Planning and Satisfiability
Planning and Satisfiability: Kautz & Selman ECAI-92, KR-96, AAAI-96

• Planning is traditionally seen as deduction:
  ■ Find a proof of the statement “Initial conditions ∧ Actions ⊃ Goals”

• Recall that satisfiability is the problem of finding a model of a set of formulas (axioms)
  ■ First-order satisfiability is not decidable (testing entailment is semi-decidable)
  ■ Propositional satisfiability is decidable (although NP-complete)

• Key insight: Encode a planning problem as a set of axioms with the following property:
  ■ Any model of the axioms corresponds to a valid plan

• Then a procedure that computes models (satisfying assignments) will effectively compute plans!

• Bonus: Fast SAT solvers will yield fast planners
Two issues:
- Encoding the planning problem
- Solving the SAT problem

– Weld, AI Magazine 20, 1999
Elements of an Encoding: Initial State

- The initial state is asserted as a conjunction of all literals at time 0 (including any closed world assumptions):

\[ garbage_0 \land cleanHands_0 \land quiet_0 \land \neg dinner_0 \land \neg present_0 \]
Elements of an Encoding: Goals

• To test for a plan of length $n$, assert all desired goal properties at time $2n$:
  For $n = 1$:
  $$\neg garbage_2 \land dinner_2 \land present_2$$
Elements of an Encoding: Action Occurrences

• The result (the plan) is a set of actions to be executed at specific times.

• For each odd time $t$ between 1 and $2n - 1$, we have a disjunction of all possible actions at that time:

$$(carry_1 \lor dolly_1 \lor cook_1 \lor wrap_1) \land$$

$$(carry_3 \lor dolly_3 \lor cook_3 \lor wrap_3) \land$$

$$\ldots$$

• Note: This can lead to many axioms if actions are parameterized.
Elements of an Encoding: Preconditions and Effects

• Actions imply their preconditions and effects: For any odd time $t$ between 1 and $2n - 1$, and for each action, an axiom states that execution of the action implies its preconditions hold at $t - 1$ and its effects hold at $t + 1$. For example:

$$(\neg cook_1 \lor \neg dinner_2) \land (\neg cook_1 \lor cleanHands_0)$$

• Note: This can lead to many axioms!
Elements of an Encoding: Frame Axioms

• Frame axioms state what isn’t changed as a result of an action; or

• Explanation axioms state what must have happened given that a change occurs (Schubert, J. Logic and Computation 4(5), 1994; Gerevini & Schubert, AAAI-98)

• For every odd time $t$ between 1 and $2n - 1$, for example:

$$\neg \text{present}_0 \land \neg \text{wrap}_1 \supset \neg \text{present}_2$$

$$\text{garbage}_0 \land \neg (\text{carry}_1 \lor \text{dolly}_1) \supset \text{garbage}_2$$

• Note: This can lead to many, many axioms!
Elements of an Encoding: Summary

• Initial Conditions

• Goals

• Action Occurrences

• Preconditions and Effects

• Frame Axioms

• Other axioms for technical reasons (see papers)
SAT Encoding Considerations

• Complexity of SAT is exponential in the size of the formula being tested

• Parameters of an encoding:
  ■ Number of propositional variables
  ■ Number of clauses
  ■ Number of literals (in all clauses)
SAT Encoding Complexity

• Let $\mathcal{A}$ be the number of actions

• Let $\mathcal{F}$ be the number of fluents

• Linear encoding (Kautz & Selman ECAI-92):
  ■ Number of propositional variables: $O(n|A| + n|F|)$
  ■ Number of literals is dominated by frame axioms: $O(n|A|^2 + n|A||F|)$

• Operator splitting (lifting) reduces both numbers

• Explanatory frame axioms: $O(n|F|)$ (but each clause may be longer; worst-case same total size)

• Many more optimizations: see Kautz, McAllester, & Selman, KR-96
Solving SAT Problems

• Systematic (deterministic) solvers

• Stochastic solvers
Recent Advances in AI Planning: Summary

- GraphPlan precomputes constraints to improve the search for plans
- SAT Planners use the power and scalability of SAT solvers to solve large problems
- Both approaches (and they can be combined) exploit the fact that planning problems are instances of more general reasoning problems (constraint satisfaction and satisfiability, respectively)
- Although planning is still a difficult computational problem, these techniques are allowing fully automated systems to solve real-world problems with practical importance (i.e., $$$)