First-order queries on finite structures over the reals

Jan Paredaens, University of Antwerp[*]
Jan Van den Bussche, University of Limburg[†]
Dirk Van Gucht, Indiana University[‡]

Abstract

We investigate properties of finite relational structures over the reals expressed by first-order sentences whose predicates are the relations of the structure plus arbitrary polynomial inequalities, and whose quantifiers can range over the whole set of reals. In constraint programming terminology, this corresponds to Boolean real polynomial constraint queries on finite structures. The fact that quantifiers range over all reals seems crucial; however, we observe that each sentence in the first-order theory of the reals can be evaluated by letting each quantifier range over only a finite set of real numbers without changing its truth value. Inspired by this observation, we then show that when all polynomials used are linear, each query can be expressed uniformly on all finite structures by a sentence of which the quantifiers range only over the finite domain of the structure. In other words, linear constraint programming on finite structures can be reduced to ordinary query evaluation as usual in finite model theory and databases. Moreover, if only "generic" queries are taken into consideration, we show that this can be reduced even further by proving that such

[*]Dept. Math. & Computer Sci., University of Antwerp (UIA), Universiteitsplein 1, B-2610 Antwerp, Belgium. E-mail: pareda@uia.ac.be.

[†]Dept. WNI, University of Limburg (LUC), B-3590 Diepenbeek, Belgium. E-mail: vdbuss@luc.ac.be.

[‡]Computer Science Department, Indiana University, Bloomington, IN 47405-4101, USA. E-mail: vgucht@cs.indiana.edu.

queries can be expressed by sentences using as polynomial inequalities only those of the simple form $x < y$.

1 Introduction

In this paper we are motivated by two fields of computer science which heavily rely on logic: relational databases and constraint programming. We will look at the latter from the perspective of the former.

In classical relational database theory [1], a database is modeled as a relational structure. The domain of this structure is some fixed universe $\mathbf{U}$ of possible data elements (such as all strings, or all natural numbers), and is typically infinite. The relations of the structure, in contrast, are always finite as they model finite tables holding data. As a consequence, the *active* domain of the database, consisting of all data elements actually occurring in one or more of the relations, is finite as well.

A (Boolean) *query* is a mapping from databases (over some fixed relational signature) to true or false. A basic way of expressing a query is by a first-order sentence over the relational signature. For example, on a database containing information on children and hobbies, the query "does each parent have at least all hobbies of his children?" is expressed by the sentence $(\forall p)(\forall c)(\forall h)(\mathit{Child}(p,c) \wedge \mathit{Hobby}(c,h) \rightarrow \mathit{Hobby}(p,h))$.

Since the domain of each database is $\mathbf{U}$, the quantifiers in a sentence expressing a query will naturally range over the whole infinite $\mathbf{U}$. However, Aylamazyan, Gilula, Stolboushkin, and Schwartz [5] showed that in order to obtain the result of the query it suffices to let the quantifiers range over the active domain augmented with a finite set of q additional data elements, where q is the number of quantified variables in the formula expressing the query. The intuition behind this result is that all data elements outside the active domain of a given database are alike with respect to that database.

Alternatively, we can choose to let the quantifiers range over the active domain only, thus obtaining a semantics which is quite different from the natural interpretation. For example, consider databases over the single unary relation symbol P. Then the sentence $(\forall x)P(x)$ will always be false under the natural interpretation, while under the active-domain interpretation it will always be true. In fact, it is not obvious that each query expressible under the natural interpretation is also expressible under the active-domain interpretation. Hull and Su [14] established that the implication indeed holds. (The converse implication holds as well, since the active-domain interpretation can easily be simulated under the natural interpretation using bounded quantification.)

In recent years, much attention has been paid to "constraint programming languages" (e.g., [8]). In particular, in 1990, Kanellakis, Kuper and Revesz demonstrated that the idea of constraint programming also applies to database query languages by introducing the framework of "constraint query languages" [15]. An important instance of this framework is that of real polynomial constraints. Here, the universe $\mathbf{U}$ of data elements is the field $\mathbf{R}$ of real numbers. Databases then are relational structures over $\mathbf{R}$, but the database relations need no longer be finite; it suffices that they are definable as finite Boolean combinations of polynomial inequalities. In other words, each k-ary relation of the structure must be a semi-algebraic subset of $\mathbf{R}^k$ [9].

A basic way of querying real polynomial constraint databases is again by first-order sentences, which can now contain polynomial inequalities in addition to the predicate symbols of the relational signature. For example, if the database holds a set S of points in $\mathbf{R}^2$, the query "do all points in S lie on a common circle?" is expressed by $(\exists x_0)(\exists y_0)(\exists r)(\forall x)(\forall y)(S(x,y) \rightarrow (x-x_0)^2+(y-y_0)^2=r^2)$. Note that quantifiers are naturally interpreted as ranging over the whole of $\mathbf{R}$. In order to evaluate such a sentence on a database, we replace each predicate symbol in the formula by the polynomial definition of the corresponding database relation, and obtain a sentence in the pure first-order theory of the reals. As is well-known, this theory is decidable [22]; the truth value of the obtained sentence yields the result of the query. So, real polynomial constraint queries are effectively computable.

Finite relations are semi-algebraic, so that finite relational databases over the reals form an important special case of real polynomial constraint databases. For example, if we want to model a database holding a finite number of rectangles, we can either choose to store the full extents of the rectangles, resulting in the infinite set of all points on the rectangles (represented in terms of linear inequalities in the obvious way), or we can choose to store only the corner points of each rectangle, resulting in a finite relation.

In the present paper, we investigate whether the results by Aylamazyan et al. and by Hull and Su, mentioned in the beginning of this Introduction, carry over from classical first-order queries on relational databases to polynomial constraint queries on finite databases over the reals. Indeed, as in the classical case, one can give an alternative active-domain semantics to constraint sentences and again ask whether this is without loss of expressive power. Note, however, that active-domain quantification defies the very nature of constraint programming as a means to reason about intentionally defined, potentially infinite, ranges of values. Hence, it is not obvious that the results just mentioned might carry over at all.

Nonetheless, we have found a natural analog of the Aylamazyan et al. theorem, and we have been able to establish the verbatim analog of the Hull-Su theorem in the case when only linear polynomials are used. This is often the case in practice. Our result might be paraphrased by saying that on finite structures, first-order linear constraint programming can be reduced to ordinary query evaluation as usual in finite model theory and databases.

Our development is based upon the following observation. Consider a prenex normal form sentence $(Q_1 x_1)\dots(Q_n x_n)M(x_1,\dots,x_n)$ in the first-order theory of the reals. For any finite set D_0 of real numbers, there exists a sequence $D_0 \subseteq D_1 \subseteq \cdots \subseteq D_n$ of finite sets of reals such that the sentence can be evaluated by letting each quantifier Q_i range over D_i only (rather than over the whole of $\mathbf{R}$) without changing the sentence's truth value. By taking D_0 to be the active domain of a given finite database over the reals, we get the analog in the real case of the Aylamazyan et al. theorem.

The reader familiar with Collins's method for quantifier elimination in real-closed fields through cylindrical algebraic decomposition (cad) [3, 4, 11] will not be surprised by the above observation. Indeed, it follows more or less directly from an obvious adaptation of the cad construction. However, we give an alternative self-contained proof from first principles which abstracts away the purely algorithmical aspects of the cad construction and focuses on the logic behind it. Importantly, this proof provides us with a basis to show how in the case of linear polynomials, the construction of the sequence $D_1 \subseteq \cdots \subseteq D_n$ departing from the active domain D_0 can be simulated using a linear constraint formula. As a result, we obtain the analog in the real case of the Hull-Su theorem.

In a final section of this paper, we look at queries that are "generic," i.e., that do not distinguish between isomorphic databases. Genericity is a natural criterion in the context of classical relational databases [2, 10]. Perhaps this is a little less so for databases over the reals; in other work [18] we have proposed alternative, "spatial" genericity criterions based on geometrical intuitions.

Nevertheless, it remains interesting to investigate which classically generic queries can be expressed using linear constraint sentences.

Sentences that do not contain any polynomial inequalities always express generic queries, but from the moment a sentence even contains only simple inequalities of the form $x < y$ it can already be non-generic. Furthermore, examples are known (e.g., [1, Exercise 17.27]) of generic queries expressible with such simple inequalities but not without. In other words, simple inequalities, though inherently non-generic when viewed in isolation, help to express more generic queries. The natural question now is to ask whether general linear polynomial inequalities help even more. We will answer this question negatively.[1]

This paper is organized as follows. We start with a rather general Section 2 in which we introduce the notion of domain sequence on which much of our development will hinge. In Section 3 we then introduce the subject of queries on real databases. In Section 4 we focus on the linear case. In Section 5 we discuss generic queries.

After we presented the original ideas contained in the present paper at a conference [19], several researchers have been able to generalize our results. We provide a brief summary of these generalizations in Section 6.

2 Domain sequences

We will use the basic terminology from mathematical logic [12]. Let $\mathcal{A}$ be a structure over a finite relational vocabulary L. The domain of $\mathcal{A}$ is denoted by A. Let $\Phi(x_1, \dots, x_k)$ be a first-order formula over L written in prenex normal form

$$(Q_{k+1}x_{k+1}) \dots (Q_n x_n) M(x_1, \dots, x_n), \tag{$*$}$$

with each Q_i either $\exists$ or $\forall$ and M quantifier-free. If $k = 0$ then Φ is a sentence; if $k = n$ then Φ is quantifier-free. If $\bar{a} = a_1, \dots, a_k \in A$ is a tuple of elements in A then the truth of Φ in $\mathcal{A}$ with a_i substituted for x_i is denoted by $\mathcal{A} \models \Phi[\bar{a}]$.

If D_{k+1}, …, D_n are subsets of A, then we write

$$(\mathcal{A}; D_{k+1}, \dots, D_n) \models \Phi[\bar{a}]$$

[1]We thus provide a partial rectification of Kuper's original intuitions [16] (which are incorrect as stated).

if $\Phi[\bar{a}]$ evaluates to true in $\mathcal{A}$ when we let each quantifier Q_i range over D_i only rather than over the whole of A.

Example 2.1 Let $\mathcal{A}$ consist of the integers together with the predicate $y = x^2$, and let Φ be the sentence $(\forall x)(\exists y)y = x^2$. Then $(\mathcal{A}; \{-1, 2\}, \{1, 4\}) \models \Phi$, but $(\mathcal{A}; \{-1, 2\}, \{1, 3\}) \not\models \Phi$.

In this section, we prove the following theorem:

Theorem 2.2 *Let Φ be a sentence $(Q_1 x_1) \dots (Q_n x_n) M(x_1, \dots, x_n)$, and let D_0 be a finite subset of A. Then there exists an increasing sequence $D_0 \subseteq D_1 \subseteq \dots \subseteq D_n$ of finite subsets of A such that*

$$\mathcal{A} \models \Phi \quad \iff \quad (\mathcal{A}; D_1, \dots, D_n) \models \Phi.$$

Example 2.3 As a trivial illustration, let $\mathcal{A}$ consist of the integers together with the predicate $x = y^2$, and let Φ be the sentence $(\forall x_1)(\exists x_2)x_2 = (x_1)^2$. Let D_0 be the empty set. We have $\mathcal{A} \models \Phi$, and indeed for $D_1 = \{-1, 2\}$ and $D_2 = \{-1, 2, 1, 4\}$ we have $(\mathcal{A}; D_1, D_2) \models \Phi$.

To prove Theorem 2.2 we introduce various auxiliary notions on which we will also rely in later sections.

We will use the following natural equivalence relation on A^n:

Definition 2.4 Two points $\bar{a}$ and $\bar{b}$ in A^n are called equivalent, denoted $\bar{a} \equiv \bar{b}$, if for each atomic formula $F(x_1, \dots, x_n)$ we have $\mathcal{A} \models F[\bar{a}]$ iff $\mathcal{A} \models F[\bar{b}]$.

In model-theoretic terminology, $\bar{a}$ and $\bar{b}$ are equivalent if they are of the same basic type in $\mathcal{A}$.

Example 2.5 Let $\mathcal{A}$ consist of the reals together with the predicates $C(x, y)\ \theta\ 0$, $L_1(x, y)\ \theta\ 0$, and $L_2(x, y)\ \theta\ 0$, where θ is $<$, $=$, or $>$, and C, L_1 and L_2 are polynomials describing the circle and two lines depicted in Figure 1. The same figure shows that there are 19 equivalence classes in A^2: $\{a\}$, $\{b, c\}$, $\{d, e\}$, A, $B \cup D$, C, E, $F \cup H$, G, I, $J \cup L$, K, α, β, γ, $\delta \cup \lambda$, ϵ, η and κ.

We now extend this equivalence relation inductively to lower dimensions such that the equivalence classes at each dimension are "cylindrical" over the equivalence classes at the next lower dimension:

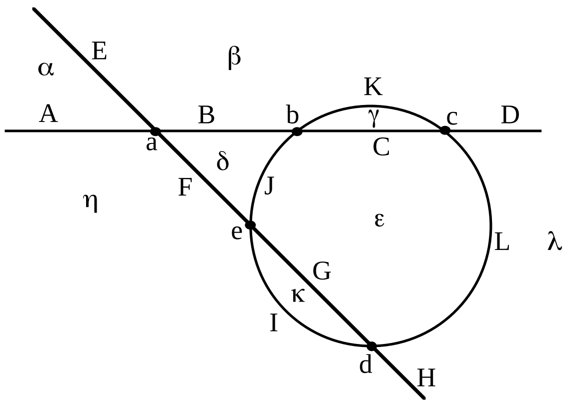

Figure 1: Equivalence classes in the plane induced by a circle and two lines.

Definition 2.6 Let $i < n$ and assume $\equiv$ is already defined on A^{i+1}. Then for $\bar{a}, \bar{b} \in A^i$ we say $\bar{a} \equiv \bar{b}$ if for each $a_{i+1} \in A$ there is a $b_{i+1} \in A$ such that $(\bar{a}, a_{i+1}) \equiv (\bar{b}, b_{i+1})$ and conversely, for each b_{i+1} there is an a_{i+1} such that $(\bar{b}, b_{i+1}) \equiv (\bar{a}, a_{i+1})$.

Example 2.7 In Figure 2 there are 12 equivalence classes in A: $\{p\}$, $\{q\}$, $\{r\}$, $\{s\}$, $\{t\}$, $\{u\}$, P, $Q \cup V$, R, S, T and U.

We note for further use:

Lemma 2.8 *For each i, $\equiv$ is of finite index on A^i.*

Proof. By downward induction on i. The base case $i = n$ is trivial since the number of atomic formulas $F(x_1, \ldots, x_n)$ is finite (we assumed a finite relational vocabulary). So assume $i < n$. For $\bar{a} \in A^i$, let $\kappa(\bar{a})$ be the set of equivalence classes in A^{i+1} intersecting the "vertical line through $\bar{a}$" $\{(\bar{a}, a_{i+1}) \mid a_{i+1} \in A\}$. Clearly, for $\bar{a}, \bar{b} \in A^i$, $\bar{a} \not\equiv \bar{b}$ implies $\kappa(\bar{a}) \neq \kappa(\bar{b})$. Since by induction, $\equiv$ is of finite index on A^{i+1}, κ can have only a finite number of possible values and hence $\equiv$ is of finite index on A^i as well. ■

The relevance of the equivalence relations just defined is demonstrated by the following lemma. We use the following notation: let $\Phi(\bar{x})$ be as in $(*)$ above. For $k \leq i \leq n$, $\Phi|_i$ stands for the formula

$$(Q_{i+1}x_{i+1}) \ldots (Q_n x_n) M(x_1, \ldots, x_n).$$

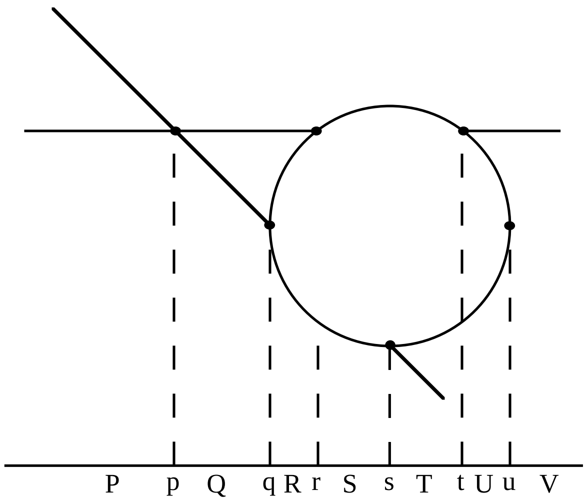

Figure 2: From equivalence classes in A^2 (Example 2.5) to equivalence classes in A.

So, $\Phi|_k$ equals Φ and $\Phi|_n$ equals M.

Lemma 2.9 *Let $k \leq i \leq n$, and let $\bar{a} \equiv \bar{b}$ be equivalent points in A^i. Then*

$$\mathcal{A} \models \Phi|_i[\bar{a}] \quad \Longleftrightarrow \quad \mathcal{A} \models \Phi|_i[\bar{b}].$$

Proof. The proof is a straightforward downward induction on i. The base case, $i = n$ and $\Phi|_n$ being quantifier-free, is obvious. Now let $k \leq i < n$. We have $\Phi|_i = (Q_{i+1}x_{i+1})\Phi|_{i+1}$. We first consider the case $Q_{i+1} = \exists$. Note that we only have to prove the implication from left to right; the other direction follows by symmetry. If $\mathcal{A} \models \Phi|_i[\bar{a}]$ then there exists $a_{i+1} \in A$ such that $\mathcal{A} \models \Phi|_{i+1}[\bar{a}, a_{i+1}]$. Since $\bar{a} \equiv \bar{b}$, there exists $b_{i+1} \in A$ such that $(\bar{a}, a_{i+1}) \equiv (\bar{b}, b_{i+1})$. By the induction hypothesis, it follows that $\mathcal{A} \models \Phi|_{i+1}[\bar{b}, b_{i+1}]$ and hence $\mathcal{A} \models \Phi|_i[\bar{b}]$.

The case $Q_{i+1} = \forall$ is similar. If $\mathcal{A} \models \Phi|_i[\bar{a}]$ then for each $a_{i+1} \in A$ we have $\mathcal{A} \models \Phi|_{i+1}[\bar{a}, a_{i+1}]$. Since $\bar{a} \equiv \bar{b}$, for each $b_{i+1} \in A$ there exists an $a_{i+1} \in A$ such that $[\bar{b}, b_{i+1}] \equiv [\bar{a}, a_{i+1}]$. By the induction hypothesis, it follows that for each b_{i+1}, $\mathcal{A} \models \Phi|_{i+1}[\bar{b}, b_{i+1}]$ and hence $\mathcal{A} \models \Phi|_i[\bar{b}]$. ■

Example 2.10 Continuing Examples 2.5 and 2.7, let $C(x, y) = x^2 + y^2 - 20x + 75$, $L_1(x, y) = x + y - 5$, and $L_2(x, y) = y - 14$. Let Φ be the sentence

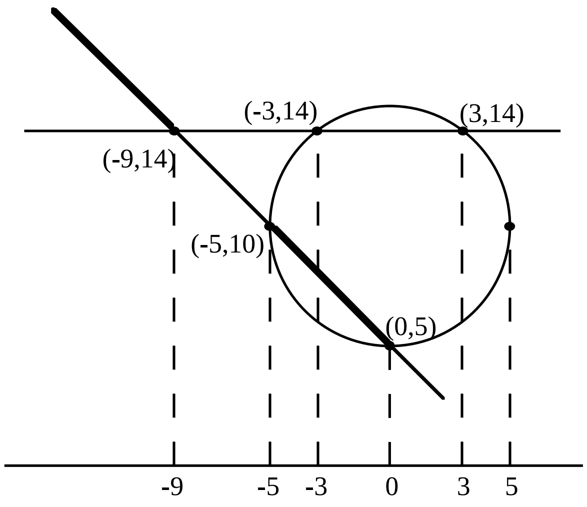

Figure 3: Equivalent points and formula satisfaction.

$(\forall x_1)(\exists x_2)(x_1+x_2-5=0\wedge(x_2-14>0\vee(x_1)^2+(x_2)^2-20x_1+75<0))$. This is illustrated in Figure 3. We have $\Phi|_0=\Phi$, $\Phi|_1=(\exists x_2)(x_1+x_2-5=0\wedge(x_2-14>0\vee(x_1)^2+(x_2)^2-20x_1+75<0))$, and $\Phi|_2=x_1+x_2-5=0\wedge(x_2-14>0\vee(x_1)^2+(x_2)^2-20x_1+75<0)$. As can be deduced from Example 2.7 the equivalence classes in A are $(-\infty,-9)$, $[-9]$, $(-9,-5)\cup(5,\infty)$, $[-5]$, $(-5,-3)$, $[-3]$, $(-3,0)$, $[0]$, $(0,3)$, $[3]$, $(3,5)$ and $[5]$. We have $\mathcal{A}\models\Phi|_1[a]$ for each $a\in(-\infty,-9)\cup(-5,-3)\cup[-3]\cup(-3,0]$.

The notion of *domain sequence* is defined next:

Definition 2.11 A sequence $D_k\subseteq D_{k+1}\subseteq\cdots\subseteq D_n$ of finite subsets of A is called a *domain sequence* if for each $k\le i<n$:

$$\forall\bar{a}\in(D_i)^i,\ \forall a_{i+1}\in A,\ \exists a'_{i+1}\in D_{i+1}:(\bar{a},a_{i+1})\equiv(\bar{a},a'_{i+1}).$$

Example 2.12 Continuing Example 2.10, from Figure 4 we see that (D_0,D_1,D_2), with

$D_0=\{0\}$,

$D_1=\{-10,-9,-7,-5,-4,-3,-2,0,1,3,4,5\}$ and

$D_2=D_1\cup\{7,14,14.5,15\}$,

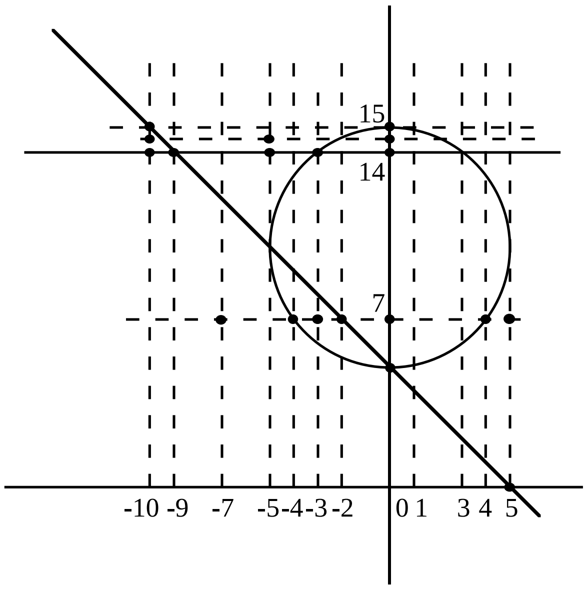

Figure 4: Domain sequence construction.

is a domain sequence.

Since $\equiv$ is of finite index (Lemma 2.8), we know:

Lemma 2.13 *For any given finite $D_k \subseteq A$, there exists a domain sequence starting from D_k.*

The following technical lemma now directly implies Theorem 2.2:

Lemma 2.14 *Let*

- *$D_k \subseteq D_{k+1} \subseteq \cdots \subseteq D_n$ be a domain sequence;*
- *$a_1, \ldots, a_k \in D_k$;*
- *$k \leq i \leq n$; and*
- *$a_j \in D_j$ for $k < j \leq i$.*

Then

$$\mathcal{A} \models \Phi|_i[a_1, \ldots, a_i] \iff (\mathcal{A}; D_{i+1}, \ldots, D_n) \models \Phi|_i[a_1, \ldots, a_i].$$

Proof. By downward induction on i. Denote $(a_1, \dots, a_i)$ by $\bar{a}$. The case $i = n$ is trivial. So assume $i < n$. We have $\Phi|_i = (Q_{i+1}x_{i+1})\Phi|_{i+1}$. Consider first the case $Q_{i+1} = \exists$. For the implication from left to right, assume $\mathcal{A} \models \Phi|_i[\bar{a}]$. Then there exists $a_{i+1} \in A$ such that $\mathcal{A} \models \Phi|_{i+1}[\bar{a}, a_{i+1}]$. According to Definition 2.11, there exists $a'_{i+1} \in D_{i+1}$ such that $(\bar{a}, a_{i+1}) \equiv (\bar{a}, a'_{i+1})$. By Lemma 2.9, we also have $\mathcal{A} \models \Phi|_{i+1}[\bar{a}, a'_{i+1}]$. By induction, $(\mathcal{A}; D_{i+2}, \dots, D_n) \models \Phi|_{i+1}[\bar{a}, a'_{i+1}]$. We can thus conclude that $(\mathcal{A}; D_{i+1}, \dots, D_n) \models \Phi|_i[\bar{a}]$.

For the implication from right to left, assume $(\mathcal{A}; D_{i+1}, \dots, D_n) \models \Phi|_i[\bar{a}]$. Then there exists $a_{i+1} \in D_{i+1}$ such that $(D_{i+2}, \dots, D_n) \models \Phi|_{i+1}[\bar{a}, a_{i+1}]$. By induction, we have $\mathcal{A} \models \Phi_{i+1}[\bar{a}, a_{i+1}]$. Since a_{i+1} is trivially in A, we can thus conclude that $\mathcal{A} \models \Phi|_i[\bar{a}]$.

Next consider the case $Q_{i+1} = \forall$. For the implication from left to right, assume $\mathcal{A} \models \Phi|_i[\bar{a}]$. Then for each $a_{i+1} \in A$ we have $\mathcal{A} \models \Phi|_{i+1}[\bar{a}, a_{i+1}]$. In particular, this holds for each $a_{i+1} \in D_{i+1}$, and by induction, we have $(\mathcal{A}; D_{i+2}, \dots, D_n) \models \Phi|_{i+1}[\bar{a}, a_{i+1}]$. We can thus conclude that $(\mathcal{A}; D_{i+1}, \dots, D_n) \models \Phi|_i[\bar{a}]$.

For the implication from right to left, assume $(\mathcal{A}; D_{i+1}, \dots, D_n) \models \Phi|_i[\bar{a}]$. Then for each $\alpha \in D_{i+1}$ we have $(\mathcal{A}; D_{i+2}, \dots, D_n) \models \Phi|_{i+1}[\bar{a}, \alpha]$, and thus, by induction, also $\mathcal{A} \models \Phi|_{i+1}[\bar{a}, \alpha]$. Now take an arbitrary $a_{i+1} \in A$. According to Definition 2.11, there exists $a'_{i+1} \in D_{i+1}$ such that $(\bar{a}, a_{i+1}) \equiv (\bar{a}, a'_{i+1})$. By Lemma 2.9, since $\mathcal{A} \models \Phi|_{i+1}[\bar{a}, a'_{i+1}]$, we also have $\mathcal{A} \models \Phi|_{i+1}[\bar{a}, a_{i+1}]$. We can thus conclude that $\mathcal{A} \models \Phi_i[\bar{a}]$. ■

Corollary 2.15 *Let Φ be a sentence $(Q_1x_1)\dots(Q_nx_n)M(x_1, \dots, x_n)$, and let $D_0 \subseteq D_1 \subseteq \dots \subseteq D_n$ be a domain sequence. Then*

$$\mathcal{A} \models \Phi \iff (\mathcal{A}; D_1, \dots, D_n) \models \Phi.$$

Proof. Set $i = k = 0$ in Lemma 2.14. ■

3 Queries on real databases

Fix a relational vocabulary σ consisting of a finite number of relation symbols S with associated arity. A *real database* $\mathcal{B}$ is a structure of type σ having the set $\mathbf{R}$ of real numbers as domain, assigning to each relation symbol S of arity a in σ a *finite* relation $S^{\mathcal{B}}$ of rank a on $\mathbf{R}$.[2] The *active domain* of $\mathcal{B}$, denoted by adom($\mathcal{B}$), is the (finite) set of all real numbers appearing in one or more relations in $\mathcal{B}$.

A *query* is a mapping from databases of type σ to true or false. A basic way of expressing queries is by *query formulas*, which are standard first-order formulas built using Boolean connectives and quantification from atomic formulas of one of the following two forms:

- $p > 0$, with p a multivariate polynomial with real coefficients;
- $S(p_1, \ldots, p_a)$, with S a relation symbol in σ of arity a, and each p_i a polynomial as in the previous item.

If $\Phi(\bar{x})$ is a query formula and $\mathcal{B}$ is a database, then the truth of Φ in $\mathcal{B}$, denoted by $\mathcal{B} \models \Phi[\bar{a}]$, is defined in the standard way. In particular, if Φ is a sentence, it expresses the query yielding true on an input database $\mathcal{B}$ iff $\mathcal{B} \models \Phi$.

Example 3.1 Assume $\sigma = \{S\}$ with $\alpha(S) = 2$. The query "do all points in S lie on a common circle?" can be expressed as

$$(\exists x_0)(\exists y_0)(\exists r)(\forall x)(\forall y)(S(x, y) \rightarrow (x - x_0)^2 + (y - y_0)^2 = r^2).$$

(Conditions of the form $p = 0$ are expressible in terms of conditions of the form $p > 0$ as $\neg(p > 0) \wedge \neg(-p > 0)$.)

The query "is there a point in S whose coordinates are greater than or equal to 1?" can be expressed as $(\exists x)(\exists y)S(x^2 + 1, y^2 + 1)$. Note that the quantifiers are naturally interpreted as ranging over the whole of $\mathbf{R}$.

Formulas that do not mention any of the relation names in σ are called *real formulas*. Let Ψ be a real formula and let all variables occurring in Ψ be among $x_1, \ldots, x_n$. Let Π be the set of all polynomials p for which the inequality $p > 0$ occurs in Ψ. For such a set Π of polynomials over the variables $x_1, \ldots, x_n$ we define:

Definition 3.2 The structure $\mathbf{R}_\Pi$ is the structure having as domain the set $\mathbf{R}$ of real numbers, and having as relations the n-ary relations $\{(r_1, \ldots, r_n) \mid p(r_1, \ldots, r_n) > 0\}$ for each $p \in \Pi$.

[2]Formally, $S^{\mathcal{B}} \subseteq \mathbf{R} \times \cdots \times \mathbf{R}$ (a times).

If Π comes from a formula Ψ as above we will also refer to $\mathbf{R}_\Pi$ as $\mathbf{R}_\Psi$. Note that Ψ can be naturally evaluated in the structure $\mathbf{R}_\Psi$.

If Φ is a query sentence and $\mathcal{B}$ is a database, then we can produce a real sentence $\Phi^{\mathcal{B}}$ in a very natural way as follows. Let $S(p_1, \ldots, p_a)$ be an atomic subformula of Φ, with S a relation symbol in σ. We know that $S^{\mathcal{B}}$ is a finite relation consisting of, say, the m tuples $\{(e_{11}, \ldots, e_{1a}), \ldots, (e_{m1}, \ldots, e_{ma})\}$. Then replace $S(p_1, \ldots, p_a)$ in Φ by $\bigvee_{i=1}^m p_1 = e_{i1} \wedge \ldots \wedge p_a = e_{ia}$. It is obvious that

$$\mathcal{B} \models \Phi \quad \Longleftrightarrow \quad \mathbf{R}_{\Phi^{\mathcal{B}}} \models \Phi^{\mathcal{B}}.$$

Now assume the query sentence Φ is in prenex normal form:

$$(Q_1 x_1) \ldots (Q_n x_n) M(x_1, \ldots, x_n). \tag{$\dagger$}$$

If $\mathcal{B}$ is a database and $D_1, \ldots, D_n$ are subsets of $\mathbf{R}$, then we say that Φ is satisfied on $(\mathcal{B}; D_1, \ldots, D_n)$, written $(\mathcal{B}; D_1, \ldots, D_n) \models \Phi$, if Φ evaluates to true on $\mathcal{B}$ when we let each quantifier Q_i range over D_i only, rather than over the whole of $\mathbf{R}$.

Corollary 2.15 immediately implies the following:

Theorem 3.3 *Let Φ be a query sentence as in ($\dagger$) above and let $\mathcal{B}$ be a real database. For each domain sequence $D_0 \subseteq D_1 \subseteq \cdots \subseteq D_n$ in the context of the structure $\mathbf{R}_{\Phi^{\mathcal{B}}}$,*

$$\mathcal{B} \models \Phi \quad \Longleftrightarrow \quad (\mathcal{B}; D_1, \ldots, D_n) \models \Phi.$$

When we choose $D_0 = \text{adom}(\mathcal{B})$, this theorem can be viewed as the analog in the real case of the Aylamazyan et al. theorem [5] mentioned in the Introduction.

4 The linear case

In this section, we focus on *linear* queries, expressed by query sentences in which all occurring polynomials are linear. We prove that each linear query is expressible by a linear query sentence wherein the quantifiers range over the active domain of the input database only. Thereto, we introduce a particular way to construct a domain sequence starting with the active domain of a database, based on Gaussian elimination. We then show that this construction can be simulated in a uniform (i.e., database-independent) way by a linear query formula.

Let Π be a set of linear polynomials on the variables $x_1, \ldots, x_n$. Recall Definition 3.2 of the structure $\mathbf{R}_\Pi$. Within the context of this structure we can consider equivalence of points in $\mathbf{R}^i$, for $i \leq n$, as defined in Definitions 2.4 and 2.6.

Each polynomial $p \in \Pi$ is of the form $c_0^p + \sum_{j=1}^n c_j^p x_j$. We define a sequence $\Pi_n, \ldots, \Pi_1$ of linear polynomials inductively as follows:

Definition 4.1 $\Pi_n = \Pi$, and for $i < n$,

$$\Pi_i = \{p \in \Pi_{i+1} \mid c_{i+1}^p = 0\} \cup \{p \cdot c_{i+1}^q - q \cdot c_{i+1}^p \mid p, q \in \Pi_{i+1},\ c_{i+1}^p \neq 0 \neq c_{i+1}^q\}.$$

In words, each Π_i is a set of linear polynomials over $x_1, \ldots, x_i$ obtained from Π_{i+1} by Gaussian elimination.

In the next proposition, equivalence of points in $\mathbf{R}^i$ with respect to $\mathbf{R}_\Pi$ will be characterized in terms of the polynomials in Π_i. Thereto we need an easy-to-prove lemma:

Lemma 4.2 *Let α_1, α_2, β_1, and β_2 be elements from some densely ordered domain. The following are equivalent:*

1. *For $(i,j) \in \{(1,2), (2,1)\}$,*

$$\alpha_i > \alpha_j \quad \Leftrightarrow \quad \beta_i > \beta_j.$$

2. *For each α there exists β such that for $i = 1, 2$,*

$$\alpha_i > \alpha \quad \Leftrightarrow \quad \beta_i > \beta,$$

and conversely, for each β there exists α such that the same holds.

Proposition 4.3 *Let $1 \leq i \leq n$ and let $\bar{a}, \bar{b} \in \mathbf{R}^i$. Then $\bar{a}$ and $\bar{b}$ are equivalent with respect to $\mathbf{R}_\Pi$ if and only if for each polynomial p in Π_i,*

$$p(\bar{a}) > 0 \quad \Leftrightarrow \quad p(\bar{b}) > 0.$$

Proof. By downward induction on i. The case $i = n$ is just the definition of equivalence of n-tuples. So assume $i < n$. Let $\bar{a} \equiv \bar{b}$. Then for each a_{i+1} there is a b_{i+1} such that $(\bar{a}, a_{i+1}) \equiv (\bar{b}, b_{i+1})$ (and conversely). Equivalently, by induction, for each a_{i+1} there is a b_{i+1} such that for each polynomial p in Π_{i+1}, $p(\bar{a}, a_{i+1}) > 0 \Leftrightarrow p(\bar{b}, b_{i+1}) > 0$. If $c^p_{i+1} = 0$ then $p \in \Pi_i$ and we get $p(\bar{a}) > 0 \Leftrightarrow p(\bar{b}) > 0$; this deals with the first kind of elements of Π_i.

For the other kind of elements of Π_i, consider $p, q \in \Pi_{i+1}$ with $c^p_{i+1} \neq 0 \neq c^q_{i+1}$. From the above, for each a_{i+1} there is a b_{i+1} such that

$$(c^p_0 + \sum_{j=1}^{i} c^p_j a_j)/c^p_{i+1} > -a_{i+1} \quad \Leftrightarrow \quad (c^p_0 + \sum_{j=1}^{i} c^p_j b_j)/c^p_{i+1} > -b_{i+1}$$

and

$$(c^q_0 + \sum_{j=1}^{i} c^q_j a_j)/c^q_{i+1} > -a_{i+1} \quad \Leftrightarrow \quad (c^q_0 + \sum_{j=1}^{i} c^q_j b_j)/c^q_{i+1} > -b_{i+1}.$$

Conversely, for each b_{i+1} there is an a_{i+1} such that the same holds. By Lemma 4.2 we thus deduce

$$(c^p_0+\sum_{j=1}^{i} c^p_j a_j)/c^p_{i+1} > (c^q_0+\sum_{j=1}^{i} c^q_j a_j)/c^q_{i+1} \quad \Leftrightarrow \quad (c^p_0+\sum_{j=1}^{i} c^p_j b_j)/c^p_{i+1} > (c^q_0+\sum_{j=1}^{i} c^q_j b_j)/c^q_{i+1}$$

and hence

$$(c^p_0+\sum_{j=1}^{i} c^p_j a_j)\cdot c^q_{i+1} > (c^q_0+\sum_{j=1}^{i} c^q_j a_j)\cdot c^p_{i+1} \quad \Leftrightarrow \quad (c^p_0+\sum_{j=1}^{i} c^p_j b_j)\cdot c^q_{i+1} > (c^q_0+\sum_{j=1}^{i} c^q_j b_j)\cdot c^p_{i+1}$$

or

$$(p \cdot c^q_{i+1} - q \cdot c^p_{i+1})(\bar{a}) > 0 \quad \Leftrightarrow \quad (p \cdot c^q_{i+1} - q \cdot c^p_{i+1})(\bar{b}) > 0,$$

which, by definition of Π_i, is what had to be proven. This argument for the 'only-if' implication can simply be reversed to prove the 'if' implication. ■

Now let Φ be a linear query sentence $(Q_1 x_1) \ldots (Q_n x_n) M$ in prenex normal form, and let $\mathcal{B}$ be a database. Recall the definition of the real formula $\Phi^{\mathcal{B}}$ described in the previous section; note that since Φ is linear, $\Phi^{\mathcal{B}}$ is linear as well.

Definition 4.4 Fix Π to be the set of all polynomials occurring in $\Phi^{\mathcal{B}}$ plus all those of the form $p_i - e$, where p_i occurs in some atomic formula $S(p_1, \ldots, p_a)$ of Φ and $e \in \mathrm{adom}(\mathcal{B})$.

We can then consider the sequence $\Pi = \Pi_n, \ldots, \Pi_1$ as in Definition 4.1. For what follows it is important to note that, since Π contains at least all polynomials occurring in $\Phi^{\mathcal{B}}$, equivalence of points w.r.t. $\mathbf{R}_\Pi$ implies equivalence w.r.t. the structure $\mathbf{R}_{\Phi^{\mathcal{B}}}$ mentioned in Theorem 3.3.

Example 4.5 Let $\Phi = (\forall x_1)(\exists x_2)(S(x_1 + x_2 + 1) \wedge (x_1 + 2x_2 + 2 = 0))$. Then $\Pi = \Pi_2 = \{1 - e + x_1 + x_2 \mid e \in \mathrm{adom}(\mathcal{B})\} \cup \{2 + x_1 + 2x_2\}$ and $\Pi_1 = \{-2e + x_1 \mid e \in \mathrm{adom}(\mathcal{B})\}$.

We observe:

Lemma 4.6 *Let $1 \le i \le n$. Then Π_i is a finite union of sets of the form*

$$\{c_0 + \sum_{j=1}^{2^{(n-i)}} d_j e_j + \sum_{j=1}^{i} c_j x_j \mid e_1, \ldots, e_{2^{(n-i)}} \in \mathrm{adom}(\mathcal{B})\}.$$

Both the number of these sets and the coefficients c_i and d_i for each set do not depend on the particular database $\mathcal{B}$.

Proof. By downward induction on i. The base case $i = n$ is clear since $\Pi_n = \Pi$ is clearly of the good form. So assume $i < n$. By definition, Π_i is a union of two sets:

$$\{p \in \Pi_{i+1} \mid c^p_{i+1} = 0\}$$

and

$$\{p \cdot c^q_{i+1} - q \cdot c^p_{i+1} \mid p, q \in \Pi_{i+1},\ c^p_{i+1} \neq 0 \neq c^q_{i+1}\}.$$

By the induction hypothesis, the first set is clearly of the good form. Also by the induction hypothesis, the second set is a finite union of sets of the form

$$\{c'_{i+1}\left(c_0 + \sum_{j=1}^{2^{(n-i-1)}} d_j e_j + \sum_{j=1}^{i+1} c_j x_j\right) - c_{i+1}\left(c'_0 + \sum_{j=1}^{2^{(n-i-1)}} d'_j e'_j + \sum_{j=1}^{i+1} c'_j x_j\right) \mid$$
$$e_1, \ldots, e_{2^{(n-i-1)}}, e'_1, \ldots, e'_{2^{(n-i-1)}} \in \mathrm{adom}(\mathcal{B})\}.$$

After simplification this is readily seen to be also of the good form. ■

We are now in a position to define a particular domain sequence with respect to the structure $\mathbf{R}_{\Phi^{\mathcal{B}}}$, based on the sequence $\Pi_1, \ldots, \Pi_n$. The sequence is inductively constructed: D_0 is empty, and D_i $(i > 0)$ is constructed as follows: first consider the set E_i of all the i-th coordinates of the i-dimensional points that are in a hyperplane of Π_i and whose first $i-1$ coordinates are in D_{i-1}. Add D_{i-1} to E_i, resulting in D'_i. Finally, to be sure to obtain a point in every equivalence class, we add the mean value of every pair of elements of D'_i, as well as every element increased by one and every element decreased by one, resulting in D_i. Formally, we define:

Definition 4.7 The *linear sequence on $\mathcal{B}$ with respect to* Φ is the sequence $\emptyset = D_0 \subseteq D_1 \subseteq \cdots \subseteq D_n$ inductively defined as follows: for $1 \leq i \leq n$, D_i equals

$$D'_i \cup \{y \mid (\exists y_1, y_2) \in D'_i : y = \frac{y_1 + y_2}{2} \vee y = y_1 - 1 \vee y = y_1 + 1\},$$

where D'_i is $D_{i-1} \cup E_i$ with

$$E_i = \Big\{-c_0/c_i - \sum_{j=1}^{2^{(n-i)}} (d_j/c_i)e_j - \sum_{j=1}^{i-1}(c_j/c_i)y_j \mid c_0 + \sum_{j=1}^{2^{(n-i)}} d_j e_j + \sum_{j=1}^{i} c_j x_j \in \Pi_i,$$
$$c_i \neq 0,\ y_1, \ldots, y_{i-1} \in D_{i-1},\ e_1, \ldots, e_{2^{(n-i)}} \in \text{adom}(\mathcal{B})\Big\}.$$

Example 4.8 In Example 4.5 we have

$$D'_1 = \{2e_1 \mid e_1 \in \text{adom}(\mathcal{B})\},$$
$$D_1 = \big\{2e_1 + \eta \mid \eta \in \{-1, 0, 1\}\big\},$$

and

$$\begin{aligned} E_2 = \Big\{&-1 + e_3 - (2e_1 + \eta), \\ &-1 + e_3 - (e_1 + e_2), \\ &-1 - (2e_1 + \eta)/2, \\ &-1 - (e_1 + e_2)/2 \mid \\ &\qquad e_1, e_2, e_3 \in \text{adom}(\mathcal{B}),\ \eta \in \{-1, 0, 1\}\Big\}. \end{aligned}$$

Proposition 4.9 *The linear sequence on $\mathcal{B}$ with respect to Φ is a domain sequence with respect to $\mathbf{R}_{\Phi^{\mathcal{B}}}$.*

Proof. According to Definition 2.11, we must show for each $1 \leq i \leq n$ that

$$\forall \bar{a} \in (D_{i-1})^{i-1},\ \forall a_i \in \mathbf{R},\ \exists a'_i \in D_i : (\bar{a}, a_i) \equiv (\bar{a}, a'_i).$$

So, let $\bar{a} \in (D_{i-1})^{i-1}$ and assume $a_i \notin D_i$. Consider the definition of D_i in terms of $D'_i = D_{i-1} \cup E_i$ from Definition 4.7 above. We distinguish the following possibilities for a_i:

1. $a_i < \min(E_i)$; then put $a'_i := \min(E_i) - 1$;

2. $a_i > \max(E_i)$; then put $a'_i := \max(E_i) + 1$;

3. $\min(E_i) < a_i < \max(E_i)$; then put $a'_i := (e_1 + e_2)/2$, where e_1 is the maximal element in E_i such that $e_1 < a_i$, and e_2 is the minimal element such that $a_i < e_2$.

It is obvious that $a'_i \in D_i$; moreover, we invite the reader to convince himself that from the way E_i is defined, it follows that all polynomials in Π_i have the same sign on $(\bar{a}, a_i)$ and $(\bar{a}, a'_i)$. Hence, by Proposition 4.3, the proposition follows. ■

After one final lemma we will be able to state and prove the main result of this section:

Lemma 4.10 *For each $0 \leq i \leq n$ there exists a finite set P of linear polynomials such that for each database $\mathcal{B}$, the i-th member D_i of the linear sequence on $\mathcal{B}$ with respect to Φ equals $\{p(y_1, \ldots, y_z) \mid y_1, \ldots, y_z \in \mathrm{adom}(\mathcal{B}) \wedge p \in P\}$, with z independent of $\mathcal{B}$.*

Proof. By induction on i. The case $i = 0$ is trivial since $D_0 = \emptyset$ (put $P := \emptyset$). So assume $i > 0$. The definition of D_i in terms of D'_i in Definition 4.7 is clearly of the form $D_i = \{p(y_1, y_2) \mid y_1, y_2 \in D'_i \wedge p \in P'\}$ where P' consists of the four polynomials $(y_1 + y_2)/2$, $y_1 - 1$, $y_1 + 1$, and y_1. We have $D'_i = D_{i-1} \cup E_i$, where E_i is clearly of the form $\{p(y_1, \ldots, y_{i-1}, e_1, \ldots, e_{2(n-i)}) \mid y_1, \ldots, y_{i-1} \in D_{i-1} \wedge e_1, \ldots, e_{2(n-i)} \in \mathrm{adom}(\mathcal{B}) \wedge p \in P''\}$, for some P'', and by induction, D_{i-1} is of the form $\{p(y_1, \ldots, y_z) \mid y_1, \ldots, y_z \in \mathrm{adom}(\mathcal{B}) \wedge p \in P'''\}$, for some P'''. By combining these expressions using a tedious but straightforward substitution process, we obtain the desired form for D_i. ■

Theorem 4.11 *For each linear query sentence Φ there is a linear query sentence Ψ, which can be effectively constructed from Φ, such that for each database $\mathcal{B}$, $\mathcal{B} \models \Phi$ if and only if $\mathcal{B} \models_{\text{adom}} \Psi$, where $\models_{\text{adom}}$ denotes that the quantifiers in Ψ range over the active domain of the database only.*

Proof. Let $\emptyset \subseteq D_1 \subseteq \cdots \subseteq D_n$ be the linear sequence on $\mathcal{B}$ with respect to Φ. By Theorem 3.3 and Proposition 4.9, we know that $\mathcal{B} \models \Phi$ iff $(\mathcal{B}; D_1, \ldots, D_n) \models \Phi$. We can write the latter explicitly as $\mathcal{B} \models (Q_1 x_1 \in D_1) \ldots (Q_n x_n \in D_n) M(x_1, \ldots, x_n)$. From Lemma 4.10 we know that D_1 can be written as $\{p(y_1, \ldots, y_z) \mid y_1, \ldots, y_z \in \text{adom}(\mathcal{B}) \wedge p \in P\}$. If Q_1 is $\exists$, we can rewrite the above formula as

$$\mathcal{B} \models (\exists y_1) \ldots (\exists y_z) \bigvee_{p \in P} (Q_2 x_2 \in D_2) \ldots (Q_n x_n \in D_n) M(p(y_1, \ldots, y_z), x_2, \ldots, x_n),$$

where each $(\exists y_i)$ ranges only over $\text{adom}(\mathcal{B})$. If Q_1 is $\forall$ we have

$$\mathcal{B} \models (\forall y_1) \ldots (\forall y_z) \bigwedge_{p \in P} (Q_2 x_2 \in D_2) \ldots (Q_n x_n \in D_n) M(p(y_1, \ldots, y_z), x_2, \ldots, x_n).$$

By replacing $Q_2, \ldots, Q_n$ in a similar manner, we obtain the desired sentence Ψ.

If $\text{adom}(\mathcal{B})$ is empty then the above strategy will not work. However, the sentence Ψ obtained above can be modified so as to test for this special case, and if this test succeeds, a fixed truth value can be returned. This fixed truth value is the result of evaluating $(\mathcal{B}_\emptyset \models \Phi)$, where $\mathcal{B}_\emptyset$ denotes the database with empty active domain.[3] ■

5 Generic queries

Two databases $\mathcal{B}$ and $\mathcal{B}'$ over the same relational signature σ are called *isomorphic* if there is a bijection $\rho : \text{adom}(\mathcal{B}) \to \text{adom}(\mathcal{B}')$ such that $\rho(S^{\mathcal{B}}) = S^{\mathcal{B}'}$ for each relation symbol S in σ. A query which yields the same result on isomorphic databases is called *generic*.

[3] An exception occurs when the signature σ contains relation symbols of arity zero. In this case, there is no unique $\mathcal{B}_\emptyset$ but rather a fixed finite number of them. The sentence can test which one it is dealing with and return the appropriate truth value.

For example, assume that σ consists of a single binary relation symbol S. Databases of type σ can be viewed as finite directed graphs whose nodes are real numbers. Of course, any query expressed in the language $\mathcal{L}$ of pure first-order sentences over σ (i.e., not containing any polynomial inequalities) is generic. Other examples of generic queries are "is the graph connected?" or "is the number of edges even?".

In the language $\mathcal{L}^<$ consisting of those query sentences where all inequalities are of the simple form $x < y$ (with x and y variables),[4] non-generic queries can be easily expressed, such as $(\forall x)(\forall y)S(x,y) \rightarrow x < y$. As pointed out in the Introduction, however, there are generic queries expressible in $\mathcal{L}^<$ but not in $\mathcal{L}$. We have been able to prove that there is no similar gain in expressiveness when moving from $\mathcal{L}^<$ to full linear query sentences:

Theorem 5.1 *For each linear query sentence Φ expressing a generic query there is a query sentence Ψ in $\mathcal{L}^<$, which can be effectively constructed from Φ, such that for each database $\mathcal{B}$, $\mathcal{B} \models_{\mathrm{adom}} \Phi$ if and only if $\mathcal{B} \models_{\mathrm{adom}} \Psi$.*

As in Theorem 4.11, $\models_{\mathrm{adom}}$ denotes that quantifiers range over the active domain only; we know by Theorem 4.11 that this active-domain interpretation is without loss of generality.

We next present an elementary proof of Theorem 5.1 based on three lemmas and one auxiliary definition.

The following fact is easy to prove:

Lemma 5.2 *Let $q(x) = \sum_{i=0}^{d} a_i x^i$ be a polynomial with real coefficients, in one variable, of degree d ($a_d \neq 0$). Let*

$$r > d \cdot \frac{\max_{0 \leq i \leq d} |a_i|}{\min_{\substack{0 \leq j \leq d \\ a_j \neq 0}} |a_j|}.$$

Then $q(r)$ has the same sign as a_d.

We define:

[4] As an aside, we would like the reader to note that Theorem 4.11 specializes to query sentences in $\mathcal{L}^<$. This follows from general results in [7], but can also be proven in a direct way using an argument similar to our proof of Theorem 4.11.

Definition 5.3 Let $p(x_1,\ldots,x_n) = \sum_{i=1}^n b_i x_i + b_0$ be a linear polynomial with real coefficients in n variables. We associate with p a function $\Xi_p : \mathbf{R}^n \to \mathbf{R}$ as follows. Consider $\bar{y} = (y_1,\ldots,y_n) \in \mathbf{R}^n$. Associate a "weight" $W_p^{\bar{y}}(y_i)$ to each y_i by

$$W_p^{\bar{y}}(y_i) := \sum_{\substack{1\le j\le n\\ y_j = y_i}} b_j.$$

If all weights are zero, define $\Xi_p(y_1,\ldots,y_n) := b_0$. Otherwise, define

$$\Xi_p(y_1,\ldots,y_n) := W_p^{\bar{y}}(y_M),$$

where y_M is maximal with non-zero weight, i.e., $y_m = \max\{y_i \mid 1 \le i \le n,\ W_p^{\bar{y}}(y_i) \neq 0\}$.

Example 5.4 We illustrate the above definition with three examples:

1. Let $p(x_1,x_2,x_3) = x_1 - 5x_2 + 3x_3 - 8$. Then
   - $\Xi_p(7,3,9) = 3$;
   - $\Xi_p(7,9,3) = -5$.
2. Let $p(x_1,x_2) = 2x_1 - 2x_2 + 5$. Then $\Xi_p(5,5) = 5$.
3. Let $p(x_1,\ldots,x_7) = 3x_1 - 3x_2 + 6x_3 - 6x_4 + 5x_5 + x_6 - 6x_7$. Then
   - $\Xi_p(4,4,2,2,2,2,2) = 0$;
   - $\Xi_p(5,5,4,4,3,2,1) = 5$;
   - $\Xi_p(5,5,4,4,1,2,3) = -6$.

The relevance of Ξ_p stems from the following observation:

Lemma 5.5 *Let $p(x_1,\ldots,x_n) = \sum_{i=1}^n b_i x_i + b_0$ and let $s > 0$ be a natural number. Then there exists a number α_p such that for all $\beta > \alpha_p$ and for any sequence of integers $\bar{z} = z_1,\ldots,z_n \in \{0,\ldots,s\}$,*

$$p(\beta^{z_1},\ldots,\beta^{z_n}) > 0 \iff \Xi_p(z_1,\ldots,z_n) > 0.$$

Proof. Note that

$$p(\beta^{z_1}, \ldots, \beta^{z_n}) = \sum_{i=1}^{n} b_i \beta^{z_i} + b_0 = \sum_j W_p^{\bar{z}}(z_j)\beta^{z_j} + b_0,$$

where j in the latter sum ranges over a set of indices consisting of one j for each distinct value z_j. Hence, the value $p(\beta^{z_1}, \ldots, \beta^{z_n})$ can be viewed as the value of a univariate polynomial q in β. The highest-degree coefficient of q is $\Xi_p(z_1, \ldots, z_n)$. The degree of q is $\max_j z_j$. Hence, if we take

$$\alpha_p = s \cdot \frac{\max_B |B|}{\min_{|B| \neq 0} |B|},$$

where B ranges over all partial sums of b_i's, then any $\beta > \alpha_p$ satisfies the condition of Lemma 5.2 and thus for such β, $p(\beta^{z_1}, \ldots, \beta^{z_n}) = q(\beta)$ has the same sign as $\Xi_p(z_1, \ldots, z_n)$. ■

As a last lemma towards the proof of Theorem 5.1 we note:

Lemma 5.6 *For a fixed polynomial p as above, the predicate $\Xi_p(y_1, \ldots, y_n) > 0$ can be expressed by a formula $\xi_p(y_1, \ldots, y_n)$ in $\mathcal{L}^<$.*

Proof. The crucial observation is that the value of $\Xi_p(y_1, \ldots, y_n)$ depends not on the actual values of the y_i's but only on their relative positions (in model-theoretic terms, the *order type* of $y_1, \ldots, y_n$). The number of possible order types (n being fixed) is finite, so the formula ξ_p simply consists of the disjunction of those order types for which the value $\Xi_p(y_1, \ldots, y_n)$ is positive. ■

Example 5.7 Let $p(x_1, x_2) = 2x_1 - 2x_2 + 5$. Then $\Xi_p(y_1, y_2) > 0$ is expressed by $y_1 = y_2 \vee y_1 > y_2$.

Proof of Theorem 5.1. Replace in Φ every inequality $p(x_1, \ldots, x_n) > 0$ by the formula $\xi_p(x_1, \ldots, x_n) > 0$ of Lemma 5.6. In this way we obtain a query sentence Ψ in $\mathcal{L}^<$. We still have to show that $\mathcal{B} \models_{\text{adom}} \Phi$ iff $\mathcal{B} \models_{\text{adom}} \Psi$. Let s be the cardinality of adom($\mathcal{B}$). Each polynomial p occurring in Φ has an associated lower bound α_p of Lemma 5.5. Let β be larger than any of these α_p's and let ρ be an order-preserving (i.e., monotone) bijection from $\mathrm{adom}(\mathcal{B})$ to $\{\beta, \beta^2, \ldots, \beta^s\}$. Then

$$\begin{aligned}\mathcal{B} \models_{\mathrm{adom}} \Phi \quad &\Leftrightarrow \quad \rho(\mathcal{B}) \models_{\mathrm{adom}} \Phi \\ &\Leftrightarrow \quad \rho(\mathcal{B}) \models_{\mathrm{adom}} \Psi \\ &\Leftrightarrow \quad \mathcal{B} \models_{\mathrm{adom}} \Psi.\end{aligned}$$

The first equivalence holds since Φ is generic, the second holds by Lemma 5.5, and the third holds since ρ is monotone and Ψ is a formula in $\mathcal{L}^<$; the truth of formulas in $\mathcal{L}^<$ is preserved under order-preserving isomorphisms. ■

Inspection of the above proof shows that Theorem 5.1 can be sharpened a little bit. Indeed, of the given that Φ expresses a generic query, we actually use only that this query yields the same result on databases that are isomorphic via an order-preserving, rather than an arbitrary, bijection.

We can conclude that all generic queries that are not expressible in $\mathcal{L}^<$ are not expressible as a linear query either (by Theorem 4.11, both under the active-domain interpretation as under the natural interpretation). In particular this holds for the queries, already mentioned at the beginning of this Section, of testing for connectivity or even cardinality of a finite graph over the reals:

Corollary 5.8 *Graph connectivity and even cardinality are not expressible by linear query sentences.*

Proof. By the above it suffices to show that these queries are not expressible in $\mathcal{L}^<$ with quantification on the active domain. But this is well-known [1]. ■

Grumbach, Su, and Tollu [13] have also obtained inexpressibility results for linear queries, using complexity arguments. In particular, they showed that in the context of the rationals $\mathbf{Q}$ rather than the reals $\mathbf{R}$, linear queries are in the complexity class AC_0, while even cardinality and connectivity are not. We would like to point out (as is readily verified) that our technical development applies equally well to the rationals.

6 Concluding remarks

After we presented the original ideas contained in the present paper at a conference [19], several researchers have been able to generalize our results:

- In this paper we have considered databases and queries over the structure $\mathbf{R}$ of the reals. One can do the same for any arbitrary fixed infinite "universe-structure". Using the Ehrenfeucht-Mostowski theorem on first-order indiscernibles, Otto and Van den Bussche [17] have shown that Theorem 5.1 generalizes from $\mathbf{R}$ to any arbitrary fixed infinite structure.

- Benedikt et al. [6] have generalized Theorem 5.1 in two senses: again to more universes than just the reals, and more importantly, to quantification on the whole universe rather than on the active domain only. One consequence of the results in [6] is a generalization of our Corollary 5.8: graph connectivity and even cardinality are not expressible by any (not necessarily linear) real query sentence, both under the natural interpretation as under the active-domain interpretation.

- Belegradek, Stolboushkin and Taitslin [20, 21] have generalized Theorem 5.1 in another sense: instead of finite databases they considered possibly infinite databases definable by real formulas involving only simple inequalities.

- Benedikt and Libkin [7] have shown that Theorem 4.11 holds in any densely ordered structure that satisfies the property of *o-minimality* and admits elimination of quantifiers. In particular, their result implies that Theorem 4.11 generalizes to the non-linear case (since the structure of the reals with addition and multiplication admits elimination of quantifiers).

In constrast to the proofs of these generalizations, the proofs of our results as given in the present paper are elementary and constructive (the results in [20] are also constructive).

Acknowledgment We are grateful to Bart Kuijpers for his careful reading of earlier drafts of the material presented in this paper, to Alex Stolboushkin for helpful comments on a first presentation of our results, and to an anonymous referee for pointing out a mistake in the submitted draft of this paper.

References

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases.* Addison-Wesley, 1994.

[2] A.V. Aho and J.D. Ullman. Universality of data retrieval languages. *Proceedings ACM Symposium on Principles of Programming Languages*, 1979.

[3] D.S. Arnon. Geometric reasoning with logic and algebra. *Artificial Intelligence* 37, 37–60 (1988).

[4] D.S. Arnon, G.E. Collins and S. McCallum. Cylindrical algebraic decomposition, I: The basic algorithm. *SIAM Journal on Computing* 13, 865–877, 1984.

[5] A.K. Aylamazyan, M.M. Gilula, A.P. Stolboushkin, and G.F. Schwartz. Reduction of the relational model with infinite domains to the case of finite domains. *Doklady Akademii Nauk SSSR*, 286(2):308–311, 1986. In Russian.

[6] M. Benedikt, G. Dong, L. Libkin, and L. Wong. Relational expressive power of constraint query languages. *Proceedings 15th ACM Symposium on Principles of Database Systems*, 1996.

[7] M. Benedikt and L. Libkin. On the structure of first-order queries in constraint query languages. *Proceedings 11th IEEE Symposium on Logic in Computer Science*, 1996.

[8] F. Benhamon and A. Colmerauer, editors. *Constraint Logic Programming: Selected Research.* The MIT Press, 1993.

[9] J. Bochnak, M. Coste, and M.-F. Roy. *Géometrie Algébrique Réelle.* Springer-Verlag, 1987.

[10] A.K. Chandra and D. Harel. Computable queries for relational data bases. *Journal of Computer and System Sciences* 21(2):156–178 (1980).

[11] G.E. Collins. Quantifier elimination for real closed fields by cylindrical algebraic decomposition. *Lecture Notes in Computer Science* 33, 134–183 (1975).

[12] H.B. Enderton. *A mathematical introduction to logic*. Academic Press, 1972.

[13] S. Grumbach, J. Su, and C. Tollu. Linear constraint databases. In D. Leivant, editor, *Logic and Computational Complexity*, *Lecture Notes in Computer Science* 960, 1995.

[14] R. Hull and J. Su. Domain independence and the relational calculus. *Acta Informatica* 31, 513–524 (1994).

[15] P.C. Kanellakis, G.M. Kuper, and P. Revesz. Constraint query languages. *Journal of Computer and System Sciences*, 51(1)26–52, 1995.

[16] G.M. Kuper. On the expressive power of the relational calculus with arithmetic constraints. *Lecture Notes in Computer Science* 470, 202–211, 1990.

[17] M. Otto and J. Van den Bussche. First-order queries on databases embedded in an infinite structure. *Information Processing Letters*, to appear.

[18] J. Paredaens, J. Van den Bussche, and D. Van Gucht. Towards a theory of spatial database queries. *Proceedings 13th ACM Symposium on Principles of Database Systems*, 1994.

[19] J. Paredaens, J. Van den Bussche, and D. Van Gucht. First-order queries on finite structures over the reals (extended abstract). *Proceedings 10th IEEE Symposium on Logic in Computer Science*, 1995.

[20] A.P. Stolboushkin and M.A. Taitslin. Linear vs. order constraints over rational databases. *Proceedings 15th ACM Symposium on Principles of Database Systems*, 1996.

[21] O.V. Belegradek, A.P. Stolboushkin, and M.A. Taitslin. On order-generic queries. DIMACS Technical Report 96-01, 1996.

[22] L. Van Den Dries. Alfred Tarski's elimination theory for real closed fields. *Journal of Symbolic Logic* 53, 7–19 (1988).