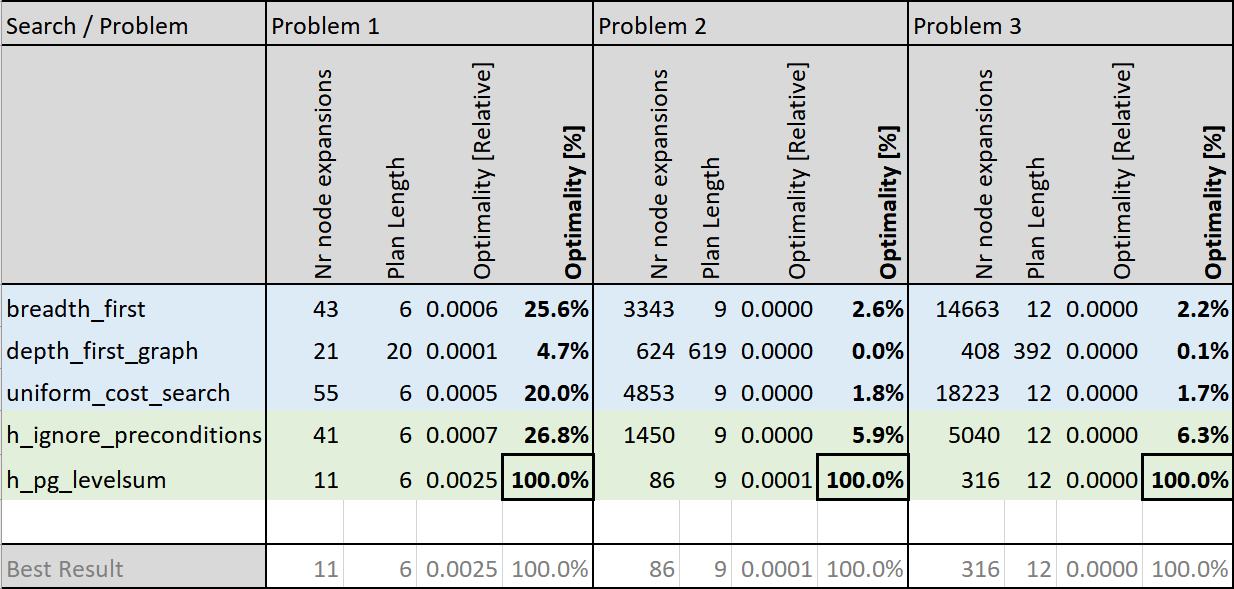


Figure 1: Optimality based on "nr nodes expansion" (ideal for memory constraint environments)

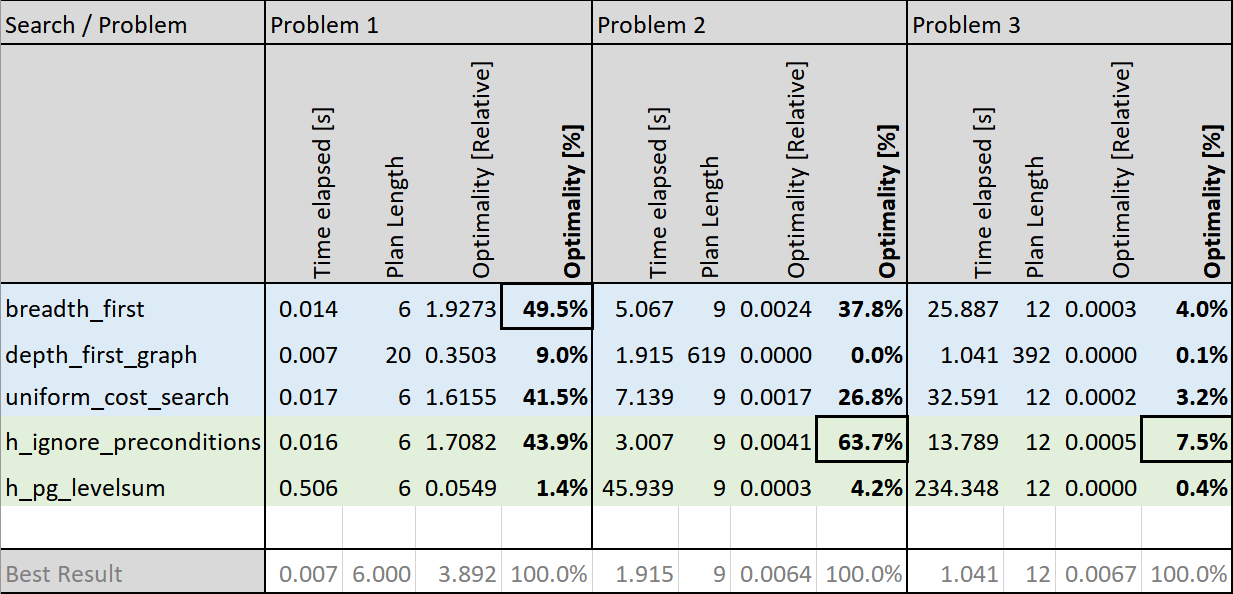


Figure 2: Optimality based on "time elapsed" (ideal for CPU constraint environments)

Figure : Scatterplot "time elapsed" vs. "nr nodes expanded" for Problem 1

Figure : Scatterplot "time elapsed" vs. "nr nodes expanded" for Problem 2

Figure : Scatterplot "time elapsed" vs. "nr nodes expanded" for Problem 3

## Discussion

The analysis showed that depending on whether we care about “nr node expansions” or “time elapsed” different searches are optimal. A\*-Search with Plangraph Levelsum heuristics is optimal if we want to minimize the number of expanded nodes. A\*-Search with Ignore Preconditions heuristics is optimal if we instead optimize time elapsed. Note: Though time elapsed is the wall time I expect the CPU time not to be hugely different. Measuring CPU-cycles instead wall time would further clarify that claim.

I conclude:

1. Depth-First-Graph Search without heuristic doesn’t work at all.
2. Non-heuristic search is fast and yields good results for very easy problems. See where Breadth First and Uniform Cost Search are close to Ignore-Preconditions.
3. To make it more efficient a good heuristic is needed (Russel & Norvig, 2016, p. 376). See and where a heuristic search is better independent of whether time elapsed or nr nodes expanded is important.
4. A\*-search with the Plangraph Levelsum heuristic is as expected giving good results even it’s inadmissible. A Plangraph increases the accuracy of the estimates (Russel & Norvig, 2016, p. 379) which can be seen in the minimal number of nodes expanded.
5. A\*-search with the Ignore Preconditions heuristic is quicker but less accurate as the Plangraph Levelsum heuristic. More nodes are therefore expanded but overall less CPU cycles are needed. This might be because no Plangraph needs to be constructed.