Chris Morse
September 13, 2017

Planning Search Heuristic Analysis

Comparison of heuristics

In this project we solve deterministic logistics planning problems for an Air Cargo transport system using a planning search agent. We use the following action schema with different 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))
```

Optimal Plan

Here are some optimal plans that came out of our project.

Problem 1

Load(C2, P2, JFK)
Load(C1, P1, SFO)
Fly(P2, JFK, SFO)
Unload(C2, P2, SFO)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)

Problem 2

Load(C2, P2, JFK)
Load(C1, P1, SFO)
Load(C3, P3, ATL)
Fly(P2, JFK, SFO)
Unload(C2, P2, SFO)
Fly(P1, SFO, JFK)
Unload(C1, P1, JFK)
Fly(P3, ATL, SFO)
Unload(C3, P3, SFO)

Problem 3

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)

Non-heuristic search result metrics

Here are the results of the non-heuristic searches.

Problem 1

Search Strategy	Optimal	Path Length	Execution Time	Node Expansions
Breadth First Search	Yes	6	0.03	43
Depth First Graph Search	No	12	0.01	12
Uniform Cost Search	Yes	6	0.04	55

Problem 2

Search Strategy	Optimal	Path Length	Execution Time	Node Expansions
Breadth First Search	Yes	9	14.9	30509
Depth First Graph Search	No	1444	13.6	14863
Uniform Cost Search	Yes	9	12.3	44030

Problem 3

Search Strategy	Optimal	Path Length	Execution Time	Node Expansions
Breadth First Search	Yes	12	114.0	129631
Depth First Graph Search	No	571	3.2	4927
Uniform Cost Search	Yes	12	54.0	159716

Analysis

Breadth First Search and Uniform Cost Search give the optimal solution for all 3 problems. This can be seen by the Path Length of each strategy. Depth First Graph Search was the fastest overall (although Uniform Cost Search was faster with Problem 2). Depth First Graph Search also expanded the least number of nodes but again was not optimal.

When comparing optimal solutions, Uniform Cost Search appears to be faster than Breadth First Search as the problem gets more complicated but requires more node expansions

Heuristic search result metrics

Problem 1

Search Strategy	Optimal	Path Length	Execution Time	Node Expansions
A* with the "ignore preconditions"	Yes	6	0.04	170
A* with the "level-sum"	Yes	6	1.01	50

Problem 2

Search Strategy	Optimal	Path Length	Execution Time	Node Expansions
A* with the "ignore preconditions"	Yes	9	4.36	13303
A* with the "level-sum"	Yes	9	185.6	841

Problem 3

Search Strategy	Optimal	Path Length	Execution Time	Node Expansions
A* with the "ignore preconditions"	Yes	12	15.9	44944
A* with the "level-sum"	Yes	12	948.5	2934

Analysis

All solutions provide an optimal path. A* with "ignore preconditions" is the fastest while A* with "level sum" expands less nodes.

Heuristic vs Non-Heuristic Search

Problem 1

Search Strategy	Optimal	Path Length	Execution Time	Node Expansions
A* with the "ignore preconditions"	Yes	6	0.04	170
Uniform Cost Search	Yes	6	0.04	55

Problem 2

Search Strategy	Optimal	Path Length	Execution Time	Node Expansions
A* with the "ignore preconditions"	Yes	9	4.36	13303
Uniform Cost Search	Yes	9	12.3	44030

Problem 3

Search Strategy	Optimal	Path Length	Execution Time	Node Expansions
A* with the "ignore preconditions"	Yes	12	15.9	44944
Uniform Cost Search	Yes	12	54.0	159716

When we compare the fastest optimal heuristic vs the fastest optimal non-heuristic we see that A* with "ignore preconditions" is both faster and takes less node expansions as the problems get more complicated. It is interesting to note that Uniform Cost Search appears to be better for smaller problems since it had less node expansions.

The results above illustrate the benefits of using the heuristic based strategy over the non-heuristic strategy when searching for an optimal plan.