Planning Heuristic Analysis

- [Optimal Plan for Problems 1, 2 and 3]
  - [PDDL Action Schema]
  - [Problem 1 initial state and goal]
  - [Problem 2 initial state and goal]
  - [Problem 3 initial state and goal]
- [Non-heuristic search result metrics]
- [Heuristic search result metrics]
- [Informed vs Uninformed Search Strategies]

Optimal Plan for Problems 1, 2 and 3

For this project, a planning search agent was implemented for an Air Cargo transport system to solve a logistic planning problem. A planning graph using A* search with domain independent heuristics was used and compared its performance against several non-heuristic search methods such as breadth-first and depth-first to name a few.

PDDL 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))
```

Non-heuristic search result metrics

This section analyzes the uninformed search metrics. Uninformed search have no additional information about states beyond that provided in the problem definition. The behavior of this type of search is to generate successors and distinguish a goal state from a non-goal state. In this section, a comparison of these strategies (as search) are reviewed in terms of execution time (as time elapsed in seconds), memory usage (as the number of nodes expanded) and optimality (Yes, if a solution of optimal length is found; No, otherwise).

Problem	Search	Plan Length	Time Elapsed In Seconds	# Nodes Expanded	Optimal
P1	Breadth First Search	6	0.03164	43	Y
P1	Depth First Graph Search	12	0.00852	12	N
P1	Uniform Cost Search	6	0.03592	55	Y
P2	Breadth First Search	9	12.96285	3343	Y
P2	Depth First Graph Search	575	3.51257	582	N
P2	Uniform Cost Search	9	11.57874	4761	Y

P3	Breadth First Search	12	108.51931	14663	Y
P3	Depth First Graph Search	596	2.98638	627	N
P3	Uniform Cost Search	12	44.46916	17513	Y

Based on the table above in terms of execution time and memory usage, Depth First Graph Search is the best option. However, this search does not generate an optimal action plan for any problem. For instance, for problem 1 the path length is 200% greater. However, for problem 2 and 3 is surprisingly bigger by 6388% and 4966% respectively.

In terms of execution time Uniform Cost Search is the best option, although, the downside is the slightly higher memory consumption. However, if the main concern is the memory consumption Breadth First Search is the best option.

Heuristic search result metrics

In this section, we compare the performance of A* Search using three different heuristics. Each strategies is evaluated in terms of speed, memory consumption and optimality.

Problem	Search	Plan Length	Time Elapsed In Seconds	# Nodes Expanded	Optimal
P1	A* with h1 heuristic	6	0.03472	55	Y
P1	A* with ignore preconditions heuristic	6	0.02836	41	Y
P1	A* with level sum heuristic	6	0.71753	11	Y
P2	A* search with h1 heuristic	9	10.80014	4761	Y
P2	A* with ignore preconditions heuristic	9	4.56849	1450	Y
P2	A* with level sum heuristic	9	64.31475	86	Y
P3	A* search with h1 heuristic	12	44.59119	17513	Y
P3	A* with ignore preconditions heuristic	12	14.00470	5022	Y

P3	A* with level sum heuristic	12	333.46678	312	Y

Based on the results depicted in the table above is clear that all the heuristics yield an optimal action plan. Is clear that A* with ignore preconditions heuristic provides the best execution time. Although, its down side is the relatively higher memory consumption in comparison with A* with level sum heuristic. However, the latest heuristic provides a the most expensive execution time among the three.

Informed vs Uninformed Search Strategies

The search strategies that generate optimal plans are Breadth First Search, Uniform Cost Search, and A* Search with all three heuristics. For uninformed search strategies as discussed earlier Uniform Cost Search provides the best execution time. Even though it has a slightly higher consumption of memory in compare with breadth first search. As for informed search strategies, A* Search with Ignore Preconditions heuristic is the fastest and uses the least memory. In the following table a comparison between Uniform Search and A* Search with Ignore Preconditions heuristic will be analyzed further.

Problem	Search	Plan Length	Time Elapsed In Seconds	# Nodes Expanded	Optimal
P1	Uniform Cost Search	6	0.03592	55	Y
P1	A* with ignore preconditions heuristic	6	0.02836	41	Y
P2	Uniform Cost Search	9	11.57874	4761	Y
P2	A* with ignore preconditions heuristic	9	4.56849	1450	Y
P3	Uniform Cost Search	12	44.46916	17513	Y
P3	A* with ignore preconditions heuristic	12	14.00470	5022	Y

It is apparent based on the results above that A* Search with Ignore Preconditions heuristic would be the best choice overall for the Air Cargo problem. Since, it provides a faster execution time and uses the less memory than Uniform Cost Search. These results describes the benefits of using informed search strategies with custom heuristics over uninformed search techniques. It is apparent that provides significant better results in terms of execution time and memory usage.