AIND-Planning Project

# Optimal Plans

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

* Compare and contrast non-heuristic search result metrics (optimality, time elapsed, number of node expansions) for Problems 1,2, and 3. Include breadth-first, depth-first, and at least one other uninformed non-heuristic search in your comparison; Your third choice of non-heuristic search may be skipped for Problem 3 if it takes longer than 10 minutes to run, but a note in this case should be included.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Search | expansions | goal tests | new nodes | plan length | time |
| breadth\_first\_search | 43 | 56 | 180 | 6 | 0.2525 |
| breadth\_first\_tree\_search | 1458 | 1459 | 5960 | 6 | 10.6024 |
| depth\_first\_graph\_search | 21 | 22 | 84 | 20 | 0.1643 |
| depth\_limited\_search | 101 | 271 | 414 | 50 | 0.5415 |
| uniform\_cost\_search | 55 | 57 | 224 | 6 | 0.2941 |

In problem 1, depth\_first variants failed to deliver optimal plans, even though they perform well in term expansions, goal tests and new nodes. Overall, breadth\_first\_search is the winning search (optimal, fast).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Search | expansions | goal tests | new nodes | plan length | time |
| breadth\_first\_search | 3343 | 4609 | 30509 | 9 | 79.2501 |
| depth\_first\_graph\_search | 624 | 625 | 5602 | 619 | 17.2183 |
| uniform\_cost\_search | 4852 | 4854 | 44030 | 9 | 119.1123 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Search | expansions | goal tests | new nodes | plan length | time |
| breadth\_first\_search | 14663 | 19098 | 129631 | 12 | 538.4931 |
| depth\_first\_graph\_search | 408 | 409 | 3364 | 392 | 12.5751 |
| uniform\_cost\_search | 18223 | 18225 | 159618 | 12 | 661.3171 |

We see similar results for problems 2 and 3. Depth\_first explored the fewest numbers of nodes, which in turn had the shortest execution time, however, failed to deliver optimal plans. In problem 2, the returned plan's length was 619, comprised of many repeated steps. Breadth\_first\_search overall outperforms and delivers optimal plans. This illustrates the advantage of BFS over DFS in smaller problems, where a solution is relatively near the root, but DFS get stuck exploring the leftmost branches. We could see that there's a need for 'smarter' (heuristic-based) searches since execution time is getting too long.

* Compare and contrast heuristic search result metrics using A\* with the "ignore preconditions" and "level-sum" heuristics for Problems 1, 2, and 3.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Problem 1** | expansions | goal tests | new nodes | plan length | time |
| recursive\_best\_first\_search h\_1 | 4229 | 4230 | 17023 | 6 | 21.8154 |
| greedy\_best\_first\_graph\_search h\_1 | 7 | 9 | 28 | 6 | 0.0371 |
| astar\_search h\_1 | 55 | 57 | 224 | 6 | 0.2855 |
| astar\_search h\_ignore\_preconditions | 41 | 43 | 170 | 6 | 0.4099 |
| astar\_search h\_pg\_levelsum | 11 | 13 | 50 | 6 | 0.6041 |
| **Problem 2** | expansions | goal tests | new nodes | plan length | time |
| greedy\_best\_first\_graph\_search h\_1 | 990 | 992 | 8910 | 21 | 30.644 |
| astar\_search h\_1 | 4852 | 4854 | 44030 | 9 | 153.2734 |
| astar\_search h\_ignore\_preconditions | 1450 | 1452 | 13303 | 9 | 43.2556 |
| astar\_search h\_pg\_levelsum | 86 | 88 | 841 | 9 | 10.21 |
| **Problem 3** | expansions | goal tests | new nodes | plan length | time |
| greedy\_best\_first\_graph\_search h\_1 | 5578 | 5580 | 49150 | 22 | 193.103 |
| astar\_search h\_1 | 18223 | 18225 | 159618 | 12 | 665.9928 |
| astar\_search h\_ignore\_preconditions | 5040 | 5042 | 44944 | 12 | 175.4908 |
| astar\_search h\_pg\_levelsum | 325 | 327 | 3002 | 12 | 44.5952 |

Most of the search strategies deliver optimal plans (except for greedy\_best\_first\_graph in Problems 2 and 3). The recursive\_best\_first strategy reached optimal plan, but had prohibitive running time and node expansions (relatively).

Among the A\*, we can see a clear pattern of improvement from h\_1 to h\_ignore\_preconditions to h\_pg\_levelsum. This reflects the degrees of admissibility among the heuristics (i.e., h\_pg\_levelsum dominates h\_ignore\_preconditions, in turn, dominates h\_1, almost always). However, there is a reverse trend in execution time in problem 1. This is possibly due to the increase in heuristic complexity outweighing the numbers of nodes explored as these are small.

* What was the best heuristic used in these problems? Was it better than non-heuristic search planning methods for all problems? Why or why not?

A\* with pg\_levelsum heuristics outperform everything else, shortest execution time in general, fewest nodes explored, and deliver optimal plans.

Note: Some searches were not performed due to prohibitive execution time (hours+)

Reference

Peter Norvig, Stuart Russell. Artificial Intelligence: A Modern Approach, 3rd ed., 2009.