# Combined Decision Techniques for the Existential Theory of the Reals *

Grant Olney Passmore and Paul B. Jackson
{s0793114, pbj}@inf.ed.ac.uk

LFCS, University of Edinburgh

**Abstract.** Methods for deciding quantifier-free non-linear arithmetical conjectures over $\mathbb{R}$ are crucial in the formal verification of many real-world systems and in formalised mathematics. While non-linear (rational function) arithmetic over $\mathbb{R}$ is decidable, it is fundamentally infeasible: any general decision method for this problem is worst-case exponential in the dimension (number of variables) of the formula being analysed. This is unfortunate, as many practical applications of real algebraic decision methods require reasoning about high-dimensional conjectures. Despite their inherent infeasibility, a number of different decision methods have been developed, most of which have "sweet spots" – e.g., types of problems for which they perform much better than they do in general. Such "sweet spots" can in many cases be heuristically combined to solve problems that are out of reach of the individual decision methods when used in isolation. RAHD ("Real Algebra in High Dimensions") is a theorem prover that works to combine a collection of real algebraic decision methods in ways that exploit their respective "sweet-spots." We discuss high-level mathematical and design aspects of RAHD and illustrate its use on a number of examples.

## 1 Introduction

RAHD ("Real Algebra in High Dimensions") is a tool for proving high-dimensional (many variable) quantifier-free non-linear theorems in the language of ordered fields over real closed fields (RCF)[1]. While the elementary theory of the real numbers in this language[2] is decidable, it is fundamentally infeasible: any general decision method must take time exponential in the dimension of the formula

* The authors would like to thank Bruno Dutertre, Sam Owre, John Rushby, N. Shankar, Hassen Saïdi, and Ashish Tiwari of SRI International for their ever helpful support and guidance for this project, including a visiting fellowship for the first author under which this work was originated. This fellowship was supported by NASA Cooperative Agreement NNX08AC59A and by NSF SGER Grant No. CNS-0823086.

[1] A real closed field is a structure elementarily equivalent to the real number line with respect to a language of quantified boolean combinations of real polynomial equations and inequalities. This language is often referred to as the *language of ordered rings*.

[2] Classical work on RCF decision problems usually takes place over the language of *ordered rings*, not the language of *ordered fields,* as partial functions such as

being analysed. This is unfortunate, as many important applications of decision methods over RCF require reasoning about high-dimensional conjectures. To combat this difficulty, we focus not on the general decision problem, but instead upon deciding certain classes of sentences that arise in practice. We exploit the fact that most RCF decision methods have "sweet spots," e.g. types of problems for which they perform much better than they do in general, and such "sweet spots" can be heuristically combined to solve problems that are out of reach of the individual decision methods when used in isolation.

For the examples in this article, we focus especially upon the combination of a "sweet-spot" in the cylindrical algebraic decomposition procedure (for topologically open constraints), Gröbner basis calculations, Sturm chains, simple Positivstellensatz witnesses, and dimensional reduction techniques stemming from sound approximations to computations induced by real radical ideals.

### 1.1 Background

Since Tarski [17] established that the full elementary theory of RCF admits quantifier elimination (QE) by giving a QE procedure of non-elementary complexity, perhaps the most important practical[3] break-through for theorem proving over RCF has been the cylindrical algebraic decomposition (CAD) algorithm devised by Collins in the 1970s [3]. Collins' community of students have been prolific in their theoretical and practical improvements to CAD over the last twenty-five years, culminating in Brown's actively supported QEPCAD-B [1] system. In addition to QEPCAD-B, versions of CAD have also been implemented in Mathematica, REDLOG/Reduce, and Maple, and a perusal of the literature shows CAD implementations finding vigorous use in many sciences, both of applied and theoretical character. In addition to CAD, a number of other RCF QE procedures have been developed and implemented in working tools since the 1980s, including Weispfenning's method of virtual term substitution [21] (as implemented in Reduce/Redlog), and the Harrison-McLaughlin proof producing version of the Cohen-Hörmander method (in the HOL Light proof assistant) [15]. The version of Weispfenning's method available in Reduce/Redlog (implemented and enhanced by Dolzmann and Sturm [7]) performs especially well on many difficult high-dimensional problems (see Section 3).

While work on improved RCF QE methods is of lasting importance, for many practical applications, full elementary QE is overkill. For these domains (such as

real division complicate the model theory. RAHD supports the language of ordered fields by pre-processing away division in literals in terms of equivalent multiplicative constraints.

[3] This is not to say that Collins's break-through was only of a practical nature: The geometrical insight contained within the CAD procedure has led to huge advances in the topological and model-theoretic understanding of ordered structures admitting quantifier elimination (e.g. o-minimality theory and tame topology). In these cases, the properties of RCF exploited by the CAD procedure have been generalized into the notion of "cellularly decomposable structures" and now bear rich mathematical fruits. [20]

program analysis [19], hardware verification [11], hybrid systems [18], and even ongoing large-scale projects in formalised mathematics [10]), simply *deciding the satisfiability* of boolean combinations of polynomial equations and inequalities over the real numbers is often sufficient. This problem is equivalent to QE for the purely $\exists$ (dually purely $\forall$) sentential fragment of the elementary theory of RCF, in which all formulas considered are sentences consisting only of a single block of non-alternating quantifiers. As will be discussed in Section 1.2, this fragment of the elementary theory of RCF admits an exponential speed-up over general RCF QE, though this fact has unfortunately not led to algorithms for this fragment that are in practice superior to the known general QE ones.

That said, the fundamental observation driving our current research is the following: While all RCF decision methods are constrained by the known complexity lower-bounds, most decision methods have types of problems for which they perform much better than their worst-case time complexity analysis would suggest. We refer to these more feasible fragments of a decision method's input domain as "sweet spots" of the decision method under investigation. We work in RAHD to orchestrate the heuristic combination of a number of decision methods for different RCF fragments by attempting to automatically massage difficult problem instances into equisatisfiable sequences of simpler problems that fit within known "sweet spots" of the decision methods RAHD provides. We will describe RAHD in more detail in Section 2.1.

### 1.2 Existential Decisions over Real Closed Fields

Let us make the fundamental decision problem in which we are interested precise.

**Question 1 (Fundamental Decision Problem)** *Let* $t_1(\boldsymbol{x}), \ldots, t_k(\boldsymbol{x}) \in \mathbb{R}(\boldsymbol{x})$ *where* $\mathbb{R}(\boldsymbol{x}) = \{\frac{p}{q} \mid p, q \in \mathbb{R}[x_1, \ldots, x_n], q \neq 0\}$. *Let* $\varphi$ *be a quantifier-free boolean combination of atoms of the form* $(t_i \odot 0)$ *with* $\odot \in \{<, \leq, =, \geq, >\}$ $(1 \leq i \leq k)$. *Is* $\varphi$ *satisfiable over* $\mathbb{R}^n$*? That is, does*

$$\langle \mathbb{R}, +, -, *, <, 0, 1 \rangle \models \exists x_1 \ldots x_n(\varphi)?$$

Though this decision problem has long been known to have a positive solution, available general purpose decision methods, such as the aforementioned CAD [3] or the Cohen-Hörmander procedure [2], have a very high worst-case time complexity. For instance, when given an $n$-dimensional existential RCF conjecture such as $\exists \boldsymbol{x} \varphi(\boldsymbol{x})$, the computing time for the CAD algorithm is dominated by a function doubly exponential in $n$. For the full theory of RCF (e.g., with arbitrary quantification), such doubly-exponential lower-bounds are known to be tight for quantifier elimination: Due to Davenport-Heinz [4], there are known families of $n$-dimensional RCF formulas of length $O(n)$ whose only quantifier-free equivalences must contain polynomials of degree $2^{2^{\Omega(n)}}$ and of length $2^{2^{\Omega(n)}}$.

For the question of decidability over the purely existential fragment, a number of more efficient algorithms have been proposed, including those of Grigor'ev-Vorobjov [9] and Renegar [16]. Both of these decision methods deliver a theoretical exponential speed-up over CAD for the existential fragment of RCF, requiring

time dominated by a function only singly exponential in the underlying dimension. Despite their apparent complexity-theoretic advantages, neither procedure appears to have been implemented. Analysis by Hong [12] suggests that even though the procedures of Grigor'ev-Vorobjov and Renegar are theoretically advantageous over CAD (e.g., for sufficiently large inputs), for practically sized examples, CAD remains superior. This is due to infeasibly large constant factors lurking behind the asymptotic analyses of these singly exponential procedures.

## 2 The RAHD System

In this section, we begin by touching upon the mathematics of some of RAHD's different proof techniques. Then, we turn to the question of how these techniques can be fruitfully combined to solve problems beyond their reaches when they are used in isolation. This leads us to a discussion of design and control aspects of RAHD's automatic heuristic proof procedure, the so-called "waterfall."

### 2.1 RCF decision-theoretic "sweet spots"

We begin by describing some decision-theoretic "sweet spots" that are combined and exploited in RAHD.

**Full and Fragmentary Open CAD (CAD-MD)** In 1993, McCallum showed how CAD could be heavily modified and made much more efficient if the semi-algebraic set defined by the formula being analysed was known to be an open set in the Euclidean topology on $\mathbb{R}^n$ [14]. This can be guaranteed if the formula under analysis consists only of strict inequality relations. The basic idea is that in these cases, rational sample points can be selected from the cells in the CAD, avoiding the need for costly irrational algebraic number computations that are in general required during normal CAD operation.

This more feasible fragment of CAD is used heavily in RAHD, and as will be discussed in Section 2.3, RAHD's design is centered around breaking difficult problems into simpler ones that are in a precise sense closer to being able to make use of this "sweet spot." When RAHD encounters a problem that it can see falls into this fragment, Brown's QEPCAD-B is used in a special mode[4] designed for making decisions about the emptiness of open sets.

**Gröbner bases** Though one usually associates Gröbner basis calculations with decisions over algebraically closed fields, much use can be made out of them for existential RCF decisions. The reason is two-fold: First, if it can be shown that a collection of equational constraints in a formula being analysed is unsatisfiable over $\mathbb{C}^n$, then this collection is of course unsatisfiable over $\mathbb{R}^n$. Even if one is

[4] To use this mode, one replaces all existential quantifiers in the constraint in question with special "exists infinitely many" quantifiers which are equivalent over open sets and cause many irrational algebraic number calculations to be avoided.

not so lucky, all term reductions induced by the equational constraints when interpreted over $\mathbb{C}^n$ are still valid over $\mathbb{R}^n$, and so one can make use of Gröbner basis calculations to simplify polynomials in the midst of RCF decisions. In practice, these reductions are often a tremendous boon, leading to simplified terms that are then more amenable to subsequently invoked decision methods. As will be seen in Section 2.2, RAHD also exploits Gröbner basis calculations for a number of other techniques centered around reducing a problem's dimension, and these techniques can derive new equations which then further enhance the reduction power of the Gröbner bases for the problem. RAHD uses a caching mechanism for sharing already computed Gröbner bases and term reductions among these different system components. RAHD has its own implementation of algorithms for computing with ideals that tend to work well for non-linear problems in six or less variables. When a non-linear problem is handed to RAHD in greater than six variables, RAHD will attempt to use the CoCoA computer algebra system for its ideal computations if it is available.

**Simple Positivstellensatz certificates** Recent work exploiting Stengle's Weak Positivstellensatz, an RCF analogue of Hilbert's Nullstellensatz for algebraically closed fields, has led to a number of computational advances for the fundamental decision problem, and a simplified form of this result is currently used in RAHD.

**Theorem 1 (Stengle's Weak Positivstellensatz)** *Given* $p_i(\boldsymbol{x}), q_i(\boldsymbol{x}), s_i(\boldsymbol{x}) \in \mathbb{R}[\boldsymbol{x}] = \mathbb{R}[x_1, \ldots, x_n]$*, the conjunctive constraint system*

$$\varphi = \begin{pmatrix} p_1(\boldsymbol{x}) = 0 \wedge \ldots \wedge p_k(\boldsymbol{x}) = 0 \\ \wedge\ q_1(\boldsymbol{x}) \geq 0 \wedge \ldots \wedge q_l(\boldsymbol{x}) \geq 0 \\ \wedge\ s_1(\boldsymbol{x}) > 0 \wedge \ldots \wedge s_m(\boldsymbol{x}) > 0 \end{pmatrix}$$

*is unsatisfiable over* $\mathbb{R}^n$ *iff*

$$\begin{gathered} \exists \mathcal{P}(\boldsymbol{x}) \in Ideal(p_1, \ldots, p_k) \\ \exists \mathcal{Q}(\boldsymbol{x}) \in Cone(q_1, \ldots, q_l) \\ \exists \mathcal{R}(\boldsymbol{x}) \in Mon(s_1, \ldots, s_m) \\ s.t.\ \ \mathcal{P} + \mathcal{Q} + \mathcal{R}^2 = -1 \end{gathered}$$

*where*

$$
\begin{aligned}
Ideal(p_1, \ldots, p_k) &= \left\{ \sum_{i=1}^{k} p_i q_i \mid q_i \in \mathbb{R}[\boldsymbol{x}] \right\}, \\
Cone(q_1, \ldots, q_l) &= \left\{ r + \sum_{i=1}^{v} t_i u_i \mid r, t_i \in \sum (\mathbb{R}[\boldsymbol{x}])^2,\ u_i \in Mon(q_1, \ldots, q_v) \mid v \in \mathbb{N} \right\}, \\
Mon(s_1, \ldots, s_m) &= \left\{ \prod_{i=1}^{m} (s_i)^j \mid j \in \mathbb{N} \right\}, \\
\sum (\mathbb{R}[\boldsymbol{x}])^2 &= \left\{ \sum_{i=1}^{v} (p_i)^2 \mid p_i \in \mathbb{R}[\boldsymbol{x}] \mid v \in \mathbb{N} \right\}.
\end{aligned}
$$

Given such an unsatisfiable $\varphi$, the equation $\mathcal{P} + \mathcal{Q} + \mathcal{R}^2 = -1$ is called a *Positivstellensatz (Psatz) certificate* for $\varphi$'s unsatisfiability. It is the finding of such certificates that has seen impressive modern advances: Building upon the work of Choi, Lam, Powers, Reznick, and Wörmann, Parrilo developed in his 2000 dissertation a feasible method for finding Psatz certificates, by translating the search for them into a convex optimization (semidefinite programming) problem that is in principle amenable to polynomial-time interior point methods. The complexity-theoretic difficulty lies in the fact that each polynomial-time solvable optimization problem only searches for a Psatz certificate up to a set multivariate total degree, and the known bounds on such degrees are at least triply exponential in $\varphi$. Still, many difficult problems are routinely solved by Psatz methods. In our experience, however, it is rare[5] to find a $\varphi$ feasibly solvable by Psatz methods and not by CAD, and so we currently use only a very simplified restriction of Psatz methods in RAHD that does no convex optimization but is nevertheless useful for many practical problems.

The family of simple Positivstellensatz witnesses RAHD considers are those which contain constraints of the following form:

$$
(p = 0) \ \text{ s.t. } \ RC(p) > 0 \wedge p \in \left\{ \sum_{j=1}^{k} m^2 \mid m = \prod_{j=1}^{n} c_j x_j^{\alpha(j)} \mid c_j \in \mathbb{Q}, \alpha(j), k \in \mathbb{N} \right\}
$$

where $p \in \mathbb{Q}[\boldsymbol{x}]$ and $RC(p)$ is the degree-zero rational coefficient of $p$. RAHD also looks for related Psatz certificates when $p$ is constrainted via a strict inequality. Such certificates can be found simply by examining the degree parity of the monomials arising in $p$ after polynomial canonicalization, which is a polynomial-time process all terms in RAHD undergo before Gröbner basis calculations.

[5] For those interested in foundational theorem proving, however, Psatz methods do have a clear advantage over CAD: The fact that Psatz certificates are simple algebraic identities guarantees that if found, they can be verified and translated into foundational proof objects easily. Harrison has made use of this fact for his REAL_SOS tactic in HOL Light [11].

**Sturm chains for univariate constraints** Sturm's theorem prescribes a method for counting the number of real roots of a univariate polynomial $p$ in a half-open interval through the analysis of the number of sign changes (SC) in the so-called *Sturm chain* induced by $p$ and the interval in question. Despite the fact that Sturm chains have well-known pathological numerical properties, we have found them to be useful in RAHD on many practical problems. The family of formulas amenable to Sturm chain analysis in RAHD are those which contain constraints of the form $[(p > 0) \ \wedge \ (x > q_1) \ \wedge \ (x < q_2)]$ s.t. $p(\frac{q_2 - q_1}{2}) \leq 0, \ \#SC(p, (q_1, q_2)) = 0, \ \text{and} \ q_1 < q_2$ with $p \in \mathbb{Q}[x]$.

### 2.2 Dimensional reduction techniques

As the time complexities of many algebraic decision methods applicable to RCF formulas are at least exponential in dimension, methods for reducing the dimension of a formula under analysis (e.g. the elimination of variables) are crucial to the feasible solubility of high-dimensional problems. This process is a central component of RAHD's automatic waterfall procedure, which spawns lower-dimensional equisatisfiable subgoals and calls itself recursively upon them when the dimension of a constraint under analysis has been reduced. Because of this subgoaling process, dimensional reductions in RAHD are allowed to eliminate a variable in terms of *any finite number* of lower-dimensional values, with each such value inducing a separate subgoal to be checked for satisfiability. For example, the transformation

$$\big(x^4 - 16 = 0 \quad \wedge \quad \mathcal{P}(x, y, z)\big) \quad \rightarrow \quad (\mathcal{P}(2, y, z) \quad \vee \quad \mathcal{P}(-2, y, z))$$

is a valid[6] dimensional reduction for an RCF predicate $\mathcal{P}$ from $\mathbb{R}^3$ to $\mathbb{R}^2$.

**Approximating real radical ideals** Over $\mathbb{C}$, the correspondence between ideals and varieties is elucidated by Hilbert's Strong Nullstellensatz.

**Theorem 2 (Hilbert's Strong Nullstellensatz)**

$$\begin{aligned}\mathcal{I}_{\mathbb{C}[\boldsymbol{x}]}(\mathcal{V}_{\mathbb{C}}(\mathcal{I}_{\mathbb{C}[\boldsymbol{x}]}(p_1, \ldots, p_k))) &= \sqrt{\mathcal{I}_{\mathbb{C}[\boldsymbol{x}]}(p_1, \ldots, p_k)} \\ &= \left\{p \in \mathbb{C}[\boldsymbol{x}] \mid \exists i \in \mathbb{N} \ s.t. \ p^i \in \mathcal{I}_{\mathbb{C}[\boldsymbol{x}]}(p_1, \ldots, p_k)\right\}.\end{aligned}$$

[6] It is interesting to note that while this reduction is valid over $\mathbb{R}$, it is not valid over $\mathbb{C}$: Consider the non-trivial quartic roots of unity $e^{\pm\frac{\pi i}{2}}$. Thus, this reduction would not be computed from any Gröbner basis for $\sqrt{\mathcal{I}(x^4 - 16)}$. This motivates the need for computations over *real* radical ideals, so that such reductions can be deduced.

That is, given $p_i, q \in \mathbb{C}[\boldsymbol{x}]$ the decision problem for universal Horn formulas over $\mathbb{C}$ can be reduced to an ideal membership check for radical ideals as follows:

$$\langle \mathbb{C}, +, -, *, 0, 1 \rangle \models \left( \bigwedge_{i=1}^{k} p_i = 0 \right) \Longrightarrow q = 0$$

$$\iff$$

$$q \in \sqrt{\mathcal{I}_{\mathbb{C}[\boldsymbol{x}]}(p_1, \ldots, p_k)}$$

which can then be effectively solved using Buchberger's algorithm. Modulo ideal membership checking, the important step here is the construction, from a set of generators for an ideal $\mathfrak{I}$ over $\mathbb{C}[\boldsymbol{x}]$, to a set of generators for the *radical ideal* containing $\mathfrak{I}$. This is a classically studied problem in algebraic geometry and most modern computer algebra systems provide efficient algorithms for complex ideal radicalization [13].

Over $\mathbb{R}$, however, things are not so simple. The algebraic structure analogous to a radical ideal for real algebraic varieties, the so-called *real radical ideal*, has to take into account the order structure of $\mathbb{R}$ by incorporating polynomial summands that are sums of squares. That is, letting $\mathfrak{I} = \mathcal{I}_{\mathbb{R}[\boldsymbol{x}]}(p_1, \ldots, p_k)$,

$$\mathcal{I}_{\mathbb{R}[\boldsymbol{x}]}(\mathcal{V}_{\mathbb{R}}(\mathfrak{I})) = \quad \sqrt[\mathbb{R}]{\mathfrak{I}} = \quad \left\{ p \in \mathbb{R}[\boldsymbol{x}] \mid p^{2i} + s \in \mathfrak{I} \mid s \in \sum (\mathbb{R}[\boldsymbol{x}])^2,\ i \in \mathbb{N} \right\}.$$

Known methods for transforming an ideal into its real radical are computationally infeasible for non-trivial problems, so we seek a method in RAHD that *approximates* real radicalization to obtain some practically useful membership decisions in an efficient way. The following is an example inference rule that captures some of this desired behavior:

$$\frac{(p_1 = 0), \ldots, (p_k = 0) \in \mathcal{C} \quad \exists m \in \mathbb{N} \text{ s.t. } (x^{2m} - q) \in \mathcal{I}(p_1, \ldots, p_k) \quad \sqrt[2m]{q} \in \mathbb{Q}}{\mathcal{C} \models (x = \sqrt[2m]{q} \ \vee \ x = -\sqrt[2m]{q})}$$

where $\mathcal{C}$ is a conjunctive RCF constraint. Note that the $q$ above need not be guessed, as if the antecedent holds, one can obtain $q$ by reducing $x^{2m}$ modulo $GB(p_1, \ldots, p_k)$. This reduction process can be done incrementally for heuristically selected terms in a formula, with $m$ ranging from 1 to some degree bound computed as a function of the generators of $GB(p_1, \ldots, p_k)$. Indeed, many important dimensional reductions in RAHD are accomplished in this way.

**Reverse Rabinoswitch encodings** Over $\mathbb{R}$, the following equivalences hold:

$$pq = 0 \iff (p = 0 \ \vee \ q = 0) \tag{1}$$

$$\sum_{i=1}^{k} p_i^2 = 0 \iff \bigwedge_{i=1}^{k} p_i = 0. \tag{2}$$

Equivalence (1) can be used to reduce a constraint $\varphi$ to an equisatisfiable disjunction $\varphi_1 \vee \varphi_2$ s.t. (i) $dim(\varphi_i) < dim(\varphi)$ in the special case that $pq$ is a monomial, or (ii) each $\varphi_i$ has additional polynomial vanishing assumptions (and hence Gröbner bases with enhanced reduction power) in the general case. Equivalence (2) can be used to reduce a constraint $\varphi$ to an equisatisfiable constraint $\varphi'$ which contains new, simpler equations which enrich the equational structure of $\varphi$ and increase the reduction power of the Gröbner bases it induces. It is interesting to note that the integral domain property (1) is valid over $\mathbb{C}$, while (2) is valid only over $\mathbb{R}$. As in the restricted version of the Psatz used by RAHD, instances of (2) are recognized only for sums of squares of monomials.

**Gröbner bases induced solving for variables and interval techniques** RAHD contains a number of additional techniques that use Gröbner bases and term orderings to solve for variables by orienting equations, constructing elimination ideals, and dividing equations through by polynomials that RAHD can prove, via spawned subgoals, to be non-vanishing under constraint hypotheses. RAHD also has a number of simple interval arithmetic reasoning mechanisms that work together with Gröbner basis reductions. This is because interval-based inconsistencies in formulas are often easier to recognize when the polynomials appearing in formula inequalities have been canonicalized through Gröbner basis reductions.

### 2.3 Proof strategy

**Preprocessing of goals into goal-sets** RAHD sessions begin with the installation of a *goal*. In RAHD, a goal can be any quantifier-free formula in the language of ordered fields. All variables in RAHD goals are implicitly existentially quantified. We consider that a goal $\mathcal{G}$ in $n$ variables has been "proved" if it has been shown to be *unsatisfiable* over $\mathbb{R}^n$. Equivalently, the semialgebraic set defined by $\mathcal{G}$ (e.g., the set $S = \{\boldsymbol{x} \in \mathbb{R}^n \mid \langle \mathbb{R}, +, -, *, <, 0, 1\rangle \models \mathcal{G}(\boldsymbol{x})\}$) is empty. Mathematically, the theorem proved is the universal dual of the existential goal being analysed. If in the process of trying to prove unsatisfiability of a goal RAHD instead finds it to be satisfiable, then this is reported as a *counter-example* to the conjecture installed as the current goal.

Once a goal is installed, its satisfiability cannot be decided until it has been pre-processed into a *goal-set*, $\mathcal{GS}$. A goal-set for a goal is an equivalent formula in disjunctive normal form (DNF). We consider the DNF formula as set of *cases*, each case being one of the conjunctions of literals. This pre-processing a goal into a goal-set also involves some normalisation transformations. For example, all occurrences of division are eliminated, each non-strict inequality is transformed into an equivalent disjunction consisting of an equation and a strict inequality, and each disequality is transformed into an equivalent disjunction of two strict inequalities. The result is that every literal in a goal-set is either an equality or a strict inequality, and every arithmetic expression is a polynomial. The next paragraph explains why we do this pre-processing.

Recall two important "sweet spots" mentioned previously: First, CAD can be made much more efficient if the semialgebraic set defined by the formula being analysed is known to be an open set. This openness can be guaranteed if the relation symbols in the formula being analysed are only strict inequalities. Second, terms appearing in formulas that contain equational constraints can be simplified by injecting those terms into the quotient ring induced by the ideal generated by the equational constraints. This can be done feasibly using Gröbner basis calculations. By decomposing non-strict inequalities into their requisite strict inequality and equational cases, we get closer to being able to exploit both "sweet spots": one of the two cases of our formula is now closer to being topologically open and thus suitable for CAD-MD, while the other case now has a richer equational structure inducing a potentially larger ideal, which can be exploited by Gröbner basis calculations resulting in more substantial term reductions[7]. Moreover, as members of the goal-set are purely conjunctive, we can exploit the following property (given $c \in \mathcal{GS}$):

$$\frac{\{l_1, \ldots, l_k\} \subseteq c \wedge \{\boldsymbol{x} \in \mathbb{R}^n \mid \langle \mathbb{R}, +, -, *, <, 0, 1\rangle \models \bigwedge_{i=1}^{k} l_i(\boldsymbol{x})\} = \emptyset}{\{\boldsymbol{x} \in \mathbb{R}^n \mid \langle \mathbb{R}, +, -, *, <, 0, 1\rangle \models c(\boldsymbol{x})\} = \emptyset}.$$

That is, to prove the unsatisfiability of some case $c \in \mathcal{GS}$, it suffices to prove the unsatisfiability of *any* subset of $c$. Considering this fact in conjunction with the aforementioned "sweet spots," we see now another way this strict-inequational/equational splitting can help us use the more feasible open fragment of CAD, even on the equational branch of a split non-strict inequality. We illustrate this by an example.

*Example 1.* Let $\varphi = ((p_1 \leq 0) \wedge \psi)$ s.t. (WLOG) $\psi$ consists only of conjoined strict inequalities and equations. Let $\varphi$ be split into $\varphi_1 = ((p_1 < 0) \wedge \psi)$ and $\varphi_2 = ((p_1 = 0) \wedge \psi)$, which will both be checked for satisfiability. Observe that the ideal generated by the equations in $\varphi_2$, $\mathcal{I}(\varphi_2)$, is a (possibly non-strict) superset of the corresponding ideals of $\varphi$ and $\varphi_1$. Now, fix a term ordering and reduce all polynomials appearing in the strict inequalities in $\varphi_2$ with respect to $GB(\mathcal{I}(\varphi_2))$ to obtain an equisatisfiable formula $\varphi_2'$. Observe that the strict inequalities in $\varphi_2'$ have now been potentially enriched with information contained in the equations of $\varphi_2$. We can now use the above observation on unsatisfiable subsets of conjoined constraints and examine the satisfiability of only the strict-inequational fragment of $\varphi_2'$. As this fragment is open, we may now use CAD-MD to decide its satisfiability, and an answer of "UNSAT" would imply the unsatisfiability of the equational branch of $\varphi$, $\varphi_2$.

**Case manipulation functions** Each of the techniques discussed in Sec. 2.1 and Sec. 2.2 is embodied in one or more *case manipulation functions* (*CMF*s). A

[7] Observe that super-ideals correspond to sub-varieties, and thus increasing an ideal takes one closer to the empty variety, which is the geometric object corresponding to an unsatisfiable formula.

CMF operates on a single case, a conjunction of equalities and strict inequalities. A CMF first checks the structure of the case to see if it is of a form on which it can make progress. For example, the CMF for applying Full Open CAD first checks that all literals are strict inequalities. A CMF can have several outcomes.

- It can determine that the case is satisfiable. In this event, the initial goal is immediately also known to be satisfiable and RAHD terminates.
- It can determine that the case is unsatisfiable
- It can return the case it was applied to unchanged, if its initial structure check fails, for example.
- It can make progress and return a simplified case, consisting again of a conjunction of equalities and strict inequalities, that is equisatisfiable with the case the CMF is applied to.
- It can return some Boolean formula equisatisfiable with the case the CMF is applied to. In general this formula might contain disjunctions, non-strict inequalities and division operations.

**Sequencing proof steps** The set of cases in a goal-set can be considered the fringe of a partial proof tree of the initial goal. RAHD makes progress by applying CMFs to cases in the fringe. Our strategy for applying CMFs is simple but nevertheless effective.

We arrange our CMFs in a *master sequence* with the idea that we work on a case by applying each CMF from the master sequence in turn, so long as the CMFs return either identical cases or single simplified cases. This application of CMFs in sequence to a case extends the branch of the proof tree corresponding to that case. When the application of a CMF finds the case it is applied to unsatisfiable, the branch of the proof tree is closed or completed. In the event a general Boolean formula is output by a CMF, we apply the preprocessing step to generate new normalised cases to work on further. Usually there is more than one of these cases, so this introduces branching into the proof tree. When working on these new cases, we start again with the CMF at the beginning of the CMF sequence.

RAHD terminates either when some CMF finds the case it applied to satisfiable, in which case we have a counter-example to the original goal, or all proof branches are completed, in which case the original goal is proven.

We currently ensure that — at the level of CMFs — RAHD cannot diverge by requiring that, if a CMF outputs a more general boolean formula and restarts processing of the master sequence, it must reduce dimension: its output formula must contain fewer variables than the case it was applied to.

We refer sometimes to this master sequence of CMFs as the RAHD *waterfall* by analogy with the organisation of proof strategies in the NQTHM and ACL2 theorem provers.

**Ordering of case manipulation functions** We describe here some of the main elements of our ordering of CMFs in the master sequence. The first principle

of this ordering is that the more expensive CMFs should be postponed until later in the sequence, giving the less expensive CMFs priority in their attempts at closing and reducing cases. For instance, one should check if the univariate fragment of a case is unsatisfiable with the Sturm sequence analysis CMF before one tries any Open CAD CMF upon the case. The next principle is that given two CMFs of roughly equivalent expense, they should be ordered such that the result of one has the chance to inform and improve the result of the other, if possible. For example, the sums of squares based PD CMF, which will record the fact that a polynomial is positive definite if its canonicalization is a sum of squares of monomials plus a positive rational constant, should be run before the aforementioned Sturm sequence inequality CMF, as this has the potential to make more case inconsistencies explicit for the Sturm sequence CMF. The last principle we will mention is that a given CMF, $A$, should be included in the sequence in more than one place if the CMFs executed subsequent to $A$'s last execution have a good chance of making the cases that remain more amenable to $A$. In this way, the lightest interval analysis CMF appears in the beginning of the master sequence, then again after Gröbner basis reductions have taken place as this may make inequality inconsistencies more explicit, and then again after reductions have taken place through the real radical ideal approximation CMF for the same reason.

At a very high level, the master sequence goes as follows: light arith. simp. and interval analysis → sos/positivstellensatz search → sturm chain analysis → full Open CAD → ideal triviality checking → term canonicalization → GB based rewriting → light arith. simp. and interval analysis → sos/positivstellensatz search → fragmentary Open CAD (as described in Example 1) → real radical ideal approximation → light arith. simp. and interval analysis → reverse rabinoswitch encodings → general CAD.

## 3 Experimental Results

Table 1 shows RAHD's performance on twenty-four example problems[8] and compares this performance to that of QEPCAD-B and two quantifier elimination procedures available in Reduce/Redlog: Rlqe, which is an enhanced implementation by Dolzmann and Sturm of Weispfenning's virtual term substitution (VTS) [21] , and Rlcad, which is an implementation by Seidl, Dolzmann and Sturm of Collins-Hong's partial CAD [6]. One interesting feature of this Rlqe procedure is that it performs VTS as long as it can (e.g., as long as the degree restrictions required for the method are not violated in a way irreparable by Redlog's simplification and degree-reduction mechanisms), and then sends the resulting formula to the Rlcad procedure if VTS alone was not sufficient. This approach is referred to as *fallback quantifier elimination*. Experiments were performed on an Intel Xeon Quad Core 2.8GHz machine with 4GB physical memory.

[8] A copy of these problems may be obtained from `http://homepages.inf.ed.ac.uk/s0793114/calculemus09/`.

Table 2 presents a listing of seven of the twenty-four problems considered in Table 1.

The results of these experiments can be broadly summarized as follows:

- RAHD is able to solve a number of high-dimension, high-degree problems that QEPCAD-B, Redlog/Rlqe, and Redlog/Rlcad are not.
- Redlog/Rlqe is able to solve a number of high-dimension, high-degree problems that QEPCAD-B and Redlog/Rlcad are not.
- Redlog/Rlqe is able to solve a number of problems significantly faster than RAHD, Redlog/Rlcad, and QEPCAD-B.
- For the problems QEPCAD-B is able to solve, using QEPCAD-B directly tends to be much faster than using RAHD's waterfall.

## 4 Future Work

We see many ways RAHD can be improved. First, based upon the fact that QEPCAD-B outperforms the RAHD waterfall on many low-dimension, low-degree problems, we should develop heuristics that use structural features of a problem to evaluate *a priori* its suitability for a direct handling by QEPCAD-B, causing RAHD in those cases to bypass both its inequality splitting preprocessing and all other CMFs in the waterfall.

Second, as the Redlog/Rlqe procedure solves a number of problems much faster than all of the others, it seems fruitful to investigate heuristics for incorporating Redlog/Rlqe into the RAHD waterfall.

Finally, in terms of RAHD's inequality splitting and translation of resulting problems into a (DNF) goal-set, the potential exponential blow-up this causes will become prohibitive for problems with large boolean structure. There are many more sophisticated techniques we will need to employ if we wish for RAHD to be applicable to these types of problems. The infeasibility of normal form conversions has motivated huge algorithmic advances in the SAT and SMT communities [8], and it would be very interesting to build a new version of RAHD that uses DPLL-like [5] case-analysis mechanisms instead of an explicit DNF conversion.

## 5 Conclusion

In closing, we have shown that a thoughtfully orchestrated heterogeneous combination of decision methods for fragments of the existential theory of the reals can be made to solve problems previously beyond the reach of automatic methods. In particular, we have shown one way that ideal-theoretic computations and restricted variants of CAD for topologically open predicates can be fruitfully combined. It is interesting that while this combination involves an exponential blow-up in its reliance on a DNF normalisation, for many problems the increase in complexity caused by this blow-up is overshadowed by the decrease in complexity of the CAD computations this process induces.

**Table 1.** A comparison of RAHD, QEPCAD-B, Redlog/Rlqe and Redlog/Rlcad.

| | dim | deg | div | $\lvert\mathcal{GS}\rvert$ | $\lvert\mathcal{PT}\rvert$ | #p | #m | simp arith | simp GB | sos | strm | rad ideal | open CAD | gen CAD | RAHD | QB | RQ | RC |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| P0 | 5 | 4 | N | 1024 | 1024 | 42 | 55 | 1717 | 0 | 23 | 60 | 0 | 128 | 0 | 16.36 | 409* | 36.6 | - |
| P1 | 6 | 4 | N | 3072 | 3072 | 48 | 60 | 5371 | 3 | 99 | 156 | 0 | 378 | 0 | 74.09 | -* | - | - |
| P2 | 5 | 4 | N | 768 | 768 | 40 | 61 | 1419 | 0 | 0 | 96 | 0 | 99 | 0 | 10.54 | -* | - | - |
| P3 | 5 | 4 | N | 768 | 768 | 40 | 61 | 2187 | 0 | 0 | 96 | 0 | 99 | 0 | 10.87 | -* | - | - |
| P4 | 5 | 4 | N | 768 | 768 | 40 | 61 | 1448 | 0 | 8 | 88 | 0 | 99 | 0 | 9.84 | -* | - | - |
| P5 | 14 | 2 | N | 4 | 4 | 64 | 176 | 4 | 4 | 0 | 0 | 0 | 0 | 0 | .89 | -* | 427 | - |
| P6 | 11 | 5 | N | 8 | 14 | 24 | 31 | 124 | 20 | 6 | 0 | 6 | 0 | 2 | 28.23 | -* | <.01 | <.01 |
| P7 | 8 | 2 | N | 1 | 1 | 8 | 18 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | <.01 | .08 | <.01 | <.01 |
| P8 | 7 | 32 | N | 64 | 94 | 30 | 34 | 352 | 152 | 0 | 0 | 30 | 34 | 0 | 182.98 | 9.72 | <.01 | - |
| P9 | 7 | 16 | N | 128 | 158 | 34 | 38 | 672 | 264 | 0 | 0 | 30 | 66 | 0 | 26.9 | .29 | .01 | 18.5 |
| P10 | 7 | 12 | N | 32768 | 32795 | 66 | 165 | 78051 | 998 | 344 | 4 | 54 | 155 | 0 | 62.4 | -* | - | - |
| P11 | 6 | 2 | Y | 32 | 32 | 28 | 28 | 14 | 16 | 0 | 0 | 0 | 31 | 1 | 2.99 | .01 | .01 | .05 |
| P12 | 5 | 3 | N | 16 | 16 | 22 | 23 | 86 | 4 | 0 | 0 | 0 | 2 | 4 | .85 | .02 | .01 | .07 |
| P13 | 4 | 10 | N | 256 | 256 | 34 | 63 | 503 | 0 | 14 | 6 | 0 | 22 | 0 | 4.4 | -* | <.01 | <.01 |
| P14 | 2 | 2 | N | 256 | 259 | 32 | 54 | 889 | 57 | 0 | 30 | 7 | 5 | 10 | 2.45 | .01 | - | - |
| P15 | 4 | 3 | Y | 8 | 8 | 14 | 15 | 8 | 0 | 0 | 0 | 0 | 0 | 8 | 1.32 | .01 | .06 | .26 |
| P16 | 4 | 2 | N | 128 | 132 | 28 | 44 | 662 | 128 | 17 | 0 | 1 | 39 | 9 | 6.11 | .02 | <.01 | <.01 |
| P17 | 4 | 2 | N | 4 | 6 | 14 | 19 | 34 | 3 | 3 | 0 | 0 | 0 | 1 | .4 | .28 | .02 | .61 |
| P18 | 4 | 2 | N | 16 | 16 | 18 | 30 | 55 | 16 | 7 | 0 | 5 | 1 | 3 | 1.03 | .01 | .28 | - |
| P19 | 3 | 6 | Y | 256 | 256 | 30 | 310 | 1248 | 0 | 0 | 0 | 0 | 256 | 0 | 24.69 | .02 | .01 | .39 |
| P20 | 3 | 4 | N | 16 | 16 | 18 | 21 | 77 | 18 | 0 | 0 | 1 | 3 | 5 | 1.19 | .01 | <.01 | .23 |
| P21 | 3 | 2 | N | 64 | 64 | 26 | 31 | 179 | 0 | 0 | 12 | 0 | 7 | 0 | .6 | .02 | .04 | .27 |
| P22 | 2 | 4 | N | 2 | 2 | 8 | 10 | 3 | 1 | 0 | 0 | 0 | 1 | 0 | .09 | .01 | <.01 | .01 |
| P23 | 2 | 2 | Y | 8 | 8 | 12 | 12 | 16 | 0 | 0 | 0 | 0 | 0 | 0 | <.01 | .01 | <.01 | <.01 |

Explanation of columns:

*High-level problem features:* [**dim**] dimension, [**deg**] maximal total multivariate degree of polynomials, [**div**] problem contains division operator.

*Properties of* RAHD*'s internal translation of the problem:* [$\lvert\mathcal{GS}\rvert$] # of cases in the generated goal-set, [$\lvert\mathcal{PT}\rvert$] # of leaves in constructed proof tree, [#p] # of polynomials, [#m] # of monomials.

*Number of reduction or refutation steps made by* RAHD*CMFs:* [**simp arith**] light weight arithmetical simplifiers and interval analysis, [**simp GB**] Gröbner bases based rewriting and canonicalization, [**sos**] sums of squares / real nullstellensatz / positivstellensatz witness extraction, [**strm**] sturm chain sign change analysis, [**rad ideal**] dimensional reduction by radical ideal approximations, [**open CAD**] open CAD or open fragmentary CAD (using QEPCAD-B and $\exists_\infty$), [**gen CAD**] general CAD (using QEPCAD-B and $\exists$).

*Timing:* (in seconds) [**RAHD**] RAHD , [**QB**] QEPCAD-B, [**RQ**] Redlog/Rlqe (fallback QE), [**RC**] Redlog/Rlcad (p-CAD)

A mark of (-) in any of the timing columns means the system listed was unable to solve the problem in 600 seconds. A mark of (*) in the **QB** column means that QEPCAD-B's default resource settings were raised in order to avoid reaching resource limits. For problems involving division, the Redlog translation flag RLNZDEN was used both for Rlqe and Rlcad runs as well as for generating the multiplicative translations of the problems for QEPCAD-B.

**Table 2.** Seven of the twenty-four problems considered in Table 1.

**P0** $a^2+ab-2ac+a+21b^4-84b^3c+126b^2c^2-84bc^3+21c^4+c^2+d^2-2de+d+e^2<0 \ \wedge\ e-1\le 0 \ \wedge\ e\ge 0 \ \wedge\ d-1\le 0 \ \wedge\ d\ge 0 \ \wedge\ c-1\le 0 \ \wedge\ c\ge 0 \ \wedge\ b-1\le 0 \ \wedge\ b\ge 0 \ \wedge\ a-1\le 0 \ \wedge\ a\ge 0$

**P1** $a^2b+a^2-2ac^2+3b^2-6bc+c^4+3c^2+d^2-2de+d+e^2+f+1<0 \ \wedge\ (f-2=0 \ \vee\ f-1=0 \ \vee\ f=0) \ \wedge\ e-1\le 0 \ \wedge\ e\ge 0 \ \wedge\ d-1\le 0 \ \wedge\ d\ge 0 \ \wedge\ c-1\le 0 \ \wedge\ c\ge 0 \ \wedge\ b-1\le 0 \ \wedge\ b\ge 0 \ \wedge\ a-1\le 0 \ \wedge\ a\ge 0$

**P5** $(y_6\neq 0 \ \vee\ x_6\neq 0) \ \wedge\ x_6^2-2x_6x_7+x_7^2+y_6^2-2y_6y_7+y_7^2-4>0 \ \wedge\ x_5^2-2x_5x_7+x_7^2+y_5^2-2y_5y_7+y_7^2-4=0 \ \wedge\ x_5^2-2x_5x_6+x_6^2+y_5^2-2y_5y_6+y_6^2-4=0 \ \wedge\ x_4^2-2x_4x_7+x_7^2+y_4^2-2y_4y_7+y_7^2-4=0 \ \wedge\ x_4^2-2x_4x_6+x_6^2+y_4^2-2y_4y_6+y_6^2-4=0 \ \wedge\ x_4^2-2x_4x_5+x_5^2+y_4^2-2y_4y_5+y_5^2-4=0 \ \wedge\ x_3^2-2x_3x_7+x_7^2+y_3^2-2y_3y_7+y_7^2-4=0 \ \wedge\ x_3^2-2x_3x_6+x_6^2+y_3^2-2y_3y_6+y_6^2-4=0 \ \wedge\ x_3^2-2x_3x_5+x_5^2+y_3^2-2y_3y_5+y_5^2-4=0 \ \wedge\ x_3^2-2x_3x_4+x_4^2+y_3^2-2y_3y_4+y_4^2-4=0 \ \wedge\ x_2^2-2x_2x_7+x_7^2+y_2^2-2y_2y_7+y_7^2-4=0 \ \wedge\ x_2^2-2x_2x_6+x_6^2+y_2^2-2y_2y_6+y_6^2-4=0 \ \wedge\ x_2^2-2x_2x_5+x_5^2+y_2^2-2y_2y_5+y_5^2-4=0 \ \wedge\ x_2^2-2x_2x_4+x_4^2+y_2^2-2y_2y_4+y_4^2-4=0 \ \wedge\ x_2^2-2x_2x_3+x_3^2+y_2^2-2y_2y_3+y_3^2-4=0 \ \wedge\ x_1^2-2x_1x_7+x_7^2+y_1^2-2y_1y_7+y_7^2-4=0 \ \wedge\ x_1^2-2x_1x_6+x_6^2+y_1^2-2y_1y_6+y_6^2-4=0 \ \wedge\ x_1^2-2x_1x_5+x_5^2+y_1^2-2y_1y_5+y_5^2-4=0 \ \wedge\ x_1^2-2x_1x_4+x_4^2+y_1^2-2y_1y_4+y_4^2-4=0 \ \wedge\ x_1^2-2x_1x_3+x_3^2+y_1^2-2y_1y_3+y_3^2-4=0 \ \wedge\ x_1^2-2x_1x_2+x_2^2+y_1^2-2y_1y_2+y_2^2-4=0 \ \wedge\ x_7^2+y_7^2-4=0 \ \wedge\ x_6^2+y_6^2-4=0 \ \wedge\ x_5^2+y_5^2-4=0 \ \wedge\ x_4^2+y_4^2-4=0 \ \wedge\ x_3^2+y_3^2-4=0 \ \wedge\ x_2^2+y_2^2-4=0 \ \wedge\ x_1^2+y_1^2-4=0$

**P6** $45dxy-g+45xy=0 \ \wedge\ g-g_1-g_2-82>0 \ \wedge\ w+1<0 \ \wedge\ -x+y\ge 0 \ \wedge\ x-1\ge 0 \ \wedge\ a=0 \ \wedge\ -a+wz=0 \ \wedge\ x^3y^2-z=0 \ \wedge\ -3g_1^2g_2+12g_1x_3x_7-xy-11x\ge 0$

**P10** $(-a+fg+11f+g^2+13g+22=0 \ \vee\ a^4b^2cd^3f^2+2a^4b^2cd^3fg+a^4b^2cd^3g^2-a^4b^2cdf^2-2a^4b^2cdfg-a^4b^2cdg^2-a^4b^2d^2f^2-2a^4b^2d^2fg-a^4b^2d^2g^2+a^4b^2f^2+2a^4b^2fg+a^4b^2g^2-2a^3b^3c^2d^2f^2-4a^3b^3c^2d^2fg-2a^3b^3c^2d^2g^2+2a^3b^3c^2f^2+4a^3b^3c^2fg+2a^3b^3c^2g^2+2a^3b^3d^2f^2+4a^3b^3d^2fg+2a^3b^3d^2g^2-2a^3b^3f^2-4a^3b^3fg-2a^3b^3g^2+a^2b^4c^3df^2+2a^2b^4c^3dfg+a^2b^4c^3dg^2-a^2b^4c^2f^2-2a^2b^4c^2fg-a^2b^4c^2g^2-a^2b^4cdf^2-2a^2b^4cdfg-a^2b^4cdg^2+a^2b^4f^2+2a^2b^4fg+a^2b^4g^2-2a^2b^2c^3df^2-4a^2b^2c^3dfg-2a^2b^2c^3dg^2+4a^2b^2c^2d^2f^2+8a^2b^2c^2d^2fg+4a^2b^2c^2d^2g^2-2a^2b^2c^2f^2-4a^2b^2c^2fg-2a^2b^2c^2g^2-2a^2b^2cd^3f^2-4a^2b^2cd^3fg-2a^2b^2cd^3g^2+4a^2b^2cdf^2+8a^2b^2cdfg+4a^2b^2cdg^2-2a^2b^2d^2f^2-4a^2b^2d^2fg-2a^2b^2d^2g^2+a^2c^3d^3f^2+2a^2c^3d^3fg+a^2c^3d^3g^2-a^2c^2d^2f^2-2a^2c^2d^2fg-a^2c^2d^2g^2-a^2cd^3f^2-2a^2cd^3fg-a^2cd^3g^2+a^2d^2f^2+2a^2d^2fg+a^2d^2g^2-2abc^3d^3f^2-4abc^3d^3fg-2abc^3d^3g^2+2abc^3df^2+4abc^3dfg+2abc^3dg^2+2abcd^3f^2+4abcd^3fg+2abcd^3g^2-2abcdf^2-4abcdfg-2abcdg^2+b^2c^3d^3f^2+2b^2c^3d^3fg+b^2c^3d^3g^2-b^2c^3df^2-2b^2c^3dfg-b^2c^3dg^2-b^2c^2d^2f^2-2b^2c^2d^2fg-b^2c^2d^2g^2+b^2c^2f^2+2b^2c^2fg+b^2c^2g^2<0) \wedge f^2g+2g^5-g=0 \ \wedge\ f^4-1=0 \ \wedge\ e^3f^3+g-2=0 \ \wedge\ g-1\le 0 \ \wedge\ f-1\le 0 \ \wedge\ e-1\le 0 \ \wedge\ d-1\le 0 \ \wedge\ c-1\le 0 \ \wedge\ b-1\le 0 \ \wedge\ a-1\le 0 \ \wedge\ g\ge 0 \wedge f\ge 0 \ \wedge\ e\ge 0 \ \wedge\ d\ge 0 \ \wedge\ c\ge 0 \ \wedge\ b\ge 0 \ \wedge\ a\ge 0$

**P16** $c^2+cd-d^2+1\le 0 \ \wedge\ 2a+b-1\ge 0 \ \wedge\ a^2+ab-b^2-1\ge 0 \ \wedge\ d-1\ge 0 \wedge c\ge 0 \ \wedge\ b\ge 0 \ \wedge\ ad+bc+bd\le 0$

**P19** $x\neq 1 \ \wedge\ y\neq 1 \ \wedge\ z\neq 1 \ \wedge\ x^2/(x-1)^2+y^2/(y-1)^2+z^2/(z-1)^2<1 \ \wedge\ xyz=1$

## References


1. Christopher W. Brown. QEPCAD B: a program for computing with semi-algebraic sets using CADs. *SIGSAM Bull.*, 37(4):97–108, 2003.
2. P. J. Cohen. Decision procedures for real and p-adic fields. *Comm. Pure and Applied Mathematics*, XXII(2):131–151, March 1969.
3. G. E. Collins. Quantifier elimination for the elementary theory of real closed fields by cylindrical algebraic decomposition. *LNCS*, 33:134–183, 1975.
4. J. H. Davenport and J. Heintz. Real quantifier elimination is doubly exponential. *Journal of Symbolic Computing*, 5(1–2):29–35, Feb 1988.
5. Martin Davis and Hilary Putnam. A computing procedure for quantification theory. *J. ACM*, 7(3):201–215, 1960.
6. Andreas Dolzmann, Andreas Seidl, and Thomas Sturm. Efficient projection orders for CAD. In *ISSAC '04: Proceedings of the 2004 international symposium on Symbolic and algebraic computation*, pages 111–118. ACM, 2004.
7. Andreas Dolzmann and Thomas Sturm. Redlog: computer algebra meets computer logic. *SIGSAM Bull.*, 31(2):2–9, 1997.
8. B. Dutertre and L. de Moura. The YICES SMT solver. `http://yices.csl.sri.com/tool-paper.pdf`, 2006.
9. D. Yu Grigor'ev and N. N. Vorobjov, Jr. Solving systems of polynomial inequalities in subexponential time. *Journal of Symbolic Computation*, 5(1,2):37–64, 1988.
10. Thomas C. Hales. Formalizing the proof of the Kepler Conjecture. In *Theorem Proving in Higher Order Logics (TPHOLs)*, 2004.
11. John Harrison. Verifying nonlinear real formulas via sums of squares. In *Theorem Proving in Higher Order Logics (TPHOLs)*, volume 4732 of *LNCS*, pages 102–118, 2007.
12. Hoon Hong. Comparison of several decision algorithms for the existential theory of the reals. Technical report, RISC Linz, 1991.
13. Santiago Laplagne. An algorithm for the computation of the radical of an ideal. In *International Symposium on Symbolic and Algebraic Computation*, 2006.
14. Scott McCallum. Solving polynomial strict inequalities using cylindrical algebraic decomposition. *The Computer Journal*, 36(5), 1993.
15. Sean McLaughlin and John Harrison. A proof-producing decision procedure for real arithmetic. In Robert Nieuwenhuis, editor, *CADE-20: 20th International Conference on Automated Deduction, proceedings*, volume 3632 of *Lecture Notes in Computer Science*, pages 295–314, Tallinn, Estonia, 2005. Springer-Verlag.
16. J. Renegar. On the computational complexity and geometry of the first-order theory of the reals (Part I). Technical Report 853, Cornell University, 1989.
17. Alfred Tarski. A decision method for elementary algebra and geometry. Technical report, Rand Corporation, 1948.
18. Ashish Tiwari. HybridSAL: Modeling and abstracting hybrid systems. Technical report, SRI International, 2003.
19. Ashish Tiwari. An algebraic approach for the unsatisfiability of nonlinear constraints. In *Computer Science Logic (CSL)*, volume 3634 of *LNCS*, pages 248–262, 2005.
20. Lou van den Dries. *Tame topology and o-minimal structures*. London Mathematical Society, 1998.
21. Volker Weispfenning. Quantifier elimination for real algebra – the quadratic case and beyond. *Applicable Algebra in Engineering Communication and Computing*, 8(2):85–101, February 1997.