New Results on Quantifier Elimination Over Real Closed Fields and Applications to Constraint Databases

Saugata Basu *
Department of Mathematics
University of Michigan
Ann Arbor, Mi 48109
saugata@math.lsa.umich.edu

Abstract

In this paper we give a new algorithm for quantifier elimination in the first order theory of real closed fields that improves the complexity of the best known algorithm for this problem till now. Unlike previously known algorithms [3, 28, 22] the combinatorial part of the complexity (the part depending on the number of polynomials in the input) of this new algorithm is independent of the number of free variables. Moreover, under the assumption that each polynomial in the input depend only on a constant number of the free variables, the algebraic part of the complexity (the part depending on the degrees of the input polynomials) can also be made independent of the number of free variables. This new feature of our algorithm allows us to obtain a new algorithm for a variant of the quantifier elimination problem. We give an almost optimal algorithm for this new problem, which we call the uniform quantifier elimination problem.

Using the uniform quantifier elimination algorithm, we give an algorithm for solving a problem arising in the field of constraint databases with real polynomial constraints. We give an algorithm for converting a query with natural domain semantics to an equivalent one with active domain semantics. A non-constructive version of this result was proved in [6]. Very

*This work was done while the author was a post-doctoral fellow at the Mathematical Sciences Research Institute, Berkeley. A preliminary version of this paper appeared in the Proceedings of the 38th IEEE Symposium on the Foundations of Computer Science, 1997.

recently, a constructive proof was also given independently in [8]. However, complexity issues were not considered and no algorithm with a reasonable complexity bound was known for this latter problem till now.

We also point out interesting logical consequences of this algorithmic result, concerning the expressive power of a constraint query language over the reals. This leads to simpler and constructive proofs for these inexpressibility results than the ones known before.

Moreover, our improved algorithm for performing quantifier elimination immediately leads to improved algorithms for several problems for which quantifier elimination is a basic step, for example, the problem of computing the closure of a given semi-algebraic set.

Contents

1 Introduction

1.1 The Quantifier Elimination Problem

We are given a set, $\mathcal{P} = \{P_1, \ldots, P_s\}$, of s polynomials in $k + \ell$ variables, $X_1, \ldots, X_k, Y_1, \ldots, Y_\ell$. The degrees of the polynomials are bounded by d and their coefficients lie in a real closed field R.

We are also given a first-order formula of the form

$$(Q_\omega X^{[\omega]}) \ldots (Q_1 X^{[1]}) F(P_1, \ldots, P_s)$$

(henceforth denoted $\Phi(Y)$) where $Q_i \in \{\forall, \exists\}$, $Q_i \neq Q_{i+1}$, $Y = (Y_1, \ldots, Y_\ell)$ is a block of ℓ free variables, $X^{[i]}$ is a block of k_i variables with $\sum_{1 \leq i \leq \omega} k_i = k$, and $F(P_1, \ldots, P_s)$ is a quantifier-free Boolean formula with atomic predicates of the form $\mathrm{sign}(P_i(Y, X^{[\omega]}, \ldots, X^{[1]})) = \sigma$ where $\sigma \in \{0, 1, -1\}$,

The quantifier elimination problem is to construct a quantifier-free Boolean formula, $\Psi(Y)$, such that for any $y \in R^\ell$, $\Phi(y)$ is true if and only if $\Psi(y)$ is true.

We denote by D the smallest subring of R containing the coefficients of the input polynomials. All the computations of our algorithms take place in D. We define the complexity of our algorithms to be the number of arithmetic operations (additions, multiplications and sign determinations) in the ring D. Note that this ignores the cost of reading the input or writing the output formula (see [3], section 1.3, page 1004, for further details).

We next describe a variant of the quantifier elimination problem which was introduced in [2] motivated by a problem in constraint databases.

1.2 The Uniform Quantifier Elimination Problem

We call a sequence,

$$\{\phi_n(T_1, \ldots, T_l, Y_1, \ldots, Y_n) \mid n > 0\}$$

of first-order formulas ϕ_n in the language of ordered fields, to be a *uniform sequence* if each ϕ_n has the form,

$$\phi_n(T_1, \ldots, T_l, Y_1, \ldots, Y_n) =$$

$$Q^1_{1\leq k_1\leq n}\dots Q^\omega_{1\leq k_\omega\leq n}\phi(T_1,\dots,T_l,Y_{k_1},\dots,Y_{k_\omega}),$$

where $Q^i\in\{\bigvee,\bigwedge\}, 1\leq i\leq\omega$ and ϕ is some fixed $(l+\omega)$-ary quantifier-free first-order formula.

Thus for every n, ϕ_n is a first order formula with $l+n$ free variables. We will refer to the variables $T_1,\dots,T_l$ as *parameters*.

Given a uniform sequence of formulas, $\Phi=\{\phi_n\mid n>0\}$, where

$$\phi_n(T_1,\dots,T_l,Y_1,\dots,Y_n)=$$

$$Q^1_{1\leq k_1\leq n}\dots Q^\omega_{1\leq k_\omega\leq n}\phi(T_1,\dots,T_l,Y_{k_1},\dots,Y_{k_\omega}),$$

we define the *size* of Φ henceforth denoted $||\Phi||=|\phi|$ where $|\phi|$ is the length of the formula ϕ.

Before proceeding further we give an example.

Example:

Consider the uniform sequence of formulas,

$$\phi_n(T_1,Y_1,\dots,Y_n)=\bigwedge_{1\leq k_1\leq n}(Y_{k_1}-T_1=0),\quad n>0.$$

Now consider the sequence of quantified formulas, $(\exists T_1)\phi_n(T_1,Y_1,\dots,Y_n)$. From the fact that there exists quantifier elimination for the first order theory of real closed fields, it is clear that for every $n>0$, there exists a quantifier free formula equivalent to $(\exists T_1)\phi_n(T_1,Y_1,\dots,Y_n)$. However, it is not clear whether there exists a uniform sequence of quantifier-free formulas, $\{\Psi_n(Y_1,\dots,Y_n)\mid n>0\}$ such that for every $n>0$, $\Psi_n(Y_1,\dots,Y_n)\Leftrightarrow(\exists T_1)\phi_n(T_1,Y_1,\dots,Y_n)$. In this example, it is easily seen that letting

$$\Psi_n=\bigwedge_{1\leq k_1\leq n}\bigwedge_{1\leq k_2\leq n}(Y_{k_1}-Y_{k_2}=0),$$

we get a uniform sequence of quantifier-free formulas satisfying,

$$\Psi_n(Y_1,\dots,Y_n)\Leftrightarrow(\exists T_1)\phi_n(T_1,Y_1,\dots,Y_n)$$

for every $n>0$.

Note that, by definition the formulas of a uniform sequence do not have quantifiers. However, in practice we will often quantify over the parameters of a uniform sequence, and we will be interested in eliminating these quantifiers. The uniform quantifier elimination problem is to eliminate parameters (which are assumed to be quantified) from a uniform sequence of formulas and obtain another uniform sequence of quantifier free formulas.

1.3 History

The existence of an algorithm for quantifier elimination was first proved by Tarski [31] (see also [29]). However, the complexity of his algorithm is not elementary recursive. The first algorithm with a an elementary-recursive worst-case time bound was given by Collins [11]. His algorithm had a worst case running time doubly exponential in the number of variables.

Heintz, Roy and Solerno [22] and Renegar [28] gave quantifier elimination algorithms which were doubly exponential only in the number of quantifier alternations. See also [33] for the lower bound proof.

The best algorithm for this problem till now appeared in [3], which has a complexity of

$$s^{(\ell+1)\Pi(k_i+1)} d^{(\ell+1)\Pi O(k_i)}.$$

Moreover, the degrees of the polynomials appearing in the output formula are bounded by $d^{\Pi_i O(k_i)}$. The size of the output formula is bounded by $s^{(\ell+1)\Pi(k_i+1)} d^{(\ell+1)\Pi O(k_i)}$.

To our knowledge the uniform quantifier elimination problem was not considered by any previous researcher till now. We note that the previously known quantifier elimination algorithms are inadequate for solving this problem because their complexity depend on the total number of free variables which is unbounded in this case.

Even though the uniform quantifier elimination problem was motivated by its applications in constraint databases, we believe that it is an important generalization of the classical quantifier elimination problem over the reals and will be useful in other applications.

Note: It is an interesting problem to investigate whether other well-known theories which admit quantifier elimination, such as the theory of algebraically closed fields of a fixed characteristic, the theory of p-adic fields or the theory of differentially closed fields of a fixed caracteristic etc., also admit uniform quantifier elimination.

1.4 Semi-algebraic and Semi-linear geometry

Sets defined by a Boolean formula involving a finite number of polynomial equalities and inequalities are called *semi-algebraic sets.* If all the polynomials involved in the definition are linear then the set is called *semi-linear.* Geometrically, quantifier elimination can be thought of as the projection operator on semi-algebraic sets.

Example: Let $S \subset R^k$ be defined by, $S = \{(x_1, \ldots, x_k) \in R^k | \Phi(x_1, \ldots, x_k)\}$ where Φ is a quantifier-free Boolean formula whose atoms are of the form $P_i\{>, <, =\}0$, $1 \leq i \leq s$.

Then eliminating the quantifier from $(\exists X_k)\Phi$, is the same as computing a quantifier-free description of the projection of the set S onto the space spanned by the first $k-1$ co-ordinates. $\square$

Notice if S is a *semi-linear* set then the problem of eliminating one quantifier corresponds to the problem of projecting a given semi-linear set to one lower dimension. The classical algorithm for projecting semi-linear sets is the Fourier-Motzkin algorithm (see [34] page 35) which requires s^2 arithmetic operations. However, if we use the general quantifier elimination algorithm in [3] in this special case, the complexity is easily seen to be $O(s^{2k})$. Thus, the dependence on s is very far from optimal, and this is due to the fact that the exponent of s in the complexity depends on the number of free variables, ℓ.

One way of looking at computational problems in semi-algebraic geometry is by considering them to be generalizations of the corresponding problems for semi-linear sets where the polynomials involved are all linear. Semi-linear sets are the basic objects in computational geometry, where many problems involve arrangements of hyperplanes. The important parameter for measuring the size of the input as well as the complexity of these algorithms is the number of linear constraints s (this is usually called n in the computational geometry literature). Recently however, there have been efforts to generalize computational geometry algorithms to problems where the constraints are low degree polynomials, not just linear ones (see, for example, [1, 30, 24, 25, 5] and also the survey by Chazelle [10]).

From this point of view it makes sense to separate out the roles of the parameters s and d in the complexity analysis of algorithms in semi-algebraic geometry. We call the part that depends on s the *combinatorial part* and the part that depends on d the *algebraic part*. The combinatorial complexity of the best known algorithm for performing quantifier elimination till now is $s^{(\ell+1)\Pi(k_i+1)}$.

Our aim is to design algorithms whose combinatorial complexity matches the best known complexity for the corresponding problem over semi-linear sets. Of course, one should also try to minimize the algebraic part. However, in applications in computational geometry, the degrees of the polynomials as well as the number of variables are often assumed to be small and thus the combinatorial part becomes the most important part of the complexity (the algebraic part being bounded by a constant).

In the above example it is clear that the combinatorial complexity of the quantifier elimination algorithm does not match the corresponding bound for the semi-linear case. In this paper we give an improved algorithm for doing quantifier elimination whose combinatorial complexity matches the complexity of the corresponding problem with linear polynomials. More precisely, the combinatorial complexity of the new algorithm is $s^{\Pi(k_i+1)}$. There is no dependence on the number of free variables, ℓ.

The rest of the paper is organized as follows. In the next section we state our main results. In subsection 2.1 we state our new result on quantifier elimination. In subsection 2.2 we give an application of our technique to give an efficient algorithm for the *uniform quantifier elimination* problem. In subsection 2.3 we describe an application of the uniform quantifier elimination algorithm to constraint databases. In section 3 we recall a few subroutines and facts from real algebraic geometry used in our algorithms with appropriate pointers. In sections 4 and 5 we give the proofs of the main theorems and finally in section 6 we discuss some inexpressibility results.

2 Statement of Our Results

2.1 Quantifier Elimination

We prove the following theorem.

Theorem 1 *(Quantifier Elimination) Let $\mathcal{P} = \{P_1, \ldots, P_s\}$, be a set of s polynomials each of degree at most d, in $k + \ell$ variables, with coefficients in a real closed field R and*

$$\Phi(Y) = (Q_\omega X^{[\omega]}) \ldots (Q_1 X^{[1]}) F(P_1, \ldots, P_s),$$

a first-order formula, where $Q_i \in \{\forall, \exists\}$, $Q_i \neq Q_{i+1}$, $Y = (Y_1, \ldots, Y_\ell)$ is a block of ℓ free variables, $X^{[i]}$ is a block of k_i variables, $\sum_{1 \leq i \leq \omega} k_i = k$, and $F(P_1, \ldots, P_s)$ is a quantifier-free Boolean formula with atomic predicates of the form $P_i(Y, X^{[\omega]}, \ldots, X^{[1]})$ $\{<, >, =\}$ 0. Moreover, let every polynomial in $\mathcal{P}$ depend on at most τ of the Y_j's.

Then, there exists an equivalent quantifier-free formula, $\Psi(Y)$ of size $s^{\Pi(k_i+1)} d^{\ell' \Pi O(k_i)} |F|$, where $\ell' = \min(\ell, \tau \Pi_i (k_i + 1))$, and $|F|$ is the length of the formula F. The degrees of the polynomials appearing in $\Psi(Y)$ are bounded by $d^{\Pi_i O(k_i)}$.

Moreover, we present an algorithm to compute $\Psi(Y)$ using $s^{\Pi(k_i+1)} d^{\ell' \Pi O(k_i)}$ arithmetic operations in $\mathcal{D}$.

Note the improvement in the combinatorial complexity, which no longer depends on the number of free variables, ℓ.

Also, if each polynomial in $\mathcal{P}$ depends on only a constant number of the free variables, then the algebraic part of the complexity is also independent of ℓ. We exploit this feature of our algorithm for solving the *uniform quantifier elimination* problem (see section 2.2).

Finally, note that the size of the output now depends on the size of the input formula F. Thus, if the input formula is sparse (say consists of a single sign condition on the family $\mathcal{P}$) the output formula will reflect this and will have smaller size. This is not the case in the previous algorithms for quantifier elimination, where the bound on the size of the output formula does not depend on the size of the input. Of course, the caveat is that if the input itself is very large, the output formula by the new algorithm will be large too, while in the case of the algorithms in [3, 28] the size of the output is bounded independent of the size of the input formula. Moreover, unlike the algorithms in [3, 28] the output of our algorithm is not guaranteed to be a disjunction of *realizable* sign conditions on a family of polynomials.

The improvement in the combinatorial complexity immediately gives improved algorithms for certain problems where quantifier elimination is the basic step.

Example: Consider the problem of computing the closure of a semi-algebraic set, $S = \{x \in R^k \mid \Phi(x)\}$ where $\Phi(X)$ is a quantifier-free first order formula involving s polynomials in k variables with degrees bounded by d.

(Note that, one does not always obtain the closure of a semi-algebraic set by merely replacing all strict inequalities used in defining the set by weak ones. For example, consider the set defined by $X^2(X-1) > 0$. The closure of this set is defined by $X \geq 1$, which is not equal to the set defined by $X^2(X-1) \geq 0$.)

The closure of the set S can be expressed as, $\bar{S} = \{x \in R^k \mid \bar{\Phi}(x)\}$, where $\bar{\Phi}(Y) = (\forall \epsilon)(\exists X)(|X - Y|^2 < \epsilon^2) \wedge \Phi(X)$. Thus, in order to obtain a semi-algebraic description of the closure we have to eliminate two blocks of quantifiers of length 1 and k respectively.

Using the algorithm in [3] the complexity is, $s^{2(k+1)(k+1)} d^{O(k^2)}$.

Moreover, even if the original set is defined by a single sign condition, (say) $P_1 \geq 0, \ldots, P_s \geq 0$, the quantifier-free formula is of size $s^{2(k+1)(k+1)} d^{O(k^2)}$.

However, using the new algorithm the complexity is $s^{2(k+1)}d^{O(k^2)}$. Moreover, if the original set is given by a single sign condition the length of the quantifier free formula output is $s^{2(k+1)}d^{O(k^2)}$.

$\square$

2.2 Uniform Quantifier Elimination

Using our new technique it is possible to eliminate quantifiers from a uniform sequence of formulas and obtain another uniform sequence of quantifier free formulas. We have the following theorem.

Theorem 2 *(Uniform Quantifier Elimination) Let,*

$$\Phi = \{\phi_n(T_1, \ldots, T_l, Y_1, \ldots, Y_n) \mid n > 0\}$$

be a uniform sequence of formulas with parameters $T_1, \ldots, T_l$, where

$$\phi_n(T_1, \ldots, T_l, Y_1, \ldots, Y_n) =$$

$$Q^1_{1 \le k_1 \le n} \cdots Q^{\omega}_{1 \le k_\omega \le n} \phi(T_1, \ldots, T_l, Y_{k_1}, \ldots, Y_{k_\omega}).$$

Let the number of different $(l+\omega)$-variate polynomials appearing in ϕ be s and let their degrees be bounded by d.

Let $R_1, \ldots, R_m \in \{\exists, \forall\}$, $R_i \neq R_{i+1}$, and let $T^{[1]}, \ldots, T^{[m]}$ be a partition of the variables, $T_1, \ldots, T_l$ into m blocks of size $l_1, \ldots, l_m$, where $\sum_{1 \le i \le m} l_i = l$.

Then, there exists an algorithm that outputs a quantifier-free first order formula, $\psi(Y_{k_1}, \ldots, Y_{k_{\omega'}})$, along with $Q^i \in \{\bigvee, \bigwedge\}, 1 \le i \le \omega'$, such that for every $n > 0$,

$$\psi_n(Y_1, \ldots, Y_n) = Q^1_{1 \le k_1 \le n} \cdots Q^{\omega'}_{1 \le k_{\omega'} \le n} \psi(Y_{k_1}, \ldots, Y_{k_{\omega'}})$$

$$\Leftrightarrow (R_1 T^{[1]}) \ldots (R_m T^{[m]}) \phi_n(T_1, \ldots, T_l, Y_1, \ldots, Y_n).$$

The complexity of the algorithm is $s^{\Pi_i(l_i+1)} d^{\omega \Pi_i O(l_i^2)}$, and the size of the formula ψ is $s^{\Pi_i(l_i+1)} d^{\omega \Pi_i O(l_i^2)} |\phi|$.

2.3 Application to Constraint Databases

We now describe the application of *uniform quantifier elimination* to the theory of *constraint databases.*

In recent years, constraint databases, first introduced by Kanellakis, Kuper and Revesz [27] has attracted a lot of attention. Unlike traditional databases which are finite collections of data items, constraint databases permit infinite collections of items to be stored in the database. They are a powerful generalization of Codd's relational model [32]. Different types of constraints have been considered by various authors [27, 26, 17] including dense linear order inequalities, real polynomial inequalities etc.

Even though constraint databases permit infinite number of elements in the database, for the purposes of this paper it is enough to consider our database to be a *finite* set S of real numbers.

We first introduce some notation. Let $L = \langle 0, 1, +, \times, \leq \rangle$ be the language of ordered fields. We extend L to $L' = \langle 0, 1, +, \times, \leq, R \rangle$ by including a unary relation symbol R where $R(x) \Leftrightarrow x \in S$. A *query* is a first order L'-sentence. For instance the query,

$$\mathcal{Q} = (\forall T_1)(\forall T_2)((T_1 \neq T_2) \wedge R(T_1) \wedge R(T_2))$$
$$\Rightarrow (T_1 - T_2)^2 \geq 4,$$

expresses the fact that any two distinct members of S differ by at least 2.

Thus, $\mathcal{Q}$ will be true when $S = \{1, 5, 11\}$ but false for $S = \{1, 2\}$.

Let $\langle \Re, S \rangle$ denote the L'-structure consisting of the reals (with the usual identification of $+$ with addition, $\times$ with multiplication and so on) and a subset $S \subset \Re$ representing the unary relation R. We call two queries $\mathcal{Q}_1, \mathcal{Q}_2$ *equivalent* iff for every finite subset S of real numbers, $\langle \Re, S \rangle \models \mathcal{Q}_1 \Leftrightarrow \langle \Re, S \rangle \models \mathcal{Q}_2$.

Note that the quantified variables in the query vary over all the reals. This interpretation of a query is called *natural domain semantics.* Unless otherwise stated all queries in this paper will be interpreted in this manner. Another interpretation, called *active domain semantics*, interpret the quantified variables to vary only over the set S. Clearly, a query with active domain semantics is equivalent to another one with natural domain semantics. However, the converse is not quite obvious. Given a query with natural domain semantics, such as in the example above, does there exist a query with active domain semantics that is equivalent to it ? Actually, in the example above $\mathcal{Q}$ with active domain semantics is equivalent to itself with natural domain semantics. However, this is not always the case. Consider the query, $\mathcal{Q} = (\exists T_1)\neg R(T_1)$. Clearly, the natural and active domain semantics give different interpretation to this query. The query is always true with natural domain semantics but clearly always false with active domain semantics.

We apply our algorithmic result on uniform quantifier elimination to give an algorithm that given a query with natural domain semantics, outputs another query equivalent to it with active domain semantics.

The equivalence of natural and active domain semantics over real closed fields was proved in [6], [7]. However, their proofs are essentially non-constructive and one does not obtain an effective algorithm for converting a query with the natural domain semantics into an equivalent one with active domain semantics. Very recently, in [8] an algorithm was given for this problem, but complexity issues were not considered.

The advantage of having a query with active domain semantics over a finite database is clear. In order to evaluate such a query it suffices to go over the elements of the finite database as all the quantified variables vary only over the database. Thus, we have a uniform algorithm to evaluate such a query independent of the set S. This will allow, for example, compile time query optimization that is possible in a classical database setting [7]. We have the following theorem which is an easy consequence of theorem 2.

Theorem 3 *Let*

$$\sigma = (Q_1 X^{[1]}) \ldots (Q_\omega X^{[\omega]})\psi(X^{[1]} \ldots, X^{[\omega]}),$$

be a query with natural domain semantics, where each $Q_i \in \{\exists, \forall\}$, and ψ is an L'-formula.

Let the total number of polynomials appearing in σ be s and their degrees be bounded by d. Moreover, let the number of blocks of quantifiers be ω with the i^{th} block of length k_i, with $\sum_i k_i = k$. Then, there exists an algorithm for converting σ to an equivalent query with active domain semantics whose complexity is $s^{\Pi_i (k_i+1)} d^{\Pi_i O(k_i^2)}$.

The equivalence of the natural and active domain semantics has interesting consequences regarding the expressive power of constraint queries over the reals. The problem of determining the expressive power of constraint queries has spurred a lot of research leading to several interesting inexpressibility results for constraint queries over various structures [18, 17, 6]. The *parity* of the cardinality of a set and the *connectivity* of a finite graph are well-known examples of queries that are not definable over finite models in a first order language with equality. Their undefinability has been shown using different proof techniques such as locality [16], 0/1 laws [14, 15], Ehrenfeucht-Fraisse games [13]. These undefinability results also hold in the presence of an order relation [21].

Grumbach and Su [17] proved that parity is not definable over constraint databases with linear constraints. They also conjectured that parity is not definable over the reals with queries allowed to have polynomial equalities and inequalities.

Let us pose this question more concretely.

Question: Does there exist a query (that is a first order L'-sentence) σ such that for every finite set $S \subset \Re$ with $|S|$ even, $\langle \Re, S \rangle \models \sigma$ and for every finite set $S \subset \Re$ with $|S|$ odd, $\langle \Re, S \rangle \models \neg\sigma$?

This was answered in the negative by Benedikt et al. [6] (see also [7, 8] who used difficult techniques from non-standard analysis and model theory of ordered structures. Using the equivalence of natural domain and active domain semantics we give a simpler and elementary proof of the inexpressibility of parity.

In fact it turns out that the properties of finite sets that are not definable over $\langle \omega, \leq \rangle$, that is over natural numbers with order but no arithmetic, are also not definable over the reals. Thus having arithmetic adds no extra expressive power over having only inequalities. We note that this result is already implied in the work of Benedikt et al. [7, 8] but our proof is constructive and simpler.

3 Algorithmic Preliminaries

3.1 Thom Encodings and Univariate Representations

We give a few definitions, recall Thom's theorem, and describe the inputs, outputs and complexities of some subroutines that we will use in our algorithm. We refer the reader to [3] for a detailed description of these subroutines.

For $x \in R$ and $f \in R[X]$ we write $\sigma_{x,f}$ for the sign vector

$$(\mathrm{sign}(f^{(0)}(x)), \mathrm{sign}(f^{(1)}(x)), \ldots, \mathrm{sign}(f^{(\deg(f))}(x).$$

It is a consequence of Thom's lemma ([9], [12]) that if $f(x) = 0$ then $\sigma_{x,f}$ distinguishes x from all the other roots of f and if $\sigma_{x,f} \neq \sigma_{y,f}$ then these two sign vectors easily enable us to determine whether $x < y$ or $y < x$.

A *k-univariate representation* is a $k+2$-tuple

$$u = (f(T), g_0(T), g_1(T), \ldots, g_k(T)).$$

We write $\deg(u)$ for $\max\{\deg f(T), \deg g_0(T), \deg g_1(T), \ldots, \deg g_k(T)\}$. The point $p = (x_1, x_2, \ldots, x_k)$ in R^k (resp. $(R[i])^k$) is *associated to u* if there exists a root t of $f(T)$ in R (resp. $R[i]$) such that $x_i = g_i(t)/g_0(t)$, for $i = 1, \ldots, k$.

A *k-univariate representation with specified Thom encoding* is a pair (u, σ) where u is a k-univariate representation

$$u = (f(T), g_0(T), g_1(T), \ldots, g_k(T))$$

and $\sigma \in \{-1, 0, 1\}^{deg(f)}$ is the Thom encoding of a real root t_σ of $f(T)$. The point $p = (x_1, x_2, \ldots, x_k)$ in R^k is *associated to* (u, σ) if $x_i = g_i(t_\sigma)/g_0(t_\sigma)$, for $i = 1, \ldots, k$.

Given a polynomial $P(X_1, \ldots, X_k)$ and a k-univariate representation

$$u(T) = (f(T), g_0(T), g_1(T), \ldots, g_k(T)),$$

we denote by $P_u(T)$ the univariate polynomial obtained by substituting $X_i = g_i(T)$, for $i = 0, \ldots, k$, in the homogeneous polynomial $X_0^{d'} P(X_1/X_0, \ldots, X_k/X_0)$ where X_0 is a new variable and d' is the smallest even number larger than the total degree of P.

Given any finite family of polynomials $\mathcal{P} \subset R[X_1, \ldots, X_k]$, a *cell* of the family $\mathcal{P}$ is a semi-algebraically connected component of a realizable sign condition of $\mathcal{P}$.

In our algorithms we often make use of infinitesimals. We denote by $R\langle\epsilon\rangle$ the field of Puiseux series in ϵ with coefficients in R. We refer the reader to [3] (section 2.1, page 1008) for further details.

3.2 Parametrized Sample Points Subroutine

(For more details see [3], section 5.1.3, page 1036) The input is a set of s polynomials

$$\mathcal{P} = \{P_1, \ldots, P_s\} \subset R[Y_1, \ldots, Y_\ell][X_1, \ldots, X_k],$$

each of degree at most d.

The output is a set of $s^k d^{O(k)}$ parametrized univariate representations of the form,

$$(f(Y, \Omega, \delta, t), g_0(Y, \Omega, \delta, t), \ldots, g_k(Y, \Omega, \delta, t)),$$

where $f(Y, \Omega, \delta, t), g_i(Y, \Omega, \delta, t) \in D[\Omega, \delta, Y]$, (here $\frac{1}{\Omega} \gg \delta$ are positive infinitesimals). The set of tuples has the property that for any point $y \in R^\ell$, the union of the sets of points associated to the univariate representations $(f(y,t), g_0(y,t), \ldots, g_k(y,t))$ (provided that $g_0(y,t) \not\equiv 0$), intersects every cell of the set $\mathcal{P}(y)$ in $R\langle 1/\Omega, \delta\rangle^k$.

The complexity is $s^k d^{O(\ell k+k)}$.

3.3 Block Elimination Subroutine

(For more details see [3], section 5.2, page 1037) The input is again a set of s polynomials

$$\mathcal{P} = \{P_1, \ldots, P_s\} \subset R[Y_1, \ldots, Y_\ell][X_1, \ldots, X_k],$$

each of degree at most d. Let $Y = (Y_1, \ldots, Y_\ell)$ and $X = (X_1, \ldots, X_k)$. The algorithm eliminates the block X.

The output consists of two sets, $\mathcal{U}$ and $\mathcal{Q}$. The set $\mathcal{U} \subset D[Y, \Omega, \delta, t])^{k+2}$ consists of parametrized univariate representations

$$u(Y, \Omega, \delta, t) = (f(Y, \Omega, \delta, t), g_0(Y, \Omega, \delta, t), \ldots, g_k(Y, \Omega, \delta, t)).$$

The set $\mathcal{Q} \subset D[Y]$, has the property that for any cell, C of $\mathcal{Q}$, $\mathrm{SIGN}(\mathcal{P}(y, X_1, \ldots, X_k))$ is fixed as y varies over C, and the union of the sets of points associated with all tuples

$$(f(y, \Omega, \delta, t), g_0(y, \Omega, \delta, t), \ldots, g_k(y, \Omega, \delta, t))$$

in $\mathcal{U}$ for which $g_0(y, \Omega, \delta, t) \not\equiv 0$, intersects every cell of the set $\mathcal{P}(y)$, in $(R\langle\Omega, \delta\rangle)^k$.

The complexity of computing the univariate representations is $s^k d^{O(k\ell+k)}$. The degrees of the polynomials generated in this process are bounded by $d^{O(k)}$ (independent of ℓ), in each of the variables as well as in the variables Ω and δ.

The size of the set $\mathcal{Q}$ is $s^{k+1} d^{O(k)}$.

4 Proof of Theorem 1

In this section we describe our new algorithm for performing quantifier elimination which will also prove theorem 1. For ease of exposition we first consider the case of eliminating a single block of existential quantifiers. The main idea is already present in this simple situation.

4.1 Case of a Single Block

We follow the notation in theorem 1. Note that in this case, $\omega = 1$, $X^{[1]} = (X_1, \ldots, X_k)$ and we can assume without loss of generality, $Q_1 = \exists$.

Consider the polynomials $P_1, \ldots, P_s$ as polynomials in $X_1, \ldots, X_k$ with parameters $Y_1, \ldots, Y_l$.

The basic idea is as follows. We compute a set of parametrized univariate representations, using the parametrized sample points subroutine, such that for every value of the parameters, say $(y_1, \ldots, y_l)$, the set of points associated to these univariate representations intersect every cell of the family $\mathcal{P}(y)$. If every polynomial in $\mathcal{P}$ depended on at most τ of the free variables $Y_1, \ldots, Y_l$ then each parametrized univariate representations output by the parametrized sample points subroutine will depend on at most $k\tau$ of the Y_j's.

Also, for every value of the parameters, there exists a witness point satisfying the formula F iff there exists a witness point amongst this set of points. Thus, if we could "follow" each such point over the whole parameter space, then we could obtain a quantifier-free formula by taking a disjunction over all such points. We achieve this by following the point corresponding to the j^{th} largest real root of the polynomial, $f(Y, T)$ for every univariate representation $u = (f(Y, T), g_0(Y, T), \ldots, g_k(Y, T))$, and for every j, $1 \leq j \leq deg(f)$. We can then eliminate quantifiers separately from the $3s$ possible atoms of the form, $P_i \ \{>, <, =\} \ 0, 1 \leq i \leq s$ occurring in the formula F independently. This is the crux of the new idea that allows us to have a combinatorial complexity independent of the number of free variables ℓ.

We now describe the algorithm in more detail.

We call the parametrized sample points subroutine with input the family of polynomials $\mathcal{P} = \{P_1, \ldots, P_s\}$.

Denote the parametrized univariate representations output by the parametrized sample points subroutine $\{u_1, \ldots, u_N\}$, where

$$u_i = (f_i(Y, \Omega, \delta, T), g_{i,0}(Y, \Omega, \delta, T), \ldots, g_{i,k}(Y, \Omega, \delta, T).$$

Note that $N = s^k d^{O(k)}$ and the degrees of the polynomials, $f_i(Y, \Omega, \delta, T), g_{i,j}(Y, \Omega, \delta, T)$, are bounded by $d^{O(k)}$.

Suppose we have the following predicates (we will show how to construct them later) :

$$s_{ij}(Y_1, \ldots, Y_l, T), 1 \leq i \leq N, 1 \leq j \leq d^{O(k)},$$

such that $s_{ij}(Y_1, \ldots, Y_l, T)$ is true iff T is the j^{th} largest real root of $f_i(Y_1, \ldots, Y_l, T)$ considered as a polynomial in T.

Let $F_{i,j}(Y_1, \ldots, Y_l)$ be the formula obtained from F as follows:

We replace every atom of the form $P\ \sigma\ 0, \sigma \in \{=, >, <\}$ in F by the formula, $((\exists T)(P_{u_i}\sigma 0 \wedge s_{ij}(Y_1, \ldots, Y_l, T)))$ (following notation from section 3.1). Then it is clear that,

$$(\exists X_1, \ldots, X_k)F(Y_1, \ldots, Y_l, X_1, \ldots, X_k) \equiv$$
$$\bigvee_{1 \le i \le N} \bigvee_{1 \le j \le d^{O(k)}} F_{ij}(Y_1, \ldots, Y_l).$$

We now use the algorithm for performing quantifier elimination in [3] to independently remove the $s^{k+1}d^{O(k)}$ quantifiers on the right hand side, thus obtaining a quantifier-free formula equivalent to

$$(\exists X_1, \ldots, X_k)F(Y_1, \ldots, Y_l, X_1, \ldots, X_k).$$

Note that there are $s^k d^{O(k)}$ of the s_{ij}'s and for each of them we have to eliminate quantifiers from at most $3s$ formulas corresponding to the atoms $P_i\ \{<, >, =\}\ 0, 1 \le i \le s$.

It remains to show how we can construct the predicates s_{ij}.

Consider the polynomial $f_i(Y_1, \ldots, Y_l, T)$.

Using the block elimination subroutine compute a family of polynomials $\mathcal{Q}_i \subset D[Y_1, \ldots, Y_l]$ such that over every cell of this family the polynomial f_i has the same number of real roots.

The degrees of the polynomials in $\mathcal{Q}_i$ are bounded by $d^{O(k)}$.

Compute a point in every cell of the partition of R^l induced by $\mathcal{Q}_i$ and compute the list of Thom encodings of the real roots of f_i over each such point. The complexity of doing this is $d^{O(kl)}$.

Now using the Inverse Sign Determination Subroutine ([3], section 6.1, page 1041) compute for every list of Thom encodings (say L) obtained above, a formula, $F_L(Y_1, \ldots, Y_l)$ such that for all $(y_1, \ldots, y_l)$ satisfying F_L the list of Thom encodings of the roots of $f_i(y_1, \ldots, y_l, T)$ is L.

Finally, we can construct $s_{ij}(Y_1, \ldots, Y_l, T)$ from the F_L's in the obvious way.

Each s_{ij} will be of the form, $s_{ij} = \wedge_L(F_L \wedge \sigma_L)$, where σ_L is the Thom encoding of the jth root encoded by the list L.

Note that F_L is of size $d^{O(kl)}$ and σ_L is of the form,

$$(f_i = 0) \wedge (f_i'\ \{>, =, <\}\ 0) \wedge \cdots \wedge (f_i^{(deg(f_i))}\ \{>, =, <\}\ 0).$$

Consider the formula,

$$((\exists T)(P_{u_i}\sigma 0 \wedge s_{ij}(Y_1,\ldots,Y_l,T))).$$

Note that the F_L's are already independent of T. Thus the degrees of all the polynomials depending on T in the above is bounded by $d^{O(k)}$. Given the structure of s_{ij} and the above facts, it is clear that we can eliminate T from the above with complexity $d^{O(l'k)}$, where $l' = \min(k\tau, l)$.

There are $s^k d^{O(k)}$ of the s_{ij}'s and for each of them we have to perform the above at most $O(s)$ times. Thus the total complexity is bounded by $s^{k+1}d^{O(l'k)}$.

The dependence of the size of the output on the size of the input is also clear.

4.2 The Case of Many Blocks

In the description of the algorithm we borrow heavily from the description of the algorithm for the general decision problem in [3] (section 5.3, page 1038). We also follow the notation used there.

We use the first two phases of the algorithm for the general decision problem in [3] (section 5.3, page 1038), namely the Elimination Phase and the Substitution Phase.

We first recall a definition which appears in [3] (section 1.3, page 1006).

Let $\mathcal{P} = \{P_1,\ldots,P_s\}$ be a set of s polynomials in k variables $(X_1,\ldots,X_k)$, and let Π denote the partition of the set of variables $(X_1,\ldots,X_k)$ into blocks, $X^{[1]},\ldots,X^{[\omega]}$, where the block $X^{[i]}$ is of size $k_i, 1 \le i \le \omega$. For $x^{[\omega]} \in R^{k_\omega},\ldots,x^{[1]} \in R^{k_1}$, let

$$\mathrm{SIGN}_{\Pi,0}(\mathcal{P})(x^{[\omega]},\ldots,x^{[1]}) =$$

$$(\mathrm{sign}(P_1(x^{[\omega]},\ldots,x^{[1]})),\ldots,\mathrm{sign}(P_s(x^{[\omega]},\ldots,x^{[1]}))).$$

For $x^{[i+1]} \in R^{k_{i+1}},\ldots,x^{[\omega]} \in R^{k_\omega}$, and for all i, $0 < i \le \omega$, we recursively define,

$$\mathrm{SIGN}_{\Pi,i}(\mathcal{P})(x^{[\omega]},\ldots,x^{[i+1]}) =$$

$$\{\mathrm{SIGN}_{\Pi,i-1}(\mathcal{P})(x^{[\omega]},\ldots,x^{[i+1]},x^{[i]}) | x^{[i]} \in R^{k_i}\}.$$

Finally, we define

$$\mathrm{SIGN}_{\Pi}(\mathcal{P}) = \mathrm{SIGN}_{\Pi,\omega}(\mathcal{P}).$$

It is easy to decide the truth or falsity of the formula,

$$Q_\omega(X^{[\omega]})Q_{\omega-1}(X^{[\omega-1]})\dots Q_1(X^{[1]})F(P_1,\dots,P_s),$$

from the set $\mathrm{SIGN}_\Pi(\mathcal{P})$ which we call the *total list of signs of* $\mathcal{P}=(P_1,\dots,P_s)$ with respect to the partition Π of the variables, into the blocks, $(X^{[\omega]},\dots,X^{[1]})$.

Note that the set $\mathrm{SIGN}_\Pi(\mathcal{P})$ is a set of nested sets, where the nesting is of depth ω. When there is only one block of variables, we denote it by $\mathrm{SIGN}(\mathcal{P})$.

It is easy to decide the truth or falsity of a quantified formula, once this $\mathrm{SIGN}_\Pi(\mathcal{P})$ is constructed.

We also recall briefly the input and output of the Elimination and Substitution phases of the algorithm for the general decision problem.

The input is a set of s polynomials, $\mathcal{P}=\{P_1,\dots,P_s\}\subset D[X_1,\dots,X_k]$, and a partition, Π, of the variables into ω blocks, $X^{[1]},\dots,X^{[\omega]}$.

We denote by $\bar{X}^{[i]}$ the variables $(X^{[\omega]},\dots,X^{[i]})$.

There are ω stages in the Elimination Phase, one for each block of variables. In each stage of the Elimination Phase we output two sets $\mathcal{U}_i$, and $\mathcal{Q}_i$. The set $\mathcal{U}_i\subset(D[\bar{X}^{[i]},\Omega,\delta_i,t_i]^{k+2}$ consists of parametrized univariate representations generated by calls to the *Block Elimination Subroutine*. The set $\mathcal{Q}_i$ consists of polynomials in the variables $\bar{X}^{[i+1]}$, with the property that the set $\mathrm{SIGN}_{\Pi,i}(x^{[\omega]},\dots,x^{[i+1]})$ stays invariant, as $(x^{[\omega]},\dots,x^{[i+1]})$ varies over a cell of $\mathcal{Q}_i$. In the Substitution Phase for each $\bar{u}=(u_1,\dots,u_\omega)\in\bar{\mathcal{U}}=\Pi_{1\leq i\leq\omega}\mathcal{U}_i$ we define the *associated triangular system* and the *associated list of polynomials*, $\mathcal{T}_{\bar{u}}$ and $\mathcal{P}_{\bar{u}}$, as follows:

Let

$$u_i=(f^{[i]}(\bar{X}^{[i+1]},\Omega_i,\delta_i,t_i),g_0^{[i]}(\bar{X}^{[i+1]},\Omega_i,\delta_i,t_i)\dots,g_{k_{i+1}}^{[i]}(\bar{X}^{[i+1]},\Omega_i,\delta_i,t_i)).$$

For a polynomial $P(X^{[\omega]},\dots,X^{[1]})$, let $P_{\bar{u}}(t_1,\dots,t_\omega)$ denote the polynomial obtained by successively replacing the blocks of variables $X^{[i]}$, with the rational fractions associated with the tuple u_i.

Define ${T_{\bar{u}}}^{[i]}(t_\omega,\dots,t_i)={f^{[i]}}_{\bar{u}}$,

$$\mathcal{T}_{\bar{u}}=({T_{\bar{u}}}^{[\omega]}(t_\omega),{T_{\bar{u}}}^{[\omega-1]}(t_\omega,t_{\omega-1}),\cdots,{T_{\bar{u}}}^{[1]}(t_\omega,t_{\omega-1},\dots,t_1)),$$

and $\mathcal{P}_{\bar{u}}$ as $\{P_{\bar{u}}\mid P\in\mathcal{P}\}$.

Let $\mathcal{Z}_{\bar{u}}$ be the set

$$\{\alpha\in R\langle 1/\Omega_1,\delta_1,\dots,1/\Omega_\omega,\delta_\omega\rangle^\omega\mid\alpha\text{ a zero of }\mathcal{T}_{\bar{u}}\}.$$

We define $\mathcal{Z}$ and $\bar{\mathcal{P}}$ by $\mathcal{Z} = \cup_{\bar{u}\in\bar{\mathcal{U}}} \mathcal{Z}_{\bar{u}}$ and $\bar{\mathcal{P}} = \cup_{\bar{u}\in\bar{\mathcal{U}}} \mathcal{P}_{\bar{u}}$. For any $x = (x_\omega, \ldots, x_1) \in R^\omega$ let $\bar{x}_i$ denote $(x_\omega, \ldots, x_i)$.

For every $\bar{u} = (u_1, \ldots, u_\omega)$, let $s_{u,j_1,\ldots,j_\omega}(Y_1, \ldots, Y_l, T_1, \ldots, T_\omega)$ denote the predicate that T_i is the j_i^{th} real root of $f^{[i]}(Y_1, \ldots, Y_l, T_1, \ldots, T_\omega)$ viewed as a polynomial in T_i, for $1 \leq i \leq \omega$.

We construct these predicates in a similar way as in the previous case with one block.

We can now again make local substitutions as in the previous case, to obtain a quantifier-free formula.

It is easy to verify that the complexity of this algorithm is $s^{\Pi(k_i+1)} d^{l' \Pi O(k_i)}$. This proves theorem 1.

5 Proofs of Theorems 2 and 3

In order to prove theorem 2, notice that applying the quantifier elimination algorithm described above to a uniform sequence of formulas yields another uniform sequence. Observe that $\tau = \omega$ in this case. The number of free variables is unbounded but the complexity is bounded by

$$s^{\Pi_i(l_i+1)} d^{\omega \Pi_i(l_i+1) \Pi_i O(l_i)} = s^{\Pi_i(l_i+1)} d^{\omega \Pi_i O(l_i^2)}.$$

Notice that it is crucial in this case that the complexity of the quantifier elimination algorithm be independent of the total number of free variables.

5.1 Natural to Active Domain Semantics

We now prove theorem 3. Consider a query σ with natural domain semantics.

Let

$$\sigma = (Q_1 T_1)(Q_2 T_2) \ldots (Q_l T_l) \psi(T_1, \ldots, T_l),$$

where each $Q_i \in \{\exists, \forall\}$, where ψ is a L'-formula.

For every positive integer n we obtain a L-formula $\sigma_n(Y_1, \ldots, Y_n)$ with n free variables $Y_1, \ldots, Y_n$ from σ by replacing every occurrence of $R(T_j)$ in ψ by the disjunct,

$$R_n(T_j, Y_1, \ldots, Y_n) = \bigvee_{1 \leq i \leq n} (T_j = Y_i),$$

for $1 \leq j \leq l$.

Let,

$$\sigma_n(Y_1, \ldots, Y_n) = (Q_1 T_1)(Q_2 T_2) \ldots (Q_l T_l)\psi_n(T_1, \ldots, T_l, Y_1, \ldots, Y_n).$$

It is clear that $\{\psi_n \mid n > 0\}$ is a uniform sequence of formulas.

Using theorem 2 we can eliminate the quantifiers from σ_n, to obtain formulas, ϕ_n.

Now, let

$$\phi_n(Y_1, \ldots, Y_n) = Q^1_{1 \leq i_1 \leq n} \cdots Q^k_{1 \leq i_k \leq n} \phi(Y_{i_1}, \ldots, Y_{i_k}),$$

$Q^j \in \{\bigvee, \bigwedge\}$.

We syntactically replace every occurrence of $\bigvee_{1 \leq i_j \leq n}$ by $(\exists T_j)$ and replace every occurrence of $\bigwedge_{1 \leq i_j \leq n}$ by $(\forall T_j)$, and replace Y_{i_j} by the variable T_j.

Let the L'-sentence obtained in this manner from the sequence of formulas ϕ_n be denoted by Φ. It is clear that Φ with active domain semantics is equivalent to σ with natural domain semantics.

Let the total number of polynomials appearing in σ be s and their degrees be bounded by d. Moreover, let the number of blocks of quantifiers be m with the i^{th} block of length l_i, with $\sum_i l_i = l$. Then the complexity of the above algorithm is $s^{\Pi_i(l_i+1)} d^{\omega \Pi_i O(l_i^2)}$. This follows from the complexity bound in theorem 2.

6 On the Expressive Power of Constraint Queries over the Reals

We next use theorem 3 to prove that parity is not expressible over the reals.

First we choose a real closed field $\mathcal{R}$, with elements $\Omega_1, \Omega_2, \ldots, \Omega_n$, where Ω_1 is larger than all the integers, Ω_2 is larger than all powers of Ω_1 and so on. We can prove that such models exist by the compactness theorem. However, to be more specific we can consider the field, $R\langle\Omega_1\rangle\langle\Omega_2\rangle \cdots \langle\Omega_n\rangle$, where R is any real closed field (say the reals) and for every i, $R\langle\Omega_1\rangle\langle\Omega_2\rangle \ldots \langle\Omega_i\rangle$ is the field of Puiseux series in $\frac{1}{\Omega_i}$ with coefficients in $R\langle\Omega_1\rangle\langle\Omega_2\rangle \ldots \langle\Omega_{i-1}\rangle$. It is well known that the resulting field is real closed [9].

Then, for any l-ary L-formula, $\psi(Y_1, \ldots, Y_l)$ is equivalent to a finite Boolean formula whose atoms are of the form, $Y_j \leq Y_k, 1 \leq j, k \leq l$, for the Y_j's restricted to the set $\{\Omega_1, \ldots, \Omega_n\}$. This is clear, because the sign of any polynomial $P(\Omega_{i_1}, \ldots, \Omega_{i_l})$ with integral coefficients is determined only by the ordering of the i_j's.

We now prove that parity is not expressible over the reals. Note that it suffices to prove this for any one real closed field, as we can then deduce this for all other real closed fields, by an application of the Tarski-Seidenberg transfer principle as follows.

Suppose that there exists a first order sentence σ in the language L' which expresses parity for some real closed field $\mathcal{R}_1$. Let $\mathcal{R}_2$ be another real closed field. We claim that that σ has to express parity over $\mathcal{R}_2$ too. If not, then either there exists a finite set $S = \{y_1, \ldots, y_n\} \subset \mathcal{R}_2$, with $|S|$ even, such that $\langle \mathcal{R}_2, S \rangle \models \neg\sigma$, or there exists a finite set $S = \{y_1, \ldots, y_n\} \subset \mathcal{R}_2$, with $|S|$ odd, such that $\langle \mathcal{R}_2, S \rangle \models \sigma$.

In either case, consider the L-sentence,

$$\phi = (\exists Y_1, \ldots, Y_n) \bigwedge_{1 \le i < j \le n} (Y_i \neq Y_j) \wedge \sigma',$$

where σ' is obtained from σ by replacing every occurrence of the unary predicate $R(T)$ by the disjunction $\bigvee_{1 \le i \le n}(T = Y_i)$.

Note that ϕ being an L-sentence does not contain the unary relation symbol R. Moreover, either $\mathcal{R}_2 \models \phi$ and $\mathcal{R}_1 \models \neg\phi$, or $\mathcal{R}_2 \models \neg\phi$ but $\mathcal{R}_1 \models \phi$, which is impossible by the transfer principle.

This shows that it suffices to prove the inexpressibility of parity over any one real closed field in order to prove its inexpressibility over all real closed fields.

Theorem 4 *Parity is not expressible over the reals.*

Proof: Assume that there exists a first order sentence σ expressing parity.

Let

$$\sigma = (Q_1 T_1)(Q_2 T_2) \ldots (Q_l T_l) \phi(T_1, \ldots, T_l),$$

where each $Q_i \in \{\exists, \forall\}$.

Using theorem 3 we can assume that there exists a query Ψ which is equivalent to σ in the active domain semantics.

Let,

$$\Psi = (Q_1 T_1)(Q_2 T_2) \ldots (Q_m T_m) \psi(T_1, \ldots, T_m),$$

where each $Q_i \in \{\exists, \forall\}$.

Now consider the set, $S = \{\Omega_1, \ldots, \Omega_n\} \subset \mathcal{R}$. Now, if all the bound variables in the sentence Ψ vary only over S then in Ψ we can replace the predicate $\psi(T_1, \ldots, T_m)$ by another m-ary predicate, $\psi'(T_1, \ldots, T_m)$, with atoms of the form $T_j \leq T_k, 1 \leq j, k \leq m$, (as in the preceding paragraph) to obtain another sentence Ψ'.

Note that, for $S = \{\Omega_1, \ldots, \Omega_n\} \subset \mathcal{R}$, Ψ' is satisfied iff n is even. However, Ψ' contains no additions or multiplications. This would imply that parity is expressible over $\langle \omega, \leq \rangle$. However, it can be easily shown using Ehrenfeucht-Fraisse games that parity is not expressible over $\langle \omega, \leq \rangle$ (see [17]). This completes the proof. $\square$

It is also clear from the proof of the inexpressibility of parity over the reals described above, that the properties of finite sets that are not definable over $\langle \omega, \leq \rangle$, that is over natural numbers with order but no arithmetic, are also not definable over the reals. Thus having arithmetic adds no extra expressive power over having only inequalities.

6.1 Expressing Connectivity of Semi-algebraic Sets

One consequence of the inexpressibility of parity proved above, is a short proof that that connectivity of semi-algebraic sets is also not expressible by a first-order sentence. The fact that connectivity of semi-algebraic sets in $\Re^k, k \geq 2$, cannot be expressed by a first-order sentence was proved by Grumbach and Su [17], where they gave a reduction from the Majority query. (Note that connectivity for semi-algebraic subsets of $\Re$ is quite easily expressible by a first order sentence, since a connected semi-algebraic subset of $\Re$ must be a segment or a single point.) Below, we include a different proof which gives a direct reduction from parity but works only for $k > 2$.

More precisely, let $L = \langle 0, 1, +, \times, \leq \rangle$ be the language of ordered fields. We extend L to $L'' = \langle 0, 1, +, \times, \leq, \psi \rangle$ by including a ternary relation symbol ψ where $\psi(x, y, z) \Leftrightarrow (x, y, z) \in T$, for some semi-algebraic set $T \in \Re^3$.

We prove the following theorem.

Theorem 5 *There is no first order L''-sentence σ such that for every semi-algebraic set $T \in \Re^3$, and T semi-algebraically connected, $\langle \Re, T \rangle \models \sigma$ and for every every semi-algebraic set $T \in \Re^3$, and T not semi-algebraically connected, $\langle \Re, T \rangle \models \neg\sigma$.*

We now prove theorem 5.

Proof: Assume that there exists such a sentence σ. We will prove that this implies that there exists a L'-sentence for expressing parity.

Let R be a unary relation symbol expressing membership in a finite set $S \subset \Re$, that is $R(x) \Leftrightarrow x \in S$.

We construct a semi-algebraic set $T \subset \Re^3$ such that, membership to T can be expressed by a first-order L'-formula, and moreover T is connected iff either $|S|$ is even or $|S| = 1$.

Firstly, it is clear that there exists first-order L'-predicates $\max(x)$ (resp. $\min(x)$) expressing that x is the maximum (resp. minimum) element of S.

Let us assume that $|S| > 1$, and let $s, t \in \Re$ be the maximum and minimum elements of S.

Also, for $a \in S, a \neq t$, let $succ(a)$ denote the smallest element in S which is greater than a. It is clear that this is also first order expressible.

Finally, for $a, b \in S, a < b, (a, b) \neq (s, t)$, let $T_{a,b} \subset \Re^3$ be the set defined by,

$$X^2 + Y^2 + (Z - \frac{a+b}{2})^2 - (\frac{a-b}{2})^2 = 0,$$

$$Y - (\frac{b-s}{t-s} + 1)X = 0,$$

and let $T_{s,t} \subset \Re^3$ be the set defined by,

$$X^2 + Y^2 + (Z - \frac{s+t}{2})^2 - (\frac{s-t}{2})^2 = 0,$$

$$Y - X = 0.$$

Let,

$$T = \cup_{a \in S \text{ and } succ(succ(a)) \in S} T_{a, succ(succ(a))} \cup T_{s,t}.$$

It is clear from the construction that the predicate $\psi(x, y, z)$ expressing $(x, y, z) \in T$, is expressible in terms of the predicate R.

Moreover, it is an easy exercise to check that T is connected iff $|S|$ is even or $|S| = 1$.

Thus, by replacing each occurrence of $\psi(x, y, z)$ in σ by the formula expressing membership in T obtained above, we will obtain a first-order L'-sentence expressing parity. This is a contradiction. $\square$

Remark: The property that a semi-algebraic subset of $\Re^k$ is closed is expressible by a first-order sentence (see section 2.1). This fact, together with quantifier elimination, gives an automatic (and algorithmic) proof of the fact that the closure of a semi-algebraic set is semi-algebraic. The fact that connectivity is not expressible by a first-order sentence, implies that there is no such easy proof of the (nevertheless true) statement that the semi-algebraically connected components of a semi-algebraic set are themselves semi-algebraic. It is considerably more difficult to design algorithms to compute semi-algebraic descriptions of the semi-algebraically connected components of a given semi-algebraic set. For single exponential time algorithms to compute semi-algebraic descriptions of the connected components of semi-algebraic sets see [23, 4].

References

[1] P.K. Agarwal, J. Matousek On range searching with semi-algebraic sets, *Discrete and Computational Geometry* 11:393-418 (1994).

[2] S. Basu Uniform Quantifier Elimination and Constraint Query Processing, *Proc. International Symposium on Symbolic and Algebraic Computation, 1997, 21-27.*

[3] S. Basu, R. Pollack, M.-F. Roy On the Combinatorial and Algebraic Complexity of Quantifier Elimination , *Journal of the ACM*, Nov 1996, Vol. 43, Number 6, 1002-1045. (Extended Abstract in *Proc. 35th Symposium on Foundations of Computer Science,* (1994).)

[4] S. Basu, R. Pollack, M.-F. Roy Complexity of Computing Semi-algebraic Descriptions of the Connected Components of Semi-Algebraic Sets, *Proc. of the International Symposium on Symbolic and Algebraic Computation, (ISSAC)*, 25-29, 1998.

[5] S. Basu On the combinatorial and topological complexity of a single cell, *Proc. of the Symposium on the Foundations of Computer Science,* 606-616, 1998.

[6] M. Benedikt, G. Dong, L. Libkin, L. Wong Relational Expressive Power of Constraint Query Languages, *Proc. of 15th ACM Symposium on Principles of Database Systems,* 5-16,(1996).

[7] M. Benedikt, L. Libkin On the Structure of Queries in Constraint Query Languages, *Proc. of 11th Annual IEEE Symposium on Logic in Computer Science,* 25-34, (1996).

[8] M. Benedikt and L. Libkin Relational Languages over Interpreted Structure, *Proceedings of 15th ACM Symposium on Principles of Database Systems, 87-98, 1997.*

[9] J. Bochnak, M. Coste, M.-F. Roy Géométrie algébrique réelle. Springer-Verlag (1987).

[10] B. Chazelle Computational Geometry: A Retrospective, *Proc. 26th ACM Symp. on Theory of Computing, 75-94, (1994).*

[11] G. E. Collins, Quantifier elimination for real closed fields by cylindrical algebraic decomposition, Springer Lecture Notes in Computer Science 33, 515-532.

[12] M. Coste, M.-.F Roy Thom's lemma, the coding of real algebraic numbers and the topology of semi-algebraic sets. *J. of Symbolic Computation 5 121-129 (1988).*

[13] A. Ehrenfeucht An application of games to the completeness problem for formalized theories, *Fund. Math*, 49, 1961.

[14] R. Fagin Probabilities on finite models, *Journal of Symbolic Logic*, 41(1):50-58, 1976.

[15] R. Fagin Finite model theory - a personal perspective, *Theoretical Computer Science,* 116(1)3-31, 1993.

[16] H. Gaifman On local and non-local properties, *Proc. Herbrand Symposium Logic Colloquium,* 105-35, North Holland, 1981.

[17] S. Grumbach, J. Su Queries with Arithmetical Constraints, *Theoretical Computer Science*, 173(1):151-181, 1997.

[18] S. Grumbach, J. Su Finitely Representable databases, *Proc. of 13th ACM Symposium on Principles of Database Systems, 289-300, (1994).*

[19] S. Grumbach, J. Su Dense Order Constraint Databases, *Proc. of 14th ACM Symposium on Principles of Database Systems, 66-77, (1995).*

[20] D. Grigor'ev, The Complexity of deciding Tarski algebra *Journal of Symbolic Computation 5 (1988), 65-108.*

[21] Y. Gurevich Logic and the challenge of computer science, Current Trends in Theoretical Computer Science, 1-57, Computer Science Press, 1988.

[22] J. Heintz, M.-F. Roy, P. Solernò Sur la complexité du principe de Tarski-Seidenberg. *Bull. Soc. Math. France 118 101-126 (1990).*

[23] J. Heintz, M.-F. Roy, P. Solernò Description of the Connected Components of a Semialgebraic Set in Single Exponential Time, *Discrete and Computational Geometry* 11:121-140 (1994).

[24] D. Halperin, M. Sharir New Bounds for Lower Envelopes in Three Dimensions, with Applications to Visibility in Terrains, *Discrete and Computational Geometry* 12:313-326 (1994).

[25] D. Halperin, M. Sharir Almost Tight Upper Bounds for the Single Cell and Zone Problems in Three Dimensions, *Discrete and Computational Geometry* 14:385-410 (1995).

[26] P. Kanellakis, D. Q. Goldin Constraint Programming and Database Query Languages, *Proc. 2nd Conf on Theoretical Aspects of Computer Software (TACS),* LNCS Vol 789, Springer-Verlag, (1994).

[27] P. Kanellakis, G. Kuper, P. Revesz Constraint Query Languages, *Proc. of 9th ACM Symposium on Principles of Database Systems,* (1990).

[28] J. Renegar On the computational complexity and geometry of the first order theory of the reals, *J. of Symbolic Comput.13(3):255-352, (1992).*

[29] A. Seidenberg A new decision method for elementary algebra, *Annals of Mathematics, 60:365-374, (1954).*

[30] M. Sharir Almost Tight Upper Bounds for Lower Envelopes in Higher Dimensions, *Discrete and Computational Geometry* 12:327-345 (1994).

[31] A. Tarski A Decision method for elementary algebra and geometry, University of California Press (1951).

[32] J.D. Ullman Principles of Database Systems, Computer Science Press, 1983.

[33] V. Weispfenning The complexity of linear problems in fields, *Journal of Symbolic Computation 5(1988).*

[34] G.M. Ziegler Lectures on Polytopes Springer-Verlag, 1994.