An SMT-Compliant Solver for the Existential Fragment of Real Algebra
Work-In-Progress Presentation
CAI 2011 Contribution

Ulrich Loup, Erika Ábrahám

July 2011
Motivation: bounded model checking for hybrid systems

Model of a bouncing ball

\[\begin{align*}
x &= 16 \\
v &= 0 \\
t &= 0
\end{align*}\]

\[\begin{align*}
x &\geq 0 \\
\dot{x} &= v \\
\dot{v} &= -10 \\
\dot{t} &= 1
\end{align*}\]

\[\begin{align*}
x &= 0 \\
t &> 0 \\
\dot{x}' &= x \\
\dot{v}' &= -\frac{v}{2} \\
\dot{t}' &= 0
\end{align*}\]
Motivation: bounded model checking for hybrid systems

Model of a bouncing ball

\[ x = 16 \]
\[ v = 0 \]
\[ t = 0 \]

\[ x \geq 0 \]
\[ \dot{x} = v \]
\[ v = -10 \]
\[ \dot{t} = 1 \]

\[ x = 0 \]
\[ t > 0 \]
\[ x' = x \]
\[ v' = -\frac{v}{2} \]
\[ t' = 0 \]

Encoding a run as a formula

\[ \varphi = I(0) \land C(1) \land D(2) \land \ldots \land C(k) \land 0 < x_k < 1 \]

\text{Var: } x_i, v_i, t_i \text{ for } 0 \leq i \leq k

I(i): \quad (x_i = 16 \land v_i = 0 \land t_i = 0)

C(i): \quad (x_i \geq 0 \land t_i \geq 0 \land v_i = v_{i-1} - 10t_i \land x_i = x_{i-1} - 5t_i^2)

D(i): \quad (x_{i-1} = 0 \land t_{i-1} > 0 \land v_i = -\frac{v_{i-1}}{2} \land x_i = x_{i-1})
Satisfiability-modulo-theories (SMT) solving

$\varphi$

Boolean skeleton  \rightarrow  SAT-solver  \rightarrow  SAT/UNSAT

Add/Delete constraints  \rightarrow  Theory solver

Provide assignment/reason
Satisfiability-modulo-theories (SMT) solving

\[ \varphi \]

Boolean skeleton

SAT-solver

SAT/UNSAT

Add/Delete constraints

Theory solver

Provide assignment/reason
Satisfiability-modulo-theories (SMT) solving

Solves

\[
\begin{pmatrix}
  P_1 & \sim_1 & 0 \\
  \vdots \\
  P_m & \sim_m & 0
\end{pmatrix}
\]

where

\[P_i \in \mathbb{Z}[x_1, \ldots, x_n], n \in \mathbb{N},\]

\[\sim_i \in \{<, =, >\}\]

for \(1 \leq i \leq m\).
Exact SMT-solving over the reals today

Linear real arithmetic

- Yices
- Z3
- MathSAT
- OpenSMT
- ABsolver

The quadratic case and beyond

- ABsolver
- Virtual substitution solver [ÁCL10]
- Z3, CVC3 (under development)
Exact SMT-solving over the reals today

Linear real arithmetic

- Yices
- Z3
- MathSAT
- OpenSMT
- ABsolver

The quadratic case and beyond

- ABsolver
- Virtual substitution solver [ÁCL10]
- Z3, CVC3 (under development)

Arbitrary degree polynomials

Our contribution
Real-Algebraic Solving
Methods available

- First decision procedure [T48] (1948)
- Cylindrical algebraic decomposition (CAD) method [C75] (1975)
- Partial CAD [CH91] (1991)
Methods and Tools available

- 1948: First decision procedure [T48]
- 1975: Cylindrical algebraic decomposition (CAD) method [C75]
- 1991: Partial CAD [CH91]
- 1993: Gröbner basis approach [W93]
- 1997: Virtual Substitution [W97]
- 1997: Computing Realizable Sign Conditions [BPR97]
- 2003: Redlog [DS97]
- 2003: QEPCAD [B03]
- 2010: Virtual substitution solver (prototype) [ÁCL10]
- 2010: Z3, CVC3 (under development)
Methods and Tools available

1948: First decision procedure [T48]
1975: Cylindrical algebraic decomposition (CAD) method [C75]
1991: Partial CAD [CH91]
1993: Gröbner basis approach [W93]
1997:
   - Virtual Substitution [W97]
   - Computing Realizable Sign Conditions [BPR97]
   - Redlog [DS97]
2003: QEPCAD [B03]
2010:
   - Virtual substitution solver (prototype) [ÁCL10]
   - Z3, CVC3 (under development)
Realizable sign conditions

"constraint system" \[\begin{pmatrix} P_1 & \sim_1 & 0 \\ \vdots \\ P_m & \sim_m & 0 \end{pmatrix}\] \[\sim\rightarrow\] "sign condition" \[\begin{pmatrix} \sgn(P_1) = \sigma_1 \\ \vdots \\ \sgn(P_m) = \sigma_m \end{pmatrix}\]

\[P = (P_i)_{i=1}^m\]
\[\sigma = (\sigma_i)_{i=1}^m\]

Real-Algebraic Solving
Realizable sign conditions

\[
\begin{pmatrix}
P_1 & \sim_1 & 0 \\
\vdots \\
P_m & \sim_m & 0
\end{pmatrix}
\xrightarrow{\text{"constraint system"}}
\begin{pmatrix}
\text{sgn}(P_1) = \sigma_1 \\
\vdots \\
\text{sgn}(P_m) = \sigma_m
\end{pmatrix}
\xrightarrow{\text{"sign condition"}}
(P, \sigma)
\]

- \( (P, \sigma) \) realizable if \( \exists \ a \in \mathbb{R}^n: \forall \ 1 \leq i \leq m: \text{sgn}(P_i(a)) = \sigma_i \).

E.g. realizable sign conditions for...

- \( \ldots P = (x^2 + 2x - 1): \) \( (P, (-1)), (P, (0)), (P, (1)) \).
- \( \ldots P = (x^2 + 1): \) \( (P, (1)) \).
Computing realizable sign conditions

Two approaches [BPR10]

1. Compute \( \text{samples}(\mathcal{P}', (x_1, \ldots, x_n)) \) for any subset \( \mathcal{P}' \) of \( \mathcal{P} \).
   Complexity: \( 2^m d^{O(n)} \)

2. Compute \( \text{samples}(\tilde{\mathcal{P}}', (x_1, \ldots, x_n)) \) for sequences \( \tilde{\mathcal{P}}' \) with up to \( n \) elements containing deformed variants of polynomials in \( \mathcal{P} \).
   Complexity: \( m^{n+1} d^{O(n)} \)

\( \text{samples}(Q, (x_1, \ldots, x_n)) \) (black box!)

Input: \( Q \in \mathcal{R}[x_1, \ldots, x_n]^k \) with \( \mathcal{R} \in \{ \mathbb{Z}, \mathbb{Z}[\delta, \gamma] \} \), \( k, n \in \mathbb{N}_{\geq 1} \)

Output: samples in \( \mathbb{R}^k \) for \( Q \) in one of the approaches
SMT-Compliant Solving
SMT-solving requirements

- adding constraints incrementally
- backtracking
- generating minimal reasons

Φ

Boolean skeleton

SAT-solver

Add/Delete constraints

Provide assignment/reason

SAT/UNSAT

Theory solver
SMT-solving requirements

- adding constraints incrementally
- backtracking
- generating minimal reasons

I-RiSC:
- Incremental computation of
  - Realizable
  - Sign
  - Conditions
The I-RiSC search tree data structure

- Tree of depth $m$ (following approach 1)
- Inner node: choice whether $P_i$ selected for sample construction
- Leaves $w \in \{0, 1\}^m$

Traverse leaves w.r.t. order $\prec$: $w_1 \prec w_2 \iff |w_1|_1 < |w_2|_1$ or ($|w_1|_1 = |w_2|_1$ and $w_1 \leq_{\text{lex}} w_2$)
Adding sign conditions

1. 

2. Go back to smallest (w.r.t. $\prec$) leaf with empty samples or the previous leaf.

3. Restart search.
Backtracking

- Restore search state.
Minimal reasons

Let \( w \in \{0, 1\}^m \).

- \( P_w = (p_i \mid w_i = 1) \)
- \( \sigma_w = (\sigma_i \mid w_i = 1) \)
- \( S(w) \) set of all samples generated in \( w \) satisfying \( (P_w, \sigma_w) \)
Minimal reasons

Let $w \in \{0, 1\}^m$.

- $P_w = (p_i \mid w_i = 1)$
- $\sigma_w = (\sigma_i \mid w_i = 1)$
- $S(w)$ set of all samples generated in $w$ satisfying $(P_w, \sigma_w)$
- $w$ admissible if $S(w) \neq \emptyset$ and inadmissible otherwise.

$(P_w, \sigma_w)$ minimal reason if $w$ admissible and all inner nodes before (w.r.t. $\prec$) are admissible.
Example 1

Prerequisites

1. \( ((P_1, P_2), (\sigma_1, \sigma_2)), ((P_1, P_3), (\sigma_1, \sigma_3)) \) and \( ((P_1, P_4), (\sigma_1, \sigma_4)) \) realizable

2. \( ((P_2, P_3), (\sigma_2, \sigma_3)) \) not realizable

- Search tree after adding \( (P_1, \sigma_1) \):

```
       P_1
      / \  \
  0   1  \
  0   1
```
Example II

- Search tree after adding \((P_2, \sigma_2)\):
Example III

- Search tree after adding \((P_3, \sigma_3)\):

\[
P_3? \quad 0 \quad P_2\quad 1 \quad P_2? \quad 0 \quad P_2? \quad 1
\]

\[
P_3? \quad 0 \quad P_2? \quad 1 \quad P_2? \quad 0 \quad P_2? \quad 1
\]

\[
P_1? \quad 0 \quad P_1\quad 1 \quad P_1? \quad 1 \quad P_1? \quad 0
\]

\[
P_1? \quad 0 \quad P_1\quad 1 \quad P_1? \quad 0 \quad P_1? \quad 0
\]

\[
000 \quad 001 \quad 010 \quad 011
\]

\[
100 \quad 101 \quad 110 \quad \ldots
\]

\[
(P_{110}, \sigma_{110})
\]

not realizable
Example IV

- Search tree after backtracking 2 steps:
Example V

- Search tree after adding \((P_4, \sigma_4)\):

```
        P_4?
       /   \
      0   1
      / \
    P_1?  P_1?
   /  \
00  01 10 11
```
To conclude

Summary

▶ first approach to exact SMT-compliant solving for the whole existential fragment of real algebra

Outlook

▶ implementation into OpenSMT using GiNaCRA
▶ theory propagation
▶ coupling of I-RiSC and the virtual substitution solver
To conclude

Summary

- first approach to exact SMT-compliant solving for the whole existential fragment of real algebra

Outlook

- implementation into OpenSMT using GiNaCRA
- theory propagation
- coupling of I-RiSC and the virtual substitution solver

Contact: Ulrich Loup
RWTH Aachen University
http://www-i2.informatik.rwth-aachen.de/i2/loup/
References I


References II


