Heuristic analysis

Loay Ashraf

Problem Definition and Results

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

```
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

Load(C2, P2, JFK)

Load(C1, P1, SFO)

Fly(P2, JFK, SFO)

Unload(C2, P2, SFO)

Fly(P1, SFO, JFK)

Unload(C1, P1, JFK)

Result Metrics (non-Heuristic)

Search Algorithm	Optimal	Plan Length	Time Elapsed	Expansions	Goal Tests	New Nodes
Breadth first	Yes	6	0.043	43	56	180
Depth first	No	12	0.011	12	13	48
Uniform cost	Yes	6	0.05	55	57	224

For the first problem, Breadth first search performs the best. It gives optimum results consuming less time and less memory (nodes expanded) than the uniform cost search which also yields optimal results too. Depth First search doesn't yield an optimal result, despite consuming the least amount of time and memory compared to Breadth first and Uniform cost.

Result Metrics (Heuristic)

Search Algorithm	Optimal	Plan Length	Time Elapsed	Expansions	Goal Tests	New Nodes
A* with ignore preconditions	Yes	6	0.079	55	57	224
A* with level-sum	Yes	6	5.453	41	43	170

For the first problem there's not much of a difference between heuristic searches and non-heuristic searches in terms of memory consumption and the optimality of the results. Among the two heuristic search methods, Level Sum Heuristic takes much more time to complete, but consumes less memory, also, both heuristics converge to optimal results.

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

Load(C3, P3, ATL)

Load(C2, P2, JFK)

Load(C1, P1, SFO)

Fly(P3, ATL, SFO)

Unload(C3, P3, SFO)

Fly(P2, JFK, SFO)

Unload(C2, P2, SFO)

Fly(P1, SFO, JFK)

Unload(C1, P1, JFK)

Result Metrics (non-Heuristic)

Search Algorithm	Optimal	Plan Length	Time Elapsed	Expansions	Goal Tests	New Nodes
Breadth first	Yes	9	13.9	3401	4672	31049
Depth first	No	346	2.28	350	351	3142
Uniform cost	Yes	9	18.93	4853	4855	44041

For the second problem, we notice that Depth First Search doesn't yield optimal results, even though it runs the fastest and consumes the least memory resources. The path length of the solution output by Depth First Search is very large compared to the path length of the optimal solution. We observe that like problem #1, Breadth First Search does arrive at an optimal solution. The number of nodes expanded by Uniform Cost Search is very large compared to the other two search methods.

Result Metrics (Heuristic)

Search Algorithm	Optimal	Plan Length	Time Elapsed	Expansions	Goal Tests	New Nodes
A* with ignore preconditions	Yes	9	46.79	4853	4855	44041
A* with level-sum	No	N/A	N/A	N/A	N/A	N/A

For the second problem it's observed that the Level sum heuristic is not optimal at all as it took over 15 minutes so it was aborted manually and considered not optimal. The Ignore preconditions heuristic converges to an optimal result although it takes much more time compared to non-heuristic searches.

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

Load(C2, P2, JFK)

Load(C1, P1, SFO)

Fly(P2, JFK, ORD)

Load(C4, P2, ORD)

Fly(P1, SFO, ATL)

Load(C3, P1, ATL)

Fly(P1, ATL, JFK)

Unload(C1, P1, JFK)

Unload(C3, P1, JFK)

Fly(P2, ORD, SFO)

Unload(C2, P2, SFO)

Unload(C4, P2, SFO)

Result Metrics (non-Heuristic)

Search Algorithm	Optimal	Plan Length	Time Elapsed	Expansions	Goal Tests	New Nodes
Breadth first	Yes	12	75.5	14629	18072	129356
Depth first	No	220	46.6	2269	2270	19021
Uniform cost	Yes	12	84.75	18222	18224	159608

For the third problem, we see that Breadth first search yet again converges at an optimal solution at the cost of time and memory. The difference in the computational time required and memory consumed for Breadth first search is very apparent for this problem. Depth first search runs faster but converges to a non-optimal solution, whose path length is way off the path length of the optimal solution. Uniform cost search consumes more time and memory resources, but arrives at an optimal solution just like the Breadth first search.

Result Metrics (Heuristic)

Search Algorithm	Optimal	Plan Length	Time Elapsed	Expansions	Goal Tests	New Nodes
A* with ignore preconditions	Yes	12	197.25	18222	18224	159608
A* with level-sum	No	N/A	N/A	N/A	N/A	N/A

For the third problem -and just like the second problem- it's observed that the Level sum heuristic is not optimal at all as it took over 15 minutes so it was aborted manually and considered not optimal. The Ignore preconditions heuristic converges to an optimal result although it takes much more time compared to non-heuristic searches.

Results Justification

Breadth first search:

Finds the shortest path in terms of the least number of steps, but it will not find the shortest path in terms of the shortest total cost (sum of step costs).

Depth first search:

Starts by searching through the left most branch then backtracks multiple times till it reaches a solution but has a drawback which is it often misses an optimal solution.

Uniform cost search:

Takes longer and expands more nodes than necessary since even after finding a path to the goal state it continues searching to try and find a cheaper path that also reaches the goal state.

A* with ignore preconditions heuristic search:

Finds the goal faster since it ignores preconditions required for an action to be executed to make the problem easier in order to estimate the minimum number of actions that must be carried out from the current state in order to satisfy all of the goal conditions.

A* with level sum heuristic search:

Uses a Planning Graph but just estimates the sum of all actions that must be carried out from the current state to satisfy each individual goal condition.

Conclusion

We can conclude from the above results that Breadth first search with works best in terms of time taken and optimality. The A* search with ignore preconditions heuristic is noticed to have exactly the same results as the Uniform cost search throughout the three experiments done, both consume more time and memory resources than the Breadth first search and the Depth first search, So, in situations when memory and computational time are scarce, we should prefer Breadth first search.

The heuristic based search doesn't provide any improvement over uninformed non-heuristic based search algorithms in terms of time and memory consumption and also result optimality. For simpler problems having fewer literals, the heuristic based search performs reasonably well. If we want to guarantee optimality and speed is of more importance, we should prefer Breadth First Search for such scenarios.

References:

- AIND class videos (Search lesson)
- Norvig and Russell's textbook, "Artificial Intelligence: A Modern Approach"