1. **An optimal plan for Problem 1, 2, 3**

**An optimal plan for Problem 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)

**An optimal plan for Problem 2:**

Load(C2, P2, JFK)

Fly(P2, JFK, SFO)

Load(C1, P1, SFO)

Fly(P1, SFO, JFK)

Load(C3, P3, ATL)

Fly(P3, ATL, SFO)

UnLoad(C3, P3, SFO)

UnLoad(C1, P1, JFK)

UnLoad(C2, P2, SFO)

**An optimal plan for Problem 3:**

Load(C1, P1, SFO)

Load(C2, P2, JFK)

Fly(P1, SFO, ATL)

Load(C3, P1, ATL)

Fly(P2, JFK, ORD)

Load(C4, P2, ORD)

Fly(P2, ORD, SFO)

Fly(P1, ATL, JFK)

UnLoad(C4, P2, SFO)

UnLoad(C3, P1, JFK)

UnLoad(C1, P1, JFK)

UnLoad(C2, P2, SFO)

1. **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**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Problem** | **Search Algorithm** | **Expansions** | **Goal Tests** | **New Nodes** | **Time consuming** | **Plan length** | **Result** |
| Problem 1 | breadth\_first\_search | 43 | 56 | 180 | 0.036 | 6 | Load(C2, P2, JFK)  Load(C1, P1, SFO)  Fly(P2, JFK, SFO)  UnLoad(C2, P2, SFO)  Fly(P1, SFO, JFK)  UnLoad(C1, P1, JFK) |
| Problem 2 | breadth\_first\_search | 3343 | 4609 | 30509 | 7.2766 | 9 | Load(C2, P2, JFK)  Load(C1, P1, SFO)  Load(C3, P3, ATL)  Fly(P2, JFK, SFO)  UnLoad(C2, P2, SFO)  Fly(P1, SFO, JFK)  UnLoad(C1, P1, JFK)  Fly(P3, ATL, SFO)  UnLoad(C3, P3, SFO) |
| Problem 3 | breadth\_first\_search | 14663 | 18098 | 129631 | 37.8 | 12 | Load(C2, P2, JFK)  Load(C1, P1, SFO)  Fly(P2, JFK, ORD)  Load(C4, P2, ORD)  Fly(P1, SFO, ATL)  Load(C3, P1, ATL)  Fly(P1, ATL, JFK)  UnLoad(C1, P1, JFK)  UnLoad(C3, P1, JFK)  Fly(P2, ORD, SFO)  UnLoad(C2, P2, SFO)  UnLoad(C4, P2, SFO) |
| Problem 1 | uniform\_cost\_search | 55 | 57 | 224 | 0.04 | 6 | Load(C1, P1, SFO)  Load(C2, P2, JFK)  Fly(P1, SFO, JFK)  Fly(P2, JFK, SFO)  UnLoad(C1, P1, JFK)  UnLoad(C2, P2, SFO) |
| Problem 2 | uniform\_cost\_search | 4852 | 4854 | 44030 | 11.38 | 9 | Load(C1, P1, SFO)  Load(C2, P2, JFK)  Load(C3, P3, ATL)  Fly(P1, SFO, JFK)  Fly(P2, JFK, SFO)  Fly(P3, ATL, SFO)  UnLoad(C3, P3, SFO)  UnLoad(C1, P1, JFK)  UnLoad(C2, P2, SFO) |
| Problem 3 | uniform\_cost\_search | 18223 | 18225 | 159618 | 46.16 | 12 | Load(C1, P1, SFO)  Load(C2, P2, JFK)  Fly(P1, SFO, ATL)  Load(C3, P1, ATL)  Fly(P2, JFK, ORD)  Load(C4, P2, ORD)  Fly(P2, ORD, SFO)  Fly(P1, ATL, JFK)  UnLoad(C4, P2, SFO)  UnLoad(C3, P1, JFK)  UnLoad(C1, P1, JFK)  UnLoad(C2, P2, SFO) |
| Problem 1 | depth\_first\_graph\_search | 12 | 13 | 48 | 0.013566 | 12 | Fly(P1, SFO, JFK)  Fly(P2, JFK, SFO)  Load(C1, P2, SFO)  Fly(P2, SFO, JFK)  Fly(P1, JFK, SFO)  UnLoad(C1, P2, JFK)  Fly(P2, JFK, SFO)  Fly(P1, SFO, JFK)  Load(C2, P1, JFK)  Fly(P2, SFO, JFK)  Fly(P1, JFK, SFO)  UnLoad(C2, P1, SFO) |
| Problem 2 | depth\_first\_graph\_search | 582 | 583 | 5211 | 3.31 | 575 | Fly(P3, ATL, SFO)  Fly(P1, SFO, ATL)  Fly(P3, SFO, JFK)  Fly(P1, ATL, JFK)  Fly(P2, JFK, ATL)  Fly(P3, JFK, ATL)  Fly(P2, ATL, SFO)  Fly(P3, ATL, SFO)  Load(C1, P3, SFO)  Fly(P3, SFO, ATL)  Fly(P2, SFO, ATL)  Fly(P3, ATL, JFK)  … |
| Problem 3 | depth\_first\_graph\_search | 627 | 628 | 5176 | 3.58 | 596 | Fly(P1, SFO, ORD)  Fly(P2, JFK, ORD)  Fly(P1, ORD, ATL)  Fly(P2, ORD, ATL)  Fly(P1, ATL, JFK)  Fly(P2, ATL, SFO)  Load(C1, P2, SFO)  Fly(P2, SFO, ORD)  Fly(P1, JFK, ORD)  Fly(P2, ORD, ATL)  … |

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

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Problem** | **Search Algorithm** | **Expansions** | **Goal Tests** | **New Nodes** | **Time consuming** | **Plan length** | **Result** |
| Problem 1 | astar\_search h\_ignore\_preconditions | 33 | 35 | 136 | 0.1656 | 6 | Load(C1, P1, SFO)  Fly(P1, SFO, JFK)  Load(C2, P2, JFK)  Fly(P2, JFK, SFO)  UnLoad(C1, P1, JFK)  UnLoad(C2, P2, SFO) |
| Problem 2 | astar\_search h\_ignore\_preconditions | 1045 | 1047 | 9646 | 75.494 | 9 | Load(C2, P2, JFK)  Fly(P2, JFK, SFO)  Load(C1, P1, SFO)  Fly(P1, SFO, JFK)  Load(C3, P3, ATL)  Fly(P3, ATL, SFO)  UnLoad(C3, P3, SFO)  UnLoad(C1, P1, JFK)  UnLoad(C2, P2, SFO) |
| Problem 3 | astar\_search h\_ignore\_preconditions | 6017 | 6019 | 53851 | 721.39 | 12 | Load(C1, P1, SFO)  Load(C2, P2, JFK)  Fly(P1, SFO, ATL)  Load(C3, P1, ATL)  Fly(P2, JFK, ORD)  Load(C4, P2, ORD)  Fly(P2, ORD, SFO)  Fly(P1, ATL, JFK)  UnLoad(C4, P2, SFO)  UnLoad(C3, P1, JFK)  UnLoad(C1, P1, JFK)  UnLoad(C2, P2, SFO) |
| Problem 1 | astar\_search h\_pg\_levelsum | 32 | 34 | 138 | 0.688 | 6 | Load(C1, P1, SFO)  Fly(P1, SFO, JFK)  UnLoad(C1, P1, JFK)  Load(C2, P2, JFK)  Fly(P2, JFK, SFO)  UnLoad(C2, P2, SFO) |
| Problem 2 | astar\_search h\_pg\_levelsum | 168 | 170 | 1618 | 56.304 | 9 | Load(C1, P1, SFO)  Fly(P1, SFO, JFK)  Load(C3, P3, ATL)  Fly(P3, ATL, SFO)  UnLoad(C3, P3, SFO)  UnLoad(C1, P1, JFK)  Load(C2, P2, JFK)  Fly(P2, JFK, SFO)  UnLoad(C2, P2, SFO) |
| Problem 3 | astar\_search h\_pg\_levelsum | 940 | 942 | 8732 | 410.82 | 12 | Load(C1, P1, SFO)  Fly(P1, SFO, ATL)  Load(C2, P2, JFK)  Fly(P2, JFK, ORD)  Load(C3, P1, ATL)  Fly(P1, ATL, JFK)  Load(C4, P2, ORD)  UnLoad(C3, P1, JFK)  Fly(P2, ORD, SFO)  UnLoad(C1, P1, JFK)  UnLoad(C4, P2, SFO)  UnLoad(C2, P2, SFO) |

1. **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?**

Levelsum is the best heuristic for these problems. But it may be not always better than non-heuristic search planning methods. According to tables above breadth first search and uniform cost search are all better than heuristic search planning methods. Because in this case the first plan found by breadth first search is exactly the optimal plan, so breadth first search has less expansions, goal tests, new nodes exploring and time consuming but also got the optimal plan. Uniform cost search is also a kind of breadth first search essentially, so it also found the optimal plan quickly.

But uniform cost search should be equivalent to astar search with h(n)=0, why uniform cost search faster than astar search I have no reasonable idea.