Artificial Intelligence Foundations and Applications

Planning – Part 3

Planning as Satisfiability : SATPLAN

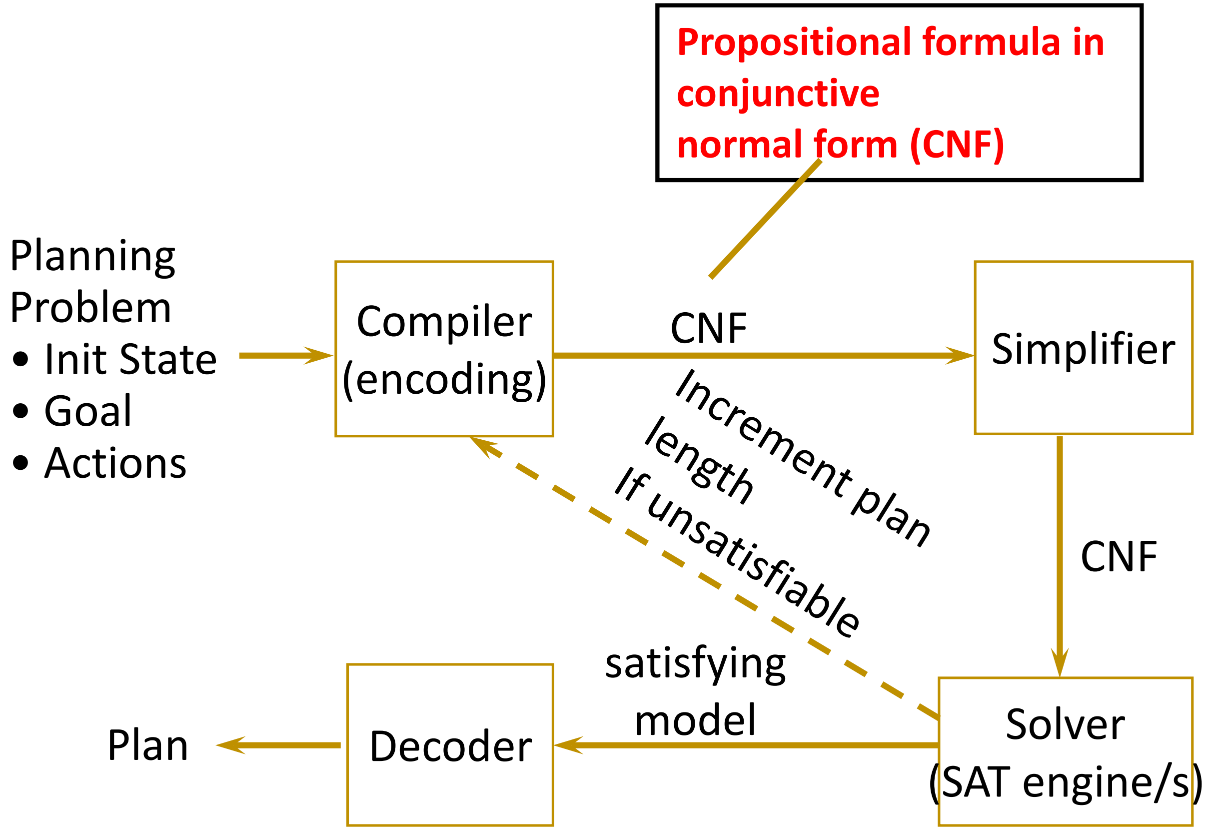
Sudeshna Sarkar

Oct 31 2022

Centre of Excellence in Artificial Intelligence, IIT Kharagpur

Planning with Propositional Logic

- The planning problem is translated into a CNF satisfiability problem
- The goal is asserted to hold at a time step T, and clauses are included for each time step up to T.
- If the clauses are satisfiable, then a plan is extracted by examining the actions that are true.
- Otherwise, we increment T and repeat



Propositional Logic: CNF

- A ***literal*** is either a proposition or the negation of a proposition
- A ***clause*** is a disjunction of literals
- A formula is in ***conjunctive normal form (CNF)*** if it is the conjunction of clauses

  $(\neg R \lor P \lor Q) \land (\neg P \lor Q) \land (\neg P \lor R)$

- Any formula can be represented in conjunctive normal form (CNF)
  - Though sometimes there may be an exponential blowup in size of CNF encoding compared to original formula
- CNF is used as a canonical representation of formulas in many algorithms

Propositional Satisfiability

- A formula is ***satisfiable*** if it is true for some truth assignment
  - e.g. $A \vee B$, C
- A formula is unsatisfiable if it is never true for any truth assignment
  - e.g. $A \wedge \neg A$
- Testing satisfiability of CNF formulas is an NP-complete problem

<u>Bounded PlanSAT</u>

Given: a planning problem, and positive integer *n*

Output: "yes" if problem is solvable in *n* steps or less, otherwise "no"

Bounded PlanSAT can be encoded as propositional satisfiability

Encoding Planning as Satisfiability

Bounded planning problem (*P*,*n*):

- *P* is a planning problem; *n* is a positive integer
- Find a solution for *P* of length *n*

Create a propositional formula that represents:

- Initial state
- Goal
- Action Dynamics

for *n* time steps

We will define the formula for (*P*,*n*) such that:
1) ***any*** satisfying truth assignment of the formula represent a solution to (*P*,*n*)
2) if (*P*,*n*) has a solution then the formula is satisfiable

for $T = 0$ to n do

cnf, mapping ← Translate2SAT (P, T)
assn ← Sat-Solver (cnf)

if assn is not null then

return Extract-soln (assn, mapping)

SATPlan: Planning as SAT

- Create a binary variable for each possible action a
  - a^i TRUE if action a is used at step i
- Create variables for each proposition that can hold at different points in time:
  - p^i TRUE if proposition p holds at step i

Constraints:

- XOR: Only one action can be executed at each time step
- At least one action must be executed at each time step
- Constraints describing effects of actions
- Maintain action: if an action does not change a prop p, then maintain action for proposition p is true
- A proposition is true at step i only if some action made it true
- Constraints for initial state and goal state

Dinner Date problem

Initial Conditions:

cleanHands, quiet, garbage

Goal:

¬garbage, dinner, present

Actions:

carry

precondition:

effect: ¬garbage, ¬cleanHands

dolly

precondition

effect: ¬garbage, ¬quiet

cook

precondition: cleanHands

effect: dinner

wrap

precondition (quiet)

effect: (present))

- Code the Initial Conditions:
  $cleanHands^0$, $quiet^0$, $garbage^0$, $\neg dinner^0$, $\neg present^0$

- Guess a time when the goal conditions will be true, and code the goal propositions:
  $\neg garb^2$, $dinner^2$, $present^2$

Building CNF formulas for planning problems

- Code the preconditions and effects for each action.
- For the action to be executed at time t, its preconditions must be true at time t, and the effects will take place at time $t + 1$.
- This must be done for every time step and for every action

$$\mathrm{cook}^0 \rightarrow \mathrm{cleanhands}^0 \wedge \mathrm{dinner}^1$$
$$\mathrm{cook}^1 \rightarrow \mathrm{cleanhands}^1 \wedge \mathrm{dinner}^2$$

$$\mathrm{wrap}^0 \rightarrow \mathrm{quiet}^0 \wedge \mathrm{present}^1$$

Building CNF formulas for planning problems

The conditions under which a proposition does not change from time t to $t+1$ must also be specified.

- Frame axioms: if a proposition p was true at time t, and an action that does not affect p is executed, then p is true at time $t+1$.

$$\mathrm{garb}^0 \wedge \mathrm{cook}^0 \rightarrow \mathrm{garb}^1$$

$$\ldots$$

- Explanatory frame axioms state which actions could have caused a proposition to change:

$$\mathrm{garb}^0 \wedge \neg\mathrm{garb}^1 \rightarrow \mathrm{dolly}^0 \vee \mathrm{carry}^0$$

$$\ldots$$

- Full frame axioms also require the at-least-one axioms to ensure that an action is executed at each time step.

$$\mathrm{cook}^0 \vee \mathrm{wrap}^0 \vee \mathrm{dolly}^0 \vee \mathrm{carry}^0$$
$$\mathrm{cook}^1 \vee \mathrm{wrap}^1 \vee \mathrm{dolly}^1 \vee \mathrm{carry}^1$$

Solving SAT problems

- Systematic solvers perform a backtracking search in the space of possible assignments
  - DPLL (Davis Putnam Logemann Loveland)
  - backtrack search + unit propagation
- Stochastic solvers perform a random search.
  - GSAT
  - Walksat (Selman, Kautz & Cohen)
    greedy local search + noise to escape minima