On Query Languages for Linear Queries Definable with Polynomial Constraints

Luc Vandeurzen[1], Marc Gyssens[1], Dirk Van Gucht[2]

[1] Dept. WNI, University of Limburg, B-3590 Diepenbeek, Belgium.
lvdeurze@luc.ac.be, gyssens@charlie.luc.ac.be.
[2] Computer Science Dept., Indiana University, Bloomington, IN 47405-4101, USA.
vgucht@cs.indiana.edu.
WWW: http://www.luc.ac.be/theocomp.

Abstract. It has been argued that the linear database model, in which semi-linear sets are the only geometric objects, is very suitable for most spatial database applications. For querying linear databases, the language FO + linear has been proposed. We present both negative and positive results regarding the expressiveness of FO+linear. First, we show that the dimension query is definable in FO + linear, which allows us to solve several interesting queries. Next, we show the non-definability of a whole class of queries that are related to sets not definable in FO+linear. This result both sharpens and generalizes earlier results independently found by Afrati et al. and the present authors, and demonstrates the need for more expressive linear query languages if we want to sustain the desirability of the linear database model. In this paper, we show how FO + linear can be strictly extended within FO + poly in a safe way. Whether any of the proposed extensions is complete for the linear queries definable in FO + poly remains open. We do show, however, that it is undecidable whether an expression in FO + poly induces a linear query.

1 Introduction

Following the seminal work by Kuper, Kanellakis, and Revesz [20] on constraint query languages with polynomial constraints (FO + poly), various researchers have introduced geometric database models and query languages within this framework [18, 22]. These researchers have studied the desirability of their models for database applications involving geometric data objects, as well as the expressiveness of the proposed geometric query languages.

An important database model that has recently been studied in this context is the *linear spatial database model* [2, 3, 26], which we adopt in this paper. The linear model allows users to define relational databases, which may, besides conventional data, contain linear geometric data objects. Formally, these objects are so-called *semi-linear sets*, which can be defined in first-order logic over the reals with addition. The class of semi-linear sets suffices for the majority of applications encountered in GIS, geometric modeling, and spatial and temporal databases [6, 23]. Furthermore, data structures and algorithms have been developed to efficiently implement a wide variety of operations on these sets [4, 11, 17, 24].

Associated with the linear model is the concept of *linear query*, which is a mapping from linear databases to linear databases. Because the linear database model is a sub-model of the polynomial database model, it is in principle possible to use the query language FO+poly to define natural linear queries, and, in fact, a vast number of important linear queries can indeed be so defined. Of course, not every query defined by an FO + poly formula induces a linear query, and, as is shown in Section 5, it is even undecidable whether an FO + poly formula induces a linear query.

Faced with this reality, several researchers [3, 26] have proposed the query language FO+linear as a natural query language to accompany the linear model. The FO+linear language is the sub-language of FO+poly wherein the polynomial constraints are restricted to *linear constraints*. Many important linear queries can be defined in FO+linear. Section 3 reviews some known results in this respect and presents some new ones. The most surprising of those is the definability of the *dimension query* which returns the topological dimension of a semi-linear set. This definability result allows us to solve some important practical queries. In particular, it follows that the *interval*-query, i.e., "Is the semi-linear set an interval?" and the *line*-query, i.e., "Is the semi-linear set a line?" are definable in FO + poly .

Unfortunately, FO + linear is *incomplete* for the linear queries definable in FO+poly as was recently shown by Afrati et al. [2]. The counter-example used by Afrati et al., however, is a technical one, and does not, in our view, adequately reveal the weaknesses of FO + linear as a language to define linear queries definable in FO + poly. In Section 4, building on the work of Afrati et al. [2] and on earlier work of the present authors [26], we show that natural FO + poly-definable linear queries, such as the query that yields the convex hull of a semi-linear set, are *not* FO + linear-definable. The conclusion we draw from these negative results is that, though FO + linear provides a good lower bound for the FO + poly-definable linear queries, FO + linear is too limited in expressiveness to be considered fully adequate to accompany the linear model.

This brings us to the last major topic of this paper. In Section 5, we introduce query languages that can only express FO+poly-definable linear queries, but that are strictly more expressive than FO+linear. These languages have some affinity with some operational languages that have been introduced in spatial database models, but that do not fall within the framework of Kuper, Kanellakis, and Revesz [20]. It is presently an open problem whether any of the query languages we propose in Section 5 is complete for the FO + poly-definable linear queries, though we conjecture this is not the case.

2 Preliminaries

In this paper we focus on the linear spatial database model as proposed in [26]. To put this paper into better perspective we briefly review some of the material in [26].

The linear model is extracted from the polynomial model, which is based on *real formulae*, i.e, formulae in first order logic over $(\mathbf{R}, \leq, +, *, 0, 1)$. Due to the work of Tarski [21], it is well known that this first order logic over the reals with inequality, addition and multiplication is a decidable theory. Every real formula $\varphi(x_1, \ldots, x_n)$ with free real variables $x_1, \ldots, x_n$ defines a geometrical figure $\{(x_1, \ldots, x_n) \mid (x_1, \ldots, x_n) \in \mathbf{R}^n \wedge \varphi(x_1, \ldots, x_n)\}$ in n-dimensional Euclidean space $\mathbf{R}^n$. Point sets defined in this way are called *semi-algebraic sets*.

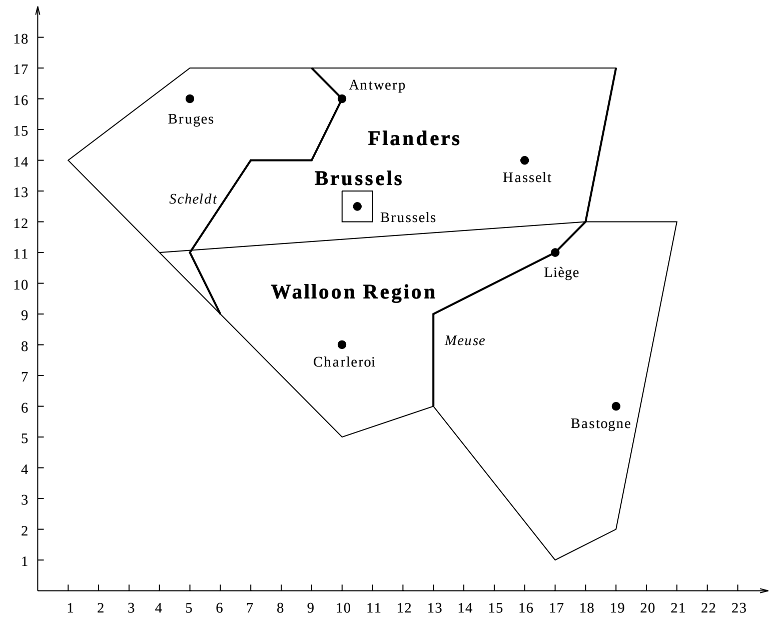
A *spatial database scheme*, $\mathcal{S}$, is a finite set of *relation names*. Each relation name, R, has a type which is a pair of natural numbers, $[m, n]$, where m denotes the number of non-spatial columns and n the dimension of the single spatial column of R. A database scheme has type $[m_1, n_1, \ldots, m_k, n_k]$ if the scheme consists of relation names, say $R_1, \ldots, R_k$, respectively of type $[m_1, n_1], \ldots, [m_k, n_k]$. A *syntactic database instance* is a mapping, $\mathcal{I}$, assigning to each relation name, R, of a scheme, $\mathcal{S}$, a syntactic relation $\mathcal{I}(R)$ of the same type. A *syntactic relation* of type $[m, n]$ is a finite set of tuples of the form $(v_1, \ldots, v_m; \varphi(x_1, \ldots, x_n))$, with $v_1, \ldots, v_m$ non-spatial values of some domain, U, and $\varphi(x_1, \ldots, x_n)$ a real formula with n free variables.

The semantics of a syntactic database instance, $\mathcal{I}$, over a database scheme, $\mathcal{S}$, is the mapping, I, assigning to each relation name, R, in $\mathcal{S}$ the semantic relation $I(\mathcal{I}(R))$. Given a syntactic relation, r, the semantic relation $I(r)$ is defined as $\bigcup_{t \in r} \{(t.v_1, \ldots, t.v_m)\} \times \{(x_1, \ldots, x_n) \mid t.\varphi(x_1, \ldots, x_n)\}$. This subset of $U^m \times \mathbf{R}^n$ can be interpreted as a possibly infinite $(m + n)$-ary relation, called *semantic relation*, the tuples of which are called *semantic tuples*.

Example 1. The example in Figure 1 shows a spatial database representing geographical information about Belgium.

We consider a query of signature $[m_1, n_1, \ldots, m_k, n_k] \rightarrow [m, n]$ to be a mapping from database instances of a spatial database scheme of type $[m_1, n_1, \ldots, m_k, n_k]$ to database instances of a spatial database scheme of type $[m, n]$ that can be regarded in a consistent way both at the syntactic and semantic level, and is computable at the syntactic level.

In this context, we define the query language FO + poly as the language obtained by adding to the language of real formulae the following: (i) a totally ordered infinite set of variables called *non-spatial variables*, disjoint from the set of real variables, (ii) atomic formulae of the form $v_1 = v_2$, with v_1 and v_2 non-spatial variables, (iii) atomic formulae of the form $R(v_1, \ldots, v_m; x_1, \ldots, x_n)$, with $v_1, \ldots, v_m$ non-spatial variables, $x_1, \ldots, x_n$ real variables, and R a relation name of type $[m, n]$, and finally (iv) universal and existential quantification of non-spatial variables. A query of signature $[m_1, n_1, \ldots, m_k, n_k] \rightarrow [m, n]$ is definable in FO + poly if there exists an FO + poly formula φ with m free value variables



Regions

Name	*Geometry*
Brussels	$(y \leq 13) \wedge (x \leq 11) \wedge (y \geq 12) \wedge (x \geq 10)$
Flanders	$(y \leq 17) \wedge (5x - y \leq 78) \wedge (x - 14y \leq -150) \wedge (x + y \geq 45) \wedge$ $(3x - 4y \geq -53) \wedge (\neg((y \leq 13) \wedge (x \leq 11) \wedge (y \geq 12) \wedge (x \geq 10)))$
Walloon Region	$((x - 14y \geq -150) \wedge (y \leq 12) \wedge (19x + 7y \leq 375) \wedge (x - 2y \leq 15) \wedge$ $(5x + 4y \geq 89) \wedge (x \geq 13)) \vee ((-x + 3y \geq 5) \wedge (x + y \geq 45) \wedge$ $(x - 14y \geq -150) \wedge (x \geq 13))$

Cities

Name	*Geometry*
Antwerp	$(x = 10) \wedge (y = 16)$
Bastogne	$(x = 19) \wedge (y = 6)$
Bruges	$(x = 5) \wedge (y = 16)$
Brussels	$(x = 10.5) \wedge (y = 12.5)$
Charleroi	$(x = 10) \wedge (y = 8)$
Hasselt	$(x = 16) \wedge (y = 14)$
Liège	$(x = 17) \wedge (y = 11)$

Rivers

Name	*Geometry*
Meuse	$((y \leq 17) \wedge (5x - y \leq 78) \wedge (y \geq 12)) \vee$ $((y \leq 12) \wedge (x - y = 6) \wedge (y \geq 11)) \vee$ $((y \leq 11) \wedge (x - 2y = -5) \wedge (y \geq 9)) \vee$ $((y \leq 9) \wedge (x = 13) \wedge (y \geq 6))$
Scheldt	$((y \leq 17) \wedge (x + y = 26) \wedge (y \geq 16)) \vee$ $((y \leq 16) \wedge (2x - y = 4) \wedge (y \geq 14)) \vee$ $((x \leq 9) \wedge (x \geq 7) \wedge (y = 14)) \vee$ $((y \leq 14) \wedge (-3x + 2y = 7) \wedge (y \geq 11)) \vee$ $((y \leq 11) \wedge (2x + y = 21) \wedge (y \geq 9))$

Fig. 1. Example of a (linear) spatial database.

and n free real variables such that, for every input database instance of signature $[m_1, n_1, \ldots, m_k, n_k]$, $\{(v_1, \ldots, v_m, x_1, \ldots, x_n) \mid \varphi(v_1, \ldots, v_m, x_1, \ldots, x_n)\}$ evaluates to the corresponding output database, which is of type $[m, n]$.

Example 2. Assuming that S is a relation of type $[0, 2]$, i.e., a semi-algebraic set in the plane, the FO + poly-formula

$$(\exists x_1)(\exists y_1)(\exists x_2)(\exists y_2)(\exists x_3)(\exists y_3)(\exists \lambda)(\exists \mu)(\exists \nu)(S(x_1, y_1) \wedge S(x_2, y_2) \wedge S(x_3, y_3) \wedge \lambda \geq 0 \wedge \mu \geq 0 \wedge \nu \geq 0 \wedge \lambda + \mu + \nu = 1 \wedge x = \lambda x_1 + \mu x_2 + \nu x_3 \wedge y = \lambda y_1 + \mu y_2 + \nu y_3.$$

defines the *convex-hull*[3] query of signature $[0, 2] \rightarrow [0, 2]$ which associates with S its convex hull.

Real formulae not containing non-linear polynomials are called *linear formulae.* Point sets defined by linear formulae are called *semi-linear sets.*

The linear spatial data model is defined in the same way as the general spatial data model above using linear formulae instead of real formulae. Similarly, linear queries can be defined. Notice that a general query induces a linear query if the query restricted to linear database instances is linear. Observe that the convex-hull query (Example 2) induces a linear query. Queries of signature $[m_1, n_1, \ldots, m_k, n_k] \rightarrow [0, 0]$ are called *Boolean* queries, because the sets $\{()\}$ and $\{\}$ can be seen as encoding the truth values *true* and *false*, respectively. Since both these sets are semi-linear, every Boolean query induces a linear query.

A very appealing linear query language for the linear spatial data model, called FO + linear, is obtained from FO + poly by only allowing linear formulae rather than real formulae.

Example 3. The following FO + linear formula defines a Boolean (and hence linear) query of signature $[0, 2] \rightarrow [0, 0]$ deciding whether S is *convex*:

$$(\forall x_1)(\forall y_1)(\forall x_2)(\forall y_2)(\forall x_3)(\forall y_3)(S(x_1, y_1) \wedge S(x_2, y_2) \wedge 2x_3 = x_1 + x_2 \wedge 2y_3 = y_1 + y_2 \Rightarrow S(x_3, y_3).$$

We prove in Section 4, however, that not every linear query definable in FO+poly is definable in FO+linear. (In particular, we will show that the convex hull query introduced in Example 2 is not definable in FO + linear.) Therefore, it makes sense to define FO+poly^{lin} as the set of FO + poly-definable queries inducing linear queries. Thus, the set of queries definable in FO+poly^{lin} is a strict subset of the set of queries definable in FO + poly.

Throughout the paper, we use vector notation to denote points. In this notation, formulae should be interpreted coordinate-wise. Hence, $\neg(\mathbf{x} = \mathbf{0})$ indicates that $\mathbf{x}$ is not the origin of the coordinate system, whereas $\mathbf{x} \neq \mathbf{0}$ denotes that *none* of the coordinates of $\mathbf{x}$ equals 0.

[3] Let $A \subseteq \mathbf{R}^n$. The convex hull of A is the smallest convex set of $\mathbf{R}^n$ containing A. In particular, the convex hull of a semi-linear set is a semi-linear set.

3 Expressiveness of FO + linear

In this section, we discuss the expressiveness of the query language FO + linear. To simplify the discussion, we focus on purely spatial queries, i.e., queries acting on linear databases consisting of relations of a type of the form $[0, n]$.

First, we briefly review some known results involving linear queries computable in FO + linear.

The following operations on semi-linear sets can be defined rather trivially in FO + linear: union, intersection, difference, complement, and projection. In general, any affine transformation of semi-linear sets can be defined in FO + linear. In [26], FO+linear expressions are given for the Boolean queries checking boundedness, convexity, and discreteness of semi-linear sets. The expressive power of FO + linear unfolds completely, however, when topological properties of geometrical objects are considered. The definitions of topological interior, boundary, and closure can indeed be translated almost straightforwardly into linear calculus expressions. Hence, for example, the regularization of a semi-linear set, defined as the closure of its interior, can be computed in FO + linear, which is of importance, since the regularized set operators union, intersection, and difference, turn out to be indispensable in most spatial database applications [10, 19, 12]. More generally, Egenhofer et al. showed in a series of papers [7, 8, 9] that a whole class of topological relationships such as *disjoint*, *in*, *contained*, *overlap*, *touch*, *equal*, and *covered* can be defined in terms of intersections between the boundary, interior, and complement of the geometrical objects.

Another property of geometrical objects often used in spatial database applications, is *dimension*. For instance, in [5], the dimension is used to further refine the class of topological relationships defined by Egenhofer et al. We now show that it can be decided in FO + linear whether a given semi-linear set has a given number as its dimension, which is the contribution of this section. Since there are only finitely many known possibilities for the dimension of a semi-linear set, it follows that the dimension can actually be *computed* in FO + linear.

Definition 1. The *dimension* of a semi-linear set S of $\mathbf{R}^n$ is the maximum value of d for which there exists an open d-dimensional cube fully contained in S. The dimension of the empty set is defined as -1.

Theorem 2. *The predicate* $\dim(S, d)$*, in which S is a semi-linear set of* $\mathbf{R}^n$ *and d is a number, and which evaluates to* true *if the dimension of S equals d, can be defined in* FO + linear.

The correctness of this theorem follows from two lemmas we present next. We will use the notation $\pi_i(S)$, with S a semi linear set of $\mathbf{R}^n$, to denote the semi-linear set $\{(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) \mid (\exists x_i) S(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n)\}$ of $\mathbf{R}^{n-1}$. Hence, $\pi_i(S)$ is the projection of S onto the i-th coordinate hyper-plane of $\mathbf{R}^n$.

Lemma 3. *The dimension of $\pi_i(S)$, with S a d-dimensional semi-linear set of $\mathbf{R}^n$, is at most d for $1 \leq i \leq n$.*

Lemma 4. *If S is a d-dimensional semi-linear set of $\mathbf{R}^n$, with $d < n$, then there exists i, $1 \leq i \leq n$, such that the dimension of $\pi_i(S)$ equals d.*

The rather technical proof of Lemma 4 is omitted due to space limitations.

Now define empty(S) as the FO + linear formula $\neg(\exists \mathbf{x})S(\mathbf{x})$, maxdim($S$) as the FO + linear formula $(\exists \mathbf{x})(\exists \boldsymbol{\epsilon})(\boldsymbol{\epsilon} \neq \mathbf{0} \wedge (\forall \mathbf{y})(\mathbf{x} - \boldsymbol{\epsilon} < \mathbf{y} < \mathbf{x} + \boldsymbol{\epsilon} \Rightarrow S(\mathbf{y})))$, and, $\max(d, d_1, \ldots, d_n)$ as the FO + linear formula (expression omitted) which evaluates to *true* if d is the maximum of $d_1, \ldots, d_n$. Then the FO + linear formula $(d = -1 \wedge \mathrm{empty}(S)) \vee (d = 1 \wedge \mathrm{maxdim}(S)) \vee (d = 0 \wedge \neg\mathrm{empty}(S) \wedge \neg\mathrm{maxdim}(S))$ clearly defines $\dim(S, d)$ in $\mathbf{R}$. In general, the FO + linear formula $(d = n \wedge \mathrm{maxdim}(S)) \vee (\neg\mathrm{maxdim}(S) \wedge \dim(\pi_1(S), d_1) \wedge \ldots \wedge \dim(\pi_n(S), d_n) \wedge \max(d, d_1, \ldots, d_n))$, inductively defined, by Lemma 3 and 4 defines $\dim(S, d)$ in $\mathbf{R}^n$.

Many interesting queries can be defined in a natural way using the dimension predicate, and are therefore also definable in FO + linear, as is illustrated by the following example.

Example 4. The Boolean query which decides whether a semi-linear set S is a line or a line segment, is definable in FO + linear, using the following expression:

$$\dim(S, 1) \wedge (\forall \mathbf{x})(\forall \mathbf{y})(S(\mathbf{x}) \wedge S(\mathbf{y}) \Rightarrow S((\mathbf{x} + \mathbf{y})/2)).$$

It should be noted that Afrati et al.[1] independently showed that the line query is definable FO + linear. Their solution does not use the dimension predicate.

A precise characterization of the expressive power of FO + linear is still open, however.

4 Limitations of FO + linear

Recently, Afrati et al. [2] established that FO+linear can not define all FO+poly definable linear queries:

Proposition 5. *The Boolean query on semi-linear sets S of $\mathbf{R}$ which evaluates to true if there exist u and v of S with $u^2+v^2 = 1$, is not definable in* FO+linear.

Even though the query in Proposition 5 involves a non-linear computation in order to evaluate it, it is nevertheless a linear query because it is boolean, and therefore suffices to establish the incompleteness of FO+linear for the FO+poly-definable linear queries. The query, however, is unsatisfactory because it provides little insight into whether more natural, non-boolean FO + poly-definable linear queries are FO + linear-definable. Two such queries are (1) the linear query from semi-linear sets of $\mathbf{R}^n$ to semi-linear sets of $\mathbf{R}^n$ computing the convex hull (discussed in Example 2 for $n = 2$); and (2) the Boolean query on semi-linear sets of $\mathbf{R}^n$ which evaluates to *true* if all points in the input are colinear. In this section, we show that the above queries are *not* definable in FO + linear if $n \geq 2$. To demonstrate this claim, we build on the following results established by Afrati et al. [2] and the present authors [26]:

Proposition 6. *Let $n \geq 2$ and $m \geq 3$. Then the following sets are* not *definable in* FO + linear*:*

1. $\{(\mathbf{u}_1, \ldots, \mathbf{u}_m) \in (\mathbf{R}^n)^m \mid \mathbf{u}_m \in \text{convex-hull}(\{\mathbf{u}_1, \ldots, \mathbf{u}_{m-1}\})\}$*; and*
2. $\{(\mathbf{u}_1, \ldots, \mathbf{u}_m) \in (\mathbf{R}^n)^m \mid \mathbf{u}_1, \ldots, \mathbf{u}_m \text{ are colinear}\}$.

Even though the undefinability in FO+linear of the *sets* defined in Proposition 6 may suggest that the related *queries* (1) and (2) mentioned earlier are also non-definable in FO + linear, this deduction is not obvious. To see the caveat, it suffices to notice that the sets defined in Proposition 6 are not even semi-linear, whereas the related queries are obviously linear. This technical gap appears to have been overlooked in both the work of Afrati et al. [2] and previous work of the present authors [26]. In what follows, however, we show that there exists a general technique to link results about the non-definability in FO+linear of sets to the non-definability of certain related linear queries.

Definition 7. Let P be a *semi-algebraic* subset of $(\mathbf{R}^n)^m$, $m, n \geq 1$. Let k be such that $0 \leq k \leq m$. Furthermore assume that P and k are such that, for each l, $1 \leq l \leq k$, for each sequence $\mathbf{u}_1, \ldots, \mathbf{u}_l$ in $\mathbf{R}^n$, and for all sequences $i_1, \ldots, i_k$ and $j_1, \ldots, j_k$ such that $1 \leq i_1, \ldots, i_k, j_1, \ldots, j_k \leq l$ and $\{\mathbf{u}_{i_1}, \ldots, \mathbf{u}_{i_k}\} = \{\mathbf{u}_{j_1}, \ldots, \mathbf{u}_{j_k}\} = \{\mathbf{u}_1, \ldots, \mathbf{u}_l\}$, the following permutation invariance property holds for all $\mathbf{u}_{k+1}, \ldots, \mathbf{u}_m$ in $\mathbf{R}^n$:

$$(\mathbf{u}_{i_1}, \ldots, \mathbf{u}_{i_k}, \mathbf{u}_{k+1}, \ldots, \mathbf{u}_m) \in P \Leftrightarrow (\mathbf{u}_{j_1}, \ldots, \mathbf{u}_{j_k}, \mathbf{u}_{k+1}, \ldots, \mathbf{u}_m) \in P.$$

The query $Q_{P,k}$ of signature $[0, n] \rightarrow [0, n(m-k)]$ is now defined as follows. If S consists of at most k points of $\mathbf{R}^n$, say $S = \{\mathbf{u}_1, \ldots, \mathbf{u}_k\}$ ($\mathbf{u}_1, \ldots, \mathbf{u}_k$ not necessarily all distinct), then

$$Q_{P,k}(S) = \{(\mathbf{u}_{k+1}, \ldots, \mathbf{u}_m) \mid (\mathbf{u}_1, \ldots, \mathbf{u}_k, \mathbf{u}_{k+1}, \ldots, \mathbf{u}_m) \in P\};$$

otherwise $Q_{P,k}(S)$ is empty.

Observe that the invariance property assumed for P and k guarantees that $Q_{P,k}$ is well-defined.

Example 5. 1. *Let P be the set*

$$\{(\mathbf{u}_1, \ldots, \mathbf{u}_m) \in (\mathbf{R}^n)^m \mid \mathbf{u}_m \in \text{convex-hull}(\{\mathbf{u}_1, \ldots, \mathbf{u}_{m-1}\})\},$$

with $n \geq 2$ and $m \geq 3$. Let $k = m - 1$. Then $Q_{P,k}$ is the linear query that associates with each set S consisting of $m-1$ points, the convex hull of S, and with every other set S the empty set. Notice that, by Property 6, the set P is not FO + linear*-definable.*

2. *Let P be the set*

$$\{(\mathbf{u}_1, \ldots, \mathbf{u}_m) \in (\mathbf{R}^n)^m \mid \mathbf{u}_1, \ldots, \mathbf{u}_m \text{ are colinear}\},$$

with $n \geq 2$ and $m \geq 3$. Let $k = m$. Then $Q_{P,k}$ can be interpreted as the Boolean query which evaluates a semi linear set S to true *if and only if S consists of m colinear points. Notice that, by Property 6, the set P is* not FO + linear*-definable.*

We must emphasize that the linear queries in Example 5 are closely related, but *not* identical, to the linear queries (1) and (2) in the beginning of this section. One can think of the queries in Example 5 as *restrictions* of the linear queries (1) and (2) to certain finite sets.

We now prove the following theorem:

Theorem 8. *Let P be a* semi-algebraic *subset of* $(\mathbf{R}^n)^m$ *m, $n \geq 1$, and let P and k satisfy the conditions of Definition 7. If P is undefinable in* $\mathrm{FO} + \mathrm{linear}$, *then the following holds:*

1. *The query $Q_{P,k}$ is undefinable in* $\mathrm{FO} + \mathrm{linear}$.
2. *If Q is a linear query from semi-linear sets of $\mathbf{R}^n$ to semi-linear sets of $(\mathbf{R}^n)^{m-k}$ such that, for every semi-linear set S of $\mathbf{R}^n$, $Q(S) = Q_{P,k}(S)$ if $Q_{P,k}(S)$ is not empty, then Q is undefinable in* $\mathrm{FO} + \mathrm{linear}$.

Proof. 1. Assume, to the contrary, that the query $Q_{P,k}$ is $\mathrm{FO} + \mathrm{linear}$-definable. Then there exists an $\mathrm{FO} + \mathrm{linear}$ formula $\varphi_{P,k}(R; \mathbf{x}_{k+1}, \ldots, \mathbf{x}_m)$, with R an appropriate predicate name, such that, for each semi-linear set S of $\mathbf{R}^n$, $Q_{P,k}(S) = \{(\mathbf{u}_{k+1}, \ldots, \mathbf{u}_m) \mid \varphi_{P,k}(S; \mathbf{u}_{k+1}, \ldots, \mathbf{u}_m)\}$. We now argue that the predicate name R must effectively occur in $\varphi_{P,k}$. If this were not the case, then the query associated with $\varphi_{P,k}$ would be a constant function. This constant function cannot yield the empty set, for, otherwise, by the definition of $Q_{P,k}$, P would also be the empty set, which is obviously $\mathrm{FO} + \mathrm{linear}$-definable, contrary the hypothesis of the theorem. The constant function cannot yield a non-empty set, either, however, since again by the definition of $Q_{P,k}$, there is an infinite number of inputs for which $Q_{P,k}$ returns the empty set. Thus R must occur in $\varphi_{P,k}$.

Given the formula $\varphi_{P,k}$, we can construct the formula $\hat{\varphi}_{P,k}$ as follows. Let $\mathbf{x}_1, \ldots, \mathbf{x}_k$ be variables that do not occur in $\varphi_{P,k}$. Now replace every literal of the form $R(\mathbf{z})$ in $\varphi_{P,k}$ by the formula $\mathbf{z} = \mathbf{x}_1 \vee \cdots \vee \mathbf{z} = \mathbf{x}_k$. Observe that the formula $\hat{\varphi}_{P,k}$ is a linear formula with free variables $\mathbf{x}_1, \ldots, \mathbf{x}_m$. Our claim is that the formula $\hat{\varphi}_{P,k}$ defines the set P, a contradiction with the hypothesis of the theorem. Consider an m-tuple $(\mathbf{u}_1, \ldots, \mathbf{u}_m) \in (\mathbf{R}^n)^m$. ¿From the definition of $Q_{P,k}$ and $\varphi_{P,k}$, we have $(\mathbf{u}_1, \ldots, \mathbf{u}_m) \in P \Leftrightarrow (\mathbf{u}_{k+1}, \ldots, \mathbf{u}_m) \in Q_{P,k}(\{\mathbf{u}_1, \ldots, \mathbf{u}_k\})$, whence $(\mathbf{u}_1, \ldots, \mathbf{u}_m) \in P \Leftrightarrow \varphi_{P,k}(\{\mathbf{u}_1, \ldots, \mathbf{u}_k\}; \mathbf{u}_{k+1}, \ldots, \mathbf{u}_m)$. It follows from the construction of $\hat{\varphi}_{P,k}$ from $\varphi_{P,k}$ that $(\mathbf{u}_1, \ldots, \mathbf{u}_m) \in P \Leftrightarrow \hat{\varphi}_{P,k}(\mathbf{u}_1, \ldots, \mathbf{u}_m)$.
2. Assume that Q is $\mathrm{FO} + \mathrm{linear}$-definable. Then there exists a formula

$$\varphi_Q(R; \mathbf{x}_{k+1}, \ldots, \mathbf{x}_m)$$

that defines Q. Given φ_Q, we can construct the formula $\hat{\varphi}_Q$:

$$\hat{\varphi}_Q(R; \mathbf{x}_{k+1}, \ldots, \mathbf{x}_m) \Leftrightarrow (|R| \leq k \wedge \varphi_Q(R; \mathbf{x}_{k+1}, \ldots, \mathbf{x}_m)) \vee (|R| > k \wedge \mathit{false}).$$

It is obvious that this expression for $\hat{\varphi}_Q$ can be translated into proper $\mathrm{FO}+\mathrm{linear}$ syntax. It now follows from the properties of Q that the formula $\hat{\varphi}_Q$ defines the query $Q_{P,k}$. Hence, it would follow that $Q_{P,k}$ is $\mathrm{FO} + \mathrm{linear}$-definable, which is impossible by the first part of the theorem.

Theorem 8 has the following corollary:

Corollary 9. *The convex hull query* (1) *and the colinearity query* (2) *are not definable in* $\mathrm{FO} + \mathrm{linear}$.

5 Extensions of FO + linear

Although, in Section 3, it is shown that a wide range of useful, complex linear queries can be defined in $\mathrm{FO} + \mathrm{linear}$, the language lacks the expressive power to define some important $\mathrm{FO{+}poly}^{lin}$ queries, as is clearly demonstrated in Section 4. Hence the search for languages that capture such queries is important. Without such languages, we would indeed be hard-pressed to substantiate the claim that the linear model is to be adopted as the fundamental model for applications involving linear geometric objects.

The obvious way to obtain a query language which is complete for the $\mathrm{FO{+}poly}^{lin}$ queries is to discover an algorithm that can decide which $\mathrm{FO} + \mathrm{poly}$ formulae induce linear queries. Unfortunately, such an algorithm does not exist:

Theorem 10. *It is undecidable whether an arbitrary* $\mathrm{FO} + \mathrm{poly}$ *formula induces a linear query.*

Proof. (Sketch.) The proof is a variation of a proof by Paredaens et al. [22] concerning undecidability of genericity in $\mathrm{FO} + \mathrm{poly}$ (Theorem 1, pp. 285). The $\forall^*$-fragment of number theory is undecidable since Hilbert's 10th problem can be reduced to it. Encode a natural number n by the one-dimensional semi-algebraic set $enc(n) := \{0, \dots, n\}$, and encode a vector of natural numbers $(n_1, \dots, n_k)$ by $enc(n_1) \cup (enc(n_2) + n_1 + 2) \cup \dots \cup (enc(n_k) + n_1 + 2 + \cdots + n_{k-1} + 2)$. The corresponding decoding is first-order. We then reduce a $\forall^*$-sentence $(\forall \mathbf{x})\varphi(\mathbf{x})$ of number theory to the following query of signature $[0, 1] \to [0, 0]$:

$$\textbf{if } R \text{ encodes a vector } \mathbf{x} \textbf{ then if } \varphi(\mathbf{x}) \textbf{ then } \emptyset \textbf{ else } \{(u, v) \mid u^2 + v^2 = 1\} \textbf{ else } \emptyset.$$

This query is definable in $\mathrm{FO} + \mathrm{poly}$ and induces a linear query if and only if the $\forall^*$-sentence is valid.

Theorem 10 shows that a top-down approach to discover a useful linear sub-query language is difficult. Observe that Theorem 10 still allows the isolation of a subset of the $\mathrm{FO} + \mathrm{poly}$ formulae that define $\mathrm{FO{+}poly}^{lin}$ queries, in the same way that the undecidability of safeness in the relational calculus is not in contradiction with the existence of a sub-language of the relational calculus which has precisely the expressive power of the safe relational calculus queries. [25]

In this section, we therefore take a bottom-up approach to discover restrictions of $\mathrm{FO{+}poly}^{lin}$ that are strictly more expressive than $\mathrm{FO} + \mathrm{linear}$. The basic idea is to extend $\mathrm{FO} + \mathrm{linear}$ with certain linear operators, such as the colinearity or the convex-hull query. It is important to observe in this respect how careful we have to be to avoid creating languages that are no longer linear.

A too liberal syntax can indeed lead to the definability of non-semi-linear sets associated to these operators, such as the sets exhibited in Proposition 6. This in turn can have as a consequence that the language obtains the full expressive power of $\mathrm{FO}+\mathrm{poly}$, as is shown by the following example [26].

Example 6. Extending $\mathrm{FO}+\mathrm{linear}$ with the convex-hull predicate (or with the colinearity predicate which can be derived from the former) as defined in [26] leads to a language with the expressive power of FO+poly. [26], the reason being that the predicate $\mathrm{product}(x, y, z)$ defined by $z = xy$ is can be expressed as[4]

$$\neg(\exists!\mathbf{u})((\mathrm{colinear}(\mathbf{x}, \mathbf{e}_2, \mathbf{u}) \wedge \mathrm{colinear}(\mathbf{y}, \mathbf{z}, \mathbf{u}))),$$

where $\mathbf{x} = (x, 0)$, $\mathbf{y} = (0, y)$, $\mathbf{z} = (z, 0)$, $\mathbf{u} = (u_1, u_2)$, and $\mathbf{e}_2 = (0, 1)$.

We now proceed with showing how $\mathrm{FO}+\mathrm{linear}$ can be extended with operators in a *safe* way. The subtle point of our definition consists in disallowing free *real* variables in set terms. So, even though a set-term might have free value variables, it is disallowed to have free real variables.

An *operator* is defined to be an FO+poly^{lin} query. The signature of an operator is the signature of that query.

Let $\mathcal{O}$ be a set of operator names O typed with a signature, each of which represents an operator $\mathrm{op}(O)$ of the same signature.

The query language $\mathrm{FO}+\mathrm{linear}+\mathcal{O}$ is then defined as an extension of $\mathrm{FO}+\mathrm{linear}$, as follows. First, we extend the terms of $\mathrm{FO}+\mathrm{linear}$ with *set terms*:

- If φ is an $\mathrm{FO}+\mathrm{linear}+\mathcal{O}$ formula with n free real variables $x_1, \ldots, x_n$ and m free value variables $v_1, \ldots, v_m$, and if $k \leq m$, then
$$\{(v_1, \ldots, v_k, x_1, \ldots, x_n) \mid \varphi(v_1, \ldots, v_m, x_1, \ldots, x_n)\}$$
is a *set term* of type $[k, n]$. Observe that of the value variables, $v_{k+1}, \ldots, v_m$ occur *free*, while *all* real variables, $x_1, \ldots, x_n$, occur *bounded* in the set term.[5]
- If O is an operator name in $\mathcal{O}$ of type $[m_1, n_1, \ldots, m_k, n_k] \rightarrow [m, n]$, and $S_1, \ldots, S_k$ are set terms of types $[m_1, n_1], \ldots, [m_k, n_k]$, respectively, then
$$O(S_1, \ldots, S_k)$$
is a *set term* of type $[m, n]$ with as free variables those in the union of all free variables in S_1 through S_k (which are all value variables).

Finally, we extend the atomic formulae of $\mathrm{FO}+\mathrm{linear}$:

- Let S be a set term of type $[m, n]$. Then $S(v_1, \ldots, v_m, x_1, \ldots, x_n)$, with $v_1, \ldots, v_m$ value variables and $x_1, \ldots, x_n$ real variables, is an *atomic formula* with free variables $v_1, \ldots, v_m$, $x_1, \ldots, x_n$ union the free (value) variables of S.

[4] The quantifier "$\exists!$" should be read as "there exists exactly one" and can be expressed in FO in a straightforward manner.

[5] Observe that this definition allows us to interpret a predicate name R of type $[k, n]$ as a set term of type $[k, n]$.

When actual values are substituted for the free variables, a set term of type $[m, n]$ represents a subset of $U^m \times \mathbf{R}^n$. Consider then an atomic formula of the form $S(v_1, \ldots, v_m, x_1, \ldots, x_n)$, this atomic formula evaluates to *true* if the evaluation of $(v_1, \ldots, v_m, x_1, \ldots, x_n)$ belongs to the set represented by the set term S. The full semantics of FO + linear + $\mathcal{O}$ is now straightforward to define.

If we constrain the operator in $\mathcal{O}$ to be FO + linear-definable, we can prove the following safety property by induction on the structure of FO + linear + $\mathcal{O}$-formulae:

Theorem 11. *The query language* FO + linear + $\mathcal{O}$ *only expresses* FO + poly^{lin}-*definable queries.*

The syntactic restriction that set terms contain only free *value* variables is essential for Theorem 11 to hold; otherwise, e.g., the formula in Example 6 could be expressed in FO + linear+colinear, whence FO + linear+colinear would have the full expressive power of FO + poly.

Without going into details, we mention that it is possible to define an algebraic query language equivalent to FO + linear + $\mathcal{O}$ by extending the linear algebra [26] with the operators represented by $\mathcal{O}$. This equivalence result forms a theoretical justification for the approach Güting has taken with the development of the ROSE-algebra. [13, 14, 15, 16], which is extending the relational algebra with a class of spatial operators.

Finally, we give an example of an FO + linear + $\mathcal{O}$ query language in which we can express the queries (1) and (2) in the beginning of Section 4. Thereto, define an infinite set of operator names seg^k of signature $[0, k] \rightarrow [0, k]$ and associate with each operator name seg^k the operator op(seg^k) defined by $\text{op}(\text{seg})^k(S) = \{\mathbf{x} \in \mathbf{R}^k \mid (\exists \mathbf{y})(\exists \mathbf{z})(S(\mathbf{y}) \wedge S(\mathbf{z}) \wedge \mathbf{x} \in [\mathbf{y}, \mathbf{z}]\})$ for each semi-linear set S of $\mathbf{R}^k$. Let $\mathcal{S}$ be the set of all seg^k, $k \geq 0$. Now let R be a predicate representing a semi-linear set of $\mathbf{R}^k$. The FO + linear + $\mathcal{S}$ formula

$$\underbrace{\text{seg}^k(\text{seg}^k(\ldots \text{seg}^k(R)))(\mathbf{x})}_{k \text{ times}}.$$

computes the convex hull of the set represented by R. Using the convex-hull query as a macro, we can express co-linearity by the following FO + linear + $\mathcal{S}$ formula:

$$(\exists d)(\dim(\{\mathbf{x} \mid \text{convex-hull}(S)(\mathbf{x})\}, d) \wedge d \leq 1).$$

6 Conclusion

In this paper we studied languages that define FO+poly^{lin} queries. Amongst these languages, the most natural one is FO + linear. For this language, we showed that non-trivial FO+poly^{lin} queries, such as the dimension query, can be defined in it, but we also demonstrated that important FO+poly^{lin} queries, such as the convex hull, cannot be defined. These latter results led us to the introduction of extensions of FO + linear with FO+poly^{lin}-definable operators.

The crucial part of this construction was requiring that operators can only be applied to set terms *without* free real variables. As we showed with a counter-example, our construction can lead to unsafe query languages if that restriction is lifted.

We conclude by mentioning the two most prominent open problems raised by this paper: (i) does there exist a syntactic restriction on FO + poly formulae that yields a sublanguage of FO + poly which is sound and complete for the FO+poly^{lin}-definable queries; and (ii) does there exist an extension of FO + linear (or other sublanguages of FO + poly) with operators that yields soundness and completeness?

References

1. F. Afrati, T. Andronikos, T.G. Kavalieros, "On the Expressiveness of First-Order Constraint Languages," in Proceedings *ESPRIT WG CONTESSA Workshop*, (Friedrichshafen, Germany), G. Kuper and M. Wallace, eds., *Lecture Notes in Computer Science*, vol. 1034, Springer-Verlag, Berlin, 1996, pp. 22–39.
2. F. Afrati, S. Cosmadakis, S. Grumbach, and G. Kuper, "Linear Versus Polynomial Constraints in Database Query Languages," in Proceedings *2nd Int'l Workshop on Principles and Practice of Constraint Programming* (Rosario, WA), A. Borning, ed., *Lecture Notes in Computer Science*, vol. 874, Springer-Verlag, Berlin, 1994, pp. 181–192.
3. A. Brodsky and Y. Kornatzky, "The LyriC Language: Querying Constraint Objects," in Proceedings *Post-ILPS'94 Workshop on Constraints and Databases* (Ithaca, NY), 1994.
4. I. Carlbom, "An Algorithm for Geometric Set Operations Using Cellular Subdivision Techniques," *IEEE Computer Graphics and Applications*, 7:5, 1987, pp. 44–55.
5. E. Clementini, P. Di Felice, and P. van Oosterom, "A Small Set of Formal Topological Relationships Suitable for End-User Interaction," in Proceedings *3nd Symposium on Advances in Spatial Databases*, *Lecture Notes in Computer Science*, vol. 692. Springer-Verlag, Berlin, 1993, pp. 277–295.
6. J. Nievergelt and M. Freeston, eds., Special issue on spatial data, *Computer Journal*, 37:1, 1994.
7. M.J. Egenhofer, "A Formal Definition of Binary Topological Relationships," in Proceedings *Foundations of Data Organization and Algorithms*, W. Litwin and H.-J. Schek, eds., *Lecture Notes in Computer Science*, vol. 367, Springer-Verlag, Berlin, 1989, pp. 457–472.
8. M.J. Egenhofer, and J. Herring, "A Mathematical Framework for the Definition of Topological Relationships," in Proceedings *Fourth International Symposium on Spatial Data Handling* , K. Brassel and H. Kishimoto, eds., Zurich, Switzerland, 1990, pp. 803–813.
9. M.J. Egenhofer, "Reasoning about Binary Topological Relations," in Proceedings *Advances in Spatial Databases*, O. Günther and H.-J. Schek, eds., *Lecture Notes in Computer Science*, vol. 525, Springer-Verlag, Berlin, 1991, pp. 143–160.
10. M.J. Egenhofer, "What's Special about Spatial? Database Requirements for Vehicle Navigation in Geographic Space," *SIGMOD Records*, 22:2, 1993, pp. 398–402.
11. O. Günther, ed., *Efficient Structures for Geometric Data Management*, in *Lecture Notes in Computer Science*, vol. 337, Springer-Verlag, Berlin, 1988.

12. O. Günther, and A. Buchmann, “Research Issues in Spatial Databases,” *SIGMOD Records*, 19:4, 1990, pp. 61–68.
13. R.H. Güting, “Geo-Relational Algebra: A Model and Query Language for Geometric Database Systems,” in *Advances in Database Technology—EDBT '88*, Proceedings *Int'l Conf. on Extending Database Technology* (Venice, Italy), J.W. Schmidt, S. Ceri, and M. Missikoff, eds., *Lecture Notes in Computer Science*, vol. 303, Springer-Verlag, Berlin, 1988, pp. 506–527.
14. R.H. Güting, “Gral: An Extensible Relational Database System for Geometric Applications,” in Proceedings *15th Int'l Conf. on Very Large Databases* (Amsterdam, the Netherlands), 1989, pp. 33–34.
15. R.H. Güting, “An Introduction to Spatial Database Systems,” *VLDB-Journal*, 3:4, 1994, pp. 357–399.
16. R.H. Güting, “Implementations of the ROSE Algebra: Efficient Algorithms for Real-Based Spatial Data Types,” in Proceedings *Advances in Spatial Databases*, M. Egenhofer and J. Herring, eds., *Lecture Notes in Computer Science*, vol. 951, Springer-Verlag, Berlin, 1995, pp. 216–239.
17. T. Huynh, C. Lassez, and J.-L. Lassez. Fourier Algorithm Revisited. In Proceedings *2nd Int'l Conf. on Algebraic an Logic Programming*, H. Kirchner and W. Wechler, eds. *Lecture Notes in Computer Science*, vol. 463. Springer Verlag, Berlin, 1990, pp. 117–131.
18. P.C. Kanellakis and D.Q. Goldin, “Constraint Programming and Database Query Languages,” in Proceedings *2nd Conf. on Theoretical Aspects of Computer Software*, M. Hagiya and J.C. Mitchell, eds., *Lecture Notes in Computer Science*, vol. 789, Springer-Verlag, Berlin, 1994.
19. A. Kemper, and M. Wallrath, “An Analysis of Geometric Modeling in Database Systems,” *Computing Surveys*, 19:1, 1987, pp. 47–91.
20. P.C. Kanellakis, G.M. Kuper and P.Z. Revesz, “Constraint Query Languages,” *Journal of Computer and System Sciences*, to appear, also in Proceedings *9th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems* (Nashville, TN), 1990, pp. 299–313.
21. A. Tarski, ”A Decision Method for Elementary Algebra and Geometry,” University of California Press, Berkeley, California, 1951.
22. J. Paredaens, J. Van den Bussche, and D. Van Gucht, “Towards a Theory of Spatial Database Queries,” in Proceedings *13th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems* (Minneapolis, MN), 1994. pp. 279–288.
23. N. Pissinou, R. Snodgrass, R. Elmasri, I. Mumick, T. Özsu, B. Pernici, A. Segef. B. Theodoulidis, and U. Dayal, “Towards an Infrastructure for Temporal Databases,” *SIGMOD Records*, 23:1, 1994, pp. 35–51.
24. L.K. Putnam and P.A. Subrahmanyam, “Boolean Operations on n-Dimensional Objects,” *IEEE Computer Graphics and Applications*, 6:6, 1986, pp. 43–51.
25. J. D. Ullman, “Principles of Database and Knowledge-base Systems,” Computer Science Press, 1988.
26. L. Vandeurzen, M. Gyssens, and D. Van Gucht, “On the Desirability and Limitations of Linear Spatial Query Languages,” in Proceedings *4th Symposium on Advances in Spatial Databases*, M. J. Egenhofer and J.R. Herring, eds. *Lecture Notes in Computer Science*, vol. 951, Springer Verlag, Berlin, 1995, pp. 14–28.

This article was processed using the LaTeX macro package with LLNCS style