**AIND Project 3: Domain Independent Planner**

# Heuristic Analysis

By:~ Aditya Paliwal

For this project, the implementation a planning search agent to solve deterministic logistics planning problems for an Air Cargo transport system. Three planning problems were given in the Air Cargo domain that use the same action schema:

Action(Load(c, p, a),

PRECOND: At(c, a) ∧ At(p, a) ∧ Cargo(c) ∧ Plane(p) ∧ Airport(a)

EFFECT: ¬ At(c, a) ∧ In(c, p))

Action(Unload(c, p, a),

PRECOND: In(c, p) ∧ At(p, a) ∧ Cargo(c) ∧ Plane(p) ∧ Airport(a)

EFFECT: At(c, a) ∧ ¬ In(c, p))

Action(Fly(p, from, to),

PRECOND: At(p, from) ∧ Plane(p) ∧ Airport(from) ∧ Airport(to)

EFFECT: ¬ At(p, from) ∧ At(p, to))

The three problems have the following initial states and goals:

|  |  |  |
| --- | --- | --- |
| 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)) | 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)) | 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)) |

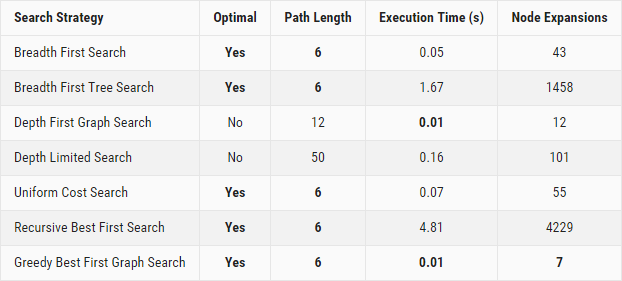
The goals above can be reached using different plans, but the **optimal plan lengths** for problems 1,2, and 3are **6, 9, and 12 actions**, respectively. Below are sample plans with optimal length:

|  |  |  |
| --- | --- | --- |
| Problem 1:  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:  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, JF | 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(P1, ATL, JFK)  Fly(P2, ORD, SFO)  Unload(C4, P2, SFO)  Unload(C3, P1, JFK)  Unload(C2, P2, SFO)  Unload(C1, P1, JFK) |

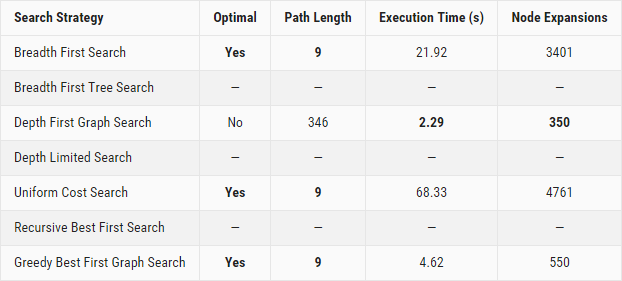
**Uninformed Search Strategies**

The following table shows the results gathered after solving the air cargo problems for this project with uninformed search. The goal of this analysis is to document the results obtained from each search type and find an optimal solution for each air cargo problem, that is; a search algorithm that finds the lowest path among all possible paths from start to goal.

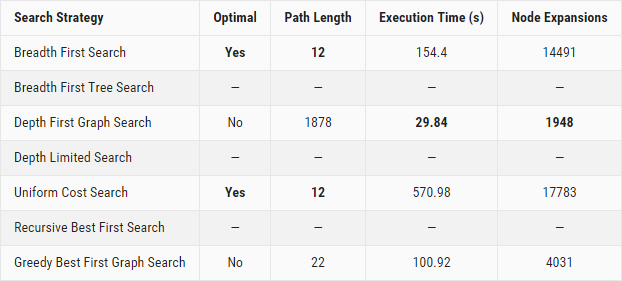
Problem 1 Results



Problem 2 Results



Problem 3 Results



Note: Both for Execution Time and Node Expansions, smaller numbers are better. Best of each category are highlighted in bold.

**Search Strategies Discussion**

With this 3-problem set, **Breadth First Search** and **Uniform Cost Search** are the only two uninformed search strategies that **yield an optimal action plan**. When it comes to execution speed and memory usage, **Depth First Graph Search** is the **fastest and uses the least memory**. However, it does not generate an optimal action plan

**Optimal Sequence of Actions**

The goals above can be reached using different plans, but the **optimal plan lengths** for problems 1, 2, and 3 are **6, 9, and 12 actions**, respectively. Below are sample plans with optimal length:

|  |  |  |
| --- | --- | --- |
| Problem 1:  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:  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, JF | 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(P1, ATL, JFK)  Fly(P2, ORD, SFO)  Unload(C4, P2, SFO)  Unload(C3, P1, JFK)  Unload(C2, P2, SFO)  Unload(C1, P1, JFK) |

**Analysis**

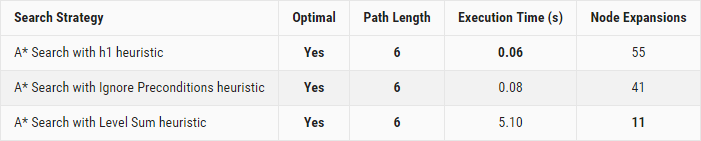
With this 3-problem set, **Breadth First Search** and **Uniform Cost Search** are the only two uninformed search strategies that **yield an optimal action plan** under the 10mn time limit. When it comes to execution speed and memory usage, **Depth First Graph Search** is the **fastest and uses the least memory**. However, it does not generate an optimal action plan (problem 1: plan length of 12 instead of 6, problem 2: plan length of 346 instead of 9, problem 3: plan length of 1878 instead of 12).

If finding the optimal path length is critical, what strategy should we use? Because it performs **faster and uses less memory** than Uniform Cost Search, **Breadth First Search** is the recommended search strategy. This isn't much of a surprise, as BFS is complete and optimal [1].

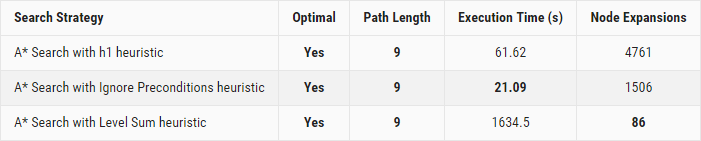
**Informed (Heuristic) Search Strategies Analysis**

The following table shows the results gathered after solving the air cargo problems for this project with informed (Heuristic) search. The goal of this analysis is to document the results obtained from each search type and find an optimal solution for each air cargo problem, that is; a search algorithm that finds the lowest path among all possible paths from start to goal

Problem 1 Results



Problem 2 Results



Problem 3 Results



**Analysis**

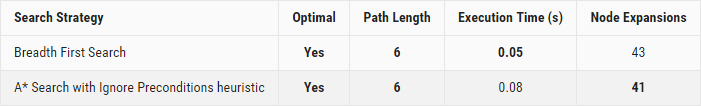
While all heuristics yield an optimal action plan, only the h1 and Ignore Preconditions heuristics return results with least execution time by the. Of the two strategies mentioned above, **A\* Search with Ignore Preconditions heuristic is the fastest**. If we let search run to completion on our machine, **A\* Search with Level Sum heuristic uses the least memory**, but its execution time is much slower

**Comparison between Informed vs Uninformed Search Strategies**

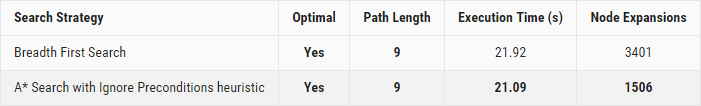
As we saw earlier, when it comes to execution speed and memory usage of uninformed search strategies, **Depth First Graph Search** is faster and uses less memory than Uniform Cost Search. As for informed search strategies, **A\* Search with Ignore Preconditions heuristic** is the fastest and uses the least memory. So, really, the choice is between Depth First Graph Search and A\* Search with Ignore Preconditions heuristic.

The following table shows the comparison results as:

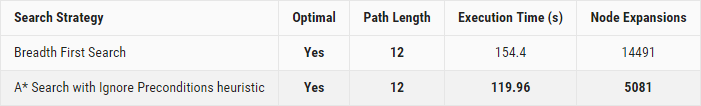
Problem 1 Results



Problem 2 Results



Problem 3 Results



**Conclusion**

The results above clearly illustrate the benefits of using informed search strategies with custom heuristics over uninformed search techniques when searching for an optimal plan. The benefits are significant both in terms of speed and memory usage.

**Reference**

1. Chapter-3, Section 3.4 (Breadth First Search), Stuart J. Russell, Peter Norvig, Artificial Intelligence: A Modern Approach.