**Heuristic Analysis**

**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))

**Optimal Plan taken from greedy\_best\_first\_graph\_search:**

Load(C1, P1, SFO)

Load(C2, P2, JFK)

Fly(P1, SFO, JFK)

Fly(P2, JFK, SFO)

Unload(C1, P1, JFK)

Unload(C2, P2, SFO)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Table – 1a Non-Heuristic Search Results | | | | | | |
| Search Algorithm | Expansions | New Nodes | Goal Tests | Plan Length | Time (seconds) | Optimal |
| breadth\_first\_search | 43 | 180 | 56 | 6 | 0.042 | Yes |
| depth\_first\_graph\_search | 12 | 48 | 13 | 12 | 0.014 | No |
| greedy\_best\_first\_graph\_search | 7 | 28 | 9 | 6 | 0.007 | Yes |

Greedy\_best\_first\_search was the top algorithm here out of the 3 noted above. It had the least number of new nodes and expansion which led to the fastest overall time. Breadth\_first\_search was the slowest among the 3 but did have an optimal plan length of 6. Even though depth\_first\_graph\_search was faster than breadth\_first\_search with lower number of expansions and new nodes, it had the least optimal plan of length 12.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Table – 1b Heuristic Search Results | | | | | | |
| Search Algorithm | Expansions | New Nodes | Goal Tests | Plan Length | Time (seconds) | Optimal |
| A\* Search with h\_1 | 55 | 224 | 57 | 6 | 0.053 |  |
| A\* Search (ignore preconditions) | 41 | 170 | 43 | 6 | 0.054 |  |
| A\* (level sum) | 11 | 50 | 13 | 6 | 1.224 |  |

Table 1a shows the performance of A\* Search with 3 different heuristics. All 3 algorithms found the optimal plan length of 6 with the level sum heuristic being the slowest amongst the 3.

**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))

**Optimal Plan taken from greedy\_best\_first\_graph\_search:**

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(C2, P2, SFO)

Unload(C1, P1, JFK)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Table – 2a Non-Heuristic Search Results | | | | | | |
| Search Algorithm | Expansions | New Nodes | Goal Tests | Plan Length | Time (seconds) | Optimal |
| breadth\_first\_search | 3401 | 31049 | 4672 | 9 | 17.744 | Yes |
| depth\_first\_graph\_search | 350 | 3142 | 351 | 346 | 1.872 | No |
| greedy\_best\_first\_graph\_search | 550 | 4950 | 552 | 9 | 1.800 | Yes |

Greedy\_best\_first\_search was the top algorithm here out of the 3 noted above. It had the optimal plan length and the fastest overall time. Breadth\_first\_search was the slowest among the 3 but did have an optimal plan length of 9. Even though depth\_first\_graph\_search was faster than breadth\_first\_search with lower number of expansions and new nodes, it had the least optimal plan of length 346.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Table – 2b Heuristic Search Results | | | | | | |
| Search Algorithm | Expansions | New Nodes | Goal Tests | Plan Length | Time (seconds) | Optimal |
| A\* Search with h\_1 | 4761 | 43206 | 4763 | 9 | 17.016 |  |
| A\* Search (ignore preconditions) | 1450 | 13303 | 1452 | 9 | 6.304 |  |
| A\* (level sum) | 86 | 841 | 88 | 9 | 268.076 |  |

Table 2b shows results for A\* search with various heuristics. All 3 heuristics found the optimal plan length but the ignore\_preconditions heuristic took 6.3 seconds vs 268.076 seconds for level\_sum.

**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))

**Optimal Plan taken from breadth\_first\_search:**

Load(C2, P2, JFK)

Load(C1, P1, SFO)

Fly(P2, JFK, ORD)

Load(C4, P2, ORD)

Fly(P2, ORD, SFO)

Unload(C2, P2, SFO)

Unload(C4, P2, SFO)

Fly(P1, SFO, ATL)

Load(C3, P1, ATL)

Fly(P1, ATL, JFK)

Unload(C1, P1, JFK)

Unload(C3, P1, JFK)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Table – 3a Non-Heuristic Search Results | | | | | | |
| Search Algorithm | Expansions | New Nodes | Goal Tests | Plan Length | Time (seconds) | Optimal |
| breadth\_first\_search | 14491 | 128184 | 17947 | 12 | 128.279 | Yes |
| depth\_first\_graph\_search | 1948 | 16253 | 1949 | 1878 | 24.475 | No |
| greedy\_best\_first\_graph\_search | 4031 | 35794 | 4033 | 22 | 15.589 | No |

Breadth\_first\_search was the only optimal algorithm here. Even though it was the slowest among the 3 with most number of expansions and new nodes, it had the most optimal plan length of 12. Even though Greedy\_best\_first\_search was the fastest amongst the 3 algorithms, it had a non-optimal plan length of 22. Depth\_first\_graph\_search was the worst performing with largest plan length of the 3 algorithms.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Table – 3b Heuristic Search Results | | | | | | |
| Search Algorithm | Expansions | New Nodes | Goal Tests | Plan Length | Time (seconds) | Optimal |
| A\* Search with h\_1 | 17783 | 155920 | 17785 | 12 | 71.684 |  |
| A\* Search (ignore preconditions) | 5003 | 44586 | 5005 | 12 | 23.109 |  |
| A\* (level sum) | 323 | 2983 | 325 | 12 | 1376.118 |  |

All 3 heuristics in Table 3b achieved the optimal plan length. While level\_sum heuristic only expanded to 323 vs 5003 for ignore\_preconditions, it took far longer with 1376.118 seconds vs 23.109 seconds for ignore\_preconditions heuristic.

**Conclusion**

The best heuristic is really dependent on the trade-off of speed vs memory. If you require the optimal plan with the fastest time and have abundant memory space then ignore\_preconditions is the way to go. On the other hand if you have limited amount of memory space and don’t care about how long it takes to find the optimal plan then level\_sum heuristic is the way to go.

Overall the ignore\_precondition heuristic is the best performing algorithm overall as it found the optimal plan length in all instances and at a comparable or faster time than the non-heuristic algorithms.