A Verified Decision Procedure for Univariate Real Arithmetic with the BKR Algorithm

Katherine Cordwell, Yong Kiam Tan, and André Platzer

Carnegie Mellon University
Problem

- Real arithmetic questions involving the $\exists$ (exists) and $\forall$ (for all) quantifiers (ranging over the reals) are difficult for computers
- **Quantifier elimination (QE):** The process of transforming a quantified statement into a *logically equivalent* quantifier-free statement
This example is taken from some of Pablo Parrilo’s lecture notes (Lecture 18 of his 2006 course, “Algebraic Techniques and Semidefinite Optimization”). Accessible through his webpage: https://www.mit.edu/~parrilo/index.html

Example

\[ \forall x. x^2 + 1 > 0 \]

QE

True

Example*

\[ \forall x \forall y. ((x^2 + ay^2 \leq 1) \Rightarrow (ax^2 - a^2 xy + 2 \geq 0)) \]

QE

\((a \geq 0) \text{ and } (a^3 - 8a - 16 \leq 0)\)

QE is identifying exactly what conditions on \(a\) will make the original formula true!
A Miraculous Result

- Algorithms for QE exist (Tarski, 1930)
- Algorithms for QE are complicated

Alfred Tarski
Terminology

- **Formulas:** Conjunctions and disjunctions of polynomial inequalities and equations (with rational coefficients)
- If a formula in a QE problem involves only one variable, we call it a **univariate** QE problem. Else it is a **multivariate** QE problem.
- **Decision problems** are problems where all variables are quantified.
Examples, Revisited

Example

\( \forall x. x^2 + 1 > 0 \)

\( \downarrow \)

True

A univariate decision problem

Example*

\( \forall x \forall y. ((x^2 + ay^2 \leq 1) \Rightarrow (ax^2 - a^2xy + 2 \geq 0)) \)

\( \downarrow \)

\( (a \geq 0) \) and \( (a^3 - 8a - 16 \leq 0) \)

A multivariate QE question

Not a decision problem

*This example is taken from some of Pablo Parrilo's lecture notes (Lecture 18 of his 2006 course, “Algebraic Techniques and Semidefinite Optimization”). Accessible through his webpage: https://www.mit.edu/~parrilo/index.html
Motivation

- Quantified statements arise in a number of applications
  - Geometry proofs
  - Stability analysis
  - Verification of cyber-physical systems (like robots!)

For more information, see:
Motivation

• Quantified statements arise in a number of applications
  ○ Geometry proofs
  ○ Stability analysis
  ○ Verification of cyber-physical systems (like robots!)

• Two conclusions
  ○ We want to know how to do QE
  ○ We want to be sure that we know how to do QE correctly
Motivation

- **What we want:** Formally verified QE algorithms
Motivation

- **What we want:** Formally verified QE algorithms
- **Problem:** Dearth of efficient verified QE support
  - CPS theorem prover KeYmaera X outsources QE to unverified software
  - This can introduce bugs
## Related Work

<table>
<thead>
<tr>
<th></th>
<th>Efficient?</th>
<th>Verified?</th>
<th>Multivariate case builds directly on univariate?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cohen-Hörmander</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Tarski</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CAD</td>
<td>✓</td>
<td></td>
<td>✗</td>
</tr>
</tbody>
</table>
## Related Work

<table>
<thead>
<tr>
<th></th>
<th>Efficient?</th>
<th>Verified?</th>
<th>Multivariate case builds directly on univariate?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cohen-Hörmander</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Tarski</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CAD</td>
<td>✓</td>
<td>[ ]</td>
<td>✗</td>
</tr>
<tr>
<td>BKR &amp; Renegar</td>
<td>[ ]</td>
<td>✗</td>
<td>✓</td>
</tr>
</tbody>
</table>

---

*Potential sweet spot!*
BKR and Renegar

- Originally BKR* was a decision procedure
- Renegar** extended BKR to a general-purpose QE algorithm
  - Explains BKR in more detail
  - Fixes an error in BKR’s multivariate complexity analysis

---


We formally verify* the univariate cases of BKR and Renegar in Isabelle/HOL.

*Available on the Archive of Formal Proofs at: https://www.isa-afp.org/entries/BenOr_Kozen_Reif.html
High-level Context

- ~7000 LOC
  - Algorithm: ~110 LOC
  - Matrix library extensions: ~1800 LOC
High-level Context

- ~7000 LOC
  - Algorithm: ~110 LOC
  - Matrix library extensions: ~1800 LOC
- Why Isabelle/HOL?
  - Well-suited to formalizing mathematics
  - Strong math libraries
  - Sledgehammer
Univariate BKR: Bird’s Eye View

- Transform the problem:
  1. Decision problems to sign determination
  2. Sign determination to restricted sign determination
  3. To solve restricted sign determination, set up a matrix equation.
Definition (sign assignment for \( \{g_1, \ldots, g_n\} \)). A mapping \( \sigma: \{g_1, \ldots, g_n\} \rightarrow \{+, -, 0\} \)
\( \sigma \) is consistent if there is a real \( x \) where, for all \( i \), the sign of \( g_i(x) \) matches \( \sigma(g_i) \).

Step 1: Decision to Sign Determination

- Solve decision problems by finding the consistent sign assignments (CSAs) for a set of polynomials (sign determination)
Step 1: Decision to Sign Determination

- Solve decision problems by finding the consistent sign assignments (CSAs) for a set of polynomials (sign determination).

Decision Problem: \( \exists x. (x^2+1 \geq 0 \wedge 3x < 0) \)

Find all consistent sign assignments for \( x^2 + 1 \) and \( 3x \)

CSAs: (+, -), (+, 0), (+, +)

CSA (+, -) indicates the existence of a point \( k \) with \( (k^2+1 \geq 0 \wedge 3k < 0) \)
Correctness Results for Step 1

```
theorem decision_procedure:
  "(∀ x::real. fml_sem fml x) ↔ decide_universal fml"
  "(∃ x::real. fml_sem fml x) ↔ decide_existential fml"
```

Canonical semantics for formulas (defines what it means for a formula to hold at x in the standard way)  

Our algorithms
Step 2: Restricted Sign Determination

- Restrict sign determination to finding all **CSAs** for a set of polynomials \( \{q_1, \ldots, q_n\} \) at the roots of an auxiliary nonzero polynomial \( p \)

Technical detail: BKR imposes some conditions on \( \{q_1, \ldots, q_n\}, p \)
Step 2: Restricted Sign Determination

- Restrict sign determination to finding all CSAs for a set of polynomials \{q_1, \ldots, q_n\} at the roots of an auxiliary nonzero polynomial p.

Diagram:
- Some root of p is less than all the roots of the q_i’s.
- p has a root in between any two roots of the q_i’s.
- Some root of p is greater than all the roots of the q_i’s.

The roots of all the q_i’s; also roots of p.
Correctness Results for Step 2

definition roots :: "real poly ⇒ real set" where "roots p = \{x. poly p x = 0\}"

definition consistent_signs_at_roots :: "real poly ⇒ real poly list ⇒ rat list set" where "consistent_signs_at_roots p qs = (sgn_vec qs) ′ (roots p)"

Plug in the roots to the q_i's, take the resulting signs

Solve for the roots of a polynomial
Correctness Results for Step 2

definition roots :: "real poly ⇒ real set" where "roots p = {x. poly p x = 0}"

definition consistent_signs_at_roots :: "real poly ⇒ real poly list ⇒ rat list set" where "consistent_signs_at_roots p qs = (sgn_vec qs) ⋆ (roots p)"

theorem find_consistent_signs_at_roots: assumes "p ≠ 0" assumes "∀ q. q ∈ set qs ⇒ coprime p q" shows "set (find_consistent_signs_at_roots p qs) = consistent_signs_at_roots p qs"

our (constructive) algorithm  the nonconstructive definition
Step 3: The Matrix Equation

- Stores all relevant information for sign determination
- Idea dates back to Tarski; similarities to Cohen and Mahboubi’s formalization*
- But BKR does it efficiently

Step 3: The Matrix Equation

Find sign assignments to $q_1, \ldots, q_n$ at the roots of $p$

$\begin{align*}
\text{Tarski} & \quad \begin{pmatrix}
\# \text{ of } (+, \ldots, +, +) \\
\# \text{ of } (+, \ldots, +, -) \\
\vdots \\
\# \text{ of } (-, \ldots, -, -)
\end{pmatrix} \\
& \quad = M^{-1} \ast \\
& \quad \begin{pmatrix}
\text{TQ subset 1} \\
\text{TQ subset 2} \\
\vdots \\
\text{TQ subset } 2^n
\end{pmatrix}
\end{align*}$

TQ stands for “Tarski query”, refers to invoking the (computational) Sturm-Tarski theorem

Invertible matrix
Size $2^n \times 2^n$
Can be computed
Step 3: The Matrix Equation

Find sign assignments to $q_1, \ldots, q_n$ at the roots of $p$

BKR builds its matrix equation (ME) inductively

- ME for $q_1$
- ME for $q_2$
- \ldots
- ME for $q_{n-1}$
- ME for $q_n$

REDUCE

- ME for $q_1, q_2$
- \ldots
- ME for $q_{n-1}, q_n$

- \ldots

REDUCE

- ME for $q_1, \ldots, q_n$
Step 3: The Matrix Equation

After each combination, remove all inconsistent sign assignments (reduction step)

\[
\begin{bmatrix}
1 & 1 & 1 & 1 \\
1 & -1 & 1 & -1 \\
1 & 1 & -1 & -1 \\
1 & -1 & -1 & 1 \\
\end{bmatrix} \cdot \begin{bmatrix}
1 \\
1 \\
1 \\
0 \\
\end{bmatrix} = \begin{bmatrix}
3 \\
1 \\
1 \\
-1 \\
\end{bmatrix}
\]

Signs: ++, + -, - +, --

\[
\begin{bmatrix}
1 & 1 & 1 & 1 \\
1 & -1 & 1 & 1 \\
1 & 1 & 1 & -1 \\
1 & 1 & 1 & -1 \\
\end{bmatrix} \cdot \begin{bmatrix}
1 \\
1 \\
1 \\
1 \\
\end{bmatrix} = \begin{bmatrix}
3 \\
1 \\
1 \\
1 \\
\end{bmatrix}
\]

Signs: ++, + -, - +, ++
Reflections on Formalizing the Matrix Equation

- Inductive construction, inductive proof!
  - It took some work to identify the right inductive invariant
  - The reduction step poses the biggest challenge
- The reduction step requires extra proofs

Reflections on Formalizing the Matrix Equation

- Isabelle/HOL has well-developed libraries
  - The Sturm-Tarski theorem is already formalized* (the key computational tool for the matrix equation)
  - A number of linear algebra libraries are available

Extending the Matrix Libraries

- We build on a matrix library by Thiemann and Yamada*
- Our additions (~1800 LOC):
  - A computational notion of the Kronecker product
  - An algorithm to extract a basis from the rows of a matrix
    - Involved proving that row rank equals column rank

Code Export and Experiments
Experiments with SML code

- We export our formally verified algorithm to SML for experimentation
- Compare to:
  - A naive (unverified) version of Tarski’s algorithm
  - Li, Passmore, and Paulson*
Experiments with SML code

- We export our formally verified algorithm to SML for experimentation
- Compare to:
  - A naive (unverified) version of Tarski’s algorithm
  - Li, Passmore, and Paulson*
- Li et. al is faster:
  - CAD is generally faster than BKR
  - Their procedure is highly optimized
  - They use Mathematica as an untrusted oracle

## Experiments with SML code

Benchmarks from [18]

<table>
<thead>
<tr>
<th>Formula</th>
<th>#Poly</th>
<th>$#N(p, q)$ (Naive)</th>
<th>$#N(p, q)$ (BKR)</th>
<th>Time (Naive)</th>
<th>Time (BKR)</th>
<th>Time ([18])</th>
</tr>
</thead>
<tbody>
<tr>
<td>ex1</td>
<td>4 (12)</td>
<td>20</td>
<td>31</td>
<td>0.003</td>
<td>0.006</td>
<td>3.020</td>
</tr>
<tr>
<td>ex2</td>
<td>5 (6)</td>
<td>576</td>
<td>180</td>
<td>5.780</td>
<td>0.442</td>
<td>3.407</td>
</tr>
<tr>
<td>ex3</td>
<td>4 (22)</td>
<td>112</td>
<td>120</td>
<td>1794.843</td>
<td>1865.313</td>
<td>3.580</td>
</tr>
<tr>
<td>ex4</td>
<td>5 (3)</td>
<td>112</td>
<td>95</td>
<td>0.461</td>
<td>0.261</td>
<td>3.828</td>
</tr>
<tr>
<td>ex5</td>
<td>8 (3)</td>
<td>576</td>
<td>219</td>
<td>28.608</td>
<td>8.333</td>
<td>3.806</td>
</tr>
<tr>
<td>ex6</td>
<td>22 (9)</td>
<td>50331648</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>6.187</td>
</tr>
<tr>
<td>ex7</td>
<td>10 (12)</td>
<td>6144</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ex1 $\land$ 2</td>
<td>9 (12)</td>
<td>2816</td>
<td>298</td>
<td>317.432</td>
<td>3.027</td>
<td>3.033</td>
</tr>
<tr>
<td>ex1 $\land$ 2 $\land$ 4</td>
<td>13 (12)</td>
<td>28672</td>
<td>555</td>
<td>-</td>
<td>51.347</td>
<td>3.848</td>
</tr>
<tr>
<td>ex1 $\land$ 2 $\land$ 5</td>
<td>16 (12)</td>
<td>131072</td>
<td>826</td>
<td>-</td>
<td>436.575</td>
<td>3.711</td>
</tr>
</tbody>
</table>

## Experiments with SML code

*Compiled with mlton*
*Run on a laptop*
*Dashes indicate timeout*
*Times in seconds*

<table>
<thead>
<tr>
<th>Formula</th>
<th>#Poly</th>
<th>#N(p, q) (Naive)</th>
<th>#N(p, q) (BKR)</th>
<th>Time (Naive)</th>
<th>Time (BKR)</th>
<th>Time ([18])</th>
</tr>
</thead>
<tbody>
<tr>
<td>ex1</td>
<td>4 (12)</td>
<td>20</td>
<td>31</td>
<td>0.003</td>
<td>0.006</td>
<td>3.020</td>
</tr>
<tr>
<td>ex2</td>
<td>5 (6)</td>
<td>576</td>
<td>180</td>
<td>5.780</td>
<td>0.442</td>
<td>3.407</td>
</tr>
<tr>
<td>ex3</td>
<td>4 (22)</td>
<td>112</td>
<td>120</td>
<td>1794.843</td>
<td>1865.313</td>
<td>3.580</td>
</tr>
<tr>
<td>ex4</td>
<td>5 (3)</td>
<td>112</td>
<td>95</td>
<td>0.461</td>
<td>0.261</td>
<td>3.828</td>
</tr>
<tr>
<td>ex5</td>
<td>8 (3)</td>
<td>576</td>
<td>219</td>
<td>28.608</td>
<td>8.333</td>
<td>3.806</td>
</tr>
<tr>
<td>ex6</td>
<td>22 (9)</td>
<td>50331648</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>6.187</td>
</tr>
<tr>
<td>ex7</td>
<td>10 (12)</td>
<td>6144</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ex1 &amp; 2</td>
<td>9 (12)</td>
<td>2816</td>
<td>298</td>
<td>317.432</td>
<td>3.027</td>
<td>3.033</td>
</tr>
<tr>
<td>ex1 &amp; 2 &amp; 4</td>
<td>13 (12)</td>
<td>28672</td>
<td>555</td>
<td>-</td>
<td>51.347</td>
<td>3.848</td>
</tr>
<tr>
<td>ex1 &amp; 2 &amp; 5</td>
<td>16 (12)</td>
<td>131072</td>
<td>826</td>
<td>-</td>
<td>436.575</td>
<td>3.711</td>
</tr>
</tbody>
</table>

## Experiments with SML code

<table>
<thead>
<tr>
<th>Formula</th>
<th>#Poly</th>
<th>#\textit{N}(p, q) \textit{(Naive)}</th>
<th>#\textit{N}(p, q) \textit{(BKR)}</th>
<th>Time \textit{(Naive)}</th>
<th>Time \textit{(BKR)}</th>
<th>Time ([18])</th>
</tr>
</thead>
<tbody>
<tr>
<td>ex1</td>
<td>4 (12)</td>
<td>20</td>
<td>31</td>
<td>0.003</td>
<td>0.006</td>
<td>3.020</td>
</tr>
<tr>
<td>ex2</td>
<td>5 (6)</td>
<td>576</td>
<td>180</td>
<td>5.780</td>
<td>0.442</td>
<td>3.407</td>
</tr>
<tr>
<td>ex3</td>
<td>4 (22)</td>
<td>112</td>
<td>120</td>
<td>1794.843</td>
<td>1865.313</td>
<td>3.580</td>
</tr>
<tr>
<td>ex4</td>
<td>5 (3)</td>
<td>112</td>
<td>95</td>
<td>0.461</td>
<td>0.261</td>
<td>3.828</td>
</tr>
<tr>
<td>ex5</td>
<td>8 (3)</td>
<td>576</td>
<td>219</td>
<td>28.608</td>
<td>8.333</td>
<td>3.806</td>
</tr>
<tr>
<td>ex6</td>
<td>22 (9)</td>
<td>50331648</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>6.187</td>
</tr>
<tr>
<td>ex7</td>
<td>10 (12)</td>
<td>6144</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ex1 \land 2</td>
<td>9 (12)</td>
<td>2816</td>
<td>298</td>
<td>317.432</td>
<td>3.027</td>
<td>3.033</td>
</tr>
<tr>
<td>ex1 \land 2 \land 4</td>
<td>13 (12)</td>
<td>28672</td>
<td>555</td>
<td>-</td>
<td>51.347</td>
<td>3.848</td>
</tr>
<tr>
<td>ex1 \land 2 \land 5</td>
<td>16 (12)</td>
<td>131072</td>
<td>826</td>
<td>-</td>
<td>436.575</td>
<td>3.711</td>
</tr>
</tbody>
</table>
Future Work and Conclusion
Future Work

- Optimizing univariate BKR
  - Add parallelism
  - Optimize Tarski queries
- Formally verified complexity analysis (ambitious!)
- Formalizing multivariate BKR
Conclusion

- We have formally verified the univariate case of BKR’s QE algorithm
  - BKR hits a potential **sweet spot** in between practicality and ease of verification
  - Contributes to Isabelle/HOL’s matrix libraries
  - Export code to SML for faster runtime

- Multivariate BKR is ongoing work
Conclusion

- We have formally verified the univariate case of BKR’s QE algorithm
  - BKR hits a potential *sweet spot* in between practicality and ease of verification
  - Contributes to Isabelle/HOL’s matrix libraries
  - Export code to SML for faster runtime
- Multivariate BKR is ongoing work

∃ Questions?