Planning as Satisfiability

Ron Parr
CPS 570

Planning vs. Satisfiability

- Planning is PSPACE-hard
- Satisfiability is NP-complete
- PSPACE-hard is (almost certainly) harder than NP-hard
- Q: How can you use satisfiability to solve planning?
- A: Bounded length planning is in NP

Propositionalizing (Grounding Out)

- Planning domains typically expressed in relational (first order/predicate) form
- SAT is propositional
- What do we do?
  - Start with situation calculus-like axioms, replacing situations with time indices
  - Create propositions for every substitution of objects into predicates
  - Each schema corresponds to a large number of propositions
- Note: This can all be done automatically, but adding some human expertise can help

Why does reducing the number of arguments help so much?

- O(n^3) vs. O(n^2)
- In general if you have n objects and your predicates take k arguments you create O(n^k) propositions for each predicate!
What does a plan look like?

• Plan will be a satisfying assignment to a (huge) SAT instance

• By checking which action variables are true at any time, we can construct a sequence of actions to achieve the goal

Models

• With situation calculus/planning as theorem proving, we ask the theorem prover if a situation exists where goal is achieved
  – Situations are created by stringing together actions
  – Goal must be necessarily true final situation for theorem prover to succeed

• Satisfiability asks if single assignment of truth values to variables satisfies a logical expression
  – Need to avoid degenerate solutions explicitly
  – False preconditions can make expression true

DP vs. GSAT

• DP is a complete algorithm
  – Exhaustive search with some tricks
  – Can work well for some large problems where tricks allow it to narrow search space
  – Can take exponential time in worst case
  – Even medium problems might take a long time

• GSAT is not complete
  – Can be very fast in some cases
  – Can fail completely in others
  – Lack of systematic search could cause failures even for small problems in some cases

Adding additional axioms

• Why this helps: Seems to reduce search space to rule out bad solutions, but...

• Other results suggests that higher clause to variable ratio leads to worse performance

• Q: Is there an inconsistency here?

• Planning problems are not randomly generated SAT instances – SAT instances that come from planning problems have more structure and symmetries
Graphplan and mutex propagation

- Graphplan is another highly competitive approach to planning
- Related to constraint satisfaction problems (CSPs) and graphical models of CSPs
- Key idea: mutex propagation
  - Narrows search space by ruling out impossible combinations
  - This idea can be generalized to other methods – similar to adding axioms for SATPlan, but can be done more aggressively than in first paper

Are there other examples of things that translate well to SAT?

- This is often- tried, sometimes with success (RP is not expert in this)
- SATPlan is one of the most notable, high profile successes
- Other fields have different canonical problems
- Operations research uses integer programs as their basic language for NP-complete problems