**Heuristics Analysis**

For Logistics Planning Problems for an Air Cargo Transport System using a planning search agent

**Problems:**

This analysis report details out the set of problems in PDDL (Planning Domain Definition Language) for Air Cargo domains and provides experimentation metrics and analysis for various search algorithms.

* Air Cargo 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))

* Problem 1 initial state and goal:

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 initial state and goal:

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 initial state and goal:

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))

**Part 1: Planning problems:**

The non-heuristics planning was done with below search algorithms and the respective results were shown in the table.

* Bread-First Search
* Depth-First Search
* Uniform-Cost-Search

**Experiment Metrics:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Problem** | **Search Type** | **Plan Length** | **Expansion** | **Goal Tests** | **Time elapsed (s)** | **Optimality** |
| P1 | breadth\_first\_search | 6 | 43 | 56 | 0.0432 | Y |
| P1 | depth\_first\_graph\_search | 12 | 12 | 13 | 0.0133 | N |
| P1 | uniform\_cost\_search | 6 | 55 | 57 | 0.0541 | Y |
| P2 | breadth\_first\_search | 9 | 3343 | 4609 | 19.3146 | Y |
| P2 | depth\_first\_graph\_search | 575 | 582 | 583 | 4.1984 | N |
| P2 | uniform\_cost\_search | 9 | 4834 | 4836 | 17.7042 | Y |
| P3 | breadth\_first\_search | 12 | 14663 | 18098 | 123.9184 | Y |
| P3 | depth\_first\_graph\_search | 195 | 3664 | 3665 | 20.981 | N |
| P3 | uniform\_cost\_search | 12 | 18167 | 18169 | 67.8777 | Y |

**Comparison Insights:**

1. **Time:** The time taken to get to the goal state by Depth-First Search was considerably less than other two searches in all three Problems.
2. **Nodes Expanded:** The nodes expanded to get to the goal state by Depth-First Search was considerably less than other two searches in all three Problems.
3. **Plan Length:** The plan length to get to the goal state by Depth-First Search was considerably more in all problems than other two searches. And also the plan length is exactly same for Breadth-First search and Uniform-First Search in all three problems.
4. **Optimality:** Looking at the nodes expanded, it is obvious that Depth-First Search is not optimal compared t other two search algorithms.
5. **Goal Tests:** Looking at the goal test, both Breadth-First Search and Uniform Cost Search take about the same range steps and Depth-First Search is lower than the other two.

**Algorithm Analysis:**

**Breadth-First Search:**

* Produces optimal plan but consumes more time and more expansion as the problem complexity increases.
* Reference: Lecture 🡪 Lesson 8: Search – 15. Breadth-First Search

**Depth-First Search:**

* Produces higher plan length in less time. As the problem complexity grows, this is not guaranteed to produce optimal results at all.
* Reference: Lecture 🡪 Lesson 8: Search – 25. Search Comparison

**Uniform-First Search:**

- Produces optimal plan in terms of time consumption and node expansion but performs not so good as breadth-First search as it keeps expanding even after reaching the goal for cheapest path.

* Reference: Lecture 🡪 Lesson 8: Search – 21. Uniform Cost Search

**Part 2: Domain-Independent Heuristics**

The heuristics planning was done with below heuristics functions and the respective results were shown in the table.

* h\_1
* h\_ignore\_preconditions
* h\_pg\_levelsum

**Experiment Metrics:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Problem** | **Search Type** | **Plan Length** | **Expansion** | **Goal Tests** | **Time elapsed (s)** | **Optimality** |
| P1 | astar\_search-h\_1 | 6 | 55 | 57 | 0.054 | Y |
| P1 | astar\_search-h\_ignore\_preconditions | 6 | 41 | 43 | 0.0401 | Y |
| P1 | astar\_search-h\_pg\_levelsum | 6 | 55 | 57 | 1.8947 | Y |
| P2 | astar\_search-h\_1 | 9 | 4834 | 4836 | 17.0342 | Y |
| P2 | astar\_search-h\_ignore\_preconditions | 9 | 1450 | 1452 | 5.1977 | Y |
| P2 | astar\_search-h\_pg\_levelsum | 9 | 4834 | 4836 | 1041.1107 | Y |
| P3 | astar\_search-h\_1 | 12 | 18167 | 18169 | 67.2693 | Y |
| P3 | astar\_search-h\_ignore\_preconditions | 12 | 5035 | 5037 | 19.7384 | Y |
| P3 | astar\_search-h\_pg\_levelsum | 12 | 388 | 390 | 562.5685 | Y |

**Comparison Insights:**

1. **Time:** The time taken to get to the goal state by h\_ignore\_preconditions heuristic was greatly less than other two heuristics.

h\_ignore\_preconditions < h\_1 < h\_pg\_levelsum

1. **Nodes Expanded:** The nodes expanded to get to the goal state by h\_ignore\_preconditions was considerably less than other two heuristics in problem 1 and 2 but pg\_lelvel\_sum was really low in problem 3 compared to other two heuristics.

P1, P2 🡪 h\_ignore\_preconditions < h\_1 = h\_pg\_levelsum

P3 🡪 h\_pg\_levelsum < h\_ignore\_preconditions < h\_1

1. **Plan Length:** The plan length to get to the goal state by all three heuristics was exactly the same in three problems.

h\_ignore\_preconditions = h\_1 = h\_pg\_levelsum

1. **Optimality:** Looking at the nodes expanded, it is obvious that all three heuristics are optimal and better than non-heuristics searches.

h\_ignore\_preconditions = h\_1 = h\_pg\_levelsum

1. **Goal Tests:** Looking at the goal test, h\_ignore\_preconditions heuristic was greatly less than other two heuristics in problem 1 and 2 but pg\_lelvel\_sum heuristics was really low in problem 3 compared to other two heuristics.

P1, P2 🡪 h\_ignore\_preconditions < h\_1 = h\_pg\_levelsum

P3 🡪 h\_pg\_levelsum < h\_ignore\_preconditions < h\_1

**Heuristics Analysis:**

**Heuristic - 1:**

* Simple heuristic as it always returns constant 1 as the estimated distance to goal on any given state. Not useful.

**Heuristic – ignore\_precondition:**

* Produces best results in terms of time and node expansion as it ignores the preconditions for actions to be executed. Great results as the problem complexity increases.

**Heuristic – pg\_levelsum:**

- Takes too much time to execute as it takes all preconditions into consideration and uses planning representation to sum of all actions to the goal state. Execution time exponentially grows as the problem complexity increases.

**Part 3: Written Analysis**

Below sections explains the best heuristics and the optimal plan obtained from it.

**Best Heuristics:**

Looking at the time taken to get the goal state and the nodes expanded, we can conclude that the heuristic h\_ignore\_preconditions is the best heuristics among all heuristics and non-heuristics searches in all three problems. The reasons this heuristic function is better than other two are,

* Time to get to goal is less
* Number of nodes expanded is less (though pg\_levelsum takes less nodes but time taken was huge)

In the non-heuristics searches, Breadth-First search shows better results in terms of less time taken to achieve goal, even though Depth-First search was less time but plan length is high in all three problems.

**Optimal plan:**

Below is the optimal plan described using h\_ignore\_preconditions heuristics approach for all three problems.

**Problem 1 using h\_ignore\_preconditions:**

Solving Air Cargo Problem 1 using astar\_search with h\_ignore\_preconditions...  
  
Expansions Goal Tests New Nodes  
 41 43 170   
  
Plan length: 6 Time elapsed in seconds: 0.04016264993697405  
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 using h\_ignore\_preconditions:**

Solving Air Cargo Problem 2 using astar\_search with h\_ignore\_preconditions...  
  
Expansions Goal Tests New Nodes  
 1450 1452 13303   
  
Plan length: 9 Time elapsed in seconds: 5.197764916811138  
Load(C3, P3, ATL)  
Fly(P3, ATL, SFO)  
Unload(C3, P3, SFO)  
Load(C2, P2, JFK)  
Fly(P2, JFK, SFO)  
Unload(C2, P2, SFO)  
Load(C1, P1, SFO)  
Fly(P1, SFO, JFK)  
Unload(C1, P1, JFK)

**Problem 3 using h\_ignore\_preconditions:**

Solving Air Cargo Problem 3 using astar\_search with h\_ignore\_preconditions...

Expansions Goal Tests New Nodes

5035 5037 44722

Plan length: 12 Time elapsed in seconds: 19.738405549898744

Load(C2, P2, JFK)

Fly(P2, JFK, ORD)

Load(C4, P2, ORD)

Fly(P2, ORD, SFO)

Unload(C4, P2, SFO)

Load(C1, P1, SFO)

Fly(P1, SFO, ATL)

Load(C3, P1, ATL)

Fly(P1, ATL, JFK)

Unload(C3, P1, JFK)

Unload(C2, P2, SFO)

Unload(C1, P1, JFK)