Heuristic Analysis

This heuristic analysis first displays potential optimal solutions for the problems that are being solved. The problems are in the area of air cargo traffic and are meant to be solved with either uninformed planning algorithms or with A\* search that applies automatic heuristics. Secondly, the performances of the different algorithms are displayed and later on compared qualitatively.

# Optimal Sequences

Problem 1:

1. Load(C1, P1, SFO)
2. Load(C2, P2, JFK)
3. Fly(P2, JFK, SFO)
4. Unload(C2, P2, SFO)
5. Fly(P1, SFO, JFK)
6. Unload(C1, P1, JFK)

Problem 2:

1. Load(C1, P1, SFO)
2. Load(C2, P2, JFK)
3. Load(C3, P3, ATL)
4. Fly(P2, JFK, SFO)
5. Unload(C2, P2, SFO)
6. Fly(P1, SFO, JFK)
7. Unload(C1, P1, JFK)
8. Fly(P3, ATL, SFO)
9. Unload(C3, P3, SFO)

Problem 3:

1. Load(C1, P1, SFO)
2. Fly(P1, SFO, ATL)
3. Load(C3, P1, ATL)
4. Fly(P1, ATL, JFK)
5. Load(C2, P1, JFK)
6. Unload(C1, P1, JFK)
7. Unload(C3, P1, JFK)
8. Fly(P1, JFK, ORD)
9. Load(C4, P1, ORD)
10. Fly(P1, ORD, SFO)
11. Unload(C2, P1, SFO)
12. Unload(C4, P1, SFO)

# Performances of Algorithms

Here the performances are being presented. “Expansions” shows how many times a node has been expanded. “Goal Tests” shows how many times it has been tested if we reached goal. “New Nodes” displays how many new nodes have been generated during the search. “Plan Length” is the length of the final plan, that was identified by the algorithm. “Time” is the time it took for the algorithm to complete. “Optimality” is a Boolean that describes if one of the optimal paths has been found.

Table 1 Problem 1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Searches | Expansions | Goal Tests | New Nodes | Plan Length | Time (s) | Optimality |
| breadth\_first\_search | 43 | 56 | 180 | 6 | 0.03 | True |
| depth\_first\_graph\_search | 21 | 22 | 84 | 20 | 0.015 | False |
| uniform\_cost\_search | 55 | 57 | 224 | 6 | 0.046 | True |
| astar\_search with h\_1 | 55 | 57 | 224 | 6 | 0.041 | True |
| astar\_search with h\_ignore\_preconditions | 41 | 43 | 170 | 6 | 0.045 | True |
| astar\_search with h\_pg\_levelsum | 11 | 13 | 50 | 6 | 1.129 | True |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Searches | Expansions | Goal Tests | New Nodes | Plan Length | Time (s) | Optimality |
| breadth\_first\_search | 3343 | 4609 | 30509 | 9 | 15.291 | True |
| depth\_first\_graph\_search | 624 | 625 | 5602 | 619 | 3.59 | False |
| uniform\_cost\_search | 4852 | 4854 | 44030 | 9 | 13.902 | True |
| astar\_search with h\_1 | 4852 | 4854 | 44030 | 9 | 13.247 | True |
| astar\_search with h\_ignore\_preconditions | 1450 | 1452 | 13303 | 9 | 4.833 | True |
| astar\_search with h\_pg\_levelsum | 86 | 88 | 841 | 9 | 179.781 | True |

Table 2 Problem 2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Searches | Expansions | Goal Tests | New Nodes | Plan Length | Time(s) | Optimality |
| breadth\_first\_search | 13063 | 16544 | 102013 | 12 | 86.775 | True |
| depth\_first\_graph\_search | 1292 | 1293 | 5744 | 875 | 3.526 | False |
| uniform\_cost\_search | 17014 | 17016 | 132148 | 12 | 51.51 | True |
| astar\_search with h\_1 | 17014 | 17016 | 132148 | 12 | 51.817 | True |
| astar\_search with h\_ignore\_preconditions | 6531 | 6533 | 51879 | 13 | 23.499 | False |

Table 3 Problem 3

# Analysis

I think the two most prominent facts are that breadth\_first\_search and uniform\_cost\_search always find an optimal solution, but depth\_first\_graph\_search does not. Depth\_first\_search is immensely faster in finding a solution though. There exists a trade-off. Depending on what is more important, a quick solution or an optimal one, either depth\_first or the other two are better.

These results are expected as depth\_first\_search just follows one branch all the way down until it finds the goal or it goes on to the next branch. There is no guarantee that an optimal solution will be found (Russell and Norvig, 2002). On the other hand, as breadth\_first\_search goes through level by level, it is guaranteed to find an optimal solution but at a timely cost (Russell and Norvig, 2002).

For problem 1 and 2 the uninformed algorithms performed better slightly better. It can be argued that with easier problems that are lower in complexity, there is no benefit from the complexity reducing methods like A\* with the heuristics. The additional computations are not worth the investment. We can see though that with increasing complexity, the algorithms with heuristics perform better than the simpler uninformed methods. Interestingly enough, the A\*\_seach\_with\_ignore\_preconditions did not find the optimal path by 1, but the it was twice as fast. For more complex problems, applying this algorithm is suggested. For simpler problems, uniform\_cost\_search is generally performing better than breadth\_first\_search, at least time wise and it is advised to use it.

# References

Russell, S., Norvig, P., 2002. Artificial Intelligence: A Modern Approach (2nd Edition). Prentice Hall.