Model-Based Reasoning about Quantified Formulas in CVC4

Andrew Reynolds

May 23, 2013

University of Iowa
Outline

• Introduction to SMT and applications
• Model-Based approach for handling quantifiers
• How can we construct good models?
• Experimental Results
Satisfiability Modulo Theories (SMT)

• SMT solvers are powerful tools
  – Used in many formal methods applications
  – Support many background theories
    • Arithmetic, bitvectors, arrays, datatypes, ...
  – May generate:
    • Proofs
      – Theorem proving, software/hardware verification
    • Models
      – Failing instances of aforementioned applications
      – Invariant synthesis, scheduling, test case generation
Satisfiability Modulo Theories

\(( f(a) = b \lor f(a) = c ) \land c+1 = b \land \forall x. f(x) = g(x) \)
Satisfiability Modulo Theories

\[( f(a) = b \lor f(a) = c ) \land c + 1 = b \land \forall x. f(x) = g(x) \]

\[\downarrow\text{ Abstract to propositional logic}\]

\[( A \lor B ) \land C \land D \]
Satisfiability Modulo Theories

\[( f(a) = b \lor f(a) = c ) \land c + 1 = b \land \forall x. f(x) = g(x) \]

\[\begin{align*}
(A \lor B) \land C \land D \land \text{true} \\
\text{true} \land \text{true} \land \text{true}
\end{align*}\]

Find satisfying assignment: A, C, D
Satisfiability Modulo Theories

\[(f(a) = b \vee f(a) = c) \land c + 1 = b \land \forall x. f(x) = g(x)\]

Find satisfying assignment: \(A, C, D\)

Check T-consistency: \(f(a) = b, c + 1 = b, \forall x. f(x) = g(x)\)
DPLL(T) Architecture

- **Formula $F$**
  - If $F$ is sat, go to SAT Solver.
  - If $F$ is unsat, go to UNSAT, proof.

- **SAT Solver**
  - If $F$ is sat, go to Theory Solvers.
  - If $F$ is unsat, go to UNSAT, proof.
  - Clauses to add to $F$.

- **Theory Solvers**
  - Satisfying assignment $A$ for $F$.
  - $A$ is T-Consistent.
  - $A$ is T-Inconsistent.

- **UNSAT, proof**
  - SAT, model.

- **SAT, model**
DPLL(T) Architecture: Challenge

Formula $F$

F is sat

SAT Solver

$A := \{ f(a) = b, c+1 = b, \forall x. f(x) = g(x) \}$

F is unsat

Theory Solvers

$A$ is T-Consistent

Clauses to add to $F$

$A$ is T-Inconsistent

SAT, model

• Challenge: What if determining the consistency of $A$ is difficult?
• For quantified formulas, determining consistency is undecidable
SMT with Quantified Formulas

• When quantified formulas are asserted
  – Most SMT solvers will:
    • Answer unsat, if they happen to find a proof
    • Run indefinitely
    • Give up, reporting “unknown”
Why Models are Important

⇒ Solver needs way of answering satisfiable when quantified formulas are asserted
Model-Based Approach for Quantifiers

• Given:
  – Set of ground formulas $F$
  – Set of quantified axioms $Q$

• Determine the satisfiability of $F \land Q$

• Idea:
  – Construct candidate models for $Q$ based on satisfying assignments for $F$
    • Ge and deMoura [2009]
Model-Based Approach for Quantifiers

SAT Solver

Theory Solvers

Model Verifier

Ground Formulas $F$

Quantified Formulas $Q$

$F$ is sat

Satisfying assignment $A$ for $F$

$A$ is T-Consistent

Candidate model $M$

$M$ is a model for $Q$

UNSAT, proof

Clauses to add to $F$

SAT, model $M$
Running Example

person₁, person₂, person₃ : Person
blue, brown, blonde : Color
eye, hair : Person -> Color

distinct(blue, brown, blonde)
hair(person₁) = brown
eye(person₂) = blue
hair(person₃) = blonde
∀ x : Person. eye(x)=blue ⇒ hair(x)=blonde
Running Example

\[
\begin{align*}
\text{distinct(brown, blue, blonde)} \\
\text{hair(person}_1\text{)} &= \text{brown} \\
\text{eye(person}_2\text{)} &= \text{blue} \\
\text{hair(person}_3\text{)} &= \text{blonde} \\
\forall x : \text{Person}. \text{eye(x)=blue} &\implies \text{hair(x)=blonde}
\end{align*}
\]
distinct(brown, blue, blonde)

\[ \text{true} \]

\[ \text{true} \]

\[ \text{true} \]

\[ \text{true} \]

\[ \forall x : \text{Person. eye}(x) = \text{blue} \Rightarrow \text{hair}(x) = \text{blonde} \]

• A is Theory-Consistent according to the theory of equality
From $\mathbb{A}$, construct candidate model $M$.

$\mathbb{A} :=$

\[
\{ \text{distinct(brown, blue, blonde)}, \\
\text{hair(person}_1\text{) = brown,} \\
\text{eye(person}_2\text{) = blue,} \\
\text{hair(person}_3\text{) = blonde} \}
\]

$M :=$

\[
\text{hair : person}_1 \rightarrow \text{brown} \\
\text{person}_3 \rightarrow \text{blonde} \\
\text{else} \rightarrow \text{brown} \\
\text{else} \rightarrow \text{blue}
\]

\[
\text{eye : person}_2 \rightarrow \text{blue} \\
\text{else} \rightarrow \text{blue}
\]
Check whether $\mathcal{M}$ is a model of $\mathcal{Q}$

$\mathcal{M} :=$

- hair : $\text{person}_1$ -> brown
- $\text{person}_3$ -> blonde
- else -> brown
- eye : $\text{person}_2$ -> blue
- else -> blue

$\mathcal{Q} :=$

- $\forall x : \text{Person. eye}(x) = \text{blue} \Rightarrow \text{hair}(x) = \text{blonde}$

- $\mathcal{Q}$ is false for $\text{person}_2$
Check whether $\mathcal{M}$ is a model of $\mathcal{Q}$

$\mathcal{M} :=$

- hair : person$_1$ -> brown
- person$_3$ -> blonde
- else -> brown
- eye : person$_2$ -> blue
- else -> blue

$\mathcal{Q} :=$

- $\forall$ x : Person. eye(x)=blue $\Rightarrow$ hair(x)=blonde
- $\mathcal{Q}$ is false for person$_2$

• Add (eye(person$_2$)=blue $\Rightarrow$ hair(person$_2$)=blonde) to $\mathcal{F}$
• Will rule out $\mathcal{M}$ on next iteration
  • Can be thought of as model “refinement”
What are good candidate models?

• Good candidate models
  – Have small domain sizes
  – Most instances of axioms $\mathcal{Q}$ are likely to be true

• For small domain sizes,
  – Use specialized theory solver within DPLL(T)

• For making most instances true,
  – Use ground solver to guide model construction

• These features are implemented in SMT solver CVC4
Finding Minimal Models in DPLL(T) Search

- Idea: try to fix domain sizes 1, 2, 3, ....
  - Prioritize decisions made by DPLL(T) search

Search for models of size=1

|Person|≤1

If none exist, search for models of size=2

|Person|≤2

distinct(brown, blue, blonde)

|Person|≤3

- |Person|≤1

- |Person|≤2

- |Person|≤3

- |Person|≤3

- etc.

- eye(person\textsubscript{2}) = blue

- hair(person\textsubscript{3}) = blonde
Finding Minimal Models in DPLL(T) Search

\[ |\text{Person}| \leq 1 \quad \rightarrow \quad \neg |\text{Person}| \leq 1 \]

Fails: \( \text{person}_1 \neq \text{person}_3 \)

Success: Can identify \( \text{person}_1 = \text{person}_2 \)

distinct(brown, blue, blonde)

\( \text{hair(person}_1) = \text{brown} \)

\( \text{eye(person}_2) = \text{blue} \)

\( \text{hair(person}_3) = \text{blonde} \)

• Implementation in CVC4 uses:
  – Splitting on demand to shrink model sizes
  – Efficient methods for clique detection

\[ \Rightarrow \text{Theory of finite cardinality constraints [CAV 2013]} \]
• Set $S$ can be very large
  – For $Q$ with $n$ variables with domain size $d$, $|S|$ can be $O(d^n)$
• Would prefer if most instances of $Q$ are true in $M$
Constructing Good Candidate Models

• Idea for axiom $\mathcal{Q}$:
  – Chose default values in model $\mathbb{M}$ based on one satisfying ground instance of $\mathcal{Q}$

$\mathcal{Q}$

- distinct(brown, blue, blonde)
- hair(person$_1$) = brown
- eye(person$_2$) = blue
- hair(person$_3$) = blonde
- $\forall x : \text{Person. }$ eye($x$)=blue $\Rightarrow$ hair($x$)=blonde

• See how $\mathcal{Q}$ is satisfied for one instance, then generalize this [CADE 2013]
Constructing Good Candidate Models

- distinct(brown, blue, blonde)
- hair(person₁) = brown
- eye(person₂) = blue
- hair(person₃) = blonde
- ∀ x : Person. eye(x)=blue ⇒ hair(x)=blonde

- Consider $Q[person₁/x]$

$Q$

![Diagram with formulas and variables](image-url)
Constructing Good Candidate Models

- distinct(brown, blue, blonde)
- hair(person₁) = brown
- eye(person₂) = blue
- hair(person₃) = blonde
- \( \forall x : \text{Person. eye}(x) = \text{blue} \implies \text{hair}(x) = \text{blonde} \)

- Find satisfying assignment
Constructing Good Candidate Models

\[ \begin{align*}
\text{distinct(brown, blue, blonde)} \\
\text{hair(person}_1\text{) = brown} \\
\text{eye(person}_2\text{) = blue} \\
\text{hair(person}_3\text{) = blonde} \\
\forall \, x : \text{Person. eye}(x) = \text{blue} \Rightarrow \text{hair}(x) = \text{blonde} \\
\text{eye(person}_1\text{)} = \text{blue} \Rightarrow \text{hair(person}_1\text{)} = \text{blonde}
\end{align*} \]

• Construct candidate model

\[ \begin{align*}
\mathcal{A} := \\
\{ \text{distinct(brown, blue, blonde)}, \\
\text{hair(person}_1\text{) = brown}, \\
\text{eye(person}_2\text{) = blue}, \\
\text{hair(person}_3\text{) = blonde}, \\
\text{eye(person}_1\text{)} \neq \text{blue} \} \\
\mathcal{M} := \text{hair : person}_1\rightarrow\text{brown} \\
& \text{person}_3\rightarrow\text{blonde} \\
& \text{else} \rightarrow \ldots \\
& \text{eye : person}_1\rightarrow\text{brown} \\
& \text{person}_2\rightarrow\text{blue} \\
& \text{else} \rightarrow \ldots
\end{align*} \]
Constructing Good Candidate Models

\( M := \begin{cases} 
\text{hair} : & \text{person}_1 \to \text{brown} \\
& \text{person}_3 \to \text{blonde} \\
& \text{else} \to \text{brown} \\
\text{eye} : & \text{person}_1 \to \text{brown} \\
& \text{person}_2 \to \text{blue} \\
& \text{else} \to \text{brown}
\end{cases} \)

\( A := \{ \text{distinct(brown, blue, blonde)}, \\
\text{hair(person}_1) = \text{brown}, \\
\text{eye(person}_2) = \text{blue}, \\
\text{hair(person}_3) = \text{blonde}, \\
\text{eye(person}_1) \neq \text{blue} \} \)
Model-Based Approach in CVC4

• CVC4 is state of the art SMT solver with
  – Support for many theories

• Features implemented in CVC4:
  – Theory solver for handling cardinality constraints
  – Techniques for constructing candidate models
  – Efficient methods for verifying candidate models
    • Not mentioned in this talk
Experiments

• DVF Benchmarks
  – Taken from verification tool DVF used by Intel
  – Both SAT/UNSAT benchmarks
    • SAT benchmarks generated by removing necessary $pf$ assumptions
  – Many theories: UF, arithmetic, arrays, datatypes
  – Quantifiers only over free sorts
    • Memory addresses, Values, Sets, ...

• TPTP Benchmarks
  – Unsorted, equality, function symbols
  – Heavy use of quantifiers
Experiments: DVF

<table>
<thead>
<tr>
<th>SAT</th>
<th>german</th>
<th>refcount</th>
<th>agree</th>
<th>apg</th>
<th>bmk</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
<td>45</td>
<td>6</td>
<td>42</td>
<td>19</td>
<td>37</td>
<td>149</td>
</tr>
<tr>
<td>cvc3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>yices</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>z3</td>
<td>45</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>46</td>
</tr>
<tr>
<td>cvc4</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>cvc4+f</td>
<td>45</td>
<td>6</td>
<td>42</td>
<td>19</td>
<td>37</td>
<td>149</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>UNSAT</th>
<th>german</th>
<th>refcount</th>
<th>agree</th>
<th>apg</th>
<th>bmk</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
<td>145</td>
<td>40</td>
<td>488</td>
<td>304</td>
<td>244</td>
<td>1221</td>
</tr>
<tr>
<td>cvc3</td>
<td>145</td>
<td>40</td>
<td>457</td>
<td>267</td>
<td>229</td>
<td>1138</td>
</tr>
<tr>
<td>yices</td>
<td>145</td>
<td>40</td>
<td>488</td>
<td>304</td>
<td>244</td>
<td>1221</td>
</tr>
<tr>
<td>z3</td>
<td>145</td>
<td>40</td>
<td>488</td>
<td>304</td>
<td>244</td>
<td>1221</td>
</tr>
<tr>
<td>cvc4</td>
<td>145</td>
<td>40</td>
<td>484</td>
<td>304</td>
<td>244</td>
<td>1217</td>
</tr>
<tr>
<td>cvc4+f</td>
<td>145</td>
<td>40</td>
<td>471</td>
<td>300</td>
<td>242</td>
<td>1198</td>
</tr>
</tbody>
</table>

- Configurations:
  - cvc4 : heuristic inst.
  - cvc4+f : model-based

- cvc4+f effective for sat
- cvc4+f solves 4 unsat that cvc4 cannot
Experiments: TPTP

• For 1995 satisfiable benchmarks:
  – Paradox solves 1305
  – iProver solves 1231
  – z3 solves 887
  – cvc4+f solves 1186
    • Includes 3 problems with rating 1.0

• For 12568 unsatisfiable benchmarks:
  – z3 solves 5934
  – iProver solves 5556
  – cvc4 solves 5415
  – cvc4+f solves 3028
    • Orthogonal to other approaches
    • 282 cannot be solved by z3
Summary

• Completed work in CVC4:
  – Ground solver for finding small models
  – Methods for constructing and verifying candidate models


• Current work:
  – Fair strategies for minimizing models for multiple sorts
  – Improve existing approaches for answering UNSAT
  – Other applications
    • Theory of Strings : bounded length
    • Integer quantification within bounded ranges
Current Work

• Extension to bounded integer quantification
  – Can use similar approach

∀ x : Int. 0 ≤ x ≤ N ⇒ P(x)
Thanks

• Collaborators:
  – Cesare Tinelli, Amit Goel, Sava Krstic, Clark Barrett, Morgan Deters, Leonardo de Moura

• Questions?