Implement a Planning Search Heuristic Analysis

Air Cargo Planning Problem Description

In this project, I defined a group of problems in classical PDDL (Planning Domain Definition Language) for the air cargo domain discussed in the lectures. The goal of this project is to set up the problems for search, experiment with various automatically generated heuristics, including planning graph heuristics, to solve the problems, and then provide an analysis of the results. All problems are in the Air Cargo domain. They have the same action schema defined, but different initial states and goals:

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))
```

Optima Sequence of Actions Identified

The goals can be reached following different plans, but the best plan in term of length for the 3 problems are:

Problem 1 (Plan length: 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 (Plan length: 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(C2, P2, SFO)
Unload(C1, P1, JFK)
```

Problem 3 (Plan length: 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(P1, ATL, JFK)
Fly(P2, ORD, SFO)
Unload(C4, P2, SFO)
Unload(C3, P1, JFK)
Unload(C2, P2, SFO)
Unload(C1, P1, JFK)
```

Performance Comparison

I approached the problem dividing the tests in 2 groups: for the first group I measured the performance of uninformed search algorithms, and for the second the heuristic search algorithms.

Uninformed Planning Algorithms

As we learned during the lecture of this course, there are search strategies that have no additional information about states rather than that provided in the problem definition. These type of search techniques generate successors, distinguishing goal states from non-goal states.

The metrics used to compare the performances among the different algorithms for the 3 problems are:

- Path length
- Expansions (this parameter can tell us about the memory used)
- Execution Time (this parameter can tell us about the CPU required)
- Optimality (True if a solution of optimal length is found. False otherwise)

The next tables show the results running some of the algorithms provided for this project (for problem 1 I tested all the algorithms provided, for problem 2 and 3 some of them because the time requested was too long, and the rubric specify to run at least three uninformed planning algorithms).

Search Algorithms	Path Length	Optimal	Expansions	Time elapsed (sec)
Breadth First Search	6	TRUE	43	0.02
Breadth First Tree Search	6	TRUE	1458	0.76
Depth First Graph Search	20	FALSE	21	0.01
Depth Limited Search	50	FALSE	101	0.07
Uniform Cost Search	6	TRUE	55	0.03
Recursive Best First Search	6	TRUE	4229	2.25
Greedy Best First Graph Search	6	TRUE	7	0.003

Table 1 - Air Cargo Problem 1 Uninformed Results

Search Algorithms	Path Length	Optimal	Expansions	Time elapsed (sec)
Breadth First Search	9	TRUE	3401	11.66
Breadth First Tree Search	N/A	N/A	N/A	N/A
Depth First Graph Search	1138	FALSE	1192	7.10
Depth Limited Search	N/A	N/A	N/A	N/A
Uniform Cost Search	9	TRUE	4761	9.97
Recursive Best First Search	N/A	N/A	N/A	N/A
Greedy Best First Graph Search	9	TRUE	550	1.14

Table 2 - Air Cargo Problem 2 Uninformed Results

Search Algorithms	Path Length	Optimal	Expansions	Time elapsed (sec)
Breadth First Search	12	TRUE	14491	82.34
Breadth First Tree Search	N/A	N/A	N/A	N/A
Depth First Graph Search	2014	FALSE	2099	17.79
Depth Limited Search	N/A	N/A	N/A	N/A
Uniform Cost Search	12	TRUE	17783	41.64
Recursive Best First Search	N/A	N/A	N/A	N/A
Greedy Best First Graph Search	22	FALSE	4031	9.44

Table 3 - Air Cargo Problem 3 Uninformed Results

Looking at the results, and considering the metrics mentioned before, only *Breadth First Search* and *Uniform Cost Search* provide an optimal action plan.

In term of speed and memory consumption Depth First Graph Search find a quick solution and requires a small amount of memory (Expansions), but as we saw during the course, it lacks optimality due the fact it expands as deep as it can one single path.

Greedy Best First Graph Search is very interesting because in problem 1 and 2 it is also optimal, and it has good performances in term of speed and memory. In a context where optimality is not mandatory, and performances are more important, this search algo should be considered.

I recommend using Breadth First Search because is optimal, faster and use less memory than Uniform Cost Search.

Informed Planning Algorithms

As we learned during the lecture of this course, there are heuristic based search strategies that can help us to get performance, especially when complexity increase, injecting some knowledge to the about features of the search space. If we could inject this knowledge in our agent so that it can direct its search towards more promising paths and find a solution more efficiently while exploring as little of the search space as possible. This is exactly what we do with A* Search with the 3 heuristics used for these tests.

Search Algorithms	Path Length	Optimal	Expansions	Time elapsed (sec)
A* Search with h1 heuristic	6	TRUE	55	0.030
A* Search with ignore preconditions heuristic	6	TRUE	41	0.031
A* Search with level sum heuristic	6	TRUE	11	0.75

Table 4 - Air Cargo Problem 1 Heuristic Results

Search Algorithms	Path Length	Optimal	Expansions	Time elapsed (sec)
A* Search with h1 heuristic	9	TRUE	4761	9.22
A* Search with ignore preconditions heuristic	9	TRUE	1450	3.38
A* Search with level sum heuristic	9	TRUE	86	126.54

Table 5 - Air Cargo Problem 2 Heuristic Results

Search Algorithms	Path Length	Optimal	Expansions	Time elapsed (sec)
A* Search with h1 heuristic	12	TRUE	17783	41.58
A* Search with ignore preconditions heuristic	12	TRUE	5003	13.52
A* Search with level sum heuristic	12	TRUE	311	852.50

Table 6 - Air Cargo Problem 3 Heuristic Results

All 3 heuristics provide an optimal solution, but *A* Search with level sum heuristic* execution time is too slow, despite the lowest memory consumption, when compared with the other 2 algorithms. I recommend using A* Search with *ignore preconditions heuristic* because is optimal and faster.

Uninformed vs Informed Planning Algorithms

In this last section, I compare Breadth First Search (the best algo selected for uninformed search tests) against A* Search with ignore preconditions heuristic (the best algo for heuristic search tests).

Search Algorithms	Path Length	Optimal	Expansions	Time elapsed (sec)
Breadth First Search	6	TRUE	43	0.02
A* Search with ignore preconditions heuristic	6	TRUE	41	0.031

Table 7 - Air Cargo Problem 1 Uninformed vs Heuristic Results

Search Algorithms	Path Length	Optimal	Expansions	Time elapsed (sec)
Breadth First Search	9	TRUE	3401	11.66
A* Search with ignore preconditions heuristic	9	TRUE	1450	3.38

Table 8 - Air Cargo Problem 2 Uninformed vs Heuristic Results

Search Algorithms	Path Length	Optimal	Expansions	Time elapsed (sec)
Breadth First Search	12	TRUE	14491	82.34
A* Search with ignore preconditions heuristic	12	TRUE	5003	13.52

Table 9 - Air Cargo Problem 3 Uninformed vs Heuristic Results

Looking at the results, A* Search with ignore preconditions is the best choice possible because is faster and use less memory.

Conclusion

This project shows the benefits of using informed search algorithms with heuristics over uninformed search algorithms, to achieve an optimal plan faster and using as less memory as possible. Uniformed search is reasonable when the domain space is small, but as soon as the complexity increase, informed algorithms make the difference.