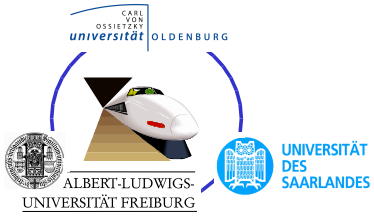
6-2009

Real World Verification

Andre Platzer
Carnegie Mellon University

Jan-David Quesel
University of Oldenburg

Philipp Rummer
University of Oxford



AVACS – Automatic Verification and Analysis of Complex Systems

REPORTS

of SFB/TR 14 AVACS

Editors: Board of SFB/TR 14 AVACS

Real World Verification

by

André Platzer Jan-David Quesel Philipp Rümmer

AVACS Technical Report No. 52
June 2009
ISSN: 1860-9821

Publisher: Sonderforschungsbereich/Transregio 14 AVACS
(Automatic Verification and Analysis of Complex Systems)
Editors: Bernd Becker, Werner Damm, Martin Fränzle, Ernst-Rüdiger Olderog,
Andreas Podelski, Reinhard Wilhelm
ATRs (AVACS Technical Reports) are freely downloadable from `www.avacs.org`

Author(s) contact: Jan-David Quesel (`quesel@informatik.uni-oldenburg.de`).

Real World Verification

André Platzer[1], Jan-David Quesel[2], and Philipp Rümmer[3]

[1] Carnegie Mellon University, Pittsburgh, PA
aplatzer@cs.cmu.edu
[2] University of Oldenburg, Germany
quesel@informatik.uni-oldenburg.de
[3] Oxford University, Computing Laboratory, UK
Philipp.Ruemmer@comlab.ox.ac.uk

Abstract. Scalable handling of real arithmetic is a crucial part of the verification of hybrid systems, mathematical algorithms, and mixed analog/digital circuits. Despite substantial advances in verification technology, complexity issues with classical decision procedures are still a major obstacle for formal verification of real-world applications, e.g., in automotive and avionic industries. To identify strengths and weaknesses, we examine state of the art symbolic techniques and implementations for the universal fragment of real-closed fields: approaches based on quantifier elimination, Gröbner Bases, and semidefinite programming for the Positivstellensatz. Within a uniform context of the verification tool KeYmaera, we compare these approaches qualitatively and quantitatively on verification benchmarks from hybrid systems, textbook algorithms, and on geometric problems. Finally, we introduce a new decision procedure combining Gröbner Bases and semidefinite programming for the real Nullstellensatz that outperforms the individual approaches on an interesting set of problems.

Keywords: real-closed fields, decision procedures, hybrid systems, software verification

1 Introduction

The field of formal verification has the important ambition to check the behavior of systems by either proving the correct functioning of the system or finding bugs in its design. For several classes of systems that come from real-world domains, reasoning about real quantities is an inherent aspect of the problem. This includes (*i*) embedded systems or complex physical systems, (*ii*) formal analysis of mixed discrete/analog effects in chip design, or (*iii*) mathematical textbook algorithms, numerical algorithms or floating point arithmetic in standard programs. For domains (*i*)–(*ii*), *hybrid systems* are a common model, i.e., systems governed by interacting discrete and continuous transitions in the state space. In these domains, the need for real arithmetic reasoning comes from the temporal evolution of the continuous part of the state space, e.g., positions, velocities, analog signals. For case (*iii*), real arithmetic occurs in the computations on program data or are used as a first approximation for floating-point arithmetic.

By a famous result due to Tarski [1], real arithmetic is decidable in the sense that (the first-order theory of) real arithmetic is equivalent to the first-order theory of real-closed fields, which is decidable by quantifier elimination (i.e., the process of replacing quantified formulas equivalently by quantifier-free formulas). Numerous algorithmic improvements have been made both for the handling of basic real arithmetic and for specific verification procedures for the problem domains (*i*)–(*iii*). However, for a large number of real-world systems, the underlying problems in real arithmetic still have a prohibitive complexity for quantifier elimination. Even numerical procedures for real arithmetic [2] suffer from the curse of dimensionality limiting their scalability.

In this paper we compare three state of the art approaches to reasoning about real-arithmetic in real-closed fields based on: quantifier elimination [3, 4] Gröbner Bases [5], and semidefinite programming [6] for the Positivstellensatz [7]. Quantifier elimination is defined for full quantified (polynomial) nonlinear real arithmetic. The other approaches are for the universal fragment, i.e., formulas with a universal quantifier prefix. We discuss strengths and weaknesses of these approaches for formal verification and compare multiple algorithms and implementations on a set of benchmarks originating from real verification problems or interesting instances of real arithmetic. To obtain representative experimental results, we integrate all these approaches within a single uniform framework of the automated theorem prover KeYmaera for hybrid systems [8].

Finally, we introduce a new decision procedure for the universal fragment of real-closed fields that combines Gröbner Basis computations with semidefinite programming for the real Nullstellensatz [7] to avoid the scalability issues with semidefinite programming for the Positivstellensatz. Our algorithm outperforms the other algorithms on an interesting set of benchmarks.

With the goal of finding out which approaches are most suitable for real world verification problems, we provide an experimental evaluation for a wide range of techniques for real arithmetic. We contrast multiple state of the art approaches and different implementations:

1. Quantifier elimination for real-closed fields in Mathematica, QEPCAD B [9], Redlog [10], and HOL Light [11];
2. Real arithmetic handling with Gröbner Bases using external procedures in Mathematica, the Orbital library, and internally with KeYmaera proof rules;
3. Semidefinite programming relaxations [6] for the Positivstellensatz [7] using the CSDP solver [12] in our own implementation and in HOL Light [13];
4. Our new algorithm combining Gröbner Bases and semidefinite programming for the real Nulstellensatz [7] using CSDP [12] and the Orbital library.

In this paper, we consider problems in the continuous world of reals that arise in real world verification problems, including hybrid systems analysis and program verification. Our contributions are a systematic quantitative and qualitative comparison of multiple techniques for handling real arithmetic within a uniform verification framework and the introduction of a novel decision procedure for universal real arithmetic that combines Gröbner Bases with semidefinite programming for the real Nullstellensatz. We further address the question how

expensive various levels of confidence in real verification are in real examples: external (unverified) blackboxes, external blackboxes producing formally checkable certificates, and internal formal reasoning within a proof system.

2 Overall Verification Approach

We briefly discuss our formal verification approach for hybrid systems and mathematical algorithms within the automated theorem prover KeYmaera [8]. It is an implementation of a Gentzen-style sequent proof calculus for hybrid systems [14] that uses deduction modulo decision procedures for handling real arithmetic. The calculus works on sequents of the form $\phi_1, \ldots, \phi_n \vdash \psi_1, \ldots, \psi_m$ with the semantics of the formula $\bigwedge_{i=1}^{n} \phi_i \rightarrow \bigvee_{i=1}^{m} \psi_i$. Among several other rules, the calculus transforms the propositional structure into a sequent representation.

The deduction modulo calculus of KeYmaera gives us a uniform context for comparing the performance of multiple approaches and implementations for real arithmetic. The input for KeYmaera is a formula given in differential dynamic logic [14]. This logic extends first-order logic over real arithmetic by constructs for reasoning about hybrid systems as well as real-valued mathematical algorithms. For the verification task, the proof calculus transforms the input formulas into first-order formulas over real-arithmetic. For details about the proof rules of this transformation we refer to [14].

In this paper we address the question of handling the resulting real arithmetic formulas. Although first-order logic over real arithmetic is decidable by quantifier elimination [1] its complexity is doubly exponential in theory and can be high in practice. The central point of this work is to examine the question which approach to handling real arithmetic is best for which class of real world examples. We further want to determine the computational cost for techniques that provide formal proof certificates.

3 Methods for Handling Real Arithmetic

We survey different approaches to handling real arithmetic in background provers for verification. We phrase these approaches in terms of reals for simplicity. Yet, all subsequent theory in Sections 3–4 generalizes from $\mathbb{R}$ to real-closed fields.

In the sequel we assume the presence of standard rules for propositional connectives. Such rules are not presented here, as propositional reasoning is orthogonal to the handling of arithmetic. The KeYmaera system uses classical propositional sequent calculus rules; see [14, 15] for details. To simplify the presentation, we further assume simple rules to normalise sequents that translate, e.g., $g \leq f$ to $f \geq g$, $f \neq g$ to $\neg(f = g)$ and $\vdash f > g$ to $f \leq g \vdash$ respectively. We assume all inequalities to be moved to the antecedent in this way.

3.1 Gröbner Bases for Real Arithmetic

Gröbner bases [5] provide a sound but incomplete procedure for proving validity of formulas in the universal fragment of equational first-order real arithmetic.

Preliminaries. Let $\mathbb{Q}[X_1, \ldots, X_n]$ be the set of *multivariate polynomials* over the indeterminates $X_1, \ldots, X_n$ with coefficients in $\mathbb{Q}$. A subset $I \subseteq \mathbb{Q}[X_1, \ldots, X_n]$ is an *ideal*, iff I is a subgroup with respect to addition and

$$rx \in I, \text{ for all } x \in I, r \in \mathbb{Q}[X_1, \ldots, X_n] \ .$$

The ideal *generated* by a set $G \subseteq \mathbb{Q}[X_1, \ldots, X_n]$ is the smallest ideal I containing G, and is denoted by (G).

The notions of Gröbner bases and polynomial reductions are relative to an admissible monomial order $\prec$, which is a strict well-order on monomials such that $uw \prec vw$ whenever $u \prec v$ for arbitrary monomials u, v, w. Admissible orders extend canonically to $\mathbb{Q}[X_1, \ldots, X_n]$ as a multiset order; see [5] for details. The monomial order determines the *leading term* in multivariate polynomials, i.e., the maximal monomial with respect to $\prec$.

Definition 1 (Reduction). *Let $f, g \in \mathbb{Q}[X_1, \ldots, X_n]$. We say that f* reduces to *g with respect to $G \subset \mathbb{Q}[X_1, \ldots, X_n]$ iff for some $m \in \mathbb{N}$ there are $f_0, f_1, \ldots, f_m$ in $\mathbb{Q}[X_1, \ldots, X_n]$ with $f_0 = f, f_m = g$ such that, for all i, $f_{i+1} = f_i - h_i g_i$ for some $h_i \in \mathbb{Q}[X_1, \ldots, X_n]$, $g_i \in G$, and $f_{i+1} \prec f_i$. We write $g = \mathrm{red}_G f$ if, in addition, g cannot be reduced further, i.e., there is no $h_{m+1} \in \mathbb{Q}[X_1, \ldots, X_n]$ and $g_{m+1} \in G$ with $g - h_{m+1}\, g_{m+1} \prec q$.*

Definition 2 (Gröbner basis). *A finite subset G of an ideal I of $\mathbb{Q}[X_1, .., X_n]$, is called* Gröbner basis *iff $I = (G)$ and $\mathrm{red}_G f$ is unique for all polynomials f. Further G is* reduced *if no $g \in G$ can be reduced further with respect to $G \setminus \{g\}$.*

There are several equivalent alternative formulations of this definition, for instance that $\mathrm{red}_G f = 0$ iff $f \in I$. This means that Gröbner bases solve the *ideal membership problem* and can, thus, directly be used as an (incomplete) proof rule for equational arithmetic.

Gröbner Basis Eliminations. The most naive use of Gröbner bases for real arithmetic is described by the rules A1, A2 in Fig. 1. The rule A1 closes a goal if the ideal G generated by equations in the antecedent contains 1, which (by Hilbert's Nullstellensatz) implies that the equations do not have common solutions (i.e., are contradictory). Similarly, A2 can be applied if the sides f, g of an equation in the succedent have the same remainder modulo G, which means $f - g \in (G)$.

The scope of the rules can be extended by testing for *radical membership* instead of ideal membership, which can prove problems like $x^2 = 0 \vdash x = 0$ that A2 cannot prove. The *radical* of an ideal I is the set

$$\sqrt{I} = \bigcup_{i=1}^{\infty} \{g \in \mathbb{Q}[X_1, \ldots, X_n] \ : \ g^i \in I\} \supseteq I$$

Because the inclusion $I \subseteq \sqrt{I}$ can be strict (e.g., $\sqrt{(x^2)} = (x)$), testing for radical membership is more liberal than ideal membership, while still being sound.

$$\text{(A1)}\ \frac{*}{\Gamma, g_1 = \tilde{g}_1, \ldots, g_n = \tilde{g}_n \vdash \Delta} \qquad \text{(A4)}\ \frac{\Gamma, f - g = z^2 \vdash \Delta}{\Gamma, f \geq g \vdash \Delta}$$

$$\text{(A2)}\ \frac{*}{\Gamma, g_1 = \tilde{g}_1, \ldots, g_n = \tilde{g}_n \vdash f = h, \Delta} \qquad \text{(A5)}\ \frac{\Gamma, (f - g)z^2 = 1 \vdash \Delta}{\Gamma, f > g \vdash \Delta}$$

$$\text{(A3)}\ \frac{\Gamma, (f - g)z = 1 \vdash \Delta}{\Gamma \vdash f = g, \Delta} \qquad \text{(A6)}\ \frac{\Gamma \vdash 1 + s_1^2 + \cdots + s_n^2 = 0, \Delta}{\Gamma \vdash \Delta}$$

In all rules, z is a fresh variable. With the Gröbner basis G of the ideal $(g_1 - \tilde{g}_1, \ldots, g_n - \tilde{g}_n)$, rule A1 is applicable if $\mathrm{red}_G\, 1 = 0$, and A2 if $\mathrm{red}_G\, f = \mathrm{red}_G\, h$. Rules similar to A2, A4 and A5 can be defined for inequalities in the succedent. In A6, the polynomials $s_1, \ldots, s_n$ can be chosen arbitrarily.

Fig. 1. Rule schemata of Gröbner calculus rules

In practice, the rule A3, which is known as *Rabinowitch's trick*, represents a simple way of testing for radical membership. It is based on the observation that $g \in \sqrt{I}$ if and only if $1 \in (I \cup \{gz - 1\})$ (where z is a fresh indeterminate). The latter property can be tested by first applying A3 and then A1.

Finally, inequalities can be translated to equations using A4, A5, which exploit the fact that a real number is positive iff it is a square (A5 is an optimized version including Rabinowitch's trick). Combined with the rules A1, A2, this encoding of inequalities is rather weak, and not able to derive simple facts like $a \leq b \wedge b \leq c \rightarrow a \leq c$. It is, however, an important preprocessing step for the complete procedure described in the next section (where we explain rule A6).

Proposition 1 (Soundness). *The Gröbner basis rules in Fig. 1 are sound. Rules A3, A4, A5 are even satisfiability-equivalent transformations, i.e., their respective premisses and conclusions are satisfiability-equivalent. (See Appendix A).*

The Gröbner basis approach gives a sound but incomplete overapproximation. To see why Gröbner bases are incomplete for real arithmetic, consider the following. Gröbner bases are a general approach and do not take into account the special properties of the reals. For instance, the sequent $x^2 = -1 \vdash$ is valid, i.e., the formula $x^2 = -1$ is unsatisfiable over $\mathbb{R}$, but the Gröbner basis of $x^2 + 1$ is $\{x^2 + 1\}$ and, in fact, $x^2 = -1$ is satisfiable over $\mathbb{C}$ but not over $\mathbb{R}$.

Implementations. We compare three implementations of the Gröbner basis rules:

GM The implementation provided by the Mathematica 7.0 computer algebra system, which can be used as a reasoning back-end by KeYmaera.

GO The implementation of Buchberger's algorithm [5] in the open-source Java-library Orbital (written by the first author of this paper).

GK An implementation of Buchberger's algorithm with (verified) proof rules that are directly defined within KeYmaera. This procedure generalizes a calculus for integer arithmetic [16] to the reals, and differs from GM and GO

in that it does not use the rules A3, A4, A5, but instead integrates the Fourier-Motzkin variable elimination rule [17] to handle inequalities (which is complete for linear arithmetic). This tight integration of the two procedures can simplify terms in inequalities using Gröbner bases, and can feed equations derived by the Fourier-Motzkin procedure back to Buchberger's algorithm. We evaluate the benefits of this cooperation in Sect. 5. Since our domain are the reals, we do not use the heuristic approach tailored to nonlinear integer inequalities from [16].

3.2 A Complete Rule using the Real Nullstellensatz

While the rules A1, A2, A3, A4, A5 only form an incomplete calculus for problems in real arithmetic, the situation is different over the complex numbers: Hilbert's Nullstellensatz tells that A1, A3 together yield a decision procedure for universal equational problems in $\mathbb{C}$. A corresponding result for real-closed fields is Stengle's real Nullstellensatz [7]; also see [13]:

Theorem 1 (Nullstellensatz [7] for real-closed fields). *Let R be a real-closed field (e.g., $R = \mathbb{R}$) and G be a finite subset of $R[X_1, \ldots, X_n]$. Then the set $\{x \in R^n \ : \ g(x) = 0$ for all $g \in G\}$ is empty if and only if there are polynomials $s_1, \ldots, s_m \in R[X_1, \ldots, X_n]$ such that $1 + s_1^2 + \cdots + s_m^2 \in (G)$. If, moreover, $G \subseteq \mathrm{Q}[X_1, \ldots, X_n]$, then also the polynomials $s_1, \ldots, s_m$ can be chosen among the elements of $\mathrm{Q}[X_1, \ldots, X_n]$.*

This theorem leads to an extremely simple, yet complete, proof method for the universal fragment of real arithmetic: in addition to the rules that we have already discussed, we add rule A6 in Fig. 1 for injecting the equation $1 + s_1^2 + \cdots + s_m^2 = 0$ into a proof goal. Any valid proof goal can then be closed in the following way: (*i*) inequalities and equations in the succedent are turned into equations in the antecedent with the help of A3, A4, A5, (*ii*) the *witness* $1 + s_1^2 + \cdots + s_m^2$ due to the real Nullstellensatz is generated using A6, and (*iii*) the goal is closed by the Gröbner Basis computations with A2.

Corollary 1 (Completeness). *Along with propositional rules, the rules in Fig. 1 are complete for the universal fragment of real arithmetic.*

Proof. Completeness follows from Theorem 1 using the satisfiability-equivalence properties for the transformation by A3, A4, A5 according to Proposition 1. □

The main difficulty with this calculus is obvious: it does not provide any guidance for choosing the witness $1 + s_1^2 + \cdots + s_m^2 = 0$. One technique to tackle the required search is semidefinite programming, following the work based on Stengle's Positivstellensatz (Sect. 3.4) in [6, 13]. We describe a new approach that combines semidefinite programming with Gröbner bases in Sect. 4.

Example 1. In Fig. 2, we show a proof for the following implication (leaving out propositional reasoning):

$$x \geq y \wedge z \geq 0 \rightarrow xz \geq yz. \tag{1}$$

$$\dfrac{\dfrac{\dfrac{}{x-y=a^2, z=b^2, (yz-xz)c^2=1 \vdash 1+(abc)^2=0}\,{}^{*}}{x-y=a^2, z=b^2, (yz-xz)c^2=1 \vdash}\,\text{A2}}{x \geq y, z \geq 0, yz > xz \vdash}\,\text{A6}\quad\text{A4,A5}$$

Fig. 2. Example proof using the real Nullstellensatz

The inequalities $x \geq y$ and $z \geq 0$ are turned into equations using A4. Proving by contradiction (or using propositional rules), the conclusion $xz \geq yz$ is considered as an assumption $yz > xz$ and subsequently eliminated with the help of A5. Once this is done, we rely on an oracle to tell us the witness $1 + (abc)^2$, which is introduced using A6. Finally, the proof can be closed by A2: the set $\{a^2 - x + y,\, b^2 - z,\, xzc^2 - yzc^2 + 1\}$ is a Gröbner basis representing the equations in the antecedent. The basis reduces the term $1 + (abc)^2$ to 0 as follows:

$$1 + a^2b^2c^2 \overset{b^2-z}{\rightsquigarrow} 1 + a^2zc^2 \overset{a^2-x+y}{\rightsquigarrow} 1 + xzc^2 - yzc^2 \rightsquigarrow 0$$

3.3 Quantifier Elimination in Real-Closed Fields

A general method for handling quantified real arithmetic is based on the seminal work by Tarski [1]. He showed that there is an algorithm computing a quantifier-free formula that is equivalent to a given formula in (first-order) real arithmetic.

Theorem 2 (Quantifier elimination [1]). *The first-order theory of reals (or of real-closed fields) admits* quantifier elimination*, i.e., to each first-order formula ϕ, a quantifier-free formula* $\mathrm{QE}(\phi)$ *can be associated effectively that is equivalent and has no additional free variables. Thus* QE *yields a decision procedure for closed formulas when evaluating the remaining quantifier-free formulas.*

Unlike the other approaches outlined in this paper, QE directly applies to full nonlinear (polynomial) real arithmetic and not just to the universal fragment. QE is also independent of propositional rules, except that computational efficiency considerations advise to combine both [18].

Example 2. For instance, QE yields the following equivalence:

$$\exists x\,(ax^2 + bx + c = 0) \;\equiv\; a \neq 0 \wedge b^2 - 4ac \geq 0 \vee a = 0 \wedge (b = 0 \rightarrow c = 0)$$

Tarski's approach has been extended to practical algorithms [3, 4], which are quite sophisticated. Unfortunately, the complexity of QE is doubly exponential in the number of quantifier alternations [19].

Implementations. We compare six implementations of QE in experiments:

QQ Partial cylindrical algebraic decomposition (PCAD) [3] in QEPCAD B [9];
QM QE based on partial CAD [3] and validated numerics [20] in Mathematica;
QR$_c$ Partial CAD [3] in Redlog [10];
QR$_s$ Virtual substitution [4] in Redlog [10], falling back to **QR**$_c$;
QC Harrison's implementation of Cohen-Hörmander quantifier elimination;
QH Proof-producing quantifier elimination [11] in HOL Light.

$$\text{(A7)}\ \frac{*}{f_1 \geq \tilde{f}_1, \ldots, f_m \geq \tilde{f}_m, g_1 = \tilde{g}_1, \ldots, g_n = \tilde{g}_n \vdash h_1 = \tilde{h}_1, \ldots, h_l = \tilde{h}_l}$$

A7 is applicable iff $s + g + m^2 = 0$ for some $s \in \mathrm{con}(\{f_1 - \tilde{f}_1, \ldots, f_m - \tilde{f}_m\})$, some $g \in (g_1 - \tilde{g}_1, \ldots, g_n - \tilde{g}_n)$, and some $m \in \mathrm{mon}(\{h_1 - \tilde{h}_1, \ldots, h_l - \tilde{h}_l\})$.

Fig. 3. Rule schemata of Positivstellensatz calculus rules

3.4 Semidefinite Programming for the Positivstellensatz

The Positivstellensatz for real-closed fields [7] is a generalisation of the real Nullstellensatz. It gives rise to a sound and complete proof method for the universal fragment of first-order real arithmetic that does not require the reductions A3, A4, A5. The Positivstellensatz has recently been exploited in combination with relaxations from semidefinite programming [6, 13].

The *multiplicative monoid* $\mathrm{mon}(H)$ generated by $H \subseteq R[X_1, \ldots, X_n]$ is the set of finite products of elements of H (including the empty product 1). The *cone* $\mathrm{con}(F)$ generated by a set $F \subseteq R[X_1, \ldots, X_n]$ is the smallest set containing F and squares s^2 of arbitrary polynomials $s \in R[X_1, \ldots, X_n]$ that is closed under addition and multiplication. For more computational representations of cones and ideals, we refer to [6, 21].

Theorem 3 (Positivstellensatz [7] for real-closed fields). *Let R be a real-closed field (e.g., $R = \mathbb{R}$) and F, G, H finite subsets of $R[X_1, \ldots, X_n]$. Then*

$$\{x \in R^n \ :\ f(x) \geq 0 \text{ for all } f \in F,\ g(x) = 0 \text{ f.a. } g \in G,\ h(x) \neq 0 \text{ f.a. } h \in H\}$$

is empty iff

$$\text{there are } s \in \mathrm{con}(F), g \in (G), m \in \mathrm{mon}(H) \text{ such that } s + g + m^2 = 0\ .$$

If, moreover, $F, G, H \subseteq \mathrm{Q}[X_1, \ldots, X_n]$, then also the polynomials s, g, m can be chosen among the elements of $\mathrm{Q}[X_1, \ldots, X_n]$.

The polynomials s, g, m are polynomial infeasibility witnesses. For bounded degree, witnesses s, g, m can be searched for using numerical semidefinite programming [6] by parameterising the resulting polynomials. As (theoretical) degree bounds exist for the certificate polynomials s, g, m, the Positivstellensatz yields a decision procedure. These bounds are, however, at least triply exponential [6]. Thus, the approach advocated by Parrilo [6] is to increase the bound successively and solve the existence of bounded degree witnesses due to the Positivstellensatz by semidefinite programming [22].

As a simple corollary to Theorem 3 we have that A7 is a sound proof rule.

Corollary 2 (Soundness). *The rule in Fig. 3 is sound.*

In contrast to the rules in Fig. 1 the only additional transformation necessary for rule A7 is a reduction from $>$ to $\geq$ via $f > g \leftrightarrow f \geq g \wedge f \neq g$. All other transformations follow from the propositional sequent calculus rules and the rewriting

rules described in the beginning of Sect. 3. Therefore, this approach does not introduce new variables, as it does not need the rules A3 – A5. Alternatively, A5 can be used in place of the $f > g$ axiomatisation as we show in the sequel.

Example 3. A proof for the implication (1) that uses the Positivstellensatz is in Fig. 4. In contrast to the proof in Fig. 2, it is now unnecessary to eliminate the inequalities $x \geq y$ and $z \geq 0$, while the rule A5 has to be used for $xz \geq yz$ (corresponding to $yz > xz$ in the antecedent). A witness for the problem is:

$$\underbrace{c^2 \cdot (x-y) \cdot z}_{s} + \underbrace{(yz-xz)c^2 - 1}_{g} + \underbrace{1}_{m^2} \;=\; 0$$

The terms $x - y$ and z in s stem from the inequalities in the sequent, while the term g is derived from the equation.

Implementations. We compare two implementations using the semidefinite programming optimization tool CSDP [12] to find witnesses for the Positivstellensatz:

PH John Harrison's implementation [13] in HOL Light.

PK Our implementation within KeYmaera directly follows the approach presented by Parrilo [6] and Harrison [13]. We follow Parrilo's enumeration of polynomials without further optimization.

4 Gröbner Bases for the Real Nullstellensatz (GRN)

We describe a new approach to turn the complete calculus based on the real Nullstellensatz (NSS, Theorem 1) into an effective proof procedure. While our method is strongly inspired by, and in parts based on, semidefinite programming for the Positivstellensatz (PSS, Theorem 3) [6, 13], there are two main motivations to deviate from this approach: (*i*) the application of the PSS requires reasoning about ideal membership (the set (G) in Theorem 3) and, thus, to solve systems of polynomial equations. This is an incentive to integrate Gröbner bases as a computational, efficient, and well-studied method to this end; (*ii*) the PSS requires constructing three witnesses s, g, m simultaneously, which makes it intricate to balance degree bounds and the number of parameters to be determined by semidefinite programming. Using a combination of Gröbner basis computations and the single witnesses of the real NSS, we avoid these issues.

In order to prove by NSS that a set G of polynomials does not have common zeroes, we need to find polynomials $s_1, \ldots, s_m$ such that $1 + s_1^2 + \cdots + s_m^2 \in (G)$.

$$\dfrac{\dfrac{}{x \geq y, z \geq 0, (yz-xz)c^2 = 1 \vdash}\,{}^{*}\,{}^{\text{A7}}}{x \geq y, z \geq 0, yz > xz \vdash}\,{}^{\text{A5}}$$

Fig. 4. Example proof using the Positivstellensatz

We reduce this problem to a search for positive semidefinite matrices with the help of the following lemma. A matrix $X \in \mathrm{R}^{k\times k}$ is called *positive semidefinite* (PSD) if it is symmetric, and if $x^t X x \geq 0$ for each vector $x \in \mathrm{R}^k$. There is a simple correspondence between PSD matrices and sums of squares:

Lemma 1. *Suppose $p \in \mathrm{Q}[X_1, \ldots, X_n]^k$ is a vector of rational polynomials. The following identities hold (see the proof in Appendix A):*

$$\begin{aligned} & \left\{ \sum_{i=1}^{l} (c_i p)^2 \ : \ l \in \mathrm{N}, c_i \in \mathrm{Q}^k \right\} \\ = & \left\{ \sum_{i=1}^{l} \alpha_i (c_i p)^2 \ : \ l \in \mathrm{N}, \alpha_i \in \mathrm{Q}, \alpha_i \geq 0, c_i \in \mathrm{Q}^k \right\} \\ = & \left\{ p^t X p \ : \ X \in \mathrm{Q}^{k\times k} \textit{ positive semidefinite} \right\} \end{aligned}$$

By combining Lemma 1 with the NSS, we see that a set G of polynomials does not have any common zeroes if and only if there is a vector p of polynomials and a PSD matrix $X \in \mathrm{R}^{k\times k}$ such that $1 + p^t X p \in (G)$. As the vector space of polynomials is generated by monomials, it is sufficient to consider vectors p of monomials.

Semidefinite programming [22] provides a simple method to determine such matrices X. A *semidefinite program* (SDP) is an optimisation problem in terms of traces (tr) of matrices:

maximise	$\mathrm{tr}(CX)$
subject to	$\mathrm{tr}(A_i X) = b_i \quad (\text{for } i \in \{1, \ldots, n\})$,
where	X positive semidefinite

where $A_i, C \in \mathrm{R}^{k\times k}$ are symmetric matrices and $b_i \in \mathrm{R}$. Such optimisation problems can be solved efficiently using numerical convex optimization [22].

The key insight underlying our method is the following: by computing a Gröbner basis B for the ideal (G), the NSS condition $1 + p^t X p \in (G)$ can be encoded as the linear side constraints $\mathrm{tr}(A_i X) = b_i$ $(i \in \{1, \ldots, n\})$ of a semidefinite program searching for X. To see this, note that both the expression $1 + p^t X p$ and the reduction $\mathrm{red}_B(1 + p^t X p)$ are linear in X. Because Gröbner bases determine unique remainders, we therefore have $1 + p^t X p \in (G)$ if and only if $\mathrm{red}_B(1 + p^t X p) = 0$. This equation is a linear constraint on X suitable for SDP.

To capture this observation formally, let Q be a symmetric $k \times k$ matrix of parameters:

$$Q \ = \ \begin{pmatrix} q_{1,1} & q_{1,2} & \cdots & q_{1,k} \\ q_{1,2} & q_{2,2} & \cdots & q_{2,k} \\ \cdots & \cdots & \cdots & \cdots \\ q_{1,k} & q_{2,k} & \cdots & q_{k,k} \end{pmatrix}$$

The polynomial $1 + p^t Q p$ is linear in Q and can be represented in the form $1 + p^t Q p = q^t C m$, where $q = (q_{1,1}, q_{1,2}, \ldots, q_{k,k})^t$ is the vector of all the Q-parameters, $m = (m_1, \ldots, m_s)^t$ is a vector of monomials over $X_1, \ldots, X_n$ (containing, at least, 1 and all products $p_i p_j$ of components of p), and $C \in \mathrm{Q}^{k^2 \times s}$

is a matrix. By computing the remainder $q^tDm = \mathrm{red}_B(q^tCm)$ of this term for a Gröbner basis B over $\mathbb{Q}[X_1, \ldots, X_n]$, we can construct the required side constraints:

Lemma 2. *Suppose that the components of m are pairwise distinct, and that q^tCm and q^tDm are two polynomials over $\mathbb{Q}[q_{1,1}, q_{1,2}, \ldots, q_{k,k}][X_1, \ldots, X_n]$ defined by the matrices $C, D \in \mathbb{Q}^{k^2 \times s}$, such that $q^tDm = \mathrm{red}_B(q^tCm)$. Then the following equation holds (see Appendix B for a proof):*

$$\{x \in \mathbb{R}^k : red_B(x^tCm) = 0\} \;=\; \{x \in \mathbb{R}^k : x^tD = 0\} \tag{2}$$

Example 4. We return to the implication (1) proven in Fig. 2 by showing that the polynomials $B = \{a^2 - x + y,\ b^2 - z,\ xzc^2 - yzc^2 + 1\}$ have no common zeroes. The witness $1 + (abc)^2$ used in the proof of Fig. 2 can be constructed systematically for a suitable set of basis monomials, say, $p = (1, a^2, abc)^t$. We need to find a PSD matrix $X \in \mathbb{Q}^{3\times 3}$ such that $1 + p^tXp \in (B)$. To do so, we compute the reduction $\mathrm{red}_B(1 + p^tQp)$ for a symbolic 3×3 parameter matrix Q:

$$\begin{aligned}
&\mathrm{red}_B(1 + p^tQp)\\
&= \mathrm{red}_B(1 + q_{1,1}1^2 + 2q_{1,2}a^2 + 2q_{1,3}abc + 2q_{2,3}a^3bc + q_{3,3}a^2b^2c^2)\\
&= 1 + q_{1,1} - q_{3,3} + 2q_{1,2}x - 2q_{1,2}y + 2q_{1,3}abc + 2q_{2,3}abcx - 2q_{2,3}abcy
\end{aligned}$$

By comparing coefficients, the constraints on Q for this polynomial to be 0 are:

$$\begin{array}{rrr}
1 + q_{1,1} - q_{3,3} = 0 & -2q_{1,2} = 0 & 2q_{2,3} = 0\\
2q_{1,2} = 0 & 2q_{1,3} = 0 & -2q_{2,3} = 0
\end{array}$$

A positive semidefinite solution of the constraints is $q_{3,3} = 1$ and $q_{i,j} = 0$ for all $(i, j) \neq (3, 3)$, which means $1 + p^tQp = 1 + (abc)^2$.

Theorem 4 (Completeness). *By enumerating all monomials for p successively, Gröbner bases for the real Nullstellensatz give a complete method for universal real arithmetic: If the original formula is valid, then, when p contains all monomials of a sufficiently large degree, the corresponding semidefinite programs will have a solution (the witness).*

Proof. The proof is a combination of Lemma 2 with Corollary 1.

4.1 Discussion and Practical Considerations

Semidefinite programming turns the search for witnesses $1 + s_1^2 + \cdots + s_m^2$ into a (simpler) search for suitable basis monomials p. As the number of basis monomials that need to be considered is finite (due to degree bounds on witnesses [6]), this yields a theoretical decision procedure. Practically, we enumerate all monomials with ascending degree. There might be more sophisticated methods, however: the number of monomials that witnesses are actually built of is usually small, and it might be possible to locate likely candidates by analyzing the

Gröbner basis B. In our experience, the number of basis monomials that are considered before a solution is found (and thus the difficulty of a problem) depends on (*i*) the number of variables in the polynomial ring, and (*ii*) the degree of the leading monomials in the Gröbner basis.

Another issue is that implementations for semidefinite programming (like the CSDP solver [12] used by us) are numerical and produce answers in floating point arithmetic. To recover precise solutions in $\mathbb{Q}$ from such answers, we use a similar approach as in [13]: We approximate floating point numbers to a certain precision by rationals (with the help of Stern-Brocot trees [23]), and check resulting solution candidate for semidefiniteness. We increase the precision successively as long as the solution candidate remains indefinite.

Optimizations. We found it essential to use preprocessing steps to reduce the number of variables in a problem, such that the number of potential basis monomials becomes tractable. Some heuristics are:

- If the Gröbner basis B contains a polynomial $x + t$ such that x does not occur in t, then x and the polynomial can be eliminated by simple rewriting.
- If B contains polynomials $xy - 1$ and $x^n + t$ such that x^n does not divide t, then x and the polynomial $xy - 1$ can be eliminated by multiplying each polynomial in B (except $xy - 1$) with a power of y and reducing w.r.t. $xy - 1$.
- Polynomials $\alpha_1 m_1^2 + \cdots + \alpha_n m_n^2 \in B$ such that $\alpha_i > 0$ for $i \in \{1, \ldots, n\}$ can be replaced by the monomials $m_1, \ldots, m_n$.
- If B contains a polynomial $\alpha_0 x^2 - \alpha_1 m_1^2 - \cdots - \alpha_n m_n^2$ such that $\alpha_i > 0$ for $i \in \{0, \ldots, n\}$ where x only occurs with even degree in B, then x can be eliminated by rewriting and the polynomial can be removed.

The last two cases are surprisingly common, due to the encoding of inequalities by quadratic terms performed by A4 and A5.

5 Experimental Results

We have integrated the techniques presented in Sect. 3–4 into KeYmaera. With the various methods for real arithmetic integrated into a common framework and real arithmetic examples from different domains, we have a solid base for our experiments. The benchmarks[4] are a collection of challenging arithmetic problems from the hybrid system world [24], the verification of invariant properties for mathematical algorithms [25, 26] and algebraic geometry [27], as well as a smaller number of synthetic problems. For the examples with mixed quantifiers, our setting applies QM to the existential quantifiers such that we can still gain insight into the scalability of the approaches that are restricted to the universal fragment on these examples. We run our experiments on a dual Intel Xeon E5430 (quad core with 2.66 GHz) and 32 gigabytes RAM.

The experimental results (see Appendix C) summarized in Fig. 5 show that, for our particular mix of examples, quantifier elimination procedures are still

[4] Available along with KeYmaera from http://symbolaris.com/info/KeYmaera.html

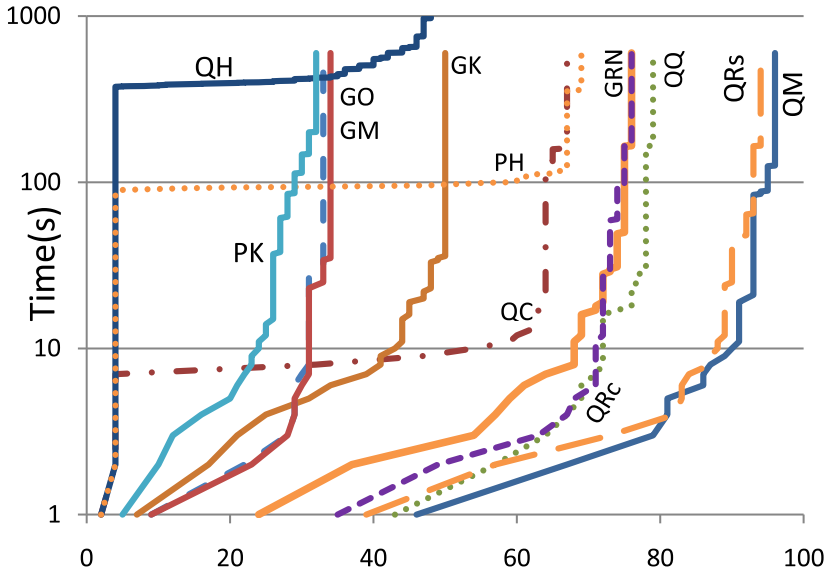


Fig. 5. Examples solved per time

faster than recent approaches with semidefinite programming relaxations for the Positivstellensatz, while Gröbner bases alone have difficulties with "real" problems. As expected, procedures tailored for real arithmetic can solve substantially more cases than Gröbner bases for general fields. Gröbner bases that integrate Fourier-Motzkin (GK) solve many more problems.

Our combination, GRN, of Gröbner bases with the real Nullstellensatz is competitive with quantifier elimination by partial CAD [3]. The experiments also show that substantial performance improvements (QR_s and QM) are still possible beyond partial CAD. Another interesting observation from the experiments is that the Positivstellensatz (PH and PK) and our GRN approach complement each other quite well. PH and GRN together can solve 84 out of 97 problems [28].

The experiments show that GM and GO are on a par. Further, QM and QR_s are very close, but clearly outperform QR_c, QQ and QC both in runtime and number of provable cases. QH is slower but competitive with the number of examples solved by GK but does not yet perform as well as other QE implementations or GRN. The performance gap between PK and PH is surprising. In part, it shows how important Harrison's optimizations [13] of Parrilo's work [6] are, but may also be caused by different heuristics for recovering rationals from floats and different enumeration orders for polynomials. This might indicate that PK, indeed, gives a more objective comparison for GRN than PH, because PK and GRN share exactly the same KeYmaera framework and rational recovering. Our new GRN procedure is a clear win compared to PK. Inevitably, performance depends on the system options and on the set of benchmarks.

6 Related Work

Nipkow [29] presented a formally verified implementations of quantifier elimination in an executable fragment of Isabelle/HOL, currently for linear real arithmetic only. McLaughlin and Harrison [11] presented a nonverified but proof-producing implementation of general quantifier elimination, so that the result of the procedure can be checked independently.

The sum of squares approach has been pioneered by Parrilo [6] and Harrison [13]. Harrison also gives optimizations for the univariate case.

Tiwari [30] presents an approach using Gröbner bases and sign conditions on variables to produce unsatisfiability witnesses for nonlinear constraints. The approach depends on appropriate heuristic variable orderings that are formed by successively introducing new variables for polynomial expressions following certain heuristics (which may not terminate). Our work and that of Tiwari share the combination of Gröbner bases with witness generation. Yet we follow semidefinite programming for the real Nullstellensatz, whereas [30] uses heuristic generation of polynomial witness expressions. Tiwari uses the Positivstellensatz to prove refutational completeness but not as part of his technique.

RSolver [2] is a numerical approach for deciding validity of (robust instances of) first-order formulas over real arithmetic extended with transcendental functions. Unlike our work, this relies on numerical stability of the input formula.

MetiTarski [31] is an interesting approach for handling special functions using a combination of resolution proving with simple QE procedures. Their focus is on handling special functions not on handling real arithmetic.

Hunt et al. [32] describe the handling of nonlinear arithmetic in ACL2, which is based on heuristic multiplication of inequalities in the style of (1) and yields an incomplete method. The method is claimed to be empirically successful, though, and can also be applied to nonlinear integer arithmetic.

7 Discussion and Conclusions

The respective approaches from Sect. 3–4 have different advantages and weaknesses for formal verification of real world problems in real arithmetic. We draw a qualitative comparison complementing the quantitative comparison from Sect. 5.

Quantifier Elimination. Quantifier elimination procedures [3] can handle full nonlinear real arithmetic, including existential quantifiers. Their implementations are quite intricate algorithms for which correctness is not easily established formally. Unfortunately, QE does not produce simple checkable certificates.

Proof-producing [11] or verified [29] QE procedures may be interesting improvements on the formal traceability of QE. Unfortunately, their performance is not yet fully competitive with other quantifier elimination implementations or our new proof-producing GRN procedure.

A compromise is reverification: Proof search [33, 18] in KeYmaera generates several problems of real arithmetic to find a proof, but only those in the final proof are soundness-critical. For soundness, it is sufficient to use a fast or

untrusted implementation of QE during the proof search and to reverify the final proof in a proof checker with a verified or proof-producing QE implementation [11, 29]. For this purpose, KeYmaera strategies are especially useful that identify the sweetspot for applying QE iteratively during the proof search [18].

Positivstellensatz. In the context of verification, a useful property of the Positivstellensatz is that it produces a witness ($s + g + m^2 = 0$) for the validity of a formula. Once the witness has been found, it is checkable by simple computations in the polynomial ring to determine whether the polynomial identity holds by comparing the coefficients. Similarly, the well-formedness of the witness can be determined by checking whether s is build from sums of squares using an extension of "completing the square" [13]. Thus, complicated numerical semidefinite programming tools [22] do not need to be part of the trusted computing base concerning soundness. Due to its enumerative nature with a large number of extra parameters, scalability with the number of variables is still limited.

Gröbner Bases. The Gröbner Basis approach does not have simple witnesses like Positivstellensatz approaches. Their working principle, however, is strictly based on symbolic computations, which can be carried out from a small set of rewrite rules within a logic. This corresponds to our built-in Gröbner basis approach GK, which is almost as efficient as external Gröbner basis implementations. Our experimental results indicate that, due to the partial ignorance of real-closed field properties, the capabilities of Gröbner bases alone are not sufficient, even in combination with Fourier-Motzkin elimination.

Real Nullstellensatz. Our new decision procedure based on Gröbner basis computations and the real Nullstellensatz share the presence of checkable witnesses with approaches based on the Positivstellensatz. Once a witness $1 + \sum_i s_i^2 = 0$ has been found, the polynomial equality check can be performed easily within a proof system using the GK rules, giving a fully formal proof. The performance in our experiments show that this new approach is promising. It outperforms most other approaches, except for highly tuned QE procedures, which lack support for formal traceability. We believe that further research in this area is likely to produce competitive but traceable solutions for real arithmetic.

Acknowledgments. We like to thank Leonardo de Moura, Nikolaj Bjørner, and John Harrison for providing benchmarks and fruitful discussions. Additionally, we like to thank Sean McLaughlin for his support in the integration of QH, PH, and QC as well as the anonymous referees for their useful comments. We also like to thank Arnold Neumaier for discussions.

References

1. Tarski, A.: A Decision Method for Elementary Algebra and Geometry. 2nd edn. University of California Press, Berkeley (1951)

2. Ratschan, S.: Efficient solving of quantified inequality constraints over the real numbers. ACM Trans. Comput. Log. **7**(4) (2006) 723–748
3. Collins, G.E., Hong, H.: Partial cylindrical algebraic decomposition for quantifier elimination. J. Symb. Comput. **12**(3) (1991) 299–328
4. Weispfenning, V.: Quantifier elimination for real algebra - the quadratic case and beyond. Appl. Algebra Eng. Commun. Comput. **8**(2) (1997) 85–101
5. Buchberger, B.: An Algorithm for Finding the Basis Elements of the Residue Class Ring of a Zero Dimensional Polynomial Ideal. PhD thesis, University of Innsbruck (1965)
6. Parrilo, P.A.: Semidefinite programming relaxations for semialgebraic problems. Math. Program. **96**(2) (2003) 293–320
7. Stengle, G.: A Nullstellensatz and a Positivstellensatz in semialgebraic geometry. Math. Ann. **207**(2) (1973) 87–97
8. Platzer, A., Quesel, J.D.: KeYmaera: A hybrid theorem prover for hybrid systems. In: IJCAR. Volume 5195 of LNCS., Springer (2008) 171–178
9. Brown, C.W.: QEPCAD B: A program for computing with semi-algebraic sets using CADs. SIGSAM Bull. **37**(4) (2003) 97–108
10. Dolzmann, A., Sturm, T.: Redlog: Computer algebra meets computer logic. ACM SIGSAM Bull. **31** (1997) 2–9
11. McLaughlin, S., Harrison, J.: A proof-producing decision procedure for real arithmetic. In Nieuwenhuis, R., ed.: CADE. Volume 3632 of LNCS., Springer (2005)
12. Borchers, B.: CSDP, a C library for semidefinite programming. Optimization Methods and Software **11**(1-4) (1999) 613–623
13. Harrison, J.: Verifying nonlinear real formulas via sums of squares. In Schneider, K., Brandt, J., eds.: TPHOLs. Volume 4732 of LNCS., Springer (2007) 102–118
14. Platzer, A.: Differential dynamic logic for hybrid systems. J. Autom. Reasoning **41**(2) (2008) 143–189
15. Beckert, B., Hähnle, R., Schmitt, P.H., eds.: Verification of Object-Oriented Software: The KeY Approach. Volume 4334 of LNCS. Springer (2007)
16. Rümmer, P.: A sequent calculus for integer arithmetic with counterexample generation. In Beckert, B., ed.: VERIFY'07 at CADE, Bremen, Germany. Volume 259 of CEUR-WS.org. (2007)
17. Schrijver, A.: Theory of Linear and Integer Programming. Wiley (1986)
18. Platzer, A.: Combining deduction and algebraic constraints for hybrid system analysis. In Beckert, B., ed.: VERIFY'07 at CADE, Bremen, Germany. Volume 259 of CEUR Workshop Proceedings., CEUR-WS.org (2007) 164–178
19. Davenport, J.H., Heintz, J.: Real quantifier elimination is doubly exponential. J. Symb. Comput. **5**(1/2) (1988) 29–35
20. Strzebonski, A.W.: Cylindrical algebraic decomposition using validated numerics. J. Symb. Comput. **41**(9) (2006) 1021–1038
21. Bochnak, J., Coste, M., Roy, M.F.: Real Algebraic Geometry. Volume 36 of Ergebnisse der Mathematik und ihrer Grenzgebiete. Springer (1998)
22. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge Univ. Press (2004)
23. Graham, R.L., Knuth, D.E., Patashnik, O.: Concrete Mathematics: A Foundation for Computer Science. Addison-Wesley Longman (1994)
24. Platzer, A., Quesel, J.D.: Logical verification and systematic parametric analysis in train control. In Egerstedt, M., Mishra, B., eds.: HSCC. LNCS, Springer (2008)
25. Kovács, L.: Aligator: A mathematica package for invariant generation (system description). In: IJCAR. Volume 5195 of LNCS., Springer (2008) 275–282
26. de Moura, L.M., Bjørner, N.: Z3: An efficient SMT solver. In Ramakrishnan, C.R., Rehof, J., eds.: TACAS. Volume 4963 of LNCS., Springer (2008) 337–340

27. Dolzmann, A., Sturm, T., Weispfenning, V.: A new approach for automatic theorem proving in real geometry. J. Autom. Reason. **21**(3) (1998) 357–380
28. Platzer, A., Quesel, J.D., Rümmer, P.: Real world verification. Reports of SFB/TR 14 AVACS 52, SFB/TR 14 AVACS (2009) ISSN: 1860-9821, http://www.avacs.org.
29. Nipkow, T.: Linear quantifier elimination. In: IJCAR. Volume 5195 of LNCS., Springer (2008)
30. Tiwari, A.: An algebraic approach for the unsatisfiability of nonlinear constraints. In Ong, C.H.L., ed.: CSL. Volume 3634 of LNCS., Springer (2005) 248–262
31. Akbarpour, B., Paulson, L.C.: Extending a resolution prover for inequalities on elementary functions. In Dershowitz, N., Voronkov, A., eds.: LPAR. Volume 4790 of LNCS., Springer (2007) 47–61
32. Warren A. Hunt, J., Krug, R.B., Moore, J.S.: Linear and nonlinear arithmetic in ACL2. In: Proceedings, Correct Hardware Design and Verification Methods, 12th IFIP Conference. Volume 2860 of LNCS., Springer (2003) 319–333
33. Platzer, A., Clarke, E.M.: Computing differential invariants of hybrid systems as fixedpoints. In Gupta, A., Malik, S., eds.: CAV. Volume 5123 of LNCS., Springer (2008) 176–189

A Soundness Proof for Gröbner Basis Rules

Proof (Proposition 1). The rules in Fig. 1 are sound. As usual for the soundness proofs we assume Γ to be true in an interpretation ν and Δ to be false as there is nothing to show otherwise. For A3, A4,A5, we show equivalence of premiss and conclusion, which implies soundness.

A2 Suppose the conclusion was false in ν, i.e., $\nu \models g_1 = \tilde{g}_1 \land \cdots \land g_n = \tilde{g}_n \land f \neq h$. Thus, $\nu \models g = 0$ for all $g \in G$. Consequently, $\nu \models g = 0$ for all polynomials g in the ideal (G) of G. As a consequence of the applicability condition, we have $\mathrm{red}_G(f - h) = 0$, which, by Def. 2, implies that $f - h$ is in the ideal of G. In combination, we have $\nu \models f - h = 0$, hence $\nu \models f = h$, which is a contradiction. □

A1 The soundness of A1 is a special case of the soundness of A2 when assuming the false formula $1 = 0$ for the succedent $f = h$.

A3 Satisfiability-equivalence of A3 is a consequence of the Rabinowitch trick equivalence $x \neq 0 \leftrightarrow \exists z\,(xz = 1)$, using $f - g$ for x. More generally, this holds in fields where non-zero elements are exactly the elements that have some inverse z. By introducing a free new variable z, we obtain that the premiss is satisfiable if and only if the conclusion is satisfiable.

A4 Satisfiability-equivalence follows from the equivalence $f \geq g \leftrightarrow \exists z\,(f - g = z^2)$ in the domain of reals. More generally, this holds in real-closed fields where squares are exactly the positive numbers. By introducing a free new variable z in the rule, this equivalence we obtain that the premiss and conclusion are satisfiability-equivalent, i.e., the premiss is satisfiable if and only if the conclusion is satisfiable.

A5 Satisfiability-equivalence follows from the equivalence $x > 0 \leftrightarrow \exists z\,(xz^2 = 1)$ in the reals, using $f - g$ for x.

A6 Since a sum of squares is nonnegative over $\mathbb{R}$, the value of $1 + s_1^2 + \cdots + s_n^2$ is strictly positive and $1 + s_1^2 + \cdots + s_n^2 = 0$ is a contradiction over the reals. Consequently, if the premiss is valid, then so is the conclusion.

Proof (Lemma 1). The first equation holds because each non-negative rational number α_i can be written as a sum of four rational squares by Lagrange's four-square theorem.

We consider the two directions of the second equation:

"$\supseteq$": This is shown (constructively) by [13, Theorem 1].

"$\subseteq$": Let $\sum_{i=1}^{l} \alpha_i (c_i p)^2$ be a sum of squares with $\alpha_i \geq 0$. We define the matrices

$$C = \begin{pmatrix} c_1^t \\ c_2^t \\ \vdots \\ c_l^t \end{pmatrix}, \qquad D = \begin{pmatrix} \alpha_1 & & & \\ & \alpha_2 & & \\ & & \ddots & \\ & & & \alpha_l \end{pmatrix}$$

and thus obtain the identity:

$$\sum_{i=1}^{l} \alpha_i (c_i p)^2 \;=\; (Cp)^t D (Cp) \;=\; p^t (C^t D C) p$$

The matrix $C^tDC = Q \in \mathrm{Q}^{k\times k}$ is positive semidefinite because of:

$$x^t(C^tDC)x \;=\; \sum_{i=1}^{l} \alpha_i (c_i x)^2 \;\geq\; 0.$$

B Completeness of Gröbner Bases for the Real Nullstellensatz

Proof (Lemma 2). First, observe that for all $x \in \mathrm{R}^n$:

$$x^tCm - x^tDm \in (B) \tag{3}$$

The proof of the lemma is as follows:

- "$\supseteq$": Suppose $x^tD = 0$. Then also $x^tDm = 0$ and, by (3), $x^tCm - x^tDm = x^tCm \in (B)$. This implies $\mathrm{red}_B(x^tCm) = 0$ because B is a Gröbner basis.
- "$\subseteq$": Suppose $\mathrm{red}_B(x^tCm) = 0$, i.e., $x^tCm \in (B)$. By (3), this implies $x^tDm \in (B)$.
  Now, observe that also the instance x^tDm is irreducible w.r.t. B: because the parametrised polynomial q^tDm is irreducible w.r.t. B, it has to be the case that the i'th component of b^tD is zero whenever the monomial m_i is reducible w.r.t. B. This means that, in this case, the i'th column of D only contains zeroes. Then also the i'th component of x^tD is zero and x^tDm cannot contain any reducible terms.
  Because B is a Gröbner basis, 0 is the only member of (B) that is irreducible w.r.t. B, which implies $x^tDm = 0$. Finally, because the elements of m are pairwise distinct and thus linearly independent, this is only possible if $x^tD = 0$.

C Full Details Experimental Results

In this section we present a table containing all experimental results produced for our evaluation of the different techniques. To get comparability, we did neither use the invariant generation techniques described in [33] but instead provided the invariants necessary a priori nor did we use our iterative background closure procedure [18].

All times are given in seconds. We write ∞ if the computation did not terminate within 600 seconds (1000 seconds for QH, because it needs $\approx$ 400 seconds for startup). If the solver terminates unsuccessfully before this timeout we mark this with –.

Example	GM	GO	GK	QQ	QM	QR_s	QR_c	QC	QH	PH	PK	GRN
aligator/conditional/lcm_gcd_dijkstra	1.8	1.9	3.2	1.9	1.5	2.3	2.1	8.7	397.5	98.6	–	2.2
aligator/conditional/simple_example_1	1.6	1.7	3.6	1.9	1.8	2.4	2.2	8.4	–	98.0	3.9	2
aligator/conditional/simple_example_2	1.7	1.6	3.7	1.8	1.4	2	1.9	8.2	394.9	94.9	3.3	1.6
aligator/conditional/simple_example_3	1.9	2	4.5	2.1	2	2.4	2.5	8.8	394.3	92.5	147.5	2.4
aligator/conditional/square_root_floor_dijkstra	2.1	2.2	5.1	2.8	2	2.7	2.5	10.2	562.2	358.3	–	2.9
aligator/conditional/square_root_zuse	1.7	2.1	4.6	2.3	1.6	2.5	2	∞	504	93.3	–	1.9
aligator/conditional/wensey_division_wegbreit	2.8	2.8	5.1	3.3	9.1	3.6	5.9	∞	∞	96.2	∞	2.6
aligator/consecutive_cubes_cohen	2.5	2.7	7.1	–	2.2	2.8	∞	∞	∞	94.1	–	2.9
aligator/division_dijkstra	0.8	0.8	1.7	0.8	0.7	1.1	0.9	7.7	389.4	95.4	–	1
aligator/fibonacci_knuth	1.9	1.8	6.3	2.2	2	2.1	2.1	9.6	–	94.5	5.6	1.9
aligator/fibonacci_stansley	1.9	1.6	4.1	1.9	1.6	1.8	1.9	8.3	396	93.6	3.9	1.8
aligator/HC_polyominoes_stansley	5.5	6.2	19.8	6.2	5.9	6.2	5.9	12.7	393.2	101.1	85.6	6.1
aligator/integer_cubic_root_knuth	2.1	2.3	6.1	–	2	2.7	2.6	9.2	408.9	95.3	–	2.2
aligator/integer_square_root_kirchner	1.6	1.9	3.7	1.6	1.4	2.1	2	8.5	–	93.6	14.1	1.8
aligator/integer_square_root_knuth	1	1	2.9	1.1	1	1.4	1.2	8.2	400.8	94.1	∞	1.1
aligator/simple_example_1	0.9	0.7	1.3	0.8	0.7	1.1	1	7.6	388.2	92.7	1.9	0.9
aligator/simple_example_2	0.8	0.8	1.8	0.8	0.6	0.9	0.9	8	379.9	91.2	1.7	0.7
aligator/simple_example_3	1.4	1.5	4.2	1.6	1.3	1.9	1.8	8.8	379.4	91.8	3.8	1.5
aligator/simple_example_4	2.1	2.5	4.9	2.3	1.9	3	2.9	13.5	∞	94.2	9.5	2.6
aligator/sum_of_powers_5_petter	1.5	1.8	7.1	1.8	1.4	1.8	1.8	8.5	380.2	93.7	–	1.8
aligator/tribonacci_stansley	3.4	3.1	9.1	3.8	3.1	3.8	3.7	9.9	426.7	94.9	11.8	3.2
ATC/roundabout/TRM-essentials	6.4	7.3	22.2	–	5.6	7.9	–	–	–	98.4	–	7
ATC/roundabout/TRM-essentials-3	35	∞	–	–	89	64.2	–	∞	–	∞	∞	∞
bouncing-ball/bouncing-ball	–	–	–	–	5.5	8	59.1	∞	∞	∞	∞	–
bouncing-ball/bouncing-ball-simple	–	–	–	–	3.6	7.2	11.5	∞	∞	∞	–	∞
complicated_arithmetic/ETCS-essentials-surprise2	–	∞	–	1.5	1.5	1.6	1.7	∞	∞	∞	∞	∞

Example	GM	GO	GK	QQ	QM	QR_s	QR_c	QC	QH	PH	PK	GRN
complicated_arithmetic/train	–	–	–	–	125.8	3.8	∞	∞	∞	∞	∞	5.4
dynamical/nonlinear2	1.6	1.6	2.7	1.7	1.5	1.5	1.7	1.6	1.6	1.7	1.6	1.5
dynamical/nonlinear4	–	–	–	165.2	84.2	165.3	165.7	171.7	∞	353.3	–	165.4
dynamical/nonlinear5	–	–	–	2.4	1.9	2.4	2.7	8.9	–	105.3	37.1	2.4
ETCS/controllability/ETCS-d-braking	–	–	–	–	1.9	2.5	2.5	∞	∞	∞	–	12
ETCS/decomposed/controllability_lemma/to_left	–	–	–	–	1.2	1.6	1.7	∞	∞	∞	∞	7.7
ETCS/decomposed/controllability_lemma/to_right	–	–	–	–	0.8	1.2	1.2	∞	∞	∞	–	7.8
ETCS/decomposed/essentials/accelerating	–	∞	–	6.9	1.4	2.2	–	∞	∞	∞	∞	∞
ETCS/decomposed/essentials/breaking	–	–	6.8	17.5	1.6	1.8	1.9	9.3	∞	∞	∞	4.4
ETCS/decomposed/essentials/invariant-initially-valid	0.2	0.2	0.3	0.2	0.3	0.2	0.2	0.2	0.3	0.2	0.2	0.2
ETCS/decomposed/essentials/use-case	–	–	–	0.7	0.5	0.8	1.2	7.6	480	93.5	∞	1.3
ETCS/decomposed/safety-lemma/initially-valid	0.3	0.3	0.4	0.3	0.4	0.3	0.4	0.4	0.3	0.4	0.4	0.3
ETCS/decomposed/safety-lemma/rbc_goal1	–	–	5	28.1	2.8	2.9	95.2	9.5	∞	118.8	∞	3.9
ETCS/decomposed/safety-lemma/rbc_goal2	1.6	1.6	2.1	1.6	1.7	1.7	1.6	1.5	1.6	1.7	1.7	1.8
ETCS/decomposed/safety-lemma/train_goal1	–	–	9.4	16.4	2.5	3.1	3	10.9	∞	∞	∞	6.2
ETCS/decomposed/safety-lemma/train_goal2	–	–	6.3	4.3	2.6	2.8	3.4	10.1	∞	∞	∞	4.5
ETCS/decomposed/safety-lemma/train_goal3	–	–	–	–	5.3	∞	∞	∞	∞	∞	∞	∞
ETCS/decomposed/safety-lemma/train_goal4	–	–	10	16.2	2.7	3.1	3.1	10	∞	∞	∞	16.7
ETCS/decomposed/safety-lemma/train_goal5	–	–	6.2	4.7	2.9	3.1	3.1	10.1	∞	∞	∞	5.7
ETCS/decomposed/safety-lemma/train_goal6	–	–	–	–	8.9	∞	∞	∞	∞	∞	∞	49.5
ETCS/decomposed/safety-lemma/use-case	–	–	–	1.7	1.3	1.6	4.2	8.2	∞	∞	∞	2
ETCS/paper/rbc-controllability-characterisation	–	–	–	3.2	2.8	3.5	5.1	11.7	∞	–	–	–
ETCS/paper/rbc-controllability-lemma	–	–	4.7	2.4	2	2.2	2.8	8.8	∞	95.8	–	2.4
ETCS/safety/binary_driver-2007-10-09	–	–	–	–	7.2	11.2	∞	∞	∞	∞	∞	∞
ETCS/safety/ETCS-essentials	–	–	–	24.2	2.9	4.1	–	∞	∞	∞	∞	∞
accel-simple	–	–	15.6	–	5.3	9.4	∞	∞	∞	∞	∞	7.4

Example	GM	GO	GK	QQ	QM	QR$_s$	QR$_c$	QC	QH	PH	PK	GRN
magnetic_field	24.2	30.4	33.8	–	19.1	47.5	∞	–	–	–	∞	30.1
magnetic_field-simplified	23.8	27.9	35.6	–	20.6	42.7	∞	–	–	–	∞	28.6
moving-point	–	–	–	2	1.7	2.2	2.3	12.6	∞	∞	–	∞
water_tank/water_tank	–	–	19.8	17.2	8.9	24.8	27.6	133	–	∞	∞	∞
weispfennig/angle	–	–	–	–	∞	1.5	∞	∞	∞	∞	∞	2.9
weispfennig/angle2	–	–	–	–	10.7	546.8	–	∞	∞	∞	–	∞
weispfennig/pedos_inequality	–	–	–	1.5	1.4	1.6	1.9	∞	∞	111.1	∞	–
realTacletPOs/decompose_mult	–	–	0.2	0.5	0.3	0.8	–	7.3	389.1	90.5	2.2	0.5
realTacletPOs/inEqSimp_contradInEq2	–	–	–	0.6	0.6	0.8	–	7.8	479.2	99.0	–	2.1
realTacletPOs/inEqSimp_contradInEq20	–	–	–	0.7	0.6	0.9	–	7.9	433.9	90.0	–	2.1
realTacletPOs/inEqSimp_exactShadow0	–	–	–	0.6	0.4	0.6	–	7.5	448.4	108.9	–	2.1
realTacletPOs/inEqSimp_exactShadow01	–	–	–	0.6	0.5	0.6	–	7.5	652.4	96.4	–	2.1
realTacletPOs/inEqSimp_subsumption2	–	–	–	0.7	0.6	0.9	–	7.7	418.4	112.8	–	2.5
realTacletPOs/inEqSimp_subsumption20	–	–	–	0.7	0.6	0.8	–	7.8	420	∞	∞	2.9
realTacletPOs/multiply_inEq0	–	–	–	0.5	0.4	0.7	–	7.4	422.7	91.2	6	1
harrison	–	–	–	0.6	0.5	0.7	0.7	8.3	754.4	95.5	7.1	7.7
harrison2	–	–	–	0.8	0.4	1.1	–	∞	∞	94.1	–	∞
harrison3	–	–	–	0.4	0.3	0.5	0.5	158.1	∞	113.0	∞	∞
harrison4	–	–	–	0.6	0.5	0.8	0.9	9.3	∞	95.7	∞	–
semi_definite_polynomials/quaternary2	–	–	–	7.7	0.7	3.4	2.5	∞	∞	93.8	–	∞
semi_definite_polynomials/quaternary4	–	–	–	0.8	0.3	1.8	∞	∞	∞	97.2	–	∞
semi_definite_polynomials/ternary1	–	–	–	0.4	0.2	0.4	0.5	∞	∞	91.0	–	∞
semi_definite_polynomials/ternary2	–	–	–	0.4	0.2	0.8	0.8	∞	∞	93.5	61.9	16.4
semi_definite_polynomials/ternary4	–	–	–	0.6	0.2	0.9	0.8	∞	∞	93.4	200.9	18.6
semi_definite_polynomials/ternary5	–	–	–	0.7	0.3	0.9	0.9	∞	–	∞	–	∞
z3/nl10	–	–	1.1	0.5	0.3	0.6	0.6	7.7	406.3	94.4	4.1	0.6

Example	GM	GO	GK	QQ	QM	QR_s	QR_c	QC	QH	PH	PK	GRN
z3/nl12	0.4	0.4	0.7	0.4	0.4	0.5	0.6	7.6	387.6	91.2	4.4	0.5
z3/nl14	–	–	1	0.6	0.5	0.6	0.7	7.3	401.2	99.0	–	0.9
z3/nl20	0.4	0.5	1	0.5	0.4	0.7	0.7	7.9	601.3	92.5	1	0.6
z3/nl21	0.5	0.5	1.1	0.5	0.4	0.6	0.6	9.1	642.9	91.6	0.8	0.5
z3/nl3	1	1.4	2.6	0.8	0.6	0.8	0.9	7.7	552.5	89.9	∞	1.5
z3/nl33	–	–	–	0.5	0.4	0.6	0.6	7.6	407.4	97.3	∞	1
z3/nl36	–	–	–	0.4	0.3	0.5	0.5	7.3	384.4	95.9	–	0.5
z3/nl4	–	–	–	0.3	0.2	0.4	0.4	7.8	602	93.8	0.5	0.3
z3/nl40	–	–	0.3	0.4	0.3	0.5	0.5	7.5	392	93.7	–	0.5
z3/nl42	–	–	–	0.5	0.4	0.6	0.7	7.6	426.1	112.6	–	∞
z3/nl43	–	–	–	0.6	0.5	0.6	0.7	8	399.6	93.1	∞	0.9
z3/nl5	–	–	–	0.5	0.3	0.5	0.5	7.3	399.3	93.4	∞	0.8
z3/nl52	–	–	1.5	0.9	0.8	0.9	1	19.3	∞	92.5	∞	3.3
z3/nl54	0.4	0.4	1	0.4	0.3	0.6	0.5	7.4	378.1	99.8	4.4	0.5
z3/nl55	–	–	1.6	0.6	0.4	0.6	0.7	7.6	390.3	90.9	113.9	0.8
z3/nl56	–	–	–	0.5	0.3	0.6	0.6	7.5	506	92.2	∞	0.8
z3/nl57	–	–	–	0.5	0.4	0.5	0.6	8.3	970.7	99.3	∞	0.7
z3/nl60	–	–	0.5	0.4	0.3	0.6	0.5	7.4	381.9	92.7	2.8	–
z3/nl9	–	–	0.4	0.4	0.3	0.5	0.5	7.5	377	94.5	1.9	0.8