Planning and Search

Classical Planning: Complexity, SATPlan

Outline

- ◇ SATPlan
- ◇ Complexity of the classical planning problem
- ◇ Summary of classic planning algorithms.

SATPlan

Reduces planning problem to classical propositional SAT problem

SAT problem: is this propositional formula satisfiable? (is there an assignment that makes it true?)

Can only find plans of fixed maximal length

To use SATPlan, PDDL planning problem description needs first to be translated to a suitable form

SATPlan informally

Making plans by logical inference

1. Construct a propositional sentence that includes

(a) description of the initial state

(b) description of the planning domain (precondition axioms, successor state axioms, mutual exclusion of actions) up to some maximum time t

(c) the assertion that the goal is achieved at time t

2. Call SAT solver to return a model for the sentence from 1.

3. If a model exists, extract the variables which represent actions at each time from 0 to t and are assigned true, and present them in order of times as a plan

SATPlan pseudocode

```
function SATPlan(init, actions, goal, T_max) returns solution or failure
   inputs: init, actions, goal, constitute a description of the problem
           T_max, an upper limit for plan length

   for t = 0 to T_max do
       cnf ← Translate-To-Sat(init, actions, goal, t)
       model ← SAT-Solver(cnf)
       if model is not null then
           return Extract-Solution(model)
   return failure
```

Converting PDDL domain description

◇ Propositionalise the actions

◇ Define the initial state: assert F^0 for every fluent F in the problem's initial state, and $\neg F^0$ for every fluent not mentioned in the initial state

◇ Propositionalise the goal: for every variable in the goal, replace it with a disjunction over constants (for example, instead of $On(A, x)$, state $On(A, B)^t \vee On(A, C)^t \vee On(A, Table)^t$).

Converting PDDL domain description 2

◇ Add successor-state axioms: for each fluent F, add an axiom

$$F^{t+1} \Leftrightarrow ActionCausesF^t \vee (F^t \wedge \neg ActionCausesNotF^t)$$

where $ActionCausesF^t$ is a disjunction of all the ground actions that have F in their add list, and $ActionCausesNotF^t$ is a disjunction of all the ground actions that have F in their delete list.

◇ Add precondition axioms: for each ground action A, add the axiom $A^t \Rightarrow \textsc{Pre}(A)^t$ (if A is taken at time t, preconditions of A must be true at t)

◇ Add action exclusion axiom: only one action is taken at each time t ($\neg A_1^t \vee \neg A_2^t$ for each pair of different actions A_1, A_2 and time t)

Example

Suppose we only have 2 blocks A and B (and the table)

Initial state is $On(A, Table) \wedge On(B, Table) \wedge Clear(A) \wedge Clear(B)$

Goal state is $On(A, B)$

The only action is

$Move(b, x, y)$:

PRECOND: $On(b, x) \wedge Clear(b) \wedge Clear(y)$

EFFECT: $On(b, y) \wedge Clear(x) \wedge \neg On(b, x) \wedge \neg Clear(y)$

Example: propositionalise actions

Propositionalise the actions (I skip moving to table actions and $Clear(Table)$ postcondition):

$Move(A, Table, B)$:

PRECOND: $On(A, Table) \wedge Clear(A) \wedge Clear(B)$

EFFECT: $On(A, B) \wedge \neg On(A, Table) \wedge \neg Clear(B)$

$Move(B, Table, A)$:

PRECOND: $On(B, Table) \wedge Clear(B) \wedge Clear(A)$

EFFECT: $On(B, A) \wedge \neg On(B, Table) \wedge \neg Clear(A)$

Example: initial state and goal

$Init^0 =$

$On(A, Table)^0 \wedge On(B, Table)^0 \wedge Clear(A)^0 \wedge Clear(B)^0 \wedge$

$\neg On(A, B)^0 \wedge \neg On(B, A)^0$

$Goal^t = On(A, B)^t$

Successor state axioms

The fluents we have are: $On(A, B)$, $On(B, A)$, $On(A, Table)$, $On(B, Table)$, $Clear(A)$, $Clear(B)$

We need $F^{t+1} \Leftrightarrow ActionCausesF^t \vee (F^t \wedge \neg ActionCausesNotF^t)$

For $t = 1$ let us call the conjunction of the following formulas SA^1:

$On(A, B)^1 \Leftrightarrow (Move(A, Table, B)^0 \vee On(A, B)^0)$

$On(B, A)^1 \Leftrightarrow (Move(B, Table, A)^0 \vee On(B, A)^0)$

$On(A, Table)^1 \Leftrightarrow On(A, Table)^0 \wedge \neg Move(A, Table, B)^0$

$On(B, Table)^1 \Leftrightarrow On(B, Table)^0 \wedge \neg Move(B, Table, A)^0$

$Clear(A)^1 \Leftrightarrow Clear(A)^0 \wedge \neg Move(B, Table, A)^0$

$Clear(B)^1 \Leftrightarrow Clear(B)^0 \wedge \neg Move(A, Table, B)^0$

Successor state axioms: remark

$$F^{t+1} \Leftrightarrow ActionCausesF^t \vee (F^t \wedge \neg ActionCausesNotF^t)$$

Note that I did not leave any actions which can make $On(A, B)$ false. If we had $Move(A, B, Table)$, then the axiom would have been

$$On(A, B)^1 \Leftrightarrow (Move(A, Table, B)^0 \vee (On(A, B)^0 \wedge \neg Move(A, B, Table)^0)$$

Precondition axioms

Let us call conjunction of the following formulas Pre^1:

$$Move^0(A, Table, B) \Rightarrow (On(A, Table)^0 \wedge Clear(A)^0 \wedge Clear(B)^0)$$

$$Move^0(B, Table, A) \Rightarrow (On(B, Table)^0 \wedge Clear(B)^0 \wedge Clear(A)^0)$$

$$Move^1(A, Table, B) \Rightarrow (On(A, Table)^1 \wedge Clear(A)^1 \wedge Clear(B)^1)$$

$$Move^1(B, Table, A) \Rightarrow (On(B, Table)^1 \wedge Clear(B)^1 \wedge Clear(A)^1)$$

Action exclusion axioms

Let us call the conjunction of the following formulas $Excl^1$

$\neg Move^0(A, Table, B) \vee \neg Move^0(B, Table, A)$

$\neg Move^1(A, Table, B) \vee \neg Move^1(B, Table, A)$

Problem description for $t = 1$

$Init^0 \wedge Goal^1 \wedge SA^1 \wedge Pre^1 \wedge Excl^1$

Actually, for this case it is enough to check that the following formula has a model:

$On(A, Table)^0 \wedge On(B, Table)^0 \wedge Clear(A)^0 \wedge Clear(B)^0 \wedge \neg On(A, B)^0 \wedge$
$\neg On(B, A)^0 \wedge$

$(On(A, B)^1 \Leftrightarrow (Move(A, Table, B)^0 \vee On(A, B)^0)) \wedge$

$(On(B, A)^1 \Leftrightarrow (Move(B, Table, A)^0 \vee On(B, A)^0)) \wedge$

$(Move(A, Table, B)^0 \Rightarrow On(A, Table)^0 \wedge Clear(A)^0 \wedge Clear(B)^0)) \wedge$

$(Move(B, Table, A)^0 \Rightarrow On(B, Table)^0 \wedge Clear(B)^0 \wedge Clear(A)^0)) \wedge$

$(\neg Move(A, Table, B)^0 \vee \neg Move(B, Table, A)^0) \wedge On(A, B)^1$

Is this formula satisfiable?

Normally would translate it in CNF and call DPLL or WalkSAT algorithms, but let us just reason informally. $On(A, B)^1$ is a conjunct. For the formula to be true, it has to be true...

$On(A, Table)^0 \wedge On(B, Table)^0 \wedge Clear(A)^0 \wedge Clear(B)^0 \wedge \neg On(A, B)^0 \wedge$
$\neg On(B, A)^0 \wedge$

$(On(A, B)^1 \Leftrightarrow (Move(A, Table, B)^0 \vee On(A, B)^0)) \wedge$

$(On(B, A)^1 \Leftrightarrow (Move(B, Table, A)^0 \vee On(B, A)^0)) \wedge$

$(Move(A, Table, B)^0 \Rightarrow On(A, Table)^0 \wedge Clear(A)^0 \wedge Clear(B)^0)) \wedge$

$(Move(B, Table, A)^0 \Rightarrow On(B, Table)^0 \wedge Clear(B)^0 \wedge Clear(A)^0)) \wedge$

$(\neg Move(A, Table, B)^0 \vee \neg Move(B, Table, A)^0) \wedge On(A, B)^1$

Is this formula satisfiable? 2

$On(A, Table)^0 \wedge On(B, Table)^0 \wedge Clear(A)^0 \wedge Clear(B)^0 \wedge \neg On(A, B)^0 \wedge$
$\neg On(B, A)^0 \wedge$

$(On(A, B)^1 \Leftrightarrow (Move(A, Table, B)^0 \vee On(A, B)^0)) \wedge$

$(On(B, A)^1 \Leftrightarrow (Move(B, Table, A)^0 \vee On(B, A)^0)) \wedge$

$(Move(A, Table, B)^0 \Rightarrow On(A, Table)^0 \wedge Clear(A)^0 \wedge Clear(B)^0)) \wedge$

$(Move(B, Table, A)^0 \Rightarrow On(B, Table)^0 \wedge Clear(B)^0 \wedge Clear(A)^0)) \wedge$

$(\neg Move(A, Table, B)^0 \vee \neg Move(B, Table, A)^0) \wedge On(A, B)^1$

Is this formula satisfiable? 3

Model: red formulas false, blue formulas true. True action at time 0 is $Move(A, Table, B)^0$.

$On(A, Table)^0 \wedge On(B, Table)^0 \wedge Clear(A)^0 \wedge Clear(B)^0 \wedge \neg On(A, B)^0 \wedge$
$\neg On(B, A)^0 \wedge$

$(On(A, B)^1 \Leftrightarrow (Move(A, Table, B)^0 \vee On(A, B)^0)) \wedge$

$(On(B, A)^1 \Leftrightarrow (Move(B, Table, A)^0 \vee On(B, A)^0)) \wedge$

$(Move(A, Table, B)^0 \Rightarrow On(A, Table)^0 \wedge Clear(A)^0 \wedge Clear(B)^0)) \wedge$

$(Move(B, Table, A)^0 \Rightarrow On(B, Table)^0 \wedge Clear(B)^0 \wedge Clear(A)^0)) \wedge$

$(\neg Move(A, Table, B)^0 \vee \neg Move(B, Table, A)^0) \wedge On(A, B)^1$

The complexity of classical planning

PlanSAT is the question whether there exists any plan that solves a given planning problem

Bounded PlanSAT is the question whether there exists a plan of length k or less

PlanSAT is about **satisficing** (want any solution, not necessarily the cheapest or the shortest)

Bounded PlanSAT can be used to ask for the **optimal** solution

If in the PDDL language we do not allow functional symbols, both problems are decidable

Complexity of both problems is PSPACE (can be solved by a Turing machine which uses polynomial amount of space)

NP $\subseteq$ PSPACE (PSPACE is even harder than NP)

Some of the top-performing systems

International planning competition winners (from Russell and Norvig, 3rd edition):

Year	Track	Winning systems (approaches)
2008	Optimal	Gamer (symbolic bi-directional search)
2008	Satisficing	LAMA (fast forward search with FF heuristic)
2006	Optimal	SATPlan, MAXPlan (boolean satisfiability)
2006	Satisficing	SGPlan (forward search, partition into independent subproblems)
2004	Optimal	SATPlan (boolean satisfiability)
2004	Satisficing	Fast Diagonally Forward (forward search with causal graph)
2002	Automated	LPG (local search, constraint satisfaction)
2002	Hand-coded	TLPLan (temporal action logic with control rules for forward search)
2000	Automated	FF (forward search)
2000	Hand-coded	TalPlanner (temporal action logic with control rules for forward search)
1998	Automated	IPP (planning graphs); HSP (forward search)

What follows for the algorithms

If a planning system based on a particular planning algorithm is very fast it does not mean necessarily that the algorithm is 'better'

Sat solvers are very fast, but one may argue that it is not very practical to propositionalise planning problems

Forward planning with good heuristics can be very fast but again may be not always practically possible

Partial order planners are considered to be very flexible and generally useful, although they don't feature in the winners table...

Next lecture

Planning with resource constraints

Russell and Norvig 3rd edition Section 11.1

Russell and Norvig 2nd edition Section 12.1