# Safe Constraint Queries

Michael A. Benedikt and Leonid Libkin

June 1998

# Safe Constraint Queries

Michael Benedikt
Bell Laboratories
1000 E Warrenville Rd
Naperville, IL 60566
E-mail: benedikt@research.bell-labs.com

Leonid Libkin
Bell Laboratories
600 Mountain Avenue
Murray Hill, NJ 07974
E-mail: libkin@research.bell-labs.com

## Abstract

We extend some of the classical characterization theorems of the relational theory — particularly those related to query safety — to the context where database elements come with fixed interpreted structure, and where formulae over elements of that structure can be used in queries. We show that the addition of common interpreted functions such as real addition and multiplication to the relational calculus preserves important *characterization theorems* of the relational calculus, and also preserves certain *combinatorial properties* of queries. Our main result of the first kind is that there is a syntactic characterization of the collection of safe queries over the relational calculus supplemented by a wide class of interpreted functions — a class that includes addition, multiplication, and exponentiation — and that this characterization gives us an interpreted analog of the concept of range-restricted query from the uninterpreted setting. Furthermore, our range-restricted queries are particularly intuitive for the relational calculus with real arithmetic, and give a natural syntax for safe queries in the presence of polynomial functions. We use these characterizations to show that safety is decidable for Boolean combinations of conjunctive queries for a large class of interpreted structures. We show a dichotomy theorem that sets a polynomial bound on the growth of the output of a query that might refer to addition, multiplication and exponentiation.

We apply the above results for finite databases to get results on finitely representable databases. We start by getting syntactic characterizations of the queries on finitely representable databases that preserve geometric conditions in the constraint data model. We consider classes of convex polytopes, polyhedra, and compact semi-linear sets, the latter corresponding to many spatial applications. We show how to give an effective syntax to safe queries, and prove that for conjunctive queries the preservation properties are decidable.

## 1 Introduction

The power of classical query languages is linked to the fact that they express a restricted class of declarative programs. The class of semantic objects expressible through queries in the relational calculus, for example, is limited in a number of helpful ways: each such query is polynomial-time computable, each is local, and each has well-defined asymptotics. Although relational calculus queries may not return finite results, a natural subclass of the relational calculus does: namely the class of *range-restricted queries.* This class gives guarantees of finite output and is complete in this respect: it captures all relational calculus queries whose outputs are always finite, the *safe* queries.

The relational theory on which these results are based deals only with *pure* relational queries: that is, those containing no interpreted predicates. Practical query languages, in contrast, contain interpreted functions such as $+$ and $*$. The resulting queries, then, make use of the domain semantics, rather than being independent of them as pure relational queries are. For example, if the underlying structure is the field of real numbers $\langle \mathbb{R}, +, *, 0, 1, < \rangle$, the extension of relational calculus is achieved by using polynomial (in)equalities. For example, the query $\varphi(x, y) \equiv \exists z. R(x, z) \wedge R(z, y) \wedge x^2 + y^2 = z$ defines a subset of the self-join with the condition that in joinable tuples $(x, z)$ and $(z, y)$, $z$ must be the sum of squares of $x$ and $y$. A natural question, then, is what sort of restrictions still apply to queries given with interpreted structure.

Clearly, many standard results fail in the presence of interpreted structure: locality and almost-sureness fail as soon as an order is available. Complexity bounds are often dependent on fragile properties of both the functions present and the encodings of the structures in some computational model. In contrast, we show here that certain kinds of *structural* properties of relational calculus queries remain when a reasonable interpreted structure is present. These include the classical equivalence of safe and range-restricted queries, decidability of safety for restricted classes of queries, as well as *combinatorial* properties of the queries; restrictions on the growth rate of the result sets, for example. A primary example of well-behaved combinatorics of these structures is a *growth dichotomy theorem*, which says that the output of a query is either polynomial in the database or infinite. We show that the well-behavedness of a structure, together with the decidability of its first-order theory, has *algorithmic* consequences; for example, the set of range-restricted formulae can be effectively computed.

A problem related to safety is that of *state-safety*, studied in [3]: for a query *and* a database, determine if the output is finite. Unlike the safety problem, which is undecidable (cf. [1]), the state-safety problem is decidable for some domains, for example, natural numbers with the order relation, see [3, 5]. However, there are interpreted structures (even having a decidable first-order theory) for which this problem is undecidable, and the class of safe queries cannot be described syntactically [32]. In contrast, we show that for many "well-behaved" structures, state-safety is decidable, safe queries can be captured by a syntactically defined class, and safety is even decidable for restricted classes of queries.

The above results are for the standard relational calculus with interpreted functions on finite structures; we then apply these results to get structural restrictions on the behavior of queries in the constraint database model of Kanellakis, Kuper and Revesz [16]. This model is motivated by new applications involving spatial and temporal data, which require storing and querying infinite collections. It generalizes the relational model by means of "generalized relations". These are possibly infinite sets defined by quantifier-free first-order formulae in the language of some underlying infinite structure $\mathcal{M} = \langle \mathcal{U}, \Omega \rangle$. Here $\mathcal{U}$ is a set (assumed to be infinite), and $\Omega$ is a signature that consists of a number of interpreted functions and predicates over $\mathcal{U}$. For example, in spatial applications, $\mathcal{M}$ is usually the real field $\langle \mathbb{R}, +, *, 0, 1, < \rangle$, and generalized relations describe sets in $\mathbb{R}^n$.

A database given by a quantifier-free formula $\alpha(x_1, \ldots, x_n)$ in the language of $\Omega$ defines a (possibly infinite) set $D_\alpha = \{\vec{a} \in \mathcal{U}^n \mid \mathcal{M} \models \alpha(\vec{a})\}$. Such databases are called *finitely representable* [12], as the formula $\alpha$ provides a finitary means for representing an infinite set. For example, if $\alpha(x, y) \equiv (x^2 + y^2 \leq 1)$, then $D_\alpha$ is the circle of radius 1 with the center in $(0, 0)$.

Relational calculus can be straightforwardly extended to this model, by incorporating atomic formulas which are $\Omega$-constraints, that is, atomic $\Omega$-formulae. For example, $\varphi(x, y) \equiv (D(x, y) \wedge y = x^2)$ is a first-order query which, on $D_\alpha$ defined above, returns the intersection of the circle with the graph of the function $y = x^2$.

One of the reasons why the constraint model can be used in spatial applications is that such queries admit a form of safety: the closed form evaluation over structures $\langle \mathbb{R}, +, -, 0, 1, < \rangle$ (linear constraints) and $\langle \mathbb{R}, +, *, 0, 1, < \rangle$ (polynomial constraints), most often used to represent spatial data. This sort of closure is a reformulation of the fact that the two structures above admit *effective quantifier-elimination*. To evaluate a query, one can replace each occurrence of a database symbol $D$ by its representation as a collection of constraints and apply the quantifier-elimination procedure to the resulting formula, to obtain a finite representation of the output. There has been work in extending these closure properties to other classes of constraint databases and other logics, and this work indicates that the existence of closure properties is often problematic. For example, for integer gap-order constraints $x <_n y$ (meaning $x < y + n$), restrictions guaranteeing safety were studied in [28] for relational calculus and stratified datalog, and the inherent incompleteness of those restrictions was later shown in [34]. However, in this paper we do obtain new positive results on closure properties for constraint queries— albeit of a different nature than [28, 34] — and we do so in domains that are quite relevant to spatial applications.

For those domains, we consider the preservation of restricted geometric classes of databases within powerful constraint query languages. In particular, we consider the behavior of queries with polynomial functions over *linear* constraint databases. Linear constraints are used to represent spatial data in many applications, see [10, 13, 25] and references therein. Linear constraints have several advantages over polynomial: the quantifier-elimination procedure is less costly, and numerous algorithms have been developed to deal with figures represented by linear constraints, cf. [22]. At the same time, the extension of relational calculus with linear constraints has severely limited power as a query language, see [2, 25]. Thus, it appears to be natural to use a more powerful language, such as relational calculus with polynomial constraints, to query databases represented by linear constraints.

As soon as the class of constraints used in queries is more general than the class used to define databases, we encounter the safety/closure property again: the output of query using polynomial constraints may fail to be definable with linear constraints alone! More generally, if spatial databases are required to have certain geometric properties, then the safety problem is whether those geometric properties are preserved by a given query language.

For relational calculus with polynomial constraints, there is a recursively enumerable class of programs that express exactly those queries that preserve the property of being definable with linear constraints; this follows from the decidability of the latter property, shown in [10]. We are interested in getting explicit and natural complete languages for preserving various geometric properties of linear-constraint databases. We give a general schema for coming up with such languages, based on the results from the first part of the paper. As applications, we consider convex polytopes, convex polyhedra, and compact semi-linear sets. We provide an effective syntax for polynomial constraint queries preserving these properties. We also show that for unions of conjunctive queries with polynomial constraints, it is decidable whether the properties of being a convex polytope or a compact semi-linear set are preserved. We find restrictions of the the growth in the number of vertices of polytopes, and use them to show that certain kinds of triangulations cannot be done even with very powerful constraints.

**Organization** In Section 2, we present the notations. Sections 3 through Section 6 all deal primarily with the standard relational calculus with interpreted functions (although generalizations to the "natural semantics" hold, and are given here, as well). Section 3 shows that the underlying interpreted structure one uses does matter: we define the concept of a *safe translation* of queries, and show that some structures admit it, and some don't. It follows that for many common structures the state-safety problem is decidable.

In Section 4, we define the concept of range-restriction, and show that the classes of safe and range-restricted queries coincide over well-behaved structures. The general concept of range-restriction is based on the notion of algebraic formulae in the underlying model. We show that for polynomial functions, these range-restricted queries have a particularly nice characterization, namely as queries that are bounded by roots of polynomials with coefficients from the database. We then show that for underlying structures admitting quantifier-elimination, it is possible to construct, effectively, a range-restricted query that coincides with a given query $Q$ on all databases for which $Q$ is safe.

Section 5 shows that for well-behaved structures, it is decidable if a Boolean combination of conjunctive queries is safe. Section 6 proves the dichotomy result: for every query $Q$ over a well-behaved structure, one can find a polynomial $p$ such that the size of $Q(D)$ is either infinite, or at most the value of $p$ on the size of $D$.

Section 7 deals with finitely representable databases. We first introduce a general schema for transferring results about query safety to the finitely-representable setting. We then give effective syntax for queries preserving geometric properties, such as the classes of convex polytopes, polyhedra and compact semi-linear sets (the latter only in two-dimensional case). We also show that it is decidable whether unions of conjunctive queries with polynomial constraints preserve the first and the third property. Concluding remarks are given in Section 8. All proofs can be found in the appendix.

# 2 Notations

The notations we use are fairly standard in the literature on constraint databases, cf. [6, 8, 24, 25]. We study databases over infinite structures. Let $\mathcal{M} = \langle \mathcal{U}, \Omega \rangle$ be an infinite structure, where $\mathcal{U}$ is an infinite set, called a carrier (in the database literature it is often called domain), and $\Omega$ is a set of interpreted functions, constants, and predicates. For example, for the real field $\langle \mathbb{R}, +, *, 0, 1, < \rangle$, the carrier is $\mathbb{R}$ (the set of real numbers), and the signature consists of the functions $+$ and $*$, constants 0 and 1, and predicate $<$.

A (relational) *database schema* $SC$ is a nonempty collection of relation names $\{S_1, \ldots, S_l\}$ with associated arities $p_1, \ldots, p_l > 0$. Given $\mathcal{M}$, an *instance* of $SC$ over $\mathcal{M}$ is a family of finite sets, $\{R_1, \ldots, R_l\}$, where $R_i \subset \mathcal{U}^{p_i}$. That is, each schema symbol $S_i$ of arity $p_i$ is interpreted as a finite $p_i$-ary relation over $\mathcal{U}$. Given an instance $D$, $adom(D)$ denotes its *active domain*, that is, the set of all elements that occur in the relations in $D$. We normally assume $adom(D) \neq \emptyset$. Although often convenient in simplifying notation, this restriction is by no means necessary, as all results straighforwardly extend to empty databases.

As our basic query language, we consider relational calculus, or *first-order logic*, FO, over the underlying models and the database schema. In what follows, $L(SC, \Omega)$ stands for the language that contains all symbols of $SC$ and $\Omega$; by $\mathrm{FO}(SC, \Omega)$ we mean the class of all first-order formulae built up from the atomic $SC$ and $\Omega$-formulae by using Boolean connectives $\vee, \wedge, \neg$ and quantifiers $\forall, \exists$ and $\forall x \in adom, \exists x \in adom$. When $\Omega$ is $(+, -, 0, 1, <)$, or $(+, *, 0, 1, <)$, or $(+, *, e^x, 0, 1, <)$, we use the standard abbreviations FO + Lin, FO + Poly and FO + Exp, often omitting the schema when it is understood. Regardless of whether we are in the 'classical' setting, where these queries are applied to finite databases, or in the constraint query setting discussed later in the paper, we will refer to the syntactic query languages as *relational calculus with $\Omega$ constraints*.

The semantics is as follows. For a structure $\mathcal{M}$ and a $SC$-instance $D$, the notion of $(\mathcal{M}, D) \models \varphi$ is defined in a standard way for $\mathrm{FO}(SC, \Omega)$ formulae, where $\exists x \in adom$ is the *active-domain* quantification. That is, $(\mathcal{M}, D) \models \exists x\ \varphi(x, \cdot)$ if for some $a \in \mathcal{U}$ we have $(\mathcal{M}, D) \models \varphi(a, \cdot)$, and $(\mathcal{M}, D) \models \exists x \in adom\ \varphi(x, \cdot)$ if for some $a \in adom(D)$ we have $(\mathcal{M}, D) \models \varphi(a, \cdot)$. If $\mathcal{M}$ is understood, we write $D \models \varphi$. The output of a query $\varphi(x_1, \ldots, x_n)$ on $D$ is $\{\vec{a} = (a_1, \ldots, a_n) \in \mathcal{U}^n \mid D \models \varphi(\vec{a})\}$, and it is denoted by $\varphi[D]$. For example, $\varphi(x, y) \equiv (S(x, y) \wedge y = x * x)$ is a $\mathrm{FO} + \mathrm{POLY}$ query over the schema that contains one binary relation $S$; and $\varphi[D]$ is the set of pairs in $(x, y)$ in $D$ where $y = x^2$.

We now, following [17, 18] say that a $\mathrm{FO}(SC, \Omega)$ query $\varphi(\vec{x})$ is *safe* for a $SC$-database $D$ if it has finitely many satisfiers for $D$; that is, $\varphi[D]$ is finite. A query is safe if it is safe for all databases.

As we explained before, we need to distinguish a class of "well-behaved" models. Following [6, 8], we use *o-minimality* [26, 36] and quantifier-elimination [9] for this purpose. We say that $\mathcal{M}$ is o-minimal, if every definable set is a finite union of points and open intervals $\{x \mid a < x < b\}$ (we assume that $<$ is in $\Omega$). Definable sets are those of the form $\{x \mid \mathcal{M} \models \varphi(x)\}$, where $\varphi$ is a first-order formula in the language of $\mathcal{M}$, possibly supplemented with symbols for constants from $\mathcal{M}$. We say that $\mathcal{M}$ admits *quantifier-elimination (QE)* if, for every formula $\varphi(\vec{x})$, there is an equivalent quantifier-free formula $\psi(\vec{x})$ such that $\mathcal{M} \models \forall \vec{x}. \varphi(\vec{x}) \leftrightarrow \psi(\vec{x})$. Below we list the most important examples, that correspond to classes of interpreted structures and constraints often used in applications.

**Linear Constraints:** $\langle \mathbb{R}, +, -, 0, 1, < \rangle$ is o-minimal, has QE, and its first-order theory is decidable.

**Polynomial Constraints:** The real field $\langle \mathbb{R}, +, *, 0, 1, < \rangle$ is o-minimal, has QE, and its first-order theory is decidable. This follows from Tarski's theorem, cf. [9, 36].

**Exponential Constraints:** $\langle \mathbb{R}, +, *, e^x, 0, 1, < \rangle$ is o-minimal, see [40].

If only quantifiers $\forall x \in adom$ and $\exists x \in adom$ are used in a query, it is called an *active-semantics* (or active-domain) query. This is the usual semantics for databases, and it will be the one used in most of the results in this paper. If quantification over the entire infinite universe is allowed, we speak of a *natural-semantics* query. Active-semantics queries admit the standard bottom-up evaluation, while for natural-semantics it is not clear a priori if they can be evaluated at all. However, in many cases one can restrict one's attention to active-semantics queries. The following result was first shown for the pure case (no interpreted structure) in [14] and for linear constraints [24], and then for a large class of structures as follows:

**Fact 1 (Natural-Active Collapse)** (see [8]) *If $\mathcal{M}$ is o-minimal and has QE, and $\varphi$ is an arbitrary $\mathrm{FO}(SC, \Omega)$ query, then there exists an equivalent $\mathrm{FO}(SC, \Omega)$ active-semantics query $\varphi_{\mathrm{act}}$. Moreover, if the first-order theory of $\mathcal{M}$ is decidable and QE is effective, then the translation $\varphi \to \varphi_{\mathrm{act}}$ is effective.* □

We now define the classes of conjunctive queries (CQ), unions of conjunctive queries (UCQ) and Boolean combination of conjunctive queries (BCCQ) in the interpreted setting. CQs are built up from atomic $SC$ formulae and *arbitrary* $\Omega$-formulae by using $\wedge$ and quantifiers $\exists x$ and $\exists x \in adom$. Note that we can always assume that parameters of each $SC$ relation are variables, as $\Omega$-terms can be eliminated by using existential (active-domain) quantifiers. It is easy to see that each CQ can be represented in the form

$$\varphi(\vec{z}) \ \equiv\ \exists \vec{x} \exists \vec{y} \in adom\ S_1(\vec{u}_1) \wedge \ldots S_k(\vec{u}_k) \wedge \gamma(\vec{x}, \vec{y}, \vec{z})$$

where $S_i$s are schema relations (not necessarily distinct), $\vec{u}_i$ is a vector of variables from $\vec{x}, \vec{y}, \vec{z}$ of appropriate arity, and $\gamma$ is a $\Omega$-formula. If $\Omega = \emptyset$ and $\vec{x} = \emptyset$, this is the usual notion of conjunctive queries. if $\gamma$ is quantifier-free, this is the notion of conjunctive queries used in [15].

We define UCQs to be built up from from atomic $SC$ formulae and *arbitrary* $\Omega$-formulae by using $\wedge$, $\vee$ and quantifiers $\exists x$ and $\exists x \in adom$. Again, it is easy to see that those are precisely the queries of the form $\varphi_1 \vee \ldots \vee \varphi_k$ where each $\varphi_i$ is a CQ. Finally, BCCQ are arbitrary Boolean combinations of CQs.

## 3 Safe translations

The main goal of this section is to show that what kind of interpreted structure one uses does matter, when one studies query safety. As a by-product, we show that the state-safety problem is decidable over certain structures. We study *safe translations*, that is, translations from arbitrary queries into safe ones that do not change the result of a query if it is finite. Formally:

**Definition 1** *We say that there is a* safe translation *of (active-semantics) first-order queries over $\mathcal{M}$ if there is a function $\varphi \to \varphi_{\text{safe}}$ on (active-semantics) formulae such that for every $\varphi$,*
*1) $\varphi_{\text{safe}}$ is safe, and*
*2) If $\varphi$ is safe for $D$, then $\varphi[D] = \varphi_{\text{safe}}[D]$.*
*A translation is canonical if $\varphi_{\text{safe}}[D] = \emptyset$ whenever $\varphi$ is not safe on $D$. A translation is recursive if the function $\varphi \to \varphi_{\text{safe}}$ is recursive.* $\square$

There are domains over which safe translations do not exist [32]. The result of [32] uses quantification over the entire universe in an essential way. Here we restrict our attention to the active-domain quantification. The following generalizes our result from [8].

**Proposition 1** *Let $\mathcal{M}$ be o-minimal, based on a dense order, admit effective QE, and have a decidable theory. Then there exists a recursive canonical safe translation of active-semantics formulae over $\mathcal{M}$.*

*Proof (sketch):* Let $\alpha(x)$ define the active domain of the output of $\varphi$. Let $\Psi_0 \equiv \neg\exists x_1, x_2\ ((x_1 < x_2) \wedge (\forall x\ x_1 < x < x_2 \to \alpha(x)))$, and let $\Psi$ be an equivalent active-semantics sentence (see Fact 2). Define $\varphi_{\text{safe}}$ as $\varphi \wedge \Psi$. Correctness follows from $D \models \Psi$ iff $\varphi[D]$ is finite. $\square$

Examples of structures to which Proposition 1 applies, are $\langle \mathbb{R}, +, -, 0, 1, < \rangle$ and $\langle \mathbb{R}, +, *, 0, 1, < \rangle$. From the proof of Proposition 1, we obtain

**Corollary 1** *Let $\mathcal{M}$ be as in Proposition 1. Then the state-safety problem over $\mathcal{M}$ is decidable. That is, given a first-order query $\varphi$, and a database $D$, it is decidable whether $\varphi[D]$ is finite.* $\square$

Safe translations (recursive or not!) need not exist even when for active-semantics queries, and all computable predicates in the signature $\Omega$.

**Proposition 2** *There is a structure $\mathcal{M} = \langle \mathbb{N}, P \rangle$, where $P$ is a computable predicate, such that there is no safe translation of active-semantics first-order queries over $\mathcal{M}$.* $\square$

In the remainder of the paper, we concentrate on "well-behaved" structures; typically, o-minimal ones. For computability, we often impose QE and decidability of first-order theory.

# 4 Range-restriction and safety

Let us informally describe the concept of range-restriction for databases over interpreted structures. It can be seen as a generalization, to arbitrary structures, of the idea of finiteness dependencies [27]. Consider a query $\varphi(x, y)$ over a database which is a finite set $S$ of real numbers:

$$\exists z\ [S(z) \wedge (x > y) \wedge (x > 0) \wedge (x * x = z) \wedge (y + y = z)]$$

This query defines a set of pairs of reals. Clearly, it is safe, as the size of its output is at most the square of the size of the input. Moreover, and this is the key observation, from the query $\varphi$ and any database $S$, one can compute an upper bound on the output $\varphi[S]$: indeed, every element in $adom(\varphi[S])$ is either $\sqrt{a}$ or $\frac{a}{2}$ for some element $a \in S$. Equivalently, in this upper bound every element is a solution to either $x^2 = a$ or $2x = a$ when $a$ ranges over $S$. That is, in this example, an upper bound on the result of a safe query can be found as the set of roots of polynomials with coefficients coming from the active domain of the database and a finite set of constants.

This is essentially the idea of range-restriction: we find, for a safe query, a set of formulae defining an upper bound on the output. A similar approach was used in [11], although the focus of [11] is different: it does not consider how the underlying structure affects safety, but instead gives a syntactically restricted class of queries with interpreted functions that can be translated into algebra expressions, along the lines of [38]. In contrast, we are interested in how the underlying structure interacts with queries.

In a general context of arbitrary structures (e.g., $\langle \mathbb{R}, +, *, e^x \rangle$), the role of roots of polynomials is played by collections of algebraic formulae, a model-theoretic notion reviewed in subsection 4.1. The range-restriction theorem is proved in subsection 4.2. We give two examples: the pure case, where our main result trivially translates into a classical relational theory result, and a much less trivial FO + POLY case, where we confirm the original intuition that the upper bound is a set of roots of polynomials.

## 4.1 Algebraic formulae over o-minimal structures

We consider formulae $\varphi(\vec{x}; \vec{y})$ in the language of $\Omega$ and constants for elements of $\mathcal{U}$, with distinguished parameter variables $\vec{y}$; we use ";" to separate those variables. Assume that $\vec{x}$ is of length $n$ and $\vec{y}$ is of length $m$. Such a formula is called *algebraic* if for each $\vec{b}$ in $\mathcal{U}^m$ there are only finitely many satisfiers of $\varphi(\vec{x}, \vec{b})$; that is, the set $\{\vec{a} \in \mathcal{U}^n \mid \mathcal{M} \models \varphi(\vec{a}; \vec{b})\}$ is finite. A collection of formulae is algebraic if each of its elements is algebraic.

There are various ways to enumerate algebraic formulae. Below we give a representation needed for characterization of safe queries. Let $\Xi = \{\xi_1(x;\vec{y}), \ldots, \xi_k(x;\vec{y})\}$ be a collection of formulae. Let $\text{same}_\Xi(x, x'; \vec{y}) \equiv \bigwedge_{i=1}^k (\xi_i(x;\vec{y}) \leftrightarrow \xi_i(x';\vec{y}))$. Now define $\beta_\Xi(x;\vec{y})$ as

$$\forall u, v.u < x < v \rightarrow (\exists z.u \leq z \leq v \wedge \neg \text{same}_\Xi(x, z; \vec{y})).$$

**Proposition 3** *Let $\mathcal{M}$ be an o-minimal structure based on a dense order. Then a formula $\varphi(x;\vec{y})$ is algebraic (over $\mathcal{M}$) iff there exists a collection of formulae $\Xi$ such that $\varphi$ is equivalent to $\beta_\Xi$. A formula $\varphi(\vec{x};\vec{y})$ is algebraic iff there exists a collection of formulae $\Xi$ in variables $(x;\vec{y})$ and a formula $\psi(\vec{x};\vec{y})$ such that $\varphi$ is equivalent to $\beta_\Xi(x_1;\vec{y}) \wedge \ldots \wedge \beta_\Xi(x_n;\vec{y}) \wedge \psi(\vec{x};\vec{y})$.* □

## 4.2 Main theorem

We start with a few definitions. For a $L(\Omega)$-formulae $\gamma(\vec{x};\vec{y})$ and database $D$ over $\mathcal{M}$, let

$$\gamma(D) \;=\; \{\vec{a} \mid \exists \vec{b} \in adom(D)^m \text{ such that } \mathcal{M} \models \gamma(\vec{a};\vec{b})\}$$

If $\Gamma$ is a collection of formulae in $\vec{x};\vec{y}$, define $\Gamma(D) = \bigcup_{\gamma \in \Gamma} \gamma(D)$. Note that if $\Gamma$ is finite and algebraic, then $\Gamma(D)$ is finite.

**Definition 2 (Range-restriction)** *Let $\mathcal{M}$ be an interpreted structure. A* range-restricted query *over $\mathcal{M}$ and a database schema $SC$ is a pair $Q = (\Gamma, \varphi(\vec{x}))$, where $\Gamma$ is a finite collection $\{\gamma_1(\vec{x};\vec{y}), \ldots, \gamma_m(\vec{x};\vec{y})\}$ of algebraic $L(\Omega)$ formulae, and $\varphi(\vec{x})$ is a $L(SC, \Omega)$ query.*

*The semantics of $Q$ is as follows:*

$$Q[D] \;=\; \{\vec{a} \in \Gamma(D) \mid D \models \varphi(\vec{a})\}$$

That is, $\Gamma$ provides an upper bound on the output of a query; within this bound, a usual first-order query is evaluated. For example, let $\varphi(x)$ be the FO + POLY query $S(x) \vee (x > 5)$. Clearly, it is not safe. Let now $\gamma(x;y) \equiv (x * x = y)$ and $Q = (\{\gamma\}, \varphi)$. Then, for any finite database $S$, $Q[S]$ is the set of those elements $a$ such that $a^2 \in S$ and either $a \in S$ or $a > 5$. Clearly, this is a finite set.

**Observation** *Every range-restricted query is safe.*

We call a range-restricted query $(\Gamma, \varphi)$ *active-semantics* if $\varphi$ is an active-semantics formula. Note that $\Gamma$ does not mention the database. It turns out that range-restricted active queries characterize all the safe active-semantics queries in the following sense.

**Theorem 1** *Let $\mathcal{M}$ be any o-minimal structure based on a dense linear order. Then there is a function* Make_Safe *that takes as input an active-domain formula $\varphi(\vec{x})$, and outputs a range-restricted active query $Q = (\Gamma, \psi)$ with the property that* Make_Safe($\varphi$) *is equivalent to $\varphi$ on all databases $D$ for which $\varphi$ is safe. Furthermore, if $\mathcal{M}$ has effective quantifier-elimination, then* Make_Safe *is recursive.*

*Proof (sketch):* We sketch the one-variable case. Let $\varphi(z) \equiv Q_1 w_1 \in adom \ldots Q_1 w_k \in adom.\alpha(z, \vec{w})$ where each $Q_i$ is $\exists$ or $\forall$ and $\alpha(z, \vec{w})$ is quantifier-free, and all atomic subformulae $R(\cdots)$ contain only variables, excluding $z$. Let $\Xi = \{\xi_i(z, \vec{w}) \mid i = 1, \ldots, k\}$ be the collection of all $\Omega$-atomic subformulae of $\alpha$. Assume that the length of $\vec{w}$ is nonzero, and that $\Xi$ is nonempty. Then Make_Safe($\varphi$) outputs $(\{\beta_\Xi(x;\vec{w})\}, \varphi)$ (see the definition in subsection 4.1). For the proof of correctness and the multi-variable case, see the appendix. □

**Corollary 2 (Range-restricted = Safe)** *For any o-minimal structure based on a dense order, the class of safe active-semantics queries is the same as the class of range-restricted queries.* □

Combining this with Fact 2, we obtain:

**Corollary 3** *Let $\mathcal{M}$ be any o-minimal structure based on a dense order that admits QE. Then there is a function* Make_Safe *(recursive if so is QE and $\mathcal{M}$ is decidable) that takes as input a natural-semantics formula $\varphi(\vec{x})$, and outputs a range-restricted active query $Q = (\Gamma, \psi)$ with the property that* Make_Safe($\varphi$) *is equivalent to $\varphi$ on all databases $D$ for which $\varphi$ is safe. In particular, over $\mathcal{M}$, the classes of safe natural-semantics queries and range-restricted queries coincide.* □

**Corollary 4** *For any o-minimal structure based on a dense order (decidable or not), the collection of safe queries is recursively enumerable.* □

The proposition below puts a bound on the active domain of the output, and is helpful in establishing decidability results in Section 5.

**Proposition 4** *Let $\mathcal{M}$ be o-minimal and based on a dense order. Let $\varphi(\vec{x})$ be a first-order query. Then there exists a set $\Gamma$ of algebraic formulae $\gamma(\vec{x};\vec{y})$ (that can be effectively constructed in $\mathcal{M}$ has effective QE and is decidable) such that, for any database $D$, if $\varphi[D]$ is finite, then $adom(\varphi[D]) \subseteq \Gamma(D)$.* □

Here we concentrated of relational calculus; in the full version, we also consider extensions to higher-order logics, and show that any query $\varphi$ that is safe over some o-minimal structure $\mathcal{M}$ based on a dense order (e.g., $\langle \mathbb{R}, +, < \rangle$) is also safe over any structure elementary equivalent to $\mathcal{M}$ (e.g., $\langle \mathbb{Q}, +, < \rangle$).

### 4.3 Examples

**The pure case** We deal with the pure relational calculus; that is, $\mathcal{M} = \mathcal{M}^\emptyset = \langle \mathcal{U}, \emptyset \rangle$. Extend $\mathcal{M}$ to $\mathcal{M}^< = \langle \mathcal{U}, < \rangle$ by adding a dense order without endpoints on $\mathcal{U}$; although it is never mentioned in queries, $\mathcal{M}^<$ is o-minimal, has QE, and Theorem 1 is applicable. We need:

**Lemma 1** *Let $\gamma(x; \vec{y})$ be algebraic over $\mathcal{M}^<$. If $\mathcal{M}^< \models \gamma(a; \vec{b})$, then $a$ coincides with one of $\vec{b}$'s components.* □

Thus, for an algebraic $\Gamma \neq \emptyset$, $\Gamma(D) = adom(D)$. Hence, the classes of safe pure relational calculus queries and relational calculus queries whose output is restricted to the active domain, coincide. Of course, this is a standard result in database theory, proved here in a rather unusual way.

**The real field:** FO + POLY Can we find a more concrete representation of range-restricted queries over the real field? Intuitively, it should be sufficient to look for roots of polynomials $p(x, \vec{a})$ where $\vec{a}$ ranges over tuples of elements of the active domain, as was suggested by the example in the beginning of the section.

Let $p(x, \vec{y})$ be a multivariate polynomial over the real field. Define $Root(p, \vec{a})$ as $\emptyset$ if $p(x, \vec{a})$ is identically zero, and the set of roots of $p(x, \vec{a})$ otherwise. Given a collection $P$ of polynomials $\{p_1(x, \vec{y}), \ldots, p_m(x, \vec{y})\}$ and a database $D$, let

$$Root_P(D) = \bigcup_{i=1}^{m} \bigcup_{\vec{a} \subseteq adom(D)} Root(p_i, \vec{a})$$

**Definition 3** *A query in* polynomial range-restricted *form is a pair $(P, \varphi)$, where $P$ is a finite collection of multivariate polynomials, and $\varphi(x_1, \ldots, x_n)$ is a* FO + POLY *query. The semantics is defined as $(P, \varphi)[D] = \varphi[D] \cap Root_P(D)^n$.*

Although not an immediate consequence of QE, the following can be shown.

**Proposition 5** *The class of safe* FO + POLY *queries (arbitrary or active-semantics) coincides with the class of queries in polynomial range-restricted form. Moreover, for every* FO + POLY *query $\varphi$, a collection of polynomials $P$ can be effectively found such that $\varphi$ and $(P, \varphi)$ are equivalent on databases on which $\varphi$ is safe.*

*Proof* is based on Theorem 1, unform bounds for o-minimal structures [26], and a characterization of semi-algebraic functions in [19]. See appendix for details. □

## 5 Deciding safety of conjunctive queries and relatives

Safety of arbitrary calculus queries is undecidable even in the pure case [39], and of course it remains undecidable when interpreted functions are present. The main goal of this section is to show that safety is *decidable* for Boolean combinations of conjunctive queries in the presence of an interpreted structure such as $\langle \mathbb{R}, +, *, 0, 1, < \rangle$. In particular, safety of FO + POLY and FO + LIN conjunctive queries is decidable.

Recall that CQ, UCQ and BCCQ stand for conjunctive, unions of conjunctive, and Boolean combinations of conjunctive queries (see Section 2 for the definition in the presence of an interpreted structure). Conjunctive queries have been widely studied from the beginning of relational theory, and they have arisen in a number of new applications as well, cf. [20, 15, 35]. Our proof will be by reduction to the containment problem, which is decidable for UCQs over certain structures. (Of course, *without* an interpreted structure, this is well known [31], as is the decidability of safety for BCCQs, cf. [1]). Note that CQs and UCQs are monotone ($D \subseteq D'$ implies $\varphi[D] \subseteq \varphi[D']$).

**Theorem 2** *Let $\mathcal{M}$ be o-minimal, based on a dense order, decidable, and admit effective QE. Then it is decidable if a given BCCQ $\varphi(\vec{x})$ over $\mathcal{M}$ is safe.*

The proof is given by two lemmas, which are of independent interest, and will be used in Section 7. By containment $\varphi \subseteq \psi$ we mean $\varphi[D] \subseteq \psi[D]$ for any $D$.

**Lemma 2** *Let $\mathcal{M}$ be o-minimal and based on a dense order, and $\varphi(\vec{x})$ be a first-order query. Then there exists an active-semantics CQ $\psi(\vec{x})$ such that $\varphi$ is safe iff $\varphi \subseteq \psi$.* □

**Lemma 3** *Let $\mathcal{M}$ be as in Theorem 2. Then containment of a BCCQ in a UCQ is decidable: for a BCCQ $\varphi(\vec{x})$ and a UCQ $\psi(\vec{x})$ it is decidable if $\varphi \subseteq \psi$.* □

**Corollary 5** *It is decidable whether any Boolean combination of* FO + LIN *or* FO + POLY *conjunctive queries is safe.* □

In the full version, we will show how to extend these results to some structures that are not o-minimal (for example, $\langle \mathbb{N}, +, < \rangle$).

Note that safety of CQs is not decidable over every structure. For example, for $\langle \mathbb{N}, +, *, 0, 1, < \rangle$, decidability of CQ safety would imply decidability of checking whether a Diophantine equation has finitely many solutions, which is known to be undecidable [?].

# 6 Dichotomy theorem and outputs of queries

The main result of this section is a simple but powerful combinatorial structure theorem, saying that over a well-behaved structure, outputs of safe queries cannot grow arbitrarily large in terms of the size of the input. In fact, we prove a *dichotomy* result: either a query $\varphi$ is not safe on $D$, or $\varphi[D]$ is at most polynomial in the size of $D$, where the bounding polynomial depends only on $\varphi$. This result shows tame behavior of relational calculus queries over some important interpreted structures, in particular those giving rise to linear, polynomial and exponential constraints. It can be used to show negative results, that is, new expressivity bounds, as well as positive results: the dichotomy theorem is a key ingredient in the decidability results of the next section.

We use the notation $\mathsf{size}(D)$ for the size of a database, measured here as the number of tuples. It can equivalently be measured as the cardinality of the active domain, or the number of tuples multiplied by their arity, and all the results below will hold.

**Theorem 3 (Dichotomy Theorem)** *Let $\mathcal{M}$ be o-minimal and based on a dense order. Let $\varphi(\vec{x})$ be a first-order query. Then there exists a a polynomial $p_\varphi : \mathbb{R} \to \mathbb{R}$ such that, for any database $D$, either $\varphi[D]$ is infinite, or $\mathsf{size}(\varphi[D]) \leq p_\varphi(\mathsf{size}(D))$.* □

Are the assumptions on a structure important for the dichotomy result? That is, can we find structures over which it fails? The following is a simple example: Let $\mathcal{M}$ be $\langle \mathbb{N}, < \rangle$, and let $\chi(x)$ be $\exists y \in adom. x < y$. Clearly, $\chi(x)$ is safe, but $\mathsf{size}(\chi[D])$ can be arbitrarily large even for one-element databases.

The dichotomy theorem can also be stated in terms of a function measuring the growth of the output size. Given a query $\varphi$, let $growth_\varphi : \mathbb{N} \to \mathbb{N} \cup \{\infty\}$ be

$$growth_\varphi(n) = \max\{\mathsf{size}(\varphi[D]) \mid \mathsf{size}(D) = n\}$$

**Corollary 6** *Let $\varphi(\vec{x})$ be a* FO + LIN, *or* FO + POLY, *or* FO + EXP *query. Then there exists a polynomial $p_\varphi$ such that, for every $n \in \mathbb{N}$, either $growth_\varphi(n) = \infty$, or $growth_\varphi(n) \leq p_\varphi(n)$.* □

For the query $\chi(x)$ over $\langle \mathbb{N}, < \rangle$ shown above, $growth_\chi(n) = \infty$ for all $n > 0$. Thus, we have a question whether Corollary 6 fails over some structures. The following is such an example:

**Proposition 6** *Let $\mathcal{M} = \langle \mathbb{N}, +, <, 1 \rangle$. Then there exists an active-semantics first-order query $\varphi(x)$ over $\mathcal{M}$ such that $growth_\varphi(n) = 2^n$ for every $n > 0$.* □

In the full paper, we show a different kind of dichotomy result for $\langle \mathbb{N}, +, <, 1 \rangle$. We will also show that $growth_\varphi$ may exist without being majorized by an exponential function.

The dichotomy theorem gives easy expressivity bounds based on the growth of the output size, in fact, somewhat surprising ones: even we use exponentiation, we still cannot express any queries with superpolynomial growth. For example, consider the following query $Q$: given a binary relation $S$ containing $n + 1$ distinct points $x_0, x_1, \ldots, x_n$ on a plane, return the vertices of the projection of an $n$-dimensional cube $[0,1]^n$, where the edges along the axes are projected onto $\vec{x_0 x_1}, \ldots, \vec{x_0 x_n}$. It is easy to see that for each fixed $n$, this query is expressible in FO + POLY. From the dichotomy theorem, we conclude that $Q$ cannot be expressed uniformly for all $n$ even as a FO + EXP query.

For monotone queries ($D_1 \subseteq D_2$ implies $\varphi[D_1] \subseteq \varphi[D_2]$), we prove a trichotomy theorem:

**Theorem 4** *Let $\mathcal{M}$ be o-minimal based on a dense order. Then, for each monotone query $\varphi(\vec{x})$, there exist two polynomials $p^1_\varphi$ and $p^2_\varphi$ such that either $growth_\varphi$ is bounded by a constant, or, for every $n$, either $p^1_\varphi(n) \leq growth_\varphi(n) \leq p^2_\varphi(n)$, or $growth_\varphi(n) = \infty$.* □

In fact, the lower polynomial bound does not require o-minimality. This gives us some new expressivity bounds. There are first-order queries (to be described in the full version) such that $growth_\varphi = O(\log n)$ or $growth_\varphi = O(\sqrt{n})$. The trichotomy theorem says that such queries cannot be defined as monotone queries (e.g., UCQs) over *any* interpreted structure.

# 7 Preserving geometric properties of constraint databases

In this section, we switch from the finite world to the infinite; that is, we deal with constraint databases that represent potentially infinite objects. The notion of safety over constraint databases is different: we are interested in identifying languages that guarantee *preservation of certain geometric properties.*

To give a very simple example, assume that spatial objects stored in a database are convex polytopes in $\mathbb{R}^n$. A simple query "return the convex hull of all the vertices $x$ with $\| x \| < 1$" does always return a convex polytope. This query must be written in an expressive language, such as FO + POLY, since FO + LIN cannot express it. Now, our question is: can we ensure in some way that a class of FO + POLY programs maps convex polytopes to convex polytopes? A related question is whether we can find, for any geometric class $\mathcal{C}$, an effective syntax for the class of FO + POLY queries that preserve the property $\mathcal{C}$.

For FO + POLY and the class of databases definable with linear constraints (semi-linear databases), [10] gave a solution, based on testing semi-linearity by a FO + POLY query. The resulting language is not quite natural, and [10] posed a problem of finding natural languages that capture queries preserving semi-linearity. Our first goal here is to present a general scheme, different from the approach of [10], for enumerating the $\mathcal{C}$-preserving queries in FO($\mathcal{M}$), for any structure $\mathcal{M}$ on the reals (not necessarily $\langle \mathbb{R}, +, *, 0, 1, < \rangle$).

The approach is based on reduction to the finite case, and then using the results on finite query safety. We reduce the finitely representable case to the finite case via *canonical codes.* This approach has also been applied in [33], to reduce a different set of questions about ordered constraint databases to ones about finite databases. We present a general result on reducing the $\mathcal{C}$-preservation question to the safety question via codes, and we apply it to three geometric classess $\mathcal{C}$: the convex polytopes, the convex polyhedra, and the compact semi-linear sets in $\mathbb{R}^2$ (the latter are perhaps the most often encountered class of constraint databases).

We then use our characterizations, together with the dichotomy theorem of Section 6, to show that for unions of conjunctive FO + POLY queries, it is *decidable* whether they preserve convex polytopes or compact semi-linear sets in $\mathbb{R}^2$.

We review some basic definitions on constraint (or finitely representable) databases. We have a language of some underlying structure $\mathcal{M}$ and a schema $SC$, but now $m$-relations in $SC$ are given by quantifier-free formulae[1] $\alpha(x_1, \ldots, x_m)$ in $L(\Omega)$. If $\mathcal{M}$ is $\langle \mathbb{R}, +, -, 0, 1, < \rangle$, then sets so defined are called *semi-linear*; for $\langle \mathbb{R}, +, *, 0, 1, < \rangle$ they are called *semi-algebraic*, cf. [36]. We use $\mathsf{SAlg}_n$ for the class of semi-algebraic sets in $\mathbb{R}^n$. As the query language, we consider FO($SC, \Omega$). We do not deal with the active-semantics quantification in this section.

If $\mathcal{M} = \langle \mathcal{U}, \Omega \rangle$ is an infinite structure, let $\mathsf{Obj}(\Omega)$ be the class of finitely representable databases over $\mathcal{M}$, that is, $\mathsf{Obj}(\Omega) = \bigcup_{n<\omega} \mathsf{Obj}_n(\Omega)$ and $\mathsf{Obj}_n(\Omega)$ is the collection of subsets of $\mathcal{U}^n$ of the form $\{\vec{x} \in \mathcal{U}^n \mid \mathcal{M} \models \alpha(\vec{x})\}$ where $\alpha$ is quantifier-free first-order formula in $L(\Omega)$.

Let $S \in SC$ be an $m$-ary relational symbol, and let $\psi(y_1, \ldots, y_n)$ be a first-order formula in the language of $S$ and $\Omega$. Then this query defines a map from $\mathsf{Obj}_m(\Omega)$ to $\mathsf{Obj}_n(\Omega)$ as follows: for any $X \in \mathsf{Obj}_m(\Omega)$, $\psi[X] = \{\vec{y} \mid (\mathcal{M}, X) \models \psi(\vec{y})\}$. Clearly $\psi[X] \in \mathsf{Obj}(\Omega)$.

Let $\mathcal{C}$ be a class of objects in $\mathsf{Obj}(\Omega)$. We say that a first-order query $\psi$ *preserves* $\mathcal{C}$ if for any $X \in \mathcal{C}$, $\psi[X] \in \mathcal{C}$.

Thus, the safety question is whether there is an effective syntax for the class of $\mathcal{C}$-preserving queries. Our approach is described below. When applicable, it leads to languages more natural than those one obtains using the decidability approach.

**Definition 4** *The class $\mathcal{C}$ has a* canonical representation *in* $\mathsf{Obj}(\Omega)$ *if there is a recursive injective function $g : \mathbb{N} \to \mathbb{N}$ with computable inverse, and for each $n$, two functions $code_n : 2^{\mathcal{U}^n} \to 2^{\mathcal{U}^m}$ and $decode_n : 2^{\mathcal{U}^m} \to 2^{\mathcal{U}^n}$, where $m = g(n)$, such that:*

1. *$decode_n \circ code_n(x) = x$ if $x \in \mathsf{Obj}_n(\Omega)$;*
2. *$\mid code_n(x) \mid < \omega$ if $x \in \mathcal{C}$; $decode_n(x) \in \mathcal{C}$ if $x$ is finite;*
3. *$code_n$ is FO($\Omega$)-definable on $\mathsf{Obj}_n(\Omega)$;*
4. *$decode_n$ is FO($\Omega$)-definable on finite sets.*

Intuitively, the canonical representation is a finite representation of $\mathcal{C}$ within $\mathsf{Obj}(\Omega)$ that can be defined in first-order logic over $\mathcal{M}$. For example, an approach to obtaining a canonical representation of convex polytopes would be to compute their vertices. This suffices

[1] For simplicity, we do not assume relational attributes, as in [10, 23] and some other papers. They do not affect our results, but would make notation heavier.

to reconstruct the polytope, and the vertices are first-order definable. The actual coding (Proposition 7) is indeed based on computing the vertices.

**Theorem 5** *Let $\mathcal{M} = \langle \mathcal{U}, \Omega \rangle$ be o-minimal, based on a dense order, decidable, and have effective QE. Suppose $\mathcal{C}$ is a class that has a canonical representation in $\mathsf{Obj}(\Omega)$. Then there is an effective syntax for $\mathcal{C}$-preserving* $\mathrm{FO}(\Omega)$ *queries; that is, there exists a r.e. set of* $\mathrm{FO}(\Omega)$ *queries that express exactly all $\mathcal{C}$-preserving* $\mathrm{FO}(\Omega)$ *queries.*

*Proof (sketch):* Corollary 2 gives us an enumeration of all safe $\mathrm{FO}(\Omega)$ queries $\langle \varphi_i \rangle$. Let $\varphi$ use a relational symbol of arity $m$, and assume that $n$ is such that $g(n) = m$. Let $\varphi_i$ have $l$ parameters, and again let $k$ be such that $g(k) = l$. If $n$ and $k$ are found for a given $\varphi_i$, consider $decode_k \circ \varphi_i \circ code_n$. This produces the required enumeration. See apenndix for details. □

We now turn to examples in the case when $\Omega = (+, *, 0, 1, <)$; that is, we are looking for canonical representations in $\mathsf{SAlg}$. Let $\mathcal{CPH}$ be the class of convex polyhedra (intersections of a finite number of closed halfspaces) and $\mathcal{CPT}$ be the class of convex polytopes (bounded polyhedra). The proofs of two propositions below are based on various representation theorems from convex analysis [29].

**Proposition 7** *The class $\mathcal{CPT}$ has canonical representation in* $\mathsf{SAlg}$. □

**Proposition 8** *The class $\mathcal{CPH}$ has canonical representation in* $\mathsf{SAlg}$. □

Let $\mathsf{SLinComp}$ be the class of *compact* (closed and bounded) semi-linear sets. We resolve this case for dimension 2. The proof combines the results on triangulation in 2-dimensional case (cf. [22]) and a characterization of vertices of semi-linear sets [21] that can be turned into a FO + POLY-definition.

**Proposition 9** *The class* $\mathsf{SLinComp}_2$ *has canonical representation in* $\mathsf{SAlg}_2$. □

The coding used in the proof of Proposition 9 cannot be used in higher dimensions. We used the fact there is a triangulation of a 2-dimensional polyhedron that does not introduce new vertices, which is not true in in the 3-dimensional case, cf. [30].

Summing up, we have

**Theorem 6** *There exists a recursively enumerable class of* FO + POLY *queries that captures the class of $\mathcal{CPT}$ ($\mathcal{CPH}$ and* $\mathsf{SLinComp}_2$*, respectively) preserving queries.* □

**Decidability results and geometric bounds** While the classes of FO + POLY queries preserving certain properties have been shown to be r.e., in general, testing nontrivial preservation properties for arbitrary first-order queries is undecidable. For example, it is shown in [37] that it is undecidable whether a FO + POLY-query preserves semi-linearity. Here, we show that for a restricted class of FO + POLY queries – unions of conjunctive queries – preserving two of the properties considered here is decidable. The proofs are based on the representation theorems of this section, and the dichotomy theorem of the previous section. We first give the following bound on the behavior of unions of conjunctive queries on convex polytopes.

**Lemma 4** *Let $\varphi(x_1, \ldots, x_n)$ be a union of* FO + POLY *CQs that mention one $m$-ary relational symbol $S$. Then one can effectively find two numbers $k$ and $l$ such that $\varphi$ is $\mathcal{CPT}$-preserving iff for every convex polytope $D$ in $\mathbb{R}^m$ with at most $k$ vertices, the output $\varphi[D]$ is a convex polytope with at most $l$ vertices in $\mathbb{R}^n$.* □

A similar bound exists for compact semi-linear sets in $\mathbb{R}^2$. When we speak of a triangle, we mean convex hull of three points in $\mathbb{R}^2$. In particular, a degenerate triangle can be a segment or a point.

**Lemma 5** *Let $\varphi(x, y)$ be a union of conjunctive* FO + POLY *queries that mention one binary relational symbol $S$. Then one can effectively find two numbers $k$ and $l$ such that $\varphi$ is* $\mathsf{SLinComp}_2$*-preserving iff for every set $D \subseteq \mathbb{R}^2$ which is a union of at most $k$ triangles in $\mathbb{R}^2$, $\varphi[D]$ is a union of at most $l$ triangles in $\mathbb{R}^2$.* □

The promised decidability results now follow from the bounds established in the lemmas above.

**Theorem 7** *The following two properties of unions of conjunctive* FO + POLY *queries are decidable:*
 *(a) being $\mathcal{CPT}$-preserving;*
 *(b) being* $\mathsf{SLinComp}_2$*-preserving.* □

**New expressivity bounds** These can be obtained by combining the dichotomy theorem with the idea of canonical representation. As a consequence of the techniques of Proposition 7, Lemma 4, Lemma 5, and the dichotomy theorem, we obtain:

**Corollary 7** *a) Let $\varphi(\vec{x})$ be a* FO+POLY *or* FO+EXP *$\mathcal{CPT}$-preserving query. Then there exists a polynomial $p_\varphi$ such that, whenever $D$ is a convex polytope with $n$ vertices, $\varphi[D]$ has at most $p_\varphi(n)$ vertices.*
*b) Let $\varphi(x, y)$ be a* FO+POLY *or* FO+EXP *query that is* $\mathsf{SLinComp}_2$*-preserving. Then there exists a polynomial $p_\varphi$ such that, whenever $D$ is a compact semi-linear set with $n$ vertices, $\varphi[D]$ has at most $p_\varphi(n)$ vertices.* □

As another application to expressive power, consider the following problem: given a polyhedron $P$ and $\epsilon > 0$, find a triangulation of $P$ of mesh $< \epsilon$. That is, the diameter of each simplex (triangle in dimension 2) must be $< \epsilon$. It is well know that each polyhedron admits such a triangulation [4]. The output of such a query can be structured in several ways; we only require that the vertices of the triangulation be computable.

**Proposition 10** *There is no* FO + EXP *query that finds a triangulation of a given polygon with a given mesh.* □

## 8 Conclusion and future work

The main conclusion of the paper is that the relational calculus with interpreted functions is a nontrivial and interesting extension of the relational calculus. For well-behaved structures, interesting generalizations of classical characterization theorems hold.

We now discuss extensions to other settings. The characterization theorems given here are by no means limited to functions on real or even rational domains. For well-behaved structures over the integers, such as linear integer constraints, modifications of the safety characterization results still hold, and algorithmic consequences, such as the decidability results for conjunctive queries, are still valid. We focused mainly on the relational calculus here, but many of the results on range-restriction and safety generalize to higher-order logics (fixpoint, second-order). Still, the safety question for higher-order logics is quite intricate, and we lack a full picture of what interpreted structures and recursion constructs permit a well-behaved theory of query safety.

Papers such as [15] give more detailed algorithmic analyses for specific structures. It still remains to give complexity-theoretic analyses of the safe translation problem and the query safety problem for conjunctive queries in the case of polynomial and linear constraints. A related question is the complexity of deciding preservation of geometric properties for conjunctive queries.

We are working on extending the growth bound theorems of Section 6. This includes characterizations of structures for which the dichotomy theorem holds, and characterizations of structures for which the growth function is bounded not by a polynomial, but by other definable functions of the input size (e.g., $e^x$).

The second part of the paper deals with applying our results on finite databases to finitely-representable ones, with the main technique being canonical codes. The main point of our codings was to facilitate this transfer of results. We are working on refining the results to get natural codings for larger geometric classes, and on studying these codes in themselves. We think that the approach here – taking range-restricted queries on finite databases and applying them to canonical representations of finitely-representable ones – is the most promising one for arriving at useful languages for queries that preserve geometric structure, and in particular for solving the problem (posed by [10]) of obtaining a natural subquery language of FO + POLY for manipulating semi-linear databases.

**Acknowledgements** We thank Ken Clarkson, Steve Fortune and Jianwen Su for helpful discussions.

## References

[1] S. Abiteboul, R. Hull and V. Vianu. *Foundations of Databases.* Addison-Wesley, 1995.

[2] F. Afrati, S. Cosmadakis, S. Grumbach and G. Kuper. Linear vs. polynomial constraints in database query languages. In *CP'94*.

[3] A.K. Ailamazyan, M.M. Gilula, A.P. Stolboushkin and G.F. Shvarts. Reduction of a relational model with infinite domains to the finite-domain case. *Doklady Akademii Nauk SSSR*, 286 (1) (1986), 308–311. Translation in *Soviet Physics – Doklady*, 31 (1986), 11–13.

[4] P.S. Aleksandrov. *Combinatorial Topology.* Graylock Press, 1956.

[5] A. Avron and J. Hirshfeld. On first order database query languages. In *LICS'91*, pages 226–231.

[6] M. Benedikt, G. Dong, L. Libkin and L. Wong. Relational expressive power of constraint query languages. *J. ACM*, to appear. Extended abstract in *PODS'96.*

[7] M. Benedikt and L. Libkin. Languages for relational databases over interpreted structures. In *PODS'97*, pages 87–98.

[8] C.C. Chang and H.J. Keisler. *Model Theory.* North Holland, 1990.

[9] M. Davis. On the number of solutions of Diophantine equations. *Proc. AMS* 35 (1972), 552–554.

[10] F. Dumortier, M. Gyssens, L. Vandeurzen, and D. Van Gucht. On the decidability of semi-linearity of

semi-algebraic sets, and its implications for spatial databases. In *PODS'97*, pages 68–77.

[11] M. Escobar-Molano, R. Hull and D. Jacobs. Safety and translation of calculus queries with scalar functions. In *PODS'93*, pages 253–264.

[12] S. Grumbach and J. Su. Finitely representable databases, *JCSS* 55 (1997), 273–298.

[13] S. Grumbach, J. Su, and C. Tollu. Linear constraint databases. In *LCC'94*, pages 426–446.

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

[15] O.H. Ibarra and J. Su. On the containment and equivalence of database queries with linear constraints. In *PODS'97*, pages 32–43.

[16] P. Kanellakis, G. Kuper, and P. Revesz. Constraint query languages. *JCSS*, 51 (1995), 26–52. Extended abstract in *PODS'90*, pages 299–313.

[17] M. Kifer. On safety, domain independence, and capturability of database queries. In *Proc. Data and Knowledge Base*, Jerusalem, 1988.

[18] M. Kifer, R. Ramakrishnan and A. Silberschatz. An axiomatic approach to deciding query safety in deductive databases. In *PODS'88*, pages 52–60.

[19] D. Marker, M. Messmer and A. Pillay. *Model Theory of Fields*. Springer Verlag, 1996.

[20] A. Levy, D. Suciu. Deciding containment for queries with complex objects. In *PODS'97*, pages 20–31.

[21] W. Nef. *Beiträge zur Theorie der Polyeder*. Herbert Lang, Bern, 1978.

[22] J. O'Rourke. *Computational Geometry in C*. Cambridge Univ. Press, 1994.

[23] J. Paredaens, J. Van den Bussche, and D. Van Gucht. Towards a theory of spatial database queries. In *PODS'94*, pages 279–288.

[24] J. Paredaens, J. Van den Bussche, and D. Van Gucht. First-order queries on finite structures over the reals. In *LICS'95*, pages 79–87.

[25] J. Paredaens, B. Kuijpers, G. Kuper and L. Vandeurzen. Euclid, Tarski and Engeler encompassed. In *DBPL'97*, Springer LNCS, to appear.

[26] A. Pillay, C. Steinhorn. Definable sets in ordered structures. III. *Trans. AMS* 309 (1988), 469–476.

[27] R. Ramakrishnan, F. Bancilhon and A. Silberschatz. Safety of recursive Horn clauses with infinite relations. In *PODS'87*, pages 328–339.

[28] P. Revesz. Safe query languages for constraint databases. *ACM TODS*, to appear.

[29] R.T. Rockafellar. *Convex Analysis*. Princeton Univ. Press, 1970.

[30] J. Ruppert and R. Seidel. On the difficulty of triangulating three-dimensional nonconvex polyhedra. *Discrete and Computational Geometry* 7 (1992), 227–254.

[31] Y. Sagiv and M. Yannakakis. Equivalence among relational expressions with the union and difference operators. *J. ACM* 27 (1980), 633–655.

[32] A. Stolboushkin and M. Tsaitlin. Finite queries do not have effective syntax. In *PODS'95*, pages 277–285.

[33] A. Stolboushkin and M.Tsaitlin. Linear vs. order constraint queries over rational databases. In *PODS'96*, pages 17–27.

[34] A. Stolboushkin and M. Tsaitlin. Safe stratified datalog with integer order does not have syntax. *ACM TODS*, to appear.

[35] J. Ullman. Information integration using logical views. In *ICDT'97*, pages 19–40.

[36] L. van den Dries. O-minimal structures. In *Logic: From Foundations to Applications*, W. Hodges, et al, eds, Clarendon Press, 1996.

[37] L. Vandeurzen, M. Gyssens and D. Van Gucht. On query languages for linear queries definable with polynomial constraints. In *CP'96*, pages 468–481.

[38] A. van Gelder and R. Topor. Safety and translation of relational calculus queries. *ACM TODS*, 16 (1991), 235–278.

[39] M. Vardi. The decision problem for database dependencies. *Inf. Proc. Let.*, 12 (1981), 251–254.

[40] A.J. Wilkie. Model completeness results for expansions of the ordered field of real numbers by restricted Pfaffian functions and the exponential function. *J. Amer. Math. Soc.* 9 (1996), 1051–1094.

# APPENDIX

## Proofs

We will need the following result about o-minimal structures, that will be used numerous times in proofs:

**Fact 2 (Uniform Bounds)** (see [26]) *If $\mathcal{M}$ is o-minimal, and $\varphi(x, \vec{y})$ is a first-order formula in the language of $\mathcal{M}$, possibly supplemented with symbols for constants from $\mathcal{M}$, then there is an integer $K$ such that, for each vector $\vec{a}$ from $\mathcal{M}$, the set $\{x \mid \mathcal{M} \models \varphi(x, \vec{a})\}$ is composed of fewer than $K$ intervals.*

### Proof of proposition 1

*Proof:* Let $\alpha(x)$ be an active-semantics formula defining the active domain of the output of $\varphi$. Let $\Psi_0$ be the following natural-semantics sentence

$$\neg\exists x_1, x_2 \ ((x_1 < x_2) \wedge (\forall x \ x_1 < x < x_2 \rightarrow \alpha(x)))$$

and let $\Psi$ be an equivalent active-semantics sentence (which exists by Fact 2). We then define $\varphi_{\text{safe}}$ as $\varphi \wedge \Psi$. The proposition then follows from the following claim: $D \models \Psi$ iff $\varphi[D]$ is finite.

First, assume that $D \models \neg\Psi$; then clearly $\varphi[D]$ is infinite because $<$ is dense. Suppose $D \models \Psi$. We then look at all occurrences of $SC$ predicates in $\alpha$ and replace each of them with a disjunction of tuples. This results in $\alpha'(x)$ in the language of $\Omega$ and constants for elements of $\mathcal{U}$; further, $D \models \alpha(a)$ iff $\mathcal{M} \models \alpha'(a)$. Let $\Psi'$ be obtained from $\Psi$ by substituting $\alpha'$ for $\alpha$. We then have $\mathcal{M} \models \Psi'$. Since $\alpha'(\mathcal{M}) = \{a \mid \mathcal{M} \models \alpha'(a)\}$ is a finite union of points and intervals, and since $\mathcal{M} \models \Psi'$, it follows that no nondegenerate interval is in $\alpha'(\mathcal{M})$. Thus, from o-minimality, we get that $\alpha'(\mathcal{M})$ is a finite union of points. Hence, $\{a \mid D \models \alpha(a)\}$ is finite, thus showing finiteness of $\varphi[D]$. $\square$

### Proof of proposition 2

Let $P$ be a ternary predicate defined as: $P(i, j, k)$ iff the $i$th Turing machine on the input $j$ makes at least $k$ moves (assuming some standard encoding of machines and inputs). Consider the schema that consists of a single binary relation $U$. Assume to the contrary that there is a safe translation over $\mathcal{M}$. Let $\varphi(k) \equiv \exists i, j \in adom\ U(i,j) \wedge P(i,j,k)$, and let $\psi(k)$ be $\varphi_{\text{safe}}$. Note that $\psi$ is an active-domain formula in the language of $U$ and $P$. We now show how to use $\psi$ to decide the halting problem.

Suppose we are given the $i$th machine $M_i$ and the input $j$. We assume without loss of generality that $M_i$ makes at least one move on $j$. Define $D$ that consists of a single pair $(i, j)$. Since we know that $\psi$ is safe, we then compute the minimum number $l$ such that $D \not\models \psi(l)$. It is computable since a) it exists, and b) for each $k$, it is decidable whether $D \models \psi(k)$.

Assume that $D \models \varphi(l)$. Then $M_i$ does not halt on $j$. Indeed, if $M_i$ halts, then $\varphi[D]$ is finite, and hence $\varphi[D] = \psi[D]$, but we have $l \in \varphi[D] - \psi[D]$. Assume that $D \not\models \varphi(l)$. Then $M_i$ makes $k < l$ moves on $j$, and thus halts. Hence, $D \models \varphi(l)$ iff $M_i$ halts on $j$. Since it is decidable whether $D \models \varphi(l)$, we get a contradiction. $\square$

### Proof of proposition 3

We start with a lemma.

**Lemma 6** *Let $\mathcal{M} = \langle \mathcal{U}, \Omega \rangle$ be o-minimal and based on a dense order, and let $\gamma(\vec{x}; \vec{y})$ be algebraic. Then:*

- *There exists a number $K$ such that for any $\vec{b} \in \mathcal{U}^m$, the set $\{\vec{a} \in \mathcal{U}^n \mid \mathcal{M} \models \gamma(\vec{a};\vec{b})\}$ has fewer than $K$ elements.*

- *There is a recursively enumerable collection of algebraic formulae $\{\chi_i(\vec{x};\vec{y})\}$ such that every algebraic formula $\gamma(\vec{x};\vec{y})$ is equivalent to one of the $\chi_i$* [2]*. If $\mathcal{M}$ is decidable, then the collection of algebraic formulae is recursive.*

*Proof of the lemma:* For the first item, consider all formulae $\gamma_i(x_i;\vec{y}) \equiv \exists \vec{x}^{(i)}\ \gamma(\vec{x};\vec{y})$, where $\vec{x}^{(i)}$ contains all components of $\vec{x}$ except $x_i$. By Fact 1, there is an integer $K_i$ such that for each $\vec{b}$, $\{a \mid \mathcal{M} \models \gamma_i(a;\vec{b})\}$ is composed of fewer than $K_i$ intervals. Since $\gamma$ is algebraic and $<$ is dense, it means that $\{a \mid \mathcal{M} \models \gamma_i(a;\vec{b})\}$ has fewer than $K_i$ elements. Hence, there are at most $K = \prod_{i=1}^n K_i$ vectors $\vec{a}$ such that $\mathcal{M} \models \gamma(\vec{a};\vec{b})$.

For the second item, for each $\gamma(\vec{x};\vec{y})$, let $\chi_\gamma^k(\vec{x})$ be a first-order formula

$$\forall \vec{y} \neg \exists \vec{x}_1 \ldots \exists \vec{x}_k . \bigwedge_{i=1}^{k} \gamma(\vec{x}_i;\vec{y}) \wedge \bigwedge_{i \neq j} (\vec{x}_i \neq \vec{x}_j)$$

saying that there are fewer than $k$ vectors $\vec{x}$ that satisfy $\gamma(\vec{x};\vec{y})$, for all $\vec{y}$. Consider the r.e. collection of formulae of the form $\gamma(\vec{x};\vec{y}) \wedge \chi_\gamma^k(\vec{x})$ where $\gamma$ ranges over $\mathrm{FO}(\Omega)$ and $k$ ranges over $\mathbb{N}$. It follows from the first item that it enumerates all algebraic formulae. If $\mathcal{M}$ is decidable, it is also decidable whether $\gamma(\vec{x};\vec{y})$ is algebraic. Indeed, the latter happens iff $\gamma^*(x;\vec{y}) = \bigvee_i \gamma_i(x;\vec{y})$ is algebraic (see the proof of the first item), which in turn happens iff there is no open interval contained in $\{a \mid \mathcal{M} \models \gamma^*(a;\vec{b})\}$ for any $\vec{b}$. The latter can be formulated as a first-order sentence $\forall \vec{y} \neg \exists u, v \forall z.(u < z < v \rightarrow \gamma^*(z;\vec{y}))$. □

We prove the one-variable case of proposition 3 first.

Let $\Xi$ be a collection of formulae, and assume that $\beta_\Xi$ is not algebraic. That is, for some $\vec{b}$ over $\mathcal{U}$, $\beta_\Xi(\mathcal{M};\vec{b}) = \{a \mid \mathcal{M} \models \beta_\Xi(a;\vec{b})\}$ is infinite. Since $\mathcal{M}$ is o-minimal, $\beta_\Xi(\mathcal{M};\vec{b})$ is a finite union of points and intervals. Since $<$ is dense, it means that there exist $a_0 < b_0 \in \mathcal{U}$ such that $[a_0, b_0] \subseteq \beta_\Xi(\mathcal{M};\vec{b})$. We now consider the formulae $\xi_i'(x) = \xi_i(x;\vec{b})$ for all $\xi_i \in \Xi$. Since both $\xi_i'(\mathcal{M})$ and $\neg\xi_i'(\mathcal{M})$ are finite unions of intervals and $<$ is dense, for every non-degenerate interval $I$, it is the case that either $I \cap \xi_i'(\mathcal{M})$ or $I \cap \neg\xi_i'(\mathcal{M})$ contains an infinite (closed) interval. Using this, we construct a sequence of intervals as follows: $I_0 = [a_0, b_0]$, $I_1 \subseteq I_0$ is an interval that is contained either in $I_0 \cap \xi_1'(\mathcal{M})$ or in $I_0 \cap \neg\xi_1'(\mathcal{M})$. At the $j$th step, $I_j \subseteq I_{j-1}$ is an interval that is contained either in $I_{j-1} \cap \xi_j'(\mathcal{M})$ or in $I_{j-1} \cap \neg\xi_j'(\mathcal{M})$. Let $I = I_k$. Then, for any $c, d \in I$, $\mathcal{M} \models \xi_i(c,\vec{b}) \leftrightarrow \xi_i(d;\vec{b})$.

Since $I = [a', b'] \subseteq [a_0, b_0]$ and $\mathcal{M} \models \beta_\Xi(c;\vec{b})$ for all $c \in I$, we obtain that, for every $c \in (a', b')$, there exists $d \in [a', b']$ such that $\mathcal{M} \models \neg\mathrm{same}(c, d;\vec{b})$. That is, for some $\xi_i \in \Xi$, $\mathcal{M} \models \neg(\xi_i(c;\vec{b}) \leftrightarrow \xi_i(d;\vec{b}))$, which is impossible by construction of $I$. This proves that $\beta_\Xi$ is algebraic.

For the converse, we let, for any $\beta(x;\vec{y})$, $\Xi$ consist of just $\beta$. That is, $\beta_\Xi(x;\vec{y})$ is

$$\forall x', x''.x' < x < x'' \rightarrow (\exists z.x' \leq z \leq x'' \wedge \neg(\beta(x;\vec{y}) \leftrightarrow \beta(z;\vec{y})))$$

We claim that $\beta$ and $\beta_\Xi$ are equivalent, if $\beta$ is algebraic. Fix any $\vec{b}$ of the same length as $\vec{y}$, and assume that $\beta(a;\vec{b})$ holds. If $\beta_\Xi(a;\vec{b})$ does not hold, then there exist $a' < a < a''$ such that for every $c \in [a', a'']$, $\beta(c;\vec{b}) \leftrightarrow \beta(a;\vec{b})$ holds; thus, $\beta(c;\vec{b})$ holds for infinitely many $c$, contradicting algebraicity of $\beta$. Hence, $\beta_\Xi(a;\vec{b})$ holds. Conversely, assume that $\beta_\Xi(a;\vec{b})$ holds. If $\beta(a;\vec{b})$ does not hold, then there is an interval containing $a$ on which $\beta(\cdot;\vec{b})$ does not hold. Indeed, $\neg\beta(\mathcal{M};\vec{b})$ is a finite union of intervals, whose complement is a finite set of points, so the above observation follows from the density. We now pick $a' < a''$ such that $\beta(\cdot;\vec{b})$ does not hold on $[a', a'']$. Since $\beta_\Xi(a;\vec{b})$ holds, we find $c \in [a', a'']$ such that $\neg(\beta(a;\vec{b}) \leftrightarrow \beta(c;\vec{b}))$ holds; that is, $\beta(c;\vec{b})$ holds for $c \in [a', a'']$, which is impossible. Thus, we conclude that $\beta(a;\vec{b})$ holds, proving that $\beta$ and $\beta_\Xi$ are equivalent, and finishing the one variable case.

[2] Provided that the set of $\mathrm{FO}(\Omega)$ formulae is recursive.

For the multi-variable case, we note that algebraicity of $\beta_\Xi$ implies that $\varphi'(\vec{x};\vec{y}) = \beta_\Xi(x_1;\vec{y}) \wedge \ldots \wedge \beta_\Xi(x_n;\vec{y}) \wedge \psi(\vec{x};\vec{y})$ is algebraic. Conversely, let $\varphi(\vec{x};\vec{y})$ be algebraic. Consider

$$\varphi_i(x_i;\vec{y}) = \exists x_1 \ldots \exists x_{i-1} \exists x_{i+1} \ldots \exists x_n . \varphi(x_1, \ldots, x_i, \ldots, x_n; \vec{y})$$

Let $\chi(x;\vec{y})$ be $\varphi_1(x;\vec{y}) \vee \ldots \vee \varphi_n(x;\vec{y})$. Obviously each $\varphi_i$ is algebraic, and thus $\chi(x;\vec{y})$ is algebraic. Hence, $\chi(x;\vec{y})$ is equivalent to $\beta_\Xi(x;\vec{y})$ for some finite collection $\Xi$ of formulae in $(x;\vec{y})$. Note that if $\varphi(\vec{a};\vec{b})$ holds and $a_i$ is the $i$th component of $\vec{a}$, then $\chi(a_i;\vec{b})$ holds and thus $\beta_\Xi(a_i;\vec{b})$ holds. This shows that $\varphi$ is equivalent to $\beta_\Xi(x_1;\vec{y}) \wedge \ldots \wedge \beta_\Xi(x_n;\vec{y}) \wedge \varphi(\vec{x};\vec{y})$, thus completing the proof. □

## Proof of theorem 1

We deal with the one-variable case first. Let

$$\varphi(z) \;\equiv\; Q_1 w_1 \in adom \ldots Q_1 w_k \in adom . \alpha(z, \vec{w})$$

where each $Q_i$ is $\exists$ or $\forall$ and $\alpha(z,\vec{w})$ is quantifier-free, and all atomic subformulae $R(\cdots)$ contain only variables, excluding $z$. Any formula can be transformed into such by adding existential quantifiers, cf. [6, 8]. Let $\Xi = \{\xi_i(z,\vec{w}) \mid i = 1, \ldots, k\}$ be the collection of all $\Omega$-atomic subformulae of $\alpha$. We may assume without loss of generality that the length of $\vec{w}$ is nonzero, and that $\Xi$ is nonempty (if this is not true for $\varphi$, take $\varphi \wedge \forall w \in adom(w = w)$ and transform it to the above form).

Define $\mathrm{same}_\Xi(a, b, \vec{w})$, as before, to be $\bigwedge_{i=1}^{k} (\xi_i(a,\vec{w}) \leftrightarrow \xi_i(b,\vec{w}))$, and define $\gamma(x;\vec{w})$ to be $\beta_\Xi(x;\vec{w})$; that is, $\gamma(x,\vec{w}) \;\equiv\; \forall x', x''. x' < x < x'' \rightarrow \exists y.[x' \le y \le x'' \wedge \neg \mathrm{same}_\Xi(x,y,\vec{w})]$. Now $\Gamma$ consists just of $\gamma$, with $\vec{w}$ being distinguished parameters. We let Make_Safe($\varphi$) output $(\{\gamma\}, \varphi)$.

Since $\gamma$ is algebraic by Proposition 3, we must show that $\{a \mid DB \models \varphi(a)\} = \{a \in \Gamma(DB) \mid DB \models \varphi(a)\}$ for every nonempty database for which $\varphi$ is safe.

Assume otherwise; that is, for some nonempty $D$ for which $\varphi$ is safe, we have $D \models \varphi(a)$ but $a \notin \Gamma(D)$. Let $\vec{c}_1, \ldots, \vec{c}_M$ be an enumeration of all vectors of the length of $\vec{w}$ of elements of the active domain. Note that $M > 0$. Since $a \notin \Gamma(D)$, we have that for each $i = 1, \ldots, M$, there exist $a_i, a_i'$ such that $a_i < a < a_i'$ and $\mathcal{M} \models \mathrm{same}_\Xi(a, c, \vec{c}_i)$ for all $c \in [a_i, a_i']$.

Let $b = \max\{a_i\}, b' = \max\{a_i'\}$. We have $b < a < b'$, and for each $\vec{c}$ (of length of $\vec{w}$) over the active domain, we have $\xi_i(a;\vec{c}) \leftrightarrow \xi_i(c,\vec{c})$ for every $c \in [b, b']$. From this, by a simple induction on the structure of the formula (using the fact that $z$ does not appear in any atomic formula $R(\cdots)$), we obtain that $D \models \alpha(a,\vec{c}) \leftrightarrow \alpha(c,\vec{c})$ for every $\vec{c}$ over $adom(D)$ and every $c \in [b, b']$, and thus $D \models \varphi(a) \leftrightarrow \varphi(c)$, which implies that $\varphi$ is not safe for $D$. This contradiction proves correctness of Make_Safe for the one-variable case.

This completes the proof for the one-variable case. We handle the multivariable case by reducing to the one-variable case.

Let $\mathcal{M}' = \langle \mathcal{U}, \Omega' \rangle$ be a definable extension of $\mathcal{M}$ that has quantifier-elimination. Note that $\mathcal{M}'$ is o-minimal. Let $\varphi(z_1, \ldots, z_n)$ be given, and Define

$$\varphi_i(z_i) \;\equiv\; \exists z_1 \ldots \exists z_{i-1} \exists z_{i+1} \ldots \exists z_n . \varphi(z_1, \ldots, z_{i-1}, z_i, z_{i+1}, \ldots, z_n).$$

By [8], there is a $L(SC, \Omega')$ *active* formula $\psi_i(z_i)$ such that $D \models \forall z. \psi_i(z) \leftrightarrow \varphi_i(z)$ for all $D$. Let $(\{\gamma_i(z_i, \vec{w}_i)\}, \psi_i(z_i))$ be the output of Make_Safe on $\psi_i$. Since $\mathcal{M}''$ is a definable extension, we can assume without loss that $\gamma_i$ is a $\Omega$-formula.

We now define

$$\gamma(\vec{z}; \vec{w}_1, \ldots, \vec{w}_n) \;\equiv\; \gamma_1(z_1; \vec{w}_1) \wedge \ldots \wedge \gamma_n(z_n; \vec{w}_n),$$

where each $\vec{w}_i$ is of the same length as the vector of distinguished parameters in the formulae $\gamma_i$. Finally, Make_Safe($\varphi$) outputs $(\{\gamma\}, \varphi)$. To see that it works, first notice that algebraicity of all $\gamma_i$s implies algebraicity of $\gamma$. Now assume that $D \models \varphi(\vec{a})$ where $\vec{a} = (a_1, \ldots, a_n)$. Then $D \models \varphi_i(a_i)$, and thus for some vector $\vec{c}_i$ of elements of the active domain, we have that $\gamma_i(a_i, \vec{c}_i)$ holds. Thus, if $\vec{c}$ is the concatenation of all $\vec{c}_i$s, then $\gamma(\vec{a}, \vec{c})$ holds, showing that $\vec{a} \in \Gamma(D)$, where $\Gamma = \{\gamma\}$. This completes the proof of the multivariable case.

We finally notice that Make_Safe for one-variable formulae is recursive, no matter what $\mathcal{M}$ is. For the multivariable case, to make it recursive, we need a procedure for converting natural-quantification formulae into active-quantification formulae. Such a procedure exists by Fact 2. $\square$

## Proof of proposition 4

Assume $\varphi$ is active-semantics. Then the proof follows the proof of Theorem 1, but at the end we replace $\gamma(x_1, \ldots, x_n; \vec{y})$ by

$$\gamma'(x; \vec{y}) \;=\; \bigvee_{i=1}^{n} \exists x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n \; \gamma(x_1, \ldots, x_n; \vec{y})$$

to get a bound on the active domain, and output $(\{\gamma'\}, \varphi)$.

For an arbitrary $\varphi$, let $\mathcal{M}'$ be a definitional expansion of $\mathcal{M}$ that has QE (such an expansion always exists, cf. [9]). Since $\mathcal{M}$ and $\mathcal{M}'$ are elementary equivalent, $\mathcal{M}'$ is o-minimal [26] and the order is dense. Thus, there exists an active-semantics $\mathrm{FO}(\mathcal{M}', SC)$ query $\psi(\vec{x})$ that is equivalent to $\varphi$ (see Fact 2), and, by the above, we have a formula $\gamma_0$ in the language of $\mathcal{M}'$ such that $adom(\varphi[D]) = adom(\psi[D]) \subseteq \gamma_0(D)$. Now obtain a $L(\Omega)$ formula $\gamma$ from $\gamma_0$ by replacing each new predicate symbol from $\mathcal{M}'$ by its definition by a $L(\Omega)$ formula. Then $\gamma(D) = \gamma_0(D)$, which proves the proposition. $\square$

## Proof of lemma 1

Assume not and fix a counterexample $a, \vec{b}$ and let $B$ be the set of components of $\vec{b}$ together with $-\infty, \infty$. Let $b', b''$ in $B$ be such that $b' < a < b''$ and $(b', b'')$ contains no other member of $B$. Since $\mathcal{M}^<$ has QE, $\gamma$ is equivalent to a quantifier-free formula. For any $a' \in (b', b'')$, $(a, \vec{b})$ and $(a', \vec{b})$ satisfy the same atomic $<$-formulae, and hence $\mathcal{M}^< \models \gamma(a, \vec{b}) \leftrightarrow \gamma(a', \vec{b})$, which contradicts algebraicity. $\square$

## Proof of proposition 5

Given a query $\varphi(\vec{x})$, find effectively a collection of algebraic formulae $\Gamma = \{\gamma_j(x; \vec{y})\}_j$ such that for any $D$ for which $\varphi$ is safe, $adom(\varphi[D]) \subseteq \Gamma(D)$. For each $\vec{a}$ and each $\gamma \in \Gamma$, the set $\gamma[\vec{a}] = \{c \mid \gamma(c; \vec{a})\}$ is finite, and by o-minimality there is a uniform bound $M$ such that $card(\gamma[\vec{a}]) < M$ for all $\vec{a}$ and $\gamma \in \Gamma$.

Now let $\gamma_j^i(x; \vec{y})$, $i < M$, be defined as follows: $\mathcal{M} \models \gamma_j^i(c; \vec{a})$ if either (1) $\gamma_j[\vec{a}]$ has at least $i$ elements, and $c$ is the $i$th element in the order $<$, or (2) $\gamma_j[\vec{a}]$ is nonempty, has fewer than $i$ elements, and $c$ is the largest element of $\gamma_j[\vec{a}]$, or (3) $\gamma_j[\vec{a}]$ is empty, and $c = 0$. Note that $\gamma_j^i(x; \vec{y})$s are indeed $L(+, *, 0, 1, <)$ formulae. It is easy to see that each $\gamma_j^i(x; \vec{y})$ defines a function $f_{ij} : \mathbb{R}^m \to \mathbb{R}$, where $m$ is the length of $\vec{y}$, by $f_{ij}(\vec{a}) = c$ iff $c$ is the unique element such that $\gamma_j^i(c; \vec{a})$ holds. Furthermore, this function is semialgebraic and the following property holds: if $\varphi$ is safe for $D$, then $adom(\varphi[D])$ is contained in $\bigcup_{i,j} \bigcup_{\vec{a}} f_{ij}(\vec{a})$, where $\vec{a}$ ranges over $adom(D)$.

It follows from [19] that each $f_{ij}(\vec{y})$ is algebraic, that is, there exists a polynomial $p_{ij}(x, \vec{y})$ such that $p_{ij}(x, \vec{y}) = 0$ iff $x = f_{ij}(\vec{y})$. It is easy to see that for $P = \{p_{ij} \mid i < M, \gamma_j \in \Gamma\}$, $\Gamma(D) \subseteq Root_P(D)$ and $Root_P(D)$ is always finite. To complete the proof, we must show effectiveness. We can effectively construct $\Gamma$, and thus find $M$ (by writing formulae saying that each $\gamma(x; \vec{y})$ has fewer than $M$ satisfiers for each $\vec{y}$, and checking if it is true by applying QE; since it is true for some $M$, the process terminates). Hence, we can effectively construct $\gamma_j^i$s, and the procedure for finding $p_{ij}$s is effective (although not stated in [19], it follows from the analysis of the proof there). $\square$

## Proof of lemma 2

Proof of lemma 2 follows from Proposition 4: take $\Gamma = \{\gamma_1(x;\vec{y}), \ldots, \gamma_k(x;\vec{y})\}$ given by the proposition; let $\gamma = \bigvee_i \gamma_i$ and let $\psi(x_1, \ldots, x_n)$ be

$$\exists \vec{y}_1 \in adom \ldots \exists \vec{y}_n \in adom\ \gamma(x_1;\vec{y}_1) \wedge \ldots \wedge \gamma(x_n;\vec{y}_n)$$

If $\varphi \subseteq \psi$, then $\varphi$ is safe since all $\gamma_i$s are algebraic. If $\varphi$ is safe, then $adom(\varphi[D]) \subseteq \Gamma(D)$ for every $D$, which implies $\varphi \subseteq \psi$. □

## Proof of lemma 3 and theorem 2

We prove the following:

**Claim 1** *Given $\varphi$ and $\psi$, one can effectively find a number $k$ such that $\varphi \subseteq \psi$ iff for every database $D$ with at most $k$ tuples, $\varphi[D] \subseteq \psi[D]$.*

This clearly implies the result, as the latter condition can be expressed as a $L(\Omega)$ sentence. For example, if the schema contains one relational symbol $S$, this sentence is $\forall \vec{x}_1 \ldots \vec{x}_k \forall \vec{x}.\ \varphi(\vec{x})[\{\vec{x}_i\}/D] \rightarrow \varphi(\vec{x})[\{\vec{x}_i\}/D]$ where $\varphi(\vec{x})[\{\vec{x}_i\}/D]$ is obtained from $\varphi(\vec{x})$ by replacing each occurrence of $S(\vec{z})$ by $\bigvee_i(\vec{z} = \vec{x}_i)$, and similarly for an arbitrary schema. The decidability of $\mathcal{M}$ now implies the lemma.

The proof of the claim proceeds similarly to [15]. Note that every BCCQ $\alpha$ can be represented as $\bigvee_{i=1}^{n}(\chi_i(\vec{x}) \wedge \neg\xi_i(\vec{x}))$ where each $\chi_i$ is CQ and each $\xi_i$ is UCQ; this follows if one writes $\alpha$ as a DNF. We take $k$ to be the maximum length of $\chi_i$ (measured as the sum of the number of atomic formulae and the number of quantified variables).

Assume that $\varphi[D] \not\subseteq \psi[D]$ for some $D$; that is, we have $\vec{a} \in \varphi[D] \not\subseteq \psi[D]$. Assume $D \models \chi_i(\vec{a}) \wedge \neg\xi_i(\vec{a})$ and let $\chi_i(\vec{x}) = \exists\vec{y}\exists\vec{z} \in adom\ \bigwedge_{j=1}^{l} \alpha_j(\vec{x},\vec{y},\vec{z})$. Then, for some $\vec{b}$ over $\mathcal{U}$ and $\vec{c}$ over $adom(D)$, we get $D \models \bigwedge_{j=1}^{l} \alpha_j(\vec{a},\vec{b},\vec{c})$. Consider those $\alpha_j$s which are $SC$-formulae. For each such $\alpha_j$, which is of the form $R(\cdots)$ where $R \in SC$, there is a tuple in $D$ that satisfies it. Select one such tuple for each $SC$-atomic $\alpha_j$, and let $D'$ be $D$ restricted to those tuples. Choose a set of at most $length(\vec{c})$ tuples in $D$ contain all the components of $\vec{c}$, and add it to $D'$. Let the resulting database be $D''$. Clearly, it has at most $k$ tuples.

Note that $D'' \models \bigwedge_{j=1}^{l} \alpha_j(\vec{a},\vec{b},\vec{c})$, and thus $D'' \models \chi_i(\vec{a})$ since $\vec{c} \subseteq adom(D'')$. On the other hand, $D'' \models \neg\xi_i(\vec{a})$ by monotonicity of $\xi_i$. Thus, we get that $\vec{a} \in \varphi[D''] - \psi[D'']$ where $D''$ has at most $k$ tuples. This implies that each counterexample to containment is witnessed by a $\leq k$-element database, and finishes the proof Lemma 3.

To complete the proof of Theorem 2, note that under the assumption on $\mathcal{M}$, the CQ $\psi$ such that $\varphi \subseteq \psi$ is equivalent to safety of $\varphi$ can be effectively constructed – this follows from the procedure for constructing $\Gamma$ given in Proposition 4. The theorem now follows from Lemma 3. □

## Proof of theorem 3

Consider $\Gamma$ given by Proposition 4. Since each $\gamma \in \Gamma$ is algebraic, there exists, by Lemma 6, a number $c_\gamma$ such that $card(\{x \mid \gamma(x;\vec{y})\}) < c_\gamma$ for every $\vec{y}$. Thus, if $adom(\varphi[D])$ is finite, then its cardinality is at most

$$\sum_{\gamma\in\Gamma} c_\gamma \cdot n^{m_\gamma}$$

where $m_\gamma$ is the number of $\vec{y}$ variables in $\gamma$, and $n$ is the size of the active domain of $D$. This is clearly bounded by $card(\Gamma) \cdot C_\Gamma \cdot n^{M_\Gamma}$, where $C_\Gamma = \max c_\gamma$ and $M_\Gamma = \max m_\gamma$.

Now notice that for every schema $SC$, there exist constants $c_0, d_0, c_1 > 0$ such that $c_1 \cdot card(adom(D)) \leq \mathsf{size}(D) \leq c_0 \cdot card(adom(D))^{d_0}$ for every $D$. Hence, for appropriately chosen $c_0, d_0 > 0$, we obtain that if $\varphi[D]$

is finite, then

$$\mathsf{size}(\varphi[D]) \le c_0 \cdot card(adom(\varphi[D]))^{d_0} \le c_0 \cdot (card(\Gamma) \cdot C_\Gamma \cdot n^{M_\Gamma})^{d_0} \le cn^d,$$

where $n = card(adom(D))$ and $c, d$ are constant that depend only on $\varphi$. Hence, for some $c_1 > 0$ that depends on the schema only, we have $\mathsf{size}(\varphi[D]) \le c \cdot (\frac{n}{c_1})^d$, which proves the theorem. □

## Proof of proposition 6

Let $SC$ consist of one unary relation $S$. We show that there exists a FO($\mathcal{M}, SC$) sentence $\Psi$ such that $S \models \Psi$ iff $S$ is of the form $S_n = \{2^i \mid 1 \le i \le n\}$. This is done by letting $\Psi$ be

$$\begin{array}{ll} & (\exists x \in adom. x = 1+1 \wedge S(x)) \\ \wedge & (\forall x \in adom. x = 1+1 \vee x > 1+1) \\ \wedge & (\forall x \in adom. x = 1+1 \vee \exists y \in adom. y+y = x) \\ \wedge & (\forall x \in adom. (\forall y \in adom. y < x \vee y = x) \vee (\exists y \in adom. y = x+x)) \end{array}$$

Now define $\varphi(x)$ as $\Psi \wedge \neg(x < 1) \wedge (\exists y \in adom. x < y \vee x = y)$. Then, for $S$ not of the form $S_n$, we have $\varphi[S] = \emptyset$, and $\varphi[S_n] = \{1, 2, 3, \ldots, 2^n\}$. Since $card(S_n) = n$, this implies $growth_\varphi(n) = 2^n$ for $n > 0$. □

## Proof of theorem 4

In view of the dichotomy theorem, it remains to show that if $growth_\varphi$ is not bounded by a constant, then it is bounded below by a polynomial. Assuming $growth_\varphi$ is not bounded by a constant, we get a family of databases $\{D_i\}_{i\in\mathbb{N}}$ such that $\mathsf{size}(\varphi[D_i]) > i$ for all $i$. Because of monotonicity, we can ensure, by adding elements to $D_i$, that $\mathsf{size}(D_i) \ge i$.

In the proof of Lemma 3 we showed that there exists a constant $k$ that depends on $\varphi$ only, such that $\vec{a} \in \varphi[D]$ iff there is $D' \subseteq D$ with at most $k$ tuples such that $\vec{a} \in \varphi[D']$. Thus, there is $D_i' \subseteq D_i$ with at most $k(i+1)$ tuples such that $\mathsf{size}(\varphi[D_i']) > i$.

Now, for a given $n$, let $i$ be the maximal such that $k(i+1) \le n$. Consider $D_i'$ and extend it to contain exactly $n$ tuples. For the resulting $D_i''$, we have $\mathsf{size}(\varphi[D_i']) > i$ by monotonicity; hence $growth_\varphi(n) > i$. Since $n \le k(i+1)$, we get from this that $growth_\varphi(n) \ge \frac{n}{k} - 1$, which completes the proof. □

## Proof of theorem 5

Consider an enumeration of all safe FO($\Omega$) queries $\langle \varphi_i \rangle$ (from Corollary 2, we know that it exists). Let $\varphi$ use the extra relation symbol of arity $m$, and assume that $n$ is such that $g(n) = m$; given the assumptions, we can compute that. Let $\varphi_i$ have $l$ parameters, and again let $k$ be such that $g(k) = l$. If $n$ and $k$ are found for a given $\varphi_i$, we let $\psi$ be:

$$decode_k \circ \varphi_i \circ code_n$$

This produces the required enumeration. So we have to check that every query above preserves $\mathcal{C}$, and for every $\mathcal{C}$ preserving $\psi$, we can get $\varphi$ such that $decode \circ \varphi \circ code$ coincides with $\psi$. The first one is clear: if we have $X \in \mathcal{C}$, then $code_n(X)$ is finite, hence $\varphi_i[code_n(X)]$ is finite too, and applying $decode_k$ we get an object in $\mathcal{C}$.

For the converse, suppose we have a $\mathcal{C}$-preserving query $\psi : \mathsf{Obj}_n(\Omega) \to \mathsf{Obj}_k(\Omega)$. Define $\alpha$ as follows: $\alpha = code_k \circ \psi \circ decode_n$. That is, $\alpha$ is a query $\mathsf{Obj}_m(\Omega) \to \mathsf{Obj}_l(\Omega)$. Given this, notice that

$$decode_k \circ \alpha \circ code_n = decode_k \circ code_k \circ \psi \circ decode_n \circ code_n = \psi$$

on $\mathsf{Obj}_n(\Omega)$. Thus, it remains to show that $\alpha$ is safe, i.e. preserves finiteness. Let $X$ be a finite set in $\mathcal{U}^m$. Then $decode_n(X) \in \mathcal{C}$, $decode_n(X) \subset \mathcal{U}^n$. Since $\psi$ is $\mathcal{C}$-preserving, we get that $Y = \psi[decode_n(X)] \in \mathsf{Obj}_k(\Omega)$ is in $\mathcal{C}$, too, and thus $code_k(Y)$ is finite. This proves finiteness of $\alpha$, and concludes the proof of the theorem. □

## Proof of proposition 7

Given a convex polytope $X$ in $\mathbb{R}^n$, its vertices can be found as $V(X) = \{\vec{x} \in \mathbb{R}^n \mid \vec{x} \in X, \vec{x} \notin \mathrm{conv}(X - \vec{x})\}$. Thus, vertices of convex polytopes are definable in FO$(+,*,0,1,<)$, because the convex hull of a finite set of points is definable, and, in view of Carathéodory's theorem, we have

$$V(X) = \{\vec{x} \in \mathbb{R}^n \mid \vec{x} \in X, \forall \vec{x}_1, \ldots, \vec{x}_{n+1} \in X - \vec{x}.\ \vec{x} \notin \mathrm{conv}(\{\vec{x}_1, \ldots, \vec{x}_{n+1}\})\}$$

We now define $code_n$. To simplify the notation, we let it produce a pair of $n$-ary relations, but it can be straightforwardly coded by one relation. If $X = \mathrm{conv}(V(X))$, then $code_n(X) = (V(X), \emptyset)$; otherwise, $code_n(X) = (\mathbb{R}^n, X)$. The function $decode_n : 2^{\mathbb{R}^n} \times 2^{\mathbb{R}^n} \to 2^{\mathbb{R}^n}$ is defined as follows:

$$decode_n(Y, Z) = \begin{cases} \bigcup_{(\vec{y}_1, \ldots, \vec{y}_{n+1}) \in Y} \ \mathrm{conv}(\{\vec{y}_1, \ldots, \vec{y}_{n+1}\}) & \text{if } Y \neq \mathbb{R}^n, \\ Z & \text{otherwise} \end{cases}$$

Clearly, $decode_n \circ code_n$ is the identity function for any (semialgebraic) set; these functions are also first-order definable. If $X$ is a polytope, $V(X)$ is finite, and by Carathéodory's theorem each point of $X$ is contained in the convex hull of most $n+1$ vertices of $X$. Hence, $card(code_n(X)) \leq card(V(X))^{n+1} < \omega$. If $(Y, Z)$ is finite, then $decode_n(Y)$ is $\mathrm{conv}(Y)$, and thus a convex polytope. This proves the proposition. □

## Proof of proposition 8

We first give a brief sketch of the coding scheme. For basic results about convex polyhedra used below, see [29]. Let $X$ be a convex polyhedron in $\mathbb{R}^n$. Then $X = L + (X \cap L^\perp)$, where $L$ is its lineality space, defined as $\{\vec{y} \mid \forall \vec{x} \in S.\ \vec{y} + \vec{x} \in X\}$ (it is a subspace of $\mathbb{R}^n$) and $L^\perp$ is the orthogonal subspace $\{\vec{y} \mid \forall \vec{x} \in L.\ \langle \vec{x}, \vec{y} \rangle = 0\}$. We shall use $X_0$ for $X \cap L^\perp$ in this proof. It is known that $X_0$ is a convex polyhedron of lineality zero, that is, it contains no line. By $A + B$ we mean $\{\vec{a} + \vec{b} \mid \vec{a} \in A, \vec{b} \in B\}$. Note the difference between the translate $X - \vec{x} = \{\vec{y} - \vec{x} \mid \vec{y} \in X\}$ and the set-theoretic difference $X - x$; we use $\vec{x}$ to distinguish between them.

For $X_0$, define its vertices as $x \in X_0$ such that $x \notin \mathrm{conv}(X_0 - x)$. A direction is given by a vector $\vec{y}$ and corresponds to the equivalence class of rays which are translate of each other. Note that each direction can be canonically represented by $\vec{y}$ such that $\| \vec{y} \| = 1$. A direction $\vec{y}$ is an extreme direction of $X_0$ if for some vertex $\vec{x}$, the ray through $\vec{x}$ in the direction of $\vec{y}$, $l(\vec{x}, \vec{y}) = \{\vec{x} + \lambda \cdot \vec{y} \mid \lambda \geq 0\}$, is a face of $X_0$. Since $X_0$ is polyhedral of lineality zero, the set of vertices and extreme directions is finite. By (generalized) Carathéodory's theorem, every point $\vec{z}$ of $X_0$ is a combination of at most $n+1$ vertices and extreme directions,

$$\lambda_1 \vec{x}_1 + \ldots + \lambda_k \vec{x}_k + \mu_1 \vec{y}_1 + \ldots + \mu_m \vec{y}_m,$$

where $\lambda_i, \mu_j \geq 0, \lambda_1 + \ldots + \lambda_k = 1, k + m \leq n + 1$.

This suggests the following coding scheme. As before, for simplicity of exposition, we assume several coding relations, but they can be combined into one easily. We also do not spell out every first-order formula, but the reader should be convinced from the mathematical definitions that all the concepts we use are first-order definable over the real field. We use relations $LINEAL_k$; each such relation contains a canonical representation (roughly, an orthogonal basis) of the lineality space of $X$, provided its dimension is $k$. That is, at most one of these relations actually contains some information. We then have the relations *Vert* and *ExtDir* for storing vertices and extreme directions of $X_0$. Finally, we have a relation *Points* that contains points that do not belong to $L + X_0$ (recall that the coding scheme applies to any semi-algebraic set, so there could be such points, and we need to record them for the *decode* function).

Thus, to code (an arbitrary semi-algebraic) set $X$, we first note that its lineality space $L(X) = \{\vec{y} \mid \forall \vec{x} \in X.\ \vec{x} + \vec{y} \in X\}$ and its orthogonal $L(X)^\perp = \{\vec{y} \mid \forall \vec{x} \in L(X).\ \langle \vec{x}, \vec{y} \rangle = 0\}$ are definable in FO + POLY (note that one can define the inner product in FO + POLY). Furthermore, for each $k \leq n$, there exists a FO + POLY sentence $dim_k$ expressing the fact that $L(X)$ is a subspace of $\mathbb{R}^n$ and its dimension is $k$. This is true because in FO + POLY we can test linear independence; thus, we can check if there exists a system of $k$ linearly independent vectors in $L$ such that every vector in $L$ is a linear combination of them.

Next, we show how to compute $\mathit{LINEAL}_k$ and *VertDir*. We first sketch the coding scheme for $\mathit{LINEAL}_k$. The set $L(X)$ is FO + POLY-definable. Assume that it is a $k$-dimensional linear space (which is tested by $\mathit{dim}_k$). Let $\Delta_n$ be some canonically chosen $n$-dimensional simplex of diameter 1 such that the origin has barycentric coordinates $(\frac{1}{n}, \ldots, \frac{1}{n})$. Consider intersection of $L(X)$ with 1-dimensional faces of $\Delta_n$ (unless $L(X)$ is a line, in which case we consider its intersection with 2-dimensional faces of $\Delta_n$). If the intersection is a point, we record that point; if it contains the whole face, we record both endpoints of the face. From the selected points, find a linearly independent subsystem (note that it can be done canonically, for example, by listing the vertices and 1-dimensional faces of $\Delta_n$ in some order). It then serves as a basis of $L(X)$, which we use to code $L(X)$. Note that $L(X)$ can be reconstructed in FO + POLY from its basis.

Now that we have a representation for the lineal space of $X$, and a first-order formula defining $L^\perp$, we have a FO + POLY-formula defining $X_0$. Using it, we can compute vertices

$$V(X_0) = \{x \in X_0 \mid \neg \exists x_1, \ldots, x_{n+1} \in X_0 - x.\ x \in \mathrm{conv}(\{x_1, \ldots, x_{n+1}\})\}.$$

Clearly, this is a first-order definition. Next, we find the set

$$E(X_0) = \{\vec{y} \mid \langle \vec{y}, \vec{y} \rangle = 1 \text{ and } \exists \vec{x} \in V(X_0).\ l(\vec{x}, \vec{y}) \text{ is a face}\}.$$

A subset $Y$ of $X_0$ is a face if every closed line segment in $X_0$ with a relative interior point in $Y$ has both endpoints in $Y$. Clearly, this is first-order definable, and thus $E(X_0)$, the set of extreme directions of $X_0$, is first-order definable.

Given two sets $V$ and $E$ in $\mathbb{R}^n$, by $\mathrm{conv}(V, E)$ we denote their convex hull, that is, the set of elements of $\mathbb{R}^n$ definable as $\sum_{i=1}^k \lambda_i \cdot \vec{x}_i + \sum_{j=1}^m \mu_j \cdot \vec{y}_j$, where $\vec{x}_i \in V$, $\vec{y}_j \in E$, $\sum_{i=1}^k \lambda_i = 1$, $\lambda_i, \mu_j \geq 0$ and $k + m \leq n + 1$. Again, this can be done in FO + POLY.

We now describe $\mathit{code}_n$. For a semi-algebraic set $X$, it produces a tuple of relations

$$(\mathit{LINEAL}_0, \ldots, \mathit{LINEAL}_n, \mathit{Vert}, \mathit{ExtDir}, \mathit{Point})$$

as follows. It first determines, by computing $L(X)$, $L(X)^\perp$, $V(X_0)$ and $E(X_0)$ if it is the case that $L(X)$ is a linear subspace of $\mathbb{R}^n$ and

$$X \;=\; L(X) + \mathrm{conv}(V(X_0), E(X_0)).$$

If this is the case, then $\mathit{LINEAL}_k$, *Vert*, and *ExtDir* are produced as before, and *Point* is empty. Otherwise, *Point* coincides with $X$, and all other sets in the coding are taken to be $\mathbb{R}^n$. From the description above it follows that $\mathit{code}_n$ is FO + POLY-definable.

To compute $\mathit{decode}_n$, we first check is the first $n+2$ relations in the code coincide with $\mathbb{R}^n$, and, if this is the case, output the last relation in the code. Otherwise, we use the nonempty $\mathit{LINEAL}_k$ with least $k$ to compute a linear subspace $L$ of $\mathbb{R}^n$ generated by the vectors in $\mathit{LINEAL}_k$ (if all $\mathit{LINEAL}_k$ are empty, we let this subspace be $\{\vec{0}\}$). Next, compute $Y = \mathrm{conv}(\mathit{Vert}, \mathit{ExtDir})$. Note that both are FO + POLY-definable. Finally, return $L + Y \cap L^\perp$; this is FO + POLY-definable also.

We now sketch the proof that this coding scheme satisfies the conditions of the definition of canonical representation. Both *code* and *decode* are FO + POLY-definable. If $X \in \mathcal{CPH}$, then $L(X)$ is a linear space, $X_0$ has finitely many vertices and extreme directions, and $X = L + X_0$ implies that *Point* is empty, thus showing that *code* produces a finite set. Assume that *decode* is given a finite input $Y$. Then none of the first $n+2$ relations is $\mathbb{R}^n$, and thus the output of *decode* is the sum of a vector space and a convex hull of a finite set of vertices and directions, and thus a convex polyhedron. To show that $\mathit{decode} \circ \mathit{code}(X) = X$ for any semi-algebraic $X$, consider two cases. If $X = L(X) + \mathrm{conv}(V(X_0), E(X_0))$, then *Point* is empty, and *Vert* and *ExtDir* record all vertices and extreme directions of $X_0$, and one of $\mathit{LINEAL}_k$ codes the lineality space. Thus, *decode* applied to $\mathit{code}(X)$ will return $L + X_0 = X$. If $X \neq L(X) + \mathrm{conv}(V(X_0), E(X_0))$, then all relations but *Point* coincide with $\mathbb{R}^n$, and *Point* contains $X$, and thus *decode* returns $X$. This completes the proof. $\square$

## Proof of proposition 9

An object in $\mathsf{SLinComp}_2$ is a finite union of convex polytopes in $\mathbb{R}^2$ – this easily follows from cell decomposition. Any such object $X$ admits a triangulation that does not introduce new vertices [22]. Thus, the idea of the coding is to find the set $V(X)$ of vertices and use as the code triples of vertices (not necessarily distinct) $(\vec{x},\vec{y},\vec{z})$ with $\mathrm{conv}(\{\vec{x},\vec{y},\vec{z}\}) \subseteq X$. More precisely, a triple $(\vec{x},\vec{y},\vec{z})$ belongs to $code(X)$ if either $\vec{x},\vec{y},\vec{z} \in V(X)$ and $\mathrm{conv}(\{\vec{x},\vec{y},\vec{z}\}) \subseteq X$, or $\vec{x}=\vec{y}=\vec{z}$ and there is no triple of elements of $V(X)$ whose convex hull is contained in $X$ and contains $\vec{x}$. Thus, $code(X) \subseteq \mathbb{R}^6$. For *decode*, we use

$$decode(Y) = \bigcup_{(\vec{x},\vec{y},\vec{z})\in Y} \mathrm{conv}(\{\vec{x},\vec{y},\vec{z}\}).$$

Clearly, *decode* $\circ$ *code* is the identity, *decode* is first-order definable, and $decode(Y)$ is compact and semi-linear when $Y$ is finite. Thus, it remains to show that $V(X)$ is finite and FO + POLY-definable. The former is well-known (see [21]). For the first-order definition of $V(X)$ we use the following result from [21]. Let $X$ be a finite union of polyhedra (in $\mathbb{R}^n$) and let $B_\epsilon(\vec{x})$ be the ball of radius $\epsilon$ around $\vec{x}$. Then for each $\vec{x}$, there exists $\delta > 0$ such that for any $0 < \epsilon_1, \epsilon_2 < \delta$, we have

$$\vec{x} + \bigcup_{\lambda>0} \lambda \cdot [(X \cap B_{\epsilon_1}(\vec{x})) - \vec{x}] \quad = \quad \vec{x} + \bigcup_{\lambda>0} \lambda \cdot [(X \cap B_{\epsilon_2}(\vec{x})) - \vec{x}].$$

We denote this set by $X(\vec{x})$. Define the equivalence relation $\equiv_X$ by $\vec{y} \equiv_X \vec{z}$ if $X(\vec{y}) = X(\vec{z})$. Then the vertices of $X$ are precisely the one-element equivalence classes of $\equiv_X$. It is routine to verify that the above can be translated into a FO + POLY definition of vertices. This completes the proof. $\square$

## Proof of lemma 4

We can assume without loss of generality that $\varphi(\vec{x})$ is of the form

$$\bigvee_j \exists \vec{z} \bigwedge_i \alpha_{ij}(\vec{x},\vec{z})$$

where each $\alpha_{ij}$ is either $S(\cdots)$ or a $L(+,*,0,1,<)$ formula. Let $k_0$ be the maximal number of $S$-atomic formulae in a disjunct of $\varphi$. Then the argument made in the proof of Lemma 3 shows that for each $\vec{a} \in \varphi[D]$, there exists a subset of $D' \subseteq D$ with at most $k_0$ points such that $\vec{a} \in \varphi[D']$.

Now assume that $D$ is a convex polytope in $\mathbb{R}^m$, and $V(D)$ is the set of its vertices. Then each element of $D$ belongs to $\mathrm{conv}(V')$ where $V'$ is a subset of $V$ of cardinality at most $m+1$. Now let $k' = k_0(m+1)$. Then it follows from monotonicity that for every $\vec{a} \in \varphi[D]$, there is a subset of $V' \subseteq V(D)$ of cardinality at most $k$ such that $a \in \varphi[\mathrm{conv}(V')]$; in particular,

$$\varphi[D] \;=\; \bigcup_{V' \subseteq V(D),\, card(V') \leq k'} \varphi[\mathrm{conv}(V')]$$

We now set $k = 2k'$.

Next, consider the following FO + POLY query $\varphi'$ which uses one $m$-ary relational symbol $R$. First, $\varphi'$ constructs the convex hull of points in $\mathbb{R}^m$ which are in $R$. Then it applies $\varphi$ to the result, to get a set $Y$. Finally, it returns $V(Y)$, that is, the set $\{y \in Y \mid y \notin \mathrm{conv}(Y - y)\}$. Clearly, it is expressible in FO + POLY. From the dichotomy theorem, we know that there is a polynomial $p$ with the following property: if $R$ is finite and contains $i$ points, then either $\varphi'[R]$ is infinite, or it contains at most $p(i)$ points. We now let $l = \max\{p(i) \mid i = 1,\ldots,k\}$. Note that both $k$ and $l$ can be effectively calculated for a given $\varphi$.

It remains to show that if $\varphi$ has the property that it sends a convex polytope with $\leq k$ vertices in $\mathbb{R}^m$ into a convex polytope with $\leq l$ vertices in $\mathbb{R}^n$, then it is $\mathcal{CPT}$-preserving. First, assume that $D$ is a convex polytope with $\leq k$ vertices in $\mathbb{R}^m$. Assume $\varphi[D]$ is a convex polytope. If we apply $\varphi'$ to a relation storing vertices of $D$, then the result is a finite set of vertices of $\varphi[D]$. Hence, by the dichotomy theorem, it has at most $l$ vertices. That is, it now suffices to show that if $\varphi$ has the property that it sends a convex polytope with $\leq k$ vertices in $\mathbb{R}^m$ into a convex polytope, then it is $\mathcal{CPT}$-preserving.

Let $D$ be a convex polytope. We know that $\varphi[D] = \bigcup_{V'} \varphi[\mathrm{conv}(V')]$ where $V'$ ranges over subsets of $V(D)$ that have at most $k' \leq k$ elements. Thus, each $\varphi[\mathrm{conv}(V')]$ is a convex polytope. Assume that $\varphi[D]$ is not convex.

Then we can find two sets of vertices $V_1, V_2$, having at most $k'$ elements each, and two points $\vec{a} \in \varphi[\text{conv}(V_1)]$, $\vec{b} \in \varphi[\text{conv}(V_2)]$, and $\vec{c}$ between $\vec{a}$ and $\vec{b}$ such that $\vec{c} \notin \varphi[D]$. Let $V_0 = V_1 \cup V_2$. Then, by monotonicity of $\varphi$, $\vec{a}, \vec{b} \in \varphi[\text{conv}(V_0)]$. By the assumption, $\varphi[\text{conv}(V_0)]$ is convex (since $card(V_0) \leq k$) and thus $\vec{c} \in \varphi[D]$. Hence, we showed that $\varphi[D]$ is convex. Since it is convex and a finite union of convex polytopes, it is a convex polytope itself. This completes the proof of Lemma 4 $\square$

## Proof of lemma 5

Assume, as in the proof of Lemma 4, that $\varphi(\vec{x})$ is of the form $\bigvee_j \ \exists \vec{z} \ \bigwedge_i \alpha_{ij}(\vec{x}, \vec{z})$ where each $\alpha_{ij}$ is either $S(\cdots)$ or a $L(+, *, 0, 1, <)$ formula, and let $k$ be the maximal number of $S$-atomic formulae in a disjunct of $\varphi$. Then, by the same argument as in the proof of Lemma 4, we obtain that if $\vec{x} \in \varphi[D]$, then there exist $k$ points $\vec{z}_1, \ldots, \vec{z}_k$ in $D$ such that $\vec{x} \in \varphi[\{\vec{z}_1, \ldots, \vec{z}_k\}]$. Assume that $D$ is compact and semilinear; since $D \subseteq \mathbb{R}^2$, it can be triangulated using only vertices of $D$. Let $V(D)$ be the set of vertices of $D$, which can be computed by a FO + Poly query, as shown in the proof of Proposition 9. Since every point of $D$ is in the convex hull of a triangle whose vertices come from $V(D)$, we obtain, by monotonicity of $\varphi$, that if $\vec{x} \in \varphi[D]$, then there exists a set $V'$ of triples of elements from $V(D)$ such that $card(V') \leq k$ and

$$\vec{x} \ \in \ \varphi[\bigcup_{(\vec{u},\vec{v},\vec{w}) \in V'} \text{conv}(\{\vec{u}, \vec{v}, \vec{w}\})].$$

Next, consider the following FO + Poly query $\psi$ on finite databases. It uses one 6-ary schema relation $R$, that can be thought of as storing triples $(\vec{u}, \vec{v}, \vec{w})$ of points in $\mathbb{R}^2$. First, $\psi$ computes

$$P(R) \ = \bigcup_{(\vec{u},\vec{v},\vec{w}) \in R} \text{conv}(\{\vec{u}, \vec{v}, \vec{w}\})],$$

which is a compact semi-linear set, and then it computes the set $V(P(R))$ of vertices of $P(R)$, using the technique of [21], that we exploited in the proof of Proposition 9: for each $\vec{x}$, the set $P(R)(\vec{x}) = \vec{x} + [(P(R) \cap B_\epsilon(\vec{x})) - \vec{x}]$ does not depend on a particular value $\epsilon$ below some threshold $\delta$. We then define vertices as those $\vec{x}$ for which there is no $\vec{x}_0 \neq \vec{x}$ with $P(R)(\vec{x}) = P(R)(\vec{x}_0)$. Thus, the query computing $V(P(R))$ is definable in FO + Poly, and by the dichotomy theorem, there is a polynomial $p$ such that, for each $R$, either $V(P(R))$ is infinite, or has at most $p(n)$ vertices, where $n$ is the number of tuples in $R$. We now let $l$ be $\max\{p(i) \mid i = 1, \ldots, k\}$.

To show that these $k$ and $l$ witness the conclusion of the lemma, assume that $\varphi$ is $\mathsf{SLinComp}_2$-preserving. Then the output of every $\varphi$ on every union of $k$ triangles is a compact semi-linear set. From the construction above, it follows that such an output can have no more than $l$ vertices. Conversely, assume that the output of every union of $k$ or fewer triangles is a union of $l$ or fewer triangles. Since $\varphi[D]$ is the union of $\varphi[\bigcup_{(\vec{u},\vec{v},\vec{w}) \in V} \text{conv}(\{\vec{u}, \vec{v}, \vec{w}\})]$, where $V$ ranges over $k$-element sets of triples of vertices of $D$, we obtain that $\varphi[D]$ is a union of a finite number of triangles, and thus compact and semi-linear. This concludes the proof of the lemma. $\square$

## Proof of theorem 7

*Proof of (a):* Note that for each $i$, there is a FO + Poly query $\psi_i$ for each $i$ that tests if a set $D$ is a convex polytope with at most $i$ vertices: it checks that the set of vertices $V(D) = \{x \in D \mid x \notin \text{conv}(D - x)\}$ has at most $i$ elements, and that $D = \text{conv}(V(D))$. In order to check if a UCQ $\varphi$ in FO + Poly is $\mathcal{CPT}$-preserving, one applies Lemma 4 to compute the numbers $k$ and $l$, and then writes a sentence saying that for every $\leq k$-element set $V$ in $\mathbb{R}^m$, applying $\varphi$ to $\text{conv}(V)$ yields a polytope with at most $l$ vertices. Since conv and $\psi_l$ are definable, this property can be expressed as a FO$(+, *, 0, 1, <)$ sentence and thus it is decidable if it is true. Hence, the property of being $\mathcal{CPT}$-preserving is decidable.

*Proof of (b):* As in the proof of (a), we notice that the condition of Lemma 5 can be written as a first-order $L(+, *, 0, 1, <)$ sentence equivalent to:

$$\forall \{\vec{x}_i^j\}_{i=1,\ldots,k}^{j=1,2,3} \ \exists \{\vec{y}_s^p\}_{s=1,\ldots,l}^{p=1,2,3} \ \forall \vec{x}. \ (\vec{x} \in \bigcup_{s=1}^{l} \text{conv}(\{\vec{y}_s^p \mid p = 1, 2, 3\})) \leftrightarrow \varphi'(\vec{x})$$

where $\varphi'$ is obtained from $\varphi$ by replacing each occurrence of $S(u,v)$ with a formula expressing the fact that $(u,v) \in \bigcup_{i=1}^{k} \mathrm{conv}(\{\vec{x}_i^j \mid j = 1,2,3\})$. Since the convex hull of a finite number of points is $\mathrm{FO}+\mathrm{POLY}$-definable, the condition of the lemma is indeed definable by a $L(+,*,0,1,<)$ sentence, and thus its validity is decidable. Hence, it is decidable if a union of conjunctive $\mathrm{FO}+\mathrm{POLY}$ queries is $\mathsf{SLinComp}_2$-preserving. $\square$

## Proof of proposition 10

Suppose such a query exists; now consider a new query that does the following. Its input is one binary relation containing a set $X$ of points $\vec{x}_1, \ldots, \vec{x}_n$ on the real plane, and one unary relation containing a single real number $\epsilon > 0$. First, in $\mathrm{FO}+\mathrm{POLY}$, construct $\mathrm{conv}(X)$, and then find vertices of a triangulation with mesh $< \epsilon$. This is clearly a safe query, so by the dichotomy theorem, there exists a polynomial $p$ such that the number of vertices of the triangulation is at most $m = p(n+1)$ ($n+1$ is the size of the input). Let $d$ be the maximal distance between the points $\vec{x}_i, \vec{x}_j$ (and thus the diameter of $\mathrm{conv}(X)$). Since the segment $[\vec{x}_i, \vec{x}_j]$ with $d(\vec{x}_i, \vec{x}_j) = d$ must be covered by the simplices of the triangulation, it is possible to find a number $\epsilon$ such that it cannot be covered by fewer than $m+1$ triangles of diameter $\epsilon$, and hence the number of points in the triangulation is $> m$. This contradiction proves the proposition. $\square$