|  |  |  |
| --- | --- | --- |
| Problem 1 | Problem 2 | Problem 3 |
| Load(C2, P2, JFK)  Load(C1, P1, SFO)  Fly(P2, JFK, SFO)  Unload(C2, P2, SFO)  Fly(P1, SFO, JFK)  Unload(C1, P1, JFK) | Load(C1, P1, SFO)  Fly(P1, SFO, JFK)  Unload(C1, P1, JFK)  Load(C2, P2, JFK)  Fly(P2, JFK, SFO)  Unload(C2, P2, SFO)  Load(C3, P3, ATL)  Fly(P3, ATL, SFO)  Unload(C3, P3, 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)  Load(C2, P2, JFK)  Fly(P2, JFK, ORD)  Load(C4, P2, ORD)  Fly(P2, ORD, SFO)  Unload(C4, P2, SFO)  Unload(C2, P2, SFO) |

# Optimal Plans

# Algorithms Stats

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Metric/Algorithm** | **BF** | **BF(T)** | **DF(G)** | **DL** | **UC** | **RBF (h=1)** | **GBF (G, h=1)** | **A\* (h=1)** | **A\* (no precon)** | **A\* (level sum)** |
| **Problem #1** | | | | | | | | | | |
| **Expansions** | 43 | 1458 | 12 | 101 | 55 | 4229 | 7 | 55 | 41 | 11 |
| **Time** | 0.030 | 0.934 | 0.008 | 0.088 | 0.038 | 2.700 | 0.005 | 0.039 | 0.037 | 11.131 |
| **Goal Tests** | 56 | 1459 | 13 | 271 | 57 | 4230 | 9 | 57 | 43 | 13 |
| **New Nodes** | 180 | 5960 | 48 | 414 | 224 | 17029 | 28 | 224 | 170 | 50 |
| **Plan length** | 6 | 6 | 12 | 50 | 6 | 6 | 6 | 6 | 6 | 6 |
| **Problem #2** | | | | | | | | | | |
| **Expansions** | 3343 | #N/A | 582 | 222719 | 4604 | #N/A | 455 | 4604 | 1310 | 74 |
| **Time** | 3.965 | #N/A | 0.741 | 1036.228 | 4.239 | #N/A | 0.814 | 4.314 | 3.448 | 7.581 |
| **Goal Tests** | 4609 | #N/A | 583 | 2053741 | 4606 | #N/A | 457 | 4606 | 1312 | 76 |
| **New Nodes** | 30509 | #N/A | 5211 | 2054119 | 41828 | #N/A | 4095 | 41828 | 11979 | 720 |
| **Plan length** | 9 | #N/A | 575 | 50 | 9 | #N/A | 16 | 9 | 9 | 9 |
| **Problem #3** | | | | | | | | | | |
| **Expansions** | 14663 | #N/A | 627 | #N/A | 16963 | #N/A | 3998 | 16963 | 4444 | 229 |
| **Time** | 18.028 | #N/A | 0.775 | #N/A | 14.928 | #N/A | 3.962 | 15.808 | 7.524 | 28.264 |
| **Goal Tests** | 18098 | #N/A | 628 | #N/A | 16965 | #N/A | 4000 | 16965 | 4446 | 231 |
| **New Nodes** | 129631 | #N/A | 5176 | #N/A | 149136 | #N/A | 35002 | 149136 | 39227 | 2081 |
| **Plan length** | 12 | #N/A | 596 | #N/A | 12 | #N/A | 30 | 12 | 12 | 13 |

# Analysis

Observation 3)

Breadth-first tree search never returned for slightly more complex Problems 2 and 3. The reason is that tree search doesn’t prevent algorithms from exploring loop paths forever. In this experiment, breadth-first tree search didn’t return after running for 2 hours on a decent hardware.

Observation 4)

Depth-first Graph Search happened for find solution fast, but this solution is very suboptimal. [TODO: Is it guaranteed to find a solution?]

Observation 5)

Breadth-first time and expansions grow exponentially [TODO: From what?]

Observation 6)

UCS and BFS are very comparable, because for these specific problems all actions have the same fixed cost. So, depth of path and cost of path are essentially the same. Slight difference in number of expanded nodes, goal tests etc. between these algorithms is determined by when they do Goal-Test. UCS does goal test at the time of Popping from frontier, while BFS does it at the time of Expansion. Therefore BFS has to examine less nodes. Generally this can cause BFS to return suboptimal plan, but in this case, when all actions have the same cost, it will always return optimal plan.

Observation 7)

When problem became big enough Problem #3, DLS didn’t return value [TODO: Why exactly?]

Non-informed conclusion: BFS is the best non-informed search algorithm for these specific problems. It examines/expands least number of nodes and guarantees to return optimal solution.

Informed search:

Observation 1)

A\* with h(s) == constant is actually the same as Uniform Cost Search. Both, UCS and A\* belong to the “best-first” class of search. So when h(s) == constant evaluation functions for UCS and A\* become the same for the purposes of “best-first” algorithm, since they are different only by a constant.

Observation 2)

A\* (no precond) is faster than A\* (level sum), but it expands so many more nodes. So, if node expansion is an expensive operation, A\* (level sum) would actually run faster. Also, A\* (level sum) returned suboptimal plan for Problem #3. It found plan of length 13, while the actual optimal plan is 12 steps. The reason is that “ignore preconditions” heuristic makes problem strictly more relaxed than the original problem, so this heuristic is admissible. It’s also consistent (non-decreasing) because [TODO: Why is it consistent?]. Level sum heuristic may not be admissible, and therefore A\* based on this heuristic might return suboptimal plan.

Why GBF (G, h=1) is not the same as DF (G). If it’s using f(s) = h(s) = 1, it should essentially be the same

[TODO: Contrast GBF and RBF]

[TODO: Learn more about A\* no precond and A\* level sum]

research

STRIPS  
GraphPlan  
HSP(Heuristic Search Planner)