Towards a theory of spatial database queries
(extended abstract)

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

Abstract

A general model for spatial databases is considered, which extends the relational model by allowing as tuple components not only atomic values but also geometrical figures. The model, which is inspired by the work of Kanellakis, Kuper and Revesz on constraint query languages, includes a calculus and an algebra which are equivalent. Given this framework, the concept of spatial database query is investigated. Thereto, Chandra and Harel's well-known consistency criterion for classical relational queries is adapted. Various adaptations are proposed, depending on the kinds of geometry in which the spatial information in the database is to be interpreted. The consistency problem for calculus queries is studied. Expressiveness issues are examined. The main purpose of the paper is to open up new grounds for theoretical research in the area of spatial database systems. Consequently, many open problems are indicated.

1   Introduction.

Recently, much attention has been paid to *spatial* database systems (e.g., [Buc89, GS91, AO93]),

---

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

[†]Research Assistant of the Belgian National Fund for Scientific Research. Address: same as Jan Paredaens. E-mail: Jan.VandenBussche@uia.ac.be

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

which are capable of dealing with spatial information. Spatial database models and prototypes proposed in the literature typically focus on one specific type of spatial information (or a finite set of specific types), such as intervals for temporal applications or polygonal line segments for geographic applications. This narrow focus is often justified, since it is sufficient for many applications, and allows for tuned, efficient implementations. Nevertheless, in order to obtain a better understanding of the fundamental issues underlying spatial database management, a more general perspective is in order.

A natural approach to achieving this generality is to consider as spatial data *any* geometrical figure definable in elementary geometry, i.e., first-order logic over the reals. This wide class of figures is also known as the class of *semi-algebraic sets* in real algebraic geometry [BCR87]. The first-order theory of the reals is decidable; actually, it allows a very strong form of effective quantifier elimination (known as Cylindrical Algebraic Decomposition [Col75, Arn88]), which makes that many properties of semi-algebraic sets are decidable as well [HRR91]. Databases containing arbitrary semi-algebraic sets (i.e., their defining formulas) and a calculus for querying them were first considered by Kanellakis, Kuper and Revesz (abbr. KKR) in the context of their work on constraint query languages [KKR90].

In the present paper, it is our goal to initiate a study of the theoretical aspects of *queries* for spatial databases. As our framework, we use an orthogonal combination of the standard relational model and the KKR model. The tuples in the relations have both ordinary data attributes and spatial attributes. Correspondingly, we have an associated calculus query language that is a two-sorted combination (and a conservative extension) of the

standard relational calculus and the KKR calculus. We also define an equivalent algebra.

In the seminal paper on the theory of classical relational queries, Chandra and Harel defined a query formally as a partial recursive function from databases to databases that is invariant under permutations of the universe of atomic values ([CH80]; cf. also [AU79]). The latter consistency criterion, nowadays known as *genericity* [HY84], says informally that the query can be computed (on a Turing machine, say) regardless of how atomic values are encoded as strings: two different atomic values can only be distinguished on the basis of their logical properties. This intuition was also expressed by Tarski at a talk given in 1966 [Tar86], where he compared the situation with that of geometry. There, for example, a mathematical object (e.g., a set of points, or a function from sets of points to sets of points) is called "Euclidean" if it is invariant under similarities, or called "topological" if it is invariant under homeomorphisms. More generally, geometrical concepts are classified according to the group of geometrical transformations under which they are invariant.

This will lead us to the definition of various consistency criteria for spatial database queries, depending on the kind of geometry in which the spatial information is to be interpreted. For example, we define *isometry-generic* queries as partial recursive functions from spatial databases to spatial relations which are invariant under isometries with respect to the spatial data (and are still generic in the standard sense with respect to the ordinary data). The usefulness of thus classifying spatial relationships has already been acknowledged in the spatial database literature (e.g., [EF91]), but has never been pushed as far as the level of global queries like we do.

One of our main technical results is negative: genericity of a calculus query is undecidable. In the standard relational case (for relational calculus queries with constant symbols), this follows immediately from the undecidability of classical first-order logic on finite structures. This is no longer true however in a purely spatial setting without ordinary data. Nevertheless, we show that undecidability carries over, even for the simplest problem of testing for translation-genericity over one-dimensional spatial relations.

We also look into the expressiveness of the spatial calculus. We observe that it can be decided in the calculus whether an arbitrary semi-algebraic set is finite. From this it follows that various important integrity constraints are expressible. For instance, one can check in the calculus whether the database consists of a finite number of circles, polygonal line segments, closed polygons, lines, and points, as is often the case in practical applications.

Along the way, several open problems are indicated. For example, it is effectively decidable whether a semi-algebraic set is topologically connected. But is it decidable by a calculus query? It is in one dimension, and in at least three dimensions we show that the problem is related to another open problem, namely whether connectivity of a finite graph over the real numbers is expressible in the calculus. The case of two dimensions remains open. Another important open problem is to find query languages that are *complete* for the (different kinds of generic) spatial database queries. In this context, we briefly discuss ways to extend the calculus with an iteration mechanism.

It should be clear that our main purpose with this paper, rather than presenting a number of hard results, is to break up new grounds for the study of fundamental aspects of queries for spatial databases.

2   Preliminaries.

First-order logic of the reals.   Assume a totally ordered infinite set of variables called *real variables*. A *real term* is a polynomial in real variables with integer coefficients. A *real formula* is an arbitrary well-formed first-order formula built from *(i)* atomic formulas of the form $P \,\Theta\, Q$, where P and Q are real terms and Θ is one of $=$, $<$, $\leq$, $>$ or $\geq$; *(ii)* boolean operators; and *(iii)* quantifications $(\exists x)$ or $(\forall x)$ of real variables. The number of free variables of a formula is called its *arity*.

Let $\mathbf{R}$ be the set of real numbers. An m-ary real formula φ with free variables $x_1, \ldots, x_m$ defines a set of points $\{(x_1, \ldots, x_m) \mid \varphi\}$ in the Euclidean space $\mathbf{R}^m$ in the obvious way by letting real variables range over the real numbers. Sets thus definable are called *semi-algebraic*. Note that for $m = 0$, there are only two such sets: $\{(\,)\}$, which also stands for the Boolean constant *True*, and the

empty set which stands for *False*.

Two real formulas are *equivalent* if they define the same set. Since Tarski [Tar51] it is known that equivalence is decidable. Actually, every real formula can be effectively converted to an equivalent formula that is quantifier-free. Algorithms are known [Col75, Ren92] which perform this quantifier elimination in time which is exponential only in the number of variables of the formula (assuming prenex normal form). So, the algorithms have polynomial-time complexity if this number is constant. This will clearly be the case in our setting.

The spatial data model. Assume a set of *relation names*, where each relation name R has a *type* $\tau(R)$ which is a pair of natural numbers $[n, m]$. A *database scheme* is a finite set of relation names. Assume further a countably infinite domain $\mathbf{U}$ of atomic values.

An *intensional tuple* (or *ituple*) of type $[n, m]$ is a tuple $(a_1, \ldots, a_n; \varphi)$ with $a_1, \ldots, a_n \in \mathbf{U}$ and φ a quantifier-free real formula of arity m. An *i-instance* I of a scheme $\mathcal{S}$ now is a mapping on $\mathcal{S}$, assigning to each relation name R of $\mathcal{S}$ a finite set of ituples of type $\tau(R)$. Such a set of ituples is called an *i-relation* of type $\tau(R)$.

Intensional tuples and relations of type $[n, m]$ are finite representations of possibly infinite subsets of $\mathbf{U}^n \times \mathbf{R}^m$ which we call *extensional relations* (or *e-relations*), as follows. For an ituple $t = (a_1, \ldots, a_n; \varphi)$, let S be the semi-algebraic set defined by φ. Then t represents the e-relation $\{(a_1, \ldots, a_n)\} \times S$ which we denote by $ext(t)$. An i-relation r then represents the e-relation $ext(r) := \bigcup_{t \in r} ext(t)$. The elements of e-relations will be called *extensional tuples* (or *etuples*). *Extensional instances* and the e-instance $ext(I)$ represented by an i-instance I are now defined in the obvious way.

A relation type of the form $[n, 0]$ is called *flat*. Relations of flat type (or *flat relations*) are the standard value-based relations of classical relational databases. Similarly, we have flat schemes (containing only relation names with a flat type) and flat instances. In the flat case, there is no difference between the intensional and the extensional level.

On the other hand, relation types of the form $[0, m]$ are called *purely spatial*. Purely spatial databases are exactly those considered in the KKR

model [KKR90] mentioned in the Introduction.

The sharp distinction we make between ordinary data and spatial data allows a clean definition of the concepts introduced in the next section. This would not be possible if we would have chosen to represent the atomic values in $\mathbf{U}$ as, e.g., integers in $\mathbf{R}$, which seems what is suggested in [KKR90].

3 Generic spatial queries.

Definition 1 For a scheme $\mathcal{S}$, denote the set of i-instances (resp., e-instances) of $\mathcal{S}$ by $i\text{-}inst(\mathcal{S})$ (resp., $e\text{-}inst(\mathcal{S})$). Let $\mathcal{S}_{\text{in}}$ and $\mathcal{S}_{\text{out}}$ be disjoint schemes.

1. An *extensional query* of signature $\mathcal{S}_{\text{in}} \to \mathcal{S}_{\text{out}}$ is a partial function from $e\text{-}inst(\mathcal{S}_{\text{in}})$ to $e\text{-}inst(\mathcal{S}_{\text{out}})$ which, viewed as a binary relation on instances, is invariant under every permutation of $\mathbf{U}$.

2. An *intensional query* of signature $\mathcal{S}_{\text{in}} \to \mathcal{S}_{\text{out}}$ is a partial recursive function $Q : i\text{-}inst(\mathcal{S}_{\text{in}}) \to i\text{-}inst(\mathcal{S}_{\text{out}})$ for which there exists an e-query Q' of the same signature such that the following diagram commutes:

$$
\begin{array}{ccc}
i\text{-}inst(\mathcal{S}_{\text{in}}) & \xrightarrow{\ Q\ } & i\text{-}inst(\mathcal{S}_{\text{out}}) \\
\Big\downarrow{ext} & & \Big\downarrow{ext} \\
e\text{-}inst(\mathcal{S}_{\text{in}}) & \xrightarrow{\ Q'\ } & e\text{-}inst(\mathcal{S}_{\text{out}})
\end{array}
$$
∎

Restricted to flat schemes, i-queries as just defined coincide with the well-known *computable queries* for flat relational databases as defined by Chandra and Harel [CH80]; in particular, their *consistency criterion* (nowadays called *genericity*) is expressed by the condition in the definition of e-query. Note also that by this condition, for any (i- or e-) query Q, every atomic value appearing in $Q(I)$ already appears in I.

How is the notion of genericity best extended in the presence of spatial data, i.e., to non-flat databases? Informally, genericity says that a query has the same behavior on "isomorphic" databases. With respect to the ordinary data, isomorphisms are simply arbitrary permutations of the set of atomic values $\mathbf{U}$. But what are the isomorphisms

with respect to the spatial data? They will certainly be transformations of the Euclidean space, but precisely which kinds of transformations should be considered depends on the kind of geometry in which the spatial information is to be interpreted.

For instance, in some applications, the relative positions (e.g., above-below, left-right) among the different geometrical figures in the database form an essential part of the spatial information. This is for example the case in temporal databases dealing with time points on the real line [T+93]. For these applications, translations of the space are considered to be isomorphisms, but reflections are not. On the other hand, in geographical applications, which often deal exclusively with areas and distances, any distance-preserving transformation (i.e., any isometry) is considered an isomorphism. Still other applications may only be interested in the topological properties of the spatial information (e.g., [EF91]), and will think of isomorphisms as topological homeomorphisms. In summary, genericity of spatial database queries is best defined with respect to some general group of geometrical transformations.

Before we can give a formal definition, we need the following notion. A relation type of the form $[n, m]$ is said to be of *dimension* k if m is a multiple of k. Assume this is the case: $m = i \cdot k$. The intuition then is that in an etuple $(a_1, \ldots, a_n; u_1, \ldots, u_m)$ of type $[n, m]$, the m-tuple of real numbers $(u_1, \ldots, u_m)$ is thought of as an i-tuple of points in the k-dimensional Euclidean space $\mathbf{R}^k$. We can similarly talk about k-dimensional schemes (containing only relation names with a k-dimensional type) and instances. Given a k-dimensional instance I and a permutation g of $\mathbf{R}^k$, we can apply g to I in the obvious way.

Definition. Let $\mathcal{S}_{\mathrm{in}}$ and $\mathcal{S}_{\mathrm{out}}$ be two disjoint schemes of dimension k. Let $\mathcal{G}$ be a group of permutations of $\mathbf{R}^k$. An e-query Q of signature $\mathcal{S}_{\mathrm{in}} \to \mathcal{S}_{\mathrm{out}}$ is $\mathcal{G}$-*generic* if for any $g \in \mathcal{G}$ and any $I_1, I_2 \in e\text{-}inst(\mathcal{S}_{\mathrm{in}})$, if $g(I_1) = I_2$ then $g(Q(I_1)) = Q(I_2)$.

An i-query Q is $\mathcal{G}$-generic if its corresponding e-query Q' (cf. Definition 1) is. ∎

Let us now consider a few naturally occurring groups of transformations in a bit more detail: $\mathbf{T}$,

the group of translations; $\mathbf{D}$, the group of direct isometries; $\mathbf{I}$, the group of isometries; $\mathbf{S}$, the group of similarities; $\mathbf{A}$, the group of affinities; $\mathbf{H}$, the group of homeomorphisms; and $\mathbf{P}$, the full group of all permutations.

$\mathbf{P}$-genericity is an extreme form which enforces the standard genericity criterion of Chandra and Harel on the querying of spatial data. In other words, the points in space are considered to be abstract atomic values without any geometrical meaning, much like the atomic values in $\mathbf{U}$.

As already mentioned earlier, $\mathbf{H}$-genericity is most appropriate for applications that are interested only in the topological properties of the spatial data.

$\mathbf{A}$-genericity corresponds to applications where straightness of lines becomes important. With $\mathbf{S}$-genericity, also the length of straight line segments comes in. $\mathbf{A}$-genericity is appropriate for applications which consider any two figures with the same shape (e.g., a circle and an ellipse, or two triangles) to be isomorphic; $\mathbf{S}$-genericity captures the applications which do so only if the two figures also have the same proportions (e.g., two circles or two similar triangles).

As already mentioned earlier, $\mathbf{I}$-genericity is for applications in which the exact distances are important when considering two figures as isomorphic. $\mathbf{D}$-genericity is even stronger, also requiring that the orientation of angles is the same: direct isometries are translations or rotations. Finally, we have $\mathbf{T}$-genericity which, as already mentioned earlier, is relevant for temporal databases (in dimension 1).

Clearly, the stronger the notion of isomorphism is, the smaller the group $\mathcal{G}$ is and the more queries are $\mathcal{G}$-generic. We thus have a series of implications which we will now show to be strict. The proof of the following Proposition also serves to illustrate the different kinds of genericity by means of natural examples.

Proposition 1 $\mathbf{P}$-*generic* $\Rightarrow$ $\mathbf{H}$-*generic* $\Rightarrow$ $\mathbf{A}$-*generic* $\Rightarrow$ $\mathbf{S}$-*generic* $\Rightarrow$ $\mathbf{I}$-*generic* $\Rightarrow$ $\mathbf{D}$-*generic* $\Rightarrow$ $\mathbf{T}$-*generic. These implications hold for i-queries (whence also for e-queries) and are strict.*

Proof. (Sketch) We work in dimension 2. Consider a database scheme $\{Lives, Region\}$, where *Lives* is of type $[1, 2]$ and *Region* is of type $[0, 2]$.

On the extensional level, each tuple in *Lives* is of the form $(n; x, y)$ where n is (the name of) some person and (x, y) is a location where this person lives. *Region* contains some region, i.e., a semi-algebraic set in the Euclidean plane. The following queries have an output scheme consisting of a single relation.

The query "give the persons who live in the region" (resulting in a relation of type $[1, 0]$) is **P**-generic.

The query "give the persons who live in the topological interior of the region" is **H**-generic but not **P**-generic. Another example of a query with this property is "give the topological boundary of the region" (of result type $[0, 2]$).

The query of result type $[0, 2]$ "give the straight lines contained in the region" is **A**-generic but not **H**-generic.

The query "give the persons who live closest to the region" is **S**-generic but not **A**-generic. Another such query (of result type $[0, 0]$) is "does the region consist of two orthogonal finite-length line segments?"

The query "give the equilateral triangles with sides of length 1 contained in the region" is **I**-generic but not **S**-generic. Another such query is "give the pairs of persons living at least 10 miles apart".

Assume the region is a circle and that persons live on this circle. The query "give the pairs (n_1, n_2) such that person n_1 lives before person n_2 in clockwise order" is **D**-generic but not **I**-generic.

Finally, the query "give the pairs (n_1, n_2) such that person n_1 lives west of person n_2" is **T**-generic but not **D**-generic. ∎

We conclude this section with the following remark. Asking for the locations where John lives, though reasonable as a query, is strictly speaking not a query at all since it gives special status to the atomic value 'John' and is therefore not invariant under all permutations of **U** (cf. Definition 1). This problem goes away if we provide a constant relation *John* containing the single atomic value 'John', and rephrase the query correspondingly. This is of course merely an alternative view on the well-known notion of *C-genericity* [HY84], a slight generalization of classical genericity allowing queries which depend on some finite set of constants C.

Under this alternative view, we can generalize the notion of C-genericity to the spatial context, appropriate for a finite set of *spatial constants* (which can be arbitrary semi-algebraic sets in general). As a simple example where the constant is a single point, consider a relation R of type $[0, 2]$ and the query (of result type $[0, 2]$) which rotates R over 90° around the center $(0, 0)$. This query is not even **T**-generic since the center point is given special status. However, given two relations R_1 and R_2 of type $[0, 2]$, the query which checks whether R_2 consists of a single point p and, if so, rotates R_1 over 90° around p certainly is **T**-generic (it is actually **S**-generic).

4 Calculus and algebra.

We now present a basic mechanism for specifying spatial queries in the form of a calculus language which is an orthogonal combination of the standard relational calculus and one of the constraint query languages of [KKR90]. The operational semantics of the calculus is provided by an equivalent algebra.

Recall the language of real formulas defined in Section 2. The formulas of the calculus query language are obtained simply by adding to the former language:

- A totally ordered infinite set of variables called *value variables*, disjoint from the set of real variables;

- Atomic formulas of the form $v_1 = v_2$ or $R(v_1, \ldots, v_n; p_1, \ldots, p_m)$ where R is a relation name of type $[n, m]$, the v_i are value variables and the p_i are real terms (the latter kind of atomic formula is called a *relation atom*); and

- Quantifications $(\exists v)$ or $(\forall v)$ of value variables.

Let Φ be a calculus formula with free value variables $v_1, \ldots, v_n$ and free real variables $x_1, \ldots, x_m$. Let $\mathcal{S}$ be a scheme containing all relation names occurring in Φ. Given an e-instance I of $\mathcal{S}$, Φ defines an e-relation $\Phi(I)$ of type $[n, m]$ in the obvious way by letting value variables range over the set of all atomic values appearing in I and letting real variables range over the real numbers.

The calculus can thus be used as a declarative language for expressing queries on the extensional level. It is, of course, more interesting how-

ever to express i-queries (which are effectively computable). This turns out to be possible: every e-query expressible in the calculus corresponds (in the sense of Definition 1) to an i-query. We show this by translating calculus formulas into equivalent algebra expressions which have an operational semantics on the intensional level. The operators of this algebra are presented next.

We will use the following terminology: given an ituple $t = (a_1, \ldots, a_n; \varphi)$, we will denote the *value part* $(a_1, \ldots, a_n)$ of t by $val(t)$ and the *spatial part* φ of t by $spat(t)$.

Now let r, r_1, and r_2 be i-relations of type $[n, m]$ and let r' be an i-relation of type $[n', m']$. Without loss of generality, assume that all real formulas appearing in r, r_1 and r_2 use the same variables $x_1, \ldots, x_m$, and that no real variable is used both in r and r'.

- The *union* $r_1 \cup r_2$ is the standard set-theoretic union.

- For an arbitrary ituple t, denote the set $\{t' \in r_i \mid val(t') = val(t)\}$ by $veq_i(t)$, for $i = 1, 2$. The *difference* $r_1 - r_2$ equals $\{(val(t); \bigvee_{t' \in veq_1(t)} spat(t') \wedge \neg \bigvee_{t' \in veq_2(t)} spat(t')) \mid t \in r_1\}$.

- The *Cartesian product* $r \times r'$ equals $\{(val(t), val(t'); spat(t) \wedge spat(t')) \mid t \in r, t' \in r'\}$, of type $[n + n', m + m']$.

- The *value selection* $\sigma_{i=j}(r)$ equals $\{t \in r \mid val(t)(i) = val(t)(j)\}$.

  Let φ be a quantifier-free real formula on the variables $x_1, \ldots, x_m$. The *spatial selection* $\sigma_\varphi(r)$ equals $\{(val(t); spat(t) \wedge \varphi) \mid t \in r\}$.

- For $1 \le i_1, \ldots, i_p \le n$, the *value projection* $\pi_{i_1, \ldots, i_p}(r)$ equals

  $$\{(val(t)(i_1), \ldots, val(t)(i_p); spat(t)) \mid t \in r\},$$

  of type $[p, m]$.

  Let $1 \le i_1, \ldots, i_p \le m$ and let $y_1, \ldots, y_p$ be real variables different from each x_i. For a real formula φ with free variables $x_1, \ldots, x_m$, define $\pi_{x_{i_1}, \ldots, x_{i_p}}(\varphi)$ as the quantifier-free equivalent of the real formula

  $$(\exists x_1) \ldots (\exists x_m)(\varphi \wedge \bigwedge_{\ell=1}^{p} y_\ell = x_{i_\ell}).$$

Then the *spatial projection* $\pi_{x_{i_1}, \ldots, x_{i_p}}(r)$ equals $\{(val(t); \pi_{x_{i_1}, \ldots, x_{i_p}}(spat(t))) \mid t \in r\}$, of type $[n, p]$.

- Finally, the algebra includes the constant $\mathbf{R}^k$ for each k, standing for the "full" relation of type $[0, k]$.

Algebra expressions are obtained by applying algebra operators to relation names. Using standard techniques we can prove:

Proposition. *Every calculus formula Φ can be effectively converted into an algebra expression E such that the e-query expressed by Φ corresponds to the i-query expressed by E.*

There is also an obvious converse to this proposition which we do not state explicitly. We can thus conclude that the calculus and the algebra are equivalent.

Example. All queries mentioned in the proof of Proposition 1 are expressible in the calculus (whence also in the algebra). An easy one is "give the pairs of persons living at least 10 miles apart": $\pi_{1,2}\sigma_{(x_3-x_1)^2+(x_4-x_2)^2 \ge 100}(Lives \times Lives)$ in the algebra.

To conclude this section we point out (proof omitted) that the spatial calculus (algebra) is a conservative extension of the standard relational calculus (algebra).

Proposition. *Every calculus query of flat signature is expressible in the flat relational calculus.*

Also, it is clear that every calculus query of purely spatial signature is expressible in the KKR calculus [KKR90] mentioned in the Introduction.

5 Genericity of calculus queries.

Not all calculus queries are generic. The simplest example is the trivial query '$\{x \mid x = 0\}$' defined on the 1-dimensional scheme $\mathcal{S}_1 = \{R\}$ with $\tau(R) = [0, 1]$. This query is not **T**-generic. Actually, it is not $\mathcal{G}$-generic for any $\mathcal{G}$ which contains a transformation that does not fix 0.

Hence, there is a "consistency problem" for the calculus. The following theorem witnesses the difficulty of this problem even in the simplest possible setting:

Theorem 1 *With $\mathcal{G}$ and $\mathcal{S}_1$ as above, $\mathcal{G}$-genericity of calculus queries of signature $\mathcal{S}_1 \to \mathcal{S}_1$ is undecidable.*

Proof. (Sketch) 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 $enc(n) := \{0, \ldots, n\}$, and encode a vector of natural numbers $(n_1, \ldots, n_k)$ by $enc(n_1) \cup enc(n_2) + n_1 + 1 \cup \cdots \cup enc(n_k) + n_{k-1} + 1$. The corresponding decoding is first-order. We then reduce a $\forall^*$-sentence $(\forall \vec{x}) \varphi(\vec{x})$ of number theory to the query:

if R encodes a vector $\vec{x}$ **then**
  if $\varphi(\vec{x})$ **then** $\emptyset$
  else $\{0\}$
else $\emptyset$

This query is expressible in the calculus and is generic iff the sentence is valid. ∎

In the same way, it can be shown that satisfiability of (whence also equivalence among) calculus queries is undecidable; this was independently shown by Grumbach and Su [GS] in the context of a model and calculus for "finitely representable databases" equivalent to ours. Note that equivalence of purely spatial *tableau* calculus queries (defined in analogy to the flat tableau queries of [ASU79]) was shown decidable in [KKR90]. This result can be extended to general tableau queries (proof omitted). This leaves us with:

Open problem. *Is genericity of tableau calculus queries decidable?*

We also point out that for any group $\mathcal{G}$ of transformations that can be defined in elementary algebra (in particular for any natural subgroup of the group of affine transformations with algebraic parameters), $\mathcal{G}$-genericity of purely spatial calculus queries is co-r.e., since for any fixed transformation g, query Q, and i-instance I, the statement $g(Q(I)) = Q(g(I))$ can be written as a sentence in the decidable first-order theory of the reals.

If we disallow quantification, the consistency problem seems to become more manageable. We present one theorem that is probably typical for the kind of results that can be expected in this situation.

Consider the simple 2-dimensional scheme consisting of a single relation R of type $[n, 2]$ for some n. Call a calculus query of result type $[n', 2]$ over this scheme *simple quantifier-free* if its defining formula $\Phi(\vec{v}; x, y)$ has the form $\Phi_1 \wedge \cdots \wedge \Phi_r \wedge \Phi'$, where each Φ_i is a relation atom or its negation and Φ' is a quantifier-free formula not involving R. The class of simple quantifier-free queries captures the "global" operations (and selections, using Φ') on the spatial data, such as a translation over a constant vector $\vec{a}$ of all points in R: $\{(\vec{v}, \vec{x}) \mid R(\vec{v}; \vec{x} - \vec{a})\}$.

Theorem. *Every* **D**-*generic simple quantifier-free query must be* **P**-*generic.*

In other words, only unsophisticated (from a spatial point of view) simple quantifier-free queries can be **D**-generic. The proof (omitted) is quite ad-hoc and proceeds by elimination of polynomials.

6 Expressiveness of the calculus.

A useful property is the following:

Proposition 2 *Let $\Phi(\vec{v}; \vec{x}, \vec{y})$ be a calculus query with free value variables $\vec{v}$ and free real variables $\vec{x}, \vec{y}$. The query $\{(\vec{v}; \vec{y}) \mid \{\vec{x} \mid \Phi(\vec{v}; \vec{x}, \vec{y})\}$ is finite$\}$ is expressible in the calculus.*

Proof. (Sketch) The set $\{\vec{x} \mid \Phi(\vec{v}; \vec{x}, \vec{y})\}$ is a semi-algebraic set parameterized by $\vec{v}$ and $\vec{y}$. Since every semi-algebraic set is homeomorphic to a union of singletons and open cubes [BCR87], such a set is finite iff it does not have an accumulation point. The topological notion of accumulation point is first-order definable in metric spaces and hence expressible in the calculus. ∎

In other words, Proposition 2 says that the generalized quantifier $(\exists^{fin} \vec{x})$ ("there exist only finitely many $\vec{x}$") is expressible in the calculus.[1] So, $(\forall n)(\exists^{fin} x, y) Lives(n; x, y)$ expresses that every

---

[1] A weaker version of this fact for the first-order theory of the reals was already known [Sch79].

person lives only on a finite number of points. Importantly, $\exists^{fin}$ can also be used to recognize properties of *infinite* semi-algebraic sets. Indeed, many types of geometrical figures used in practical situations consists of an infinite number of points, but can be described using only a finite number. For instance, polygons can be described their vertices, and circles by their center and radius. Moreover, the finite description is often first-order definable (expressible in the calculus). Given any fixed collection of such types of figures, it can then be verified in the calculus whether the spatial information in the database consists of a finite number of figures of these types.

Example. As a simple example, we can verify whether a relation R of type $[0, 2]$ contains only a finite number of line segments. If the endpoints of a line segment are (x_1, y_1) and (x_2, y_2), then the subformula $(\forall x)(\forall y)(\forall r) : (0 \leq r \leq 1 \land (x, y) = (x_1, y_1) + r(x_2 - x_1, y_2 - y_1)) \Rightarrow R(x, y)$ expresses that all points lying between the endpoints are in R, and the subformula $(\exists s)(\forall x)(\forall y)(\forall r) : (s < r < 0 \land ((x, y) = (x_1, y_1) + r(x_2 - x_1, y_2 - y_1) \lor (x, y) = (x_1, y_1) + (1 - r)(x_2 - x_1, y_2 - y_1))) \Rightarrow \neg R(x, y)$ expresses that R does not extend outside the endpoints. Calling the first subformula $\Phi(x_1, y_1, x_2, y_2)$ and the second $\Psi(x_1, y_1, x_2, y_2)$, the formula we need is thus $(\exists^{fin} x_1, y_1, x_2, y_2)(\Phi \land \Psi)$.

Recently, Grumbach and Su [GS] observed that the compactness property known from Model Theory fails for (an equivalent formulation of) our model of spatial databases. An elementary corollary of compactness is that a calculus sentence having *finite* models (i.e., instances on which the sentence yields True) of arbitrary cardinality also has an infinite model. Proposition 2 shows that this corollary (whence also compactness) fails as well, since finiteness is definable in the calculus.

Since the calculus can only express i-queries in PTIME, there are many queries it cannot express. However, restricting attention to purely spatial databases, we conjecture that also many PTIME queries are not expressible in the calculus. An obvious candidate is the query to decide whether a finite e-relation of type $[0, 2]$ is connected when viewed as a directed graph. We thus have the following open problem, also stated as an open problem in [GS]:

Open problem 1 *Is graph-theoretic connectivity expressible in the calculus?*

There is also another notion of connectivity entirely different from the graph-theoretic one, which comes in naturally in the context of spatial data. Recall that an e-relation of type $[0, k]$ is a (possibly infinite) semi-algebraic set of points in k-dimensional space $\mathbf{R}^k$. This set may or may not be connected in the classical topological sense. The query to decide topological connectivity in dimension k is in PTIME [HRR91]. For $k = 1$, it is easily expressed in the calculus. However, we ask:

Open problem 2 *Let $k \geq 2$. Is topological connectivity in dimension k expressible in the calculus?*

Graph-theoretic connectivity (Open problem 1) and topological connectivity (Open problem 2) appear unrelated in general. Nevertheless, we can show:

Proposition. *Let $k \geq 3$. Graph-theoretic connectivity of a finite e-relation of type $[0, 2]$ can be reduced, in the calculus, to topological connectivity in dimension k.*

Proof. (Sketch) Consider the case $k = 3$. We define a calculus query Q working on finite e-relations R of type $[0, 2]$, such that R is graph-theoretically connected if and only if $Q(R)$, a semi-algebraic set in dimension 3, is topologically connected. Denote the set of all vertices of R (viewed as a graph) by V. The set $Q(R)$ contains a number of vertical lines situated orthogonally to the graph of the parabola $y = x^2$ in the xy-plane. For each vertex a in V, there is one such line, going through the point $(a, a^2, 0)$. Furthermore, $Q(R)$ contains horizontal line segments linking the different vertical lines. For each edge (a, b) in R, there is one such line segment, connecting the vertical a-line with the vertical b-line, situated at height $Ma/m + b$. Here, M is one plus the maximum $\max_{a \neq a' \in V} |a - a'|$, and m is the minimum $\min_{a \neq a' \in V} |a - a'|$ minus one. It can be verified that for any two vertices a and b of R, the a-line and the b-line are topologically connected if and only if there is a path in the graph R from a to b. Furthermore, Q is expressible in the calculus. Hence, the claim follows. ∎

We have thus shown that if Open problem 1 is settled in the negative, then Open problem 2 is settled in the negative as well for $k \geq 3$. The case of dimension two remains wide open.

7   Complete languages

The design of complete query languages in our framework is an important subject for further research. For flat queries, it is well-known that using the flat relational calculus as the basis of a programming language with assignment statements, untyped relation variables, and iteration, yields a complete language known as QL [CH80]. We can similarly build a language starting from the spatial calculus, and call it SpQL.

Open problem. *Is every i-query expressible in SpQL?*

In [KKR90], it was observed that the calculus cannot be extended with declarative iteration based on least fixpoints, since such an extension would no longer be closed for the spatial data model. But (partially defined) procedural iteration based on conventional while-loops can of course still be used.

Another problem is finding languages complete not for all i-queries, but for all $\mathcal{G}$-generic ones, for various groups of transformations $\mathcal{G}$ (e.g., those considered in Section 3). In view of our undecidability result, for this problem it is perhaps more appropriate to have primitives for the querying of spatial data that are more specific to $\mathcal{G}$-generic geometry than general first-order logic. For instance, in the case of **S**-genericity (corresponding to Euclidean geometry), the programming language for geometrical constructions in Euclidean geometry defined in [Eng93] might be a good starting point.

References

[AO93]   D. Abel and B.C. Ooi, editors. *Advances in spatial databases—3rd Symposium SSD'93*, volume 692 of *Lecture Notes in Computer Science*. Springer-Verlag, 1993.

[Arn88]   D.S. Arnon. Geometric reasoning with logic and algebra. *Artificial Intelligence*, 37:37–60, 1988.

[ASU79]   A.V. Aho, Y. Sagiv, and J.D. Ullman. Equivalences among relational expressions. *SIAM Journal on Computing*, 8(2):218–246, 1979.

[AU79]   A.V. Aho and J.D. Ullman. Universality of data retrieval languages. In *Proceedings of the ACM Symposium on Principles of Programming Languages*, pages 110–120, 1979.

[BCR87]   J. Bochnak, M. Coste, and M.-F. Roy. *Géométrie algébrique réelle*. Springer-Verlag, 1987.

[Buc89]   A. Buchmann, editor. *Design and implementation of large spatial databases—First Symposium SSD'89*, volume 409 of *Lecture Notes in Computer Science*. Springer-Verlag, 1989.

[CH80]   A. Chandra and D. Harel. Computable queries for relational database systems. *Journal of Computer and System Sciences*, 21(2):156–178, 1980.

[Col75]   G.E. Collins. Quantifier elimination for real closed fields by cylindrical algebraic decomposition. *Lecture Notes in Computer Science*, 33:134–183, 1975.

[EF91]   M.J. Egenhofer and R.D. Franzosa. Point-set topological spatial relations. *Int. J. Geographical Information Systems*, 5(2):161–174, 1991.

[Eng93]   E. Engeler. *Foundations of Mathematics*. Springer-Verlag, 1993.

[GS]   S. Grumbach and J. Su. Finitely representable databases. These *Proceedings*.

[GS91]   O. Gunther and H.-J. Schek, editors. *Advances in spatial databases—2nd Symposium SSD'91*, volume 525 of *Lecture Notes in Computer Science*. Springer-Verlag, 1991.

[HRR91]   J. Heintz, T. Recio, and M.-F. Roy. Algorithms in real algebraic geometry and applications to computational geometry. In W. Steiger J. Goodman, R. Pollack,

editor, *Discrete and computational geometry*, volume 6 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*. AMS-ACM, 1991.

[HY84]   R. Hull and C.K. Yap. The format model, a theory of database organization. *Journal of the ACM*, 31(3):518–537, 1984.

[KKR90] P.C. Kanellakis, G.M. Kuper, and P.Z. Revesz. Constraint query languages. *Journal of Computer and System Sciences*, to appear. See also *Proceedings 9th ACM Symposium on Principles of Database Systems*, pages 299–313, 1990.

[Ren92] J. Renegar. On the computational complexity and geometry of the first-order theory of the reals. *Journal of Symbolic Computation*, 13, 1992.

[Sch79] W. Schwabhäuser. Addendum. *Fund. Math.*, 103:101, 1979.

[T⁺93]   A. Tansel et al., editors. *Temporal Databases: Theory, Design, and Implementation*. Benjamin/Cummings, 1993.

[Tar51] A. Tarski. *A Decision Method for Elementary Algebra and Geometry*. University of California Press, 1951.

[Tar86] A. Tarski. What are logical notions? *History and Philosophy of Logic*, 7:143–154, 1986. Edited by J. Corcoran.