A Counterexample Based Approach for Quantifier Instantiation in SMT

Andrew Reynolds, University of Iowa
MVD, September 30, 2011
Overview

- Introduction to Satisfiability Modulo Theories (SMT)
- Extending SMT to Quantifiers
- Approaches to Quantifier Instantiation
  - E-Matching
  - Model-Based Quantifier Instantiation
  - New: Counterexample-Based Approach
- Current Work
Satisfiability Modulo Theories (SMT)

- SMT extends boolean satisfiability problems to theories

\[ F = \{ ( f(c) = a \lor c + 4 > a ), ( a = g(b) ) \} \]

- Construct satisfying assignment \( M \) for set of clauses \( F \)
  - i.e. \( M = \{ f(c) = a, a = g(b) \} \)

- Is this assignment consistent according to theory reasoning?
DPLL(T) Architecture

- SMT uses DPLL(T) architecture
- Operates on states of the form \( M \parallel F \)
  - F is a set of clauses
  - M is a set of asserted theory literals “L”
    - Literals may be decisions “L^d”
DPLL(T) Architecture

- For a DPLL(T) state $M \parallel F$,
  - SMT solver can answer UNSAT if:
    - Some clause in $F$ is falsified by $M$, and
    - $M$ contains no decision literals $L_d$
  - SMT solver can answer SAT if:
    - Each clause in $F$ is satisfied in $M$, and
    - $M$ is T-consistent
DPLL(T) Architecture

- Clauses to add to F
- SAT Solver
- Theory Solvers

F is UNSAT

M is T-Inconsistent

Satisfying assignment M

M is T-Consistent

F is SAT

SAT
Role of Theory Solver T in SMT

- Accepts a set of theory literals M
  - Determine if M is T-consistent
  - If not, add lemmas C to F, where each C is T-valid
- Typically, use SMT for decidable logics
  - Quantifier-free UF, Linear Real Arithmetic, etc.
- Also may be interested in other logics
  - Non-linear arithmetic, quantified logics, etc.
Quantifiers in SMT

- Universal and existential quantifiers
  - $\forall x. \phi$, $\exists x. \phi$
  - Treated as literals by the SAT solver
- Relegate these literals to quantifiers module
  - Role is similar to theory solver
  - Checking $T$-consistency is undecidable
    - When $\forall x. \phi$ is asserted, cannot answer SAT
- When asked whether $M$ is $T$-consistent, and there is a $\forall x. \phi$ asserted in $M$, either:
  - Answer UNKNOWN
  - Add (instantiation) clause ($\neg \forall x. \phi \lor \phi[s/x]$) to $M$
Quantifiers in SMT: Challenges

(1) Finding relevant instantiations
   ▸ How do we determine ground terms?

(2) Deciding when providing instantiations is no longer worthwhile
   ▸ When should we answer UNKNOWN?

(3) Determining if all necessary instantiations have been applied
   ▸ Can we answer SAT?
Related Work: E-matching

• **Address challenge (1)**
  - Find relevant instantiations by matching terms in quantifiers $t[x]$ to ground terms $t[s/x]$

• **To construct instantiation for $\forall x.\phi$ :**
  - Find *trigger* $t$, where $x$ is in $\text{FV}(t)$
  - Find ground term $g$
  - Find substitution $[s/x]$ such that $t[s/x]$ is equivalent to $g$ modulo set of equalities $E$
    - “$t \text{ E-matches } g$”
  - Use $s$ to instantiate $\forall x.\phi$
Related Work: Model-Based Quantifier Instantiation (MBQI)

- **Address challenges (1) and (3)**
  - Determine if some model satisfies all quantifiers. If so, answer SAT. Otherwise, use values for which model fails to instantiate quantifiers.

- **Given asserted quantified formula $\forall x. \phi$:**
  - Build explicit model $M^I$ for ground clauses $F$
  - Replace uninterpreted symbols in $\phi$ to generate $\phi^I$
  - Determine the satisfiability of $R \land \neg \phi^I[e/x]$
  - If UNSAT, then $\forall x. \phi$ is valid in current context
    - Otherwise, model for $R \land \neg \phi^I[e/x]$ is used to instantiate $\forall x. \phi$
    - Rules out $M^I$ on subsequent iterations
MBQI Example

- Check satisfiability of $F \land \phi$
  
  $F$: $w \geq v + 2 \land f(v) \leq 1 \land f(w) \leq 3$
  
  $\phi$: $\forall \ i \ j. \ (i \leq j \Rightarrow f(i) \leq f(j))$

- Model $M^I$ for $F$:
  
  $v \rightarrow 0, w \rightarrow 2, f \rightarrow [0 \rightarrow 1, 2 \rightarrow 3, \text{else} \rightarrow 4]$

- Check satisfiability of $\neg \phi^I[e_i/i, e_j/j]$:
  
  $e_i \leq e_j \land \text{ite}(e_i=0, 1, \text{ite}(e_i=2, 3, 4)) = \text{ite}(e_j=0, 1, \text{ite}(e_j=2, 3, 4))$
Alternative Approach to MBQI

- MBQI builds explicit models $M^l$
  - Check sat for $R \land \neg \phi^l[e/x]$
- *Instead:* Reason about counterexample $e$ directly
  - Add clause containing $\neg \phi[e/x]$ to SMT solver
- Potential advantages:
  - Do not need to generate explicit models $M^l$
  - Reason about $\neg \phi[e/x]$ incrementally, using the same instance of SMT solver
Counterexample Lemma

- Write $\perp^\phi$ to denote literal meaning:
  "a counterexample to $\phi$ exists"

- SMT solver finds satisfying assignment to:

  $$(\phi \lor \perp^\phi)$$

  "either $\phi$ holds or a $\phi$ has a counterexample"

  $$(\perp^\phi \iff \neg\phi[e/x])$$

  "$\phi$ has a counterexample if and only if its negation holds for some value e"
Configurations for Quantifier/CE Literal

- \( \phi \) is not asserted in \( M \)
  - We don’t care about \( \phi \)

- \( \phi^{(d)} \) and \( (\bot \phi)^d \) are asserted in \( M \)
  - \( \phi \) is true but we might find a counterexample

- \( \phi^{(d)} \) and \( \neg\bot \phi \) are asserted in \( M \)
  - \( \phi \) is true and we know it does not have a counterexample

- Requirement: Never assert \( \neg(\bot \phi)^d \)
Recognizing SAT Instances with CE Literals

- If $\bot \phi$ is asserted negatively as a non-decision, then $\phi$ is valid in the current context
  - If this is true for all quantifiers $\phi$, then we may answer SAT
- Conceptually: axiom $\phi$ does not apply in the current context

- Example: $a=0 \land (\forall x. \ a > 0 \Rightarrow P(\ a, \ x ))$
  - $\bot \phi \iff (a > 0 \land \neg P(\ a, \ e ))$
Features of Counterexample-Based Approach

- May be able to recognize SAT instances
  - Cases when no quantified axiom applies, i.e. counterexample is unsatisfiable

- Use information about “e” for finding relevant instantiations
  - Theory-specific information
After finding satisfying assignment to $\neg \phi[e/x]$

- Each theory solver has theory-specific information/constraints involving $e$

- How can we use this information?
  - Naively, find arbitrary model and use value of $e$ to instantiate $\phi$
Theory-Specific Instantiators

- Can we do better?
- For each theory, associate an *instantiator*
  - Has access to internal information stored in theory solver
Using Relationships between Triggers

- For EUF:
  - Search method for finding relevant instantiations
    - For literal $t[e/x] = s$, first try to find match $t[g/x]$ in the equivalence class of $s$
  - Criteria for judging relevance of instantiations
    - Do not consider instantiations $g$ where $e = g$ is unsatisfiable
Quantifier Instantiation for EUF

- **Multiple Iterations:**
  1. Find if $e = s$ is entailed for some ground term $s$
  2. Find if there exists some $s$ such that all requirements for $e$ are entailed by $e = s$
  3. Find if there exists some $s$ such that some requirements for $e$ are (partially) matched by $e = s$
  4. Do E-matching

- Otherwise, see if (explicit) model can be constructed
Current Work

- Optimizations
  - Computing matches efficiently (i.e. indexing, caching)
- Using splitting on demand
  - Matching failed because $c_1$ and $c_2$ are not entailed to be equal
  - Add lemma ($c_1 = c_2 \lor c_1 \neq c_2$)
- Quantifier Instantiation for Arithmetic
- Recognizing Other SAT instances
  - If no matches can be found, construct explicit model $M^l$ and see if MBQI succeeds
  - Construction of $M^l$ based on information about $e$
- Backtracking decisions
  - If stuck, explore another part of the search space
Questions?