Planning Search heuristic analysis

Hyunbyung, Park

In this project, we define a group of problems in classical PDDL for the air cargo domain. We set up the problems for search, experiment with various automatically generated heuristics, including planning graph heuristics, to solve the problems, and then **now we provide an analysis of the 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))

Solution

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

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))

Solution

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

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))

Solution

Load(C2, P2, JFK)
Fly(P2, JFK, ORD)
Load(C4, P2, ORD)
Fly(P2, ORD, SFO)
Load(C1, P1, SFO)
Fly(P1, SFO, ATL)
Load(C3, P1, ATL)
Fly(P1, ATL, JFK)
Unload(C4, P2, SFO)
Unload(C3, P1, JFK)
Unload(C2, P2, SFO)
Unload(C1, P1, JFK)

Uninformed non-heuristic search

		Optimality	Plan Length	Time Elapsed	Node Expansions
Problem 1	Breadth First Search	YES	6	0.03638	43
	Depth First Graph Search	NO	20	0.01355	21
	Uniform Cost Search	YES	6	0.03619	55
Problem 2	Breadth First Search	YES	9	8.43632	3343
	Depth First Graph Search	NO	619	4.27744	624
	Uniform Cost Search	YES	9	12.6459	4852
Problem 3	Breadth First Search	YES	12	41.7846	14663
	Depth First Graph Search	NO	392	1.73900	408
	Uniform Cost Search	YES	12	51.3935	18235

I used 3 uninformed non-heuristic search, Breadth first search, Depth First graph search, and Uniform search. As looking at the result of the metrics we can see that DFS use time and node less than other search. But has longer plan length more than about approximately 30 - 50 times so this is not optimal. We consider the optimal search so need to decide between BFS and UCS. As compare by result both has same plan length but BFS takes less time and node expansion than UCS. **BFS seems the best of this three.**

A* search with heuristic

		Optimality	Plan Length	Time Elapsed	Node Expansions
Problem 1	A* ignore preconditions	YES	6	0.04348	41
	A* level-sum	YES	6	0.68742	11
Problem 2	A* ignore preconditions	YES	9	4.15485	1450
	A* level-sum	YES	9	61.9788	86
Problem 3	A* ignore preconditions	YES	12	16.0981	5040
	A* level-sum	YES	12	305.459	316

I used A* informed search used 2 heuristic, Ignore precondition & Level-sum. Both heuristic performs well at every problem and optimal. Compared each, ignore precondition is faster, and level-sum use less memory. **A* with ignore precondition seems the best in terms of time.**

Conclusion

A* ignore precondition is better than BFS in terms of memory, time.
So got conclusion that **A* ignore precondition is the best search to the problem of air cargo.**

Neither forward nor backward search is efficient without a good heuristic function. Effective heuristics can be derived by subgoal independence assumptions and by various relaxations of the planning problem. The ease of manipulating the schemas is the great advantage of the factored representation of planning problems, as compared with the atomic representation of search problems.

Artificial Intelligence A Modern Approach (3rd Edition) Chapter 10 Classical Planning