Exact and Approximate Aggregation in Constraint Query Languages

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

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

Abstract

We investigate the problem of how to extend constraint query languages with aggregate operators. We deal with standard relational aggregation, and also with aggregates specific to spatial data, such as volume. We study several approaches, including the addition of a new class of *approximate aggregate operators* which allow an error tolerance in the computation. We show how techniques based on VC-dimension can be used to give languages with approximation operators, but also show that these languages have a number of shortcomings. We then give a set of results showing that it is impossible to get constraint-based languages that admit definable aggregation operators, both for exact operators and for approximate ones. These results are quite robust, in that they show that closure under aggregation is problematic even when the class of functions permitted in constraints is expanded.

This motivates a different approach to the aggregation problem. We introduce a language FO + POLY + SUM, which permits standard discrete aggregation operators to be applied to the outputs of range-restricted constraint queries. We show that this language has a number of attractive closure and expressivity properties, and that it can compute volumes of linear-constraint databases. We also show, using techniques from machine learning, that a small extension of FO + POLY + SUM can probabilistically find approximations of volumes for polynomial-constraint databases.

1 Introduction

New applications of database technology, such as Geographical Information Systems, have spurred a considerable amount of research into generalizations of the standard relational model to deal with the manipulation of geometric or spatial data. One common approach to modeling spatial databases is to consider input databases as given by a set of well-behaved relations in euclidean space – for example, by a set of semi-linear or semi-algebraic sets. There are a number of proposed query languages that extend classical relational algebra to this setting, languages that allow the use of various geometric operations in manipulating spatial databases. One of the most well-developed models for spatial queries is the *constraint database model* [23]. In this model, spatial databases are represented as sets of linear or polynomial constraints. Databases are queried using standard relational calculus with linear (resp. polynomial) inequalities as selection criteria, see [4, 5, 6, 19, 20, 32, 38]. These languages, denoted by FO + LIN and FO + POLY, have become the dominant ones in the constraint database literature. They have a very important *closure property*: the application of a FO + LIN query to a linear constraint set yields a new set of linear constraints; similarly FO + POLY queries on sets definable with polynomial constraints produce sets that can still be defined with polynomial constraints.

Constraint Query Languages, then, give a natural analog of relational calculus in the geometric context. A crucial question, though, concerns how to extend standard aggregation constructs from the relational model to the geometric setting. This question has two components. First, we would like our languages to be able to apply standard SQL operators such as TOTAL and AVG to spatial queries, whenever these operators make sense. Since the output of queries in constraint query languages (and in other spatial query languages) may be merely finitely representable (that is, representable by some finite means, e.g., a finite set of constraints) and not finite, the aggregation operators cannot be allowed to be applied to any constraint query output. One problem then, is to design a language that allows the safe application of classical aggregates.

The second component of the 'aggregation question' concerns aggregation notions that are specific to the spatial databases. Most commonly, given a database and the output of a query over it, one wishes to form new queries about the *volume* of this output. One may also extend standard aggregates such as AVG, and ask for the *average* value of a polynomial over a spatial object. Such aggregates arise both from practical concerns of GIS, and also as the natural continuous analogs of classical aggregation queries. Thus, we would like to extend constraint query languages to incorporate the ability to calculate volumes and other aggregates arising in the spatial setting.

In attempting to add aggregation to constraint query languages, one immediately encounters some daunting obstacles. While standard constraint databases are closed under first-order operations such as join and projection, they are clearly *not* closed under taking of volumes. This fact is well-known in the literature [23, 27, 12], and stems from the fact that neither the semi-linear nor semi-algebraic sets are closed under integrals. To take an example from the semi-algebraic setting, a query asking for the volume of initial slices of the epigraph of $1/x$ outputs the graph of the ln function, while iterating volume queries in this fashion would give as output transcendental functions that are not even expressible using field operations, logarithms and exponents. Thus, one cannot hope to add a general volume operator to existing first-order constraint query languages such as $\mathrm{FO} + \mathrm{POLY}$ and get a *closed* language while still remaining within the domain of polynomial constraint databases.

There are several approaches to the volume problem mentioned above. First, one could weaken the requirement that volumes be computed *exactly* and instead aim only to compute *approximate volumes*. Thus a query might have a tolerance associated with each instance of a volume operator, with output required only to be correct within the given tolerance. There are a number of practical and theoretical motivations for this approach. While it is known that computing volumes of even simple geometric objects (convex polytopes) is a hard problem (#P-hard, see [14]), approximation of volumes, at least of convex sets, can be done in polynomial time by a randomized algorithm [15]. Moreover, in contrast to the well-known fact that semi-algebraic and semi-linear sets are not closed under volume operators, the papers [24, 25, 26] show that volumes of sets definable with polynomial constraints can be approximated, for any given $\epsilon > 0$, by a first-order formula with polynomial constraints. By giving up the exact volumes and settling for an approximation, one might hope to retain desirable closure properties.

A second approach to the aggregation problem would be to expand out of the domain of polynomial constraints, and add new functions to the signature of both the constraints and the query language. This would give the possibility of retaining a constraint-based representation of databases, while gaining closure under volume operators. Of course, in this approach one should expand the constraint set so that it still defines only topologically well-behaved objects.

A third approach to the volume problem is to search for languages which can compute or approximate the volumes of important classes of sets, but which may not be closed under iterative application of volume operators. For example, one could allow volume and other aggregation operators to be applied only to a subclass of the input queries. Restrictions on the nesting of volume operators would then have to be imposed.

An example of this last approach in the existing literature is [11], where it is shown that polynomial constraint query languages can express the (exact) volume for any set that admits a special condition called 'variable independence'. This condition means, informally, that in the constraint specification of sets in, say, $\mathbb{R}^2$, there is no interaction between x and y. Unfortunately, this condition is too restrictive: it excludes many of the sets that arise most often in spatial applications.

In this paper, we analyze the feasibility of each of the above approaches in detail. For the first approach, we show that techniques based on VC dimension, coming out of the work of [24, 25, 26] give us approximate volume operators that give semi-algebraic output on semi-algebraic input. However, we show a number of shortcomings of such an approach. Not only are the approximate volume operators obtained according to the technique of [24, 25, 26] sensitive to the input representation, but the blow-up in the size of the constraint databases produced in query evaluation precludes any possible use of these operators in practice.

Turning to the second approach, we show that it is completely infeasible. No first-order constraint language based on any reasonably well-behaved class of functions can express, or even approximate, volume. In the process of showing this, we develop a new set of techniques for proving inexpressibility results, techniques not based on the usual method of reduction to generic queries.

We then consider solutions that give up full closure under volume, and give a number of positive results. We present a higher-order language that allows one to calculate the volume of arbitrary semi-linear sets. Specifically, we give a language, called FO+POLY+SUM, that has attractive closure properties, remains within the domain of polynomial constraint databases, and allows the exact calculation of volumes for linear-constraint input databases. This language also has the pleasant feature that it is closed under the classical aggregation operators SUM and AVG. Since FO+POLY+SUM includes SQL aggregation, contains FO + POLY, and also allows one to make use of standard aggregation evaluation techniques in calculating volumes, it seems to be a good candidate for the constraint analog of classical aggregation languages.

We remark that another approach to the aggregation problem was considered in [12], which gave a new aggregate operator μ, under which FO + LIN is closed. However, $\mu(X) = 0$ for any bounded set X; thus, this operator cannot be used to deal with volumes.

Organization Section 2 introduces the notation. Approximability is studied in Section 3. The method of defining approximate volumes of [24, 25, 26] is analyzed, and the main difficulties in applying the approximation operators coming from this work are outlined. Section 4 shows that approximate volume operators cannot be defined in first-order constraint languages, even when the signature is expanded. Section 5 defines an extension of FO + POLY with SQL-like aggregation (summation over finite sets) and shows that this extension can express volumes of semi-linear databases. All proofs can be found in the full version [8].

2 Notation

Structures, instances, queries Most notations are fairly standard in the literature on constraint databases, cf. [5, 6, 32, 19]. Let $\mathcal{M} = \langle \mathcal{U}, \Omega \rangle$ be an infinite structure, where $\mathcal{U}$ is an infinite set, called a universe (in the database literature often called the domain), and Ω is a set of interpreted functions, constants, and predicates. In the field of constraint databases, most examples have $\mathcal{U} = \mathbb{R}$, the set of real numbers. Examples of signatures (and corresponding classes of constraints) that have been considered are:

Dense Order Constraints:	$\langle \mathbb{R}, < \rangle$;
Linear Constraints:	$\mathbf{R}_{\text{lin}} = \langle \mathbb{R}, +, -, 0, 1, < \rangle$;
Polynomial Constraints:	$\mathbf{R} = \langle \mathbb{R}, +, *, 0, 1, < \rangle$;
Exponential Constraints:	$\mathbf{R}_{\exp} = \langle \mathbb{R}, +, *, e^x, < \rangle$.

A (relational) *database schema* SC is a nonempty collection of relation names $\{S_1, \ldots, S_l\}$ with associated arities $p_1, \ldots, p_l > 0$. We shall consider both finite and finitely representable instances. Given $\mathcal{M}$, an *finite 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 a finite instance D, $\mathit{adom}(D)$ denotes its *active domain*, that is, the set of all elements that occur in the relations in D.

A *finitely-representable (f.r.) instance* of SC over $\mathcal{M}$ is a family of sets $\{X_1, \ldots, X_l\}$, with $X_i \subset \mathcal{U}^{p_i}$, such that for each X_i there exists a quantifier-free formula $\alpha_i(x_1, \ldots, x_{p_i})$ in the language of $\mathcal{M}$ with $X_i = \{\vec{a} \in \mathcal{U}^{p_i} \mid \mathcal{M} \models \alpha_i(\vec{a})\}$. Most applications of constraint databases consider f.r. instances defined over $\mathbf{R}_{\text{lin}}$ (these are called *semi-linear sets*) or over $\mathbf{R}$ (called *semi-algebraic sets*). For example, in the spatial setting, a f.r. instance interprets the schema predicates as semi-linear or semi-algebraic sets.

As our basic query language, we consider relational calculus, or *first-order logic*, FO, over the underlying structure and the database schema. In what follows, FO(SC, Ω) is the set of all first-order formulae in the language that contains all symbols of SC and Ω. That is, FO(SC, Ω) formulae are built up from the atomic SC and Ω formulae by using Boolean connectives $\vee, \wedge, \neg$ and quantifiers $\forall, \exists$.

Regardless of whether we are in the 'classical' setting, where queries are applied to finite databases, or in the constraint query setting, we will refer to the above syntactic query languages as *relational calculus with* Ω *constraints*. This will be denoted by FO + Ω. When Ω is $(+, -, 0, 1, <)$, or $(+, *, 0, 1, <)$, or $(+, *, e^x, <)$, we use the standard abbreviations FO + LIN, FO + POLY and FO + EXP.

In the case of finite databases, we shall also use the *active-domain quantifiers*: for a formula $\varphi(x, \vec{y})$, one can form formulae $\exists x \in \mathit{adom}.\varphi(x, \vec{y})$ and $\forall x \in \mathit{adom}.\varphi(x, \vec{y})$. 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 FO(SC, Ω) formulae, where $(\mathcal{M}, D) \models \exists x\ \varphi(x, \cdot)$ means that for some $a \in \mathcal{U}$ we have $(\mathcal{M}, D) \models \varphi(a, \cdot)$, and $(\mathcal{M}, D) \models \exists x \in \mathit{adom}\ \varphi(x, \cdot)$ means that for some $a \in \mathit{adom}(D)$ we have $(\mathcal{M}, D) \models \varphi(a, \cdot)$. If $\mathcal{M}$ is understood, we write $D \models \varphi$.

Given $\varphi(\vec{x}, \vec{y})$ and $\vec{a}$, we write $\varphi(\vec{a}, D)$ for $\{\vec{b} \mid D \models \varphi(\vec{a}, \vec{b})\}$; in the absence of $\vec{x}$ we just write $\varphi(D)$ for the output of φ on D.

The class of subformulae of FO that only use the active-domain quantification is denoted by FO_{act}. Over $\mathbf{R}$ and $\mathbf{R}_{\text{lin}}$, one does not lose expressiveness over finite instances by going from FO to FO_{act}, see [6, 32].

Adding aggregate operators We shall use $\textsc{Vol}(X)$ to denote the volume of a set $X \subseteq \mathbb{R}^n$. More precisely, $\textsc{Vol}(X)$ is the measure of any Lebesgue-measurable set $X \subseteq \mathbb{R}^n$. We shall not worry about dealing with non-measurable sets, as all bounded sets defined with constraints relevant for spatial applications (those listed above, plus some extensions) are measurable.

We shall consider adding volume to a query language as follows. If $\varphi(\vec{x}, \vec{y})$ is a formula, then the following is a formula with free variables $\vec{x}, z$:

$$[\textsc{Vol}\ \vec{y}.\varphi(\vec{x}, \vec{y})](\vec{x}, z)$$

Assume that a structure $\mathcal{M} = \langle \mathbb{R}, \Omega \rangle$ is fixed. Let an instance (finite or f.r.) D be given. Then

$$D \models [\textsc{Vol}\ \vec{y}.\varphi(\vec{x}, \vec{y})](\vec{a}, v) \ \text{ iff } \ v \ = \ \textsc{Vol}(\varphi(\vec{a}, D))$$

Recall that $\varphi(\vec{a}, D) = \{\vec{b} \mid D \models \varphi(\vec{a}, \vec{b})\}$.

The extension of any query language $\mathcal{L}$ with $\textsc{Vol}$ will be denoted by $\mathcal{L} + \textsc{Vol}$; for example, one can speak of $\text{FO} + \textsc{Lin} + \textsc{Vol}$ or $\text{FO} + \textsc{Poly} + \textsc{Vol}$. Of course we know that due to the nonclosure results mentioned in the introduction, $\text{FO} + \textsc{Lin} \subsetneq \text{FO} + \textsc{Lin} + \textsc{Vol}$ and $\text{FO} + \textsc{Poly} \subsetneq \text{FO} + \textsc{Poly} + \textsc{Vol}$.

As the next step, we restrict our attention to bounded sets. Without any loss of generality, we shall deal with subsets of $I^n \subset \mathbb{R}^n$, where I throughout this paper denotes $[0,1]$. We define $\textsc{Vol}_I\ \vec{y}.\varphi(\vec{x}, \vec{y})$ just as above, except that now we require that $v = \textsc{Vol}(\varphi(\vec{a}, D) \cap I^n)$. In particular, $0 \le v \le 1$. We similarly define languages $\mathcal{L} + \textsc{Vol}_I$. As with $\textsc{Vol}$, languages like $\text{FO} + \textsc{Lin}$ and $\text{FO} + \textsc{Poly}$ are not closed under $\textsc{Vol}_I$: for example, $\arctan(x) = \int_0^x \frac{dy}{y^2+1} = \textsc{Vol}_I(\{(y,z) \mid (0 \le y \le x) \wedge (0 \le z \le 1/(y^2+1))\})$, for $0 \le x \le 1$.

As standard languages are not closed under taking volume, we address the question of whether one can obtain closure by lowering one's demands. In particular, we would like to see if *approximating* the volume, rather than computing it directly, gives us a closed language. The hope that closure might be obtained in this way is motivated by the fact that for every formula $\varphi(\vec{x}, \vec{y})$ in $\mathbf{R}$ and for every $\epsilon > 0$, one can find a formula $\psi_\epsilon(\vec{x}, z)$ that gives ϵ-approximation of volumes of sets $\varphi(\vec{a}, \mathbf{R}) = \textsc{Vol}_I(\{\vec{b} \mid \ \mathbf{R} \models \varphi(\vec{a}, \vec{b})\})$, see [24, 25, 26].

We have to explain what we mean by approximating volume in this context. Clearly, we cannot hope to find $\psi_\epsilon(\vec{x}, z)$ with z defining an ϵ-interval around the real value of the volume – then the volume itself would be definable as the center of the interval! Thus, we settle for less. Similar to [24, 25, 26], we say for every $\epsilon > 0$, that an operator $\textsc{Vol}^\epsilon$ is an *ϵ-approximation operator* if for every f.r., over $\mathcal{M}$, set $A \subseteq \mathbb{R}^n \times \mathbb{R}^m$, given by a formula $\varphi(\vec{x}, \vec{y})$, $\textsc{Vol}^\epsilon$ returns a f.r. set in $\mathbb{R}^n \times \mathbb{R}$, given by $\psi_\epsilon(\vec{x}, z)$ such that :

1. For every $\vec{a} \in \mathbb{R}^n$, $\psi_\epsilon(\vec{a}, \cdot)$ must be satisfiable (that is, $\mathcal{M} \models \exists z.\psi_\epsilon(\vec{a}, z)$);
2. If $\mathcal{M} \models \psi_\epsilon(\vec{a}, v)$, then $|v - \textsc{Vol}(\varphi(\vec{a}, \mathbb{R}))| < \epsilon$.

Thus, $\textsc{Vol}^\epsilon$ must return a ψ_ϵ that is guaranteed to find an (absolute) ϵ-approximation of the volume. We next say that a query language $\mathcal{L}$ *defines* $\textsc{Vol}^\epsilon$, if there is a query in $\mathcal{L}$ that defines such an operator. That is, for each query $\varphi(\vec{x}, \vec{y})$ in $\mathcal{L}$ and $\epsilon > 0$ there is a $\mathcal{L}$-query $\psi_\epsilon(\vec{x}, z)$ such that for any database D, and any $\vec{a}$, we have

1) $D \models \exists z.\psi_\epsilon(\vec{x}, z)$, and
2) $D \models \psi_\epsilon(\vec{a}, v)$ implies $|v - \textsc{Vol}(\varphi(\vec{a}, D))| < \epsilon$.

Notice that in the last definition ψ_ϵ is independent of D. We also define ϵ-approximation operators to volume in the case where we restrict to bounded sets. As before, we use, w.l.o.g., I^n as bounding set. An ϵ-approximation operator in the bounded setting is denoted by $\textsc{Vol}_I^\epsilon$. Such an operator must satisfy the variant of condition 2) above: $|v - \textsc{Vol}(\varphi(\vec{a}, D) \cap I^n)| < \epsilon$.

These operators, and their definability in query languages, are studied in Sections 3 and 4.

O-minimality, VC dimension Many results that we prove extend beyond linear and polynomial constraints. To state them in greater generality, we shall use *o-minimality* [37], which plays an important role in the study of constraint query languages (cf. [5, 6, 7]).

A structure $\mathcal{M} = \langle \mathcal{U}, \Omega \rangle$ is o-minimal, if every definable set is a finite union of points and open intervals $(a,b) = \{x \mid a < x < b\}$, $(-\infty, a) = \{x \mid x < a\}$, and $(a, \infty) = \{x \mid x > a\}$ (we assume that $<$ is in Ω). Definable sets are those of the form $\{x \mid \mathcal{M} \models \varphi(x)\}$, where φ is a first-order formula in the language of $\mathcal{M}$, possibly supplemented with symbols for constants from $\mathcal{M}$. All the structures on the reals we mentioned so far – $\mathbf{R}_{\text{lin}}$, $\mathbf{R}$, $\mathbf{R}_{\text{exp}}$ – are o-minimal (the first two by Tarski's quantifier-elimination, the last one by [39]).

If $\mathcal{M} = \langle \mathbb{R}, \Omega \rangle$, we define $\mathcal{M}_{+,*}$ to be $\langle \mathbb{R}, \Omega, +, * \rangle$. We often require that not just $\mathcal{M}$ but also $\mathcal{M}_{+,*}$ be o-minimal. Note that this requirement is satisfied by $\mathbf{R}_{\text{lin}}, \mathbf{R}$ and $\mathbf{R}_{\text{exp}}$.

We also consider structures having finite VC dimension of definable families [3, 28] (also known as structures without the independence property [36]). VC dimension, introduced in statistics to study uniform convergence of stochastic processes, has become central to computational learning theory [3, 10], and found application in other areas, e.g., complexity [31].

Suppose X is an infinite set, and $\mathcal{C} \subseteq 2^X$. Let $F \subset X$ be finite; we say that $\mathcal{C}$ *shatters* F if the collection $\{F \cap C \mid C \in \mathcal{C}\}$ is 2^F. The *Vapnik-Chervonenkis (VC) dimension* of $\mathcal{C}$, $\mathrm{VCdim}(\mathcal{C})$, is the maximal cardinality of a finite set shattered by $\mathcal{C}$. If arbitrarily large finite sets are shattered by $\mathcal{C}$, we let $\mathrm{VCdim}(\mathcal{C}) = \infty$.

Let $\mathcal{M} = \langle \mathcal{U}, \Omega \rangle$, and let $\varphi(\vec{x}, \vec{y})$ be a formula in the language of $\mathcal{M}$ with $|\vec{x}| = n, |\vec{y}| = m$. For each $\vec{a} \in \mathcal{U}^n$, define $\varphi(\vec{a}, \mathcal{M}) = \{\vec{b} \in \mathcal{U}^m \mid \mathcal{M} \models \varphi(\vec{a}, \vec{b})\}$, and let $F_\varphi(\mathcal{M})$ be $\{\varphi(\vec{a}, \mathcal{M}) \mid \vec{a} \in \mathcal{U}^n\}$. Families of sets arising in such a way are called *definable families*. We say that $\mathcal{M}$ is a *structure with finite VC dimension* if the VC dimension of each definable family is finite. Every o-minimal structure is a structure with finite VC dimension [28], and the latter class is in fact much larger than the class of o-minimal structures.

3 Approximating aggregates in constraint query languages

The VC dimension-based implementation of approximate volume operators We now start our investigation of the expressibility of approximate volume operators. The results of [24, 25, 26] do immediately give a closed language for computing approximate volumes. We then examine those operators and show that they can blow up the size of the database enormously.

Lemma 1 *Let $\epsilon > 0$, and let $\varphi(\vec{x}, \vec{y})$ be a* $\mathrm{FO} + \mathrm{POLY}$ *query. Then for every semi-algebraic database instance D there exists a formula $\varphi_D^\epsilon(\vec{x}, z)$ such that $\varphi_D^\epsilon(\vec{a}, \cdot)$ is satisfiable for all $\vec{a}$, and $\models \varphi_D^\epsilon(\vec{a}, v)$ implies $| v - \mathrm{VOL}_I(\varphi(\vec{a}, D)) | < \epsilon$. Hence, there is a collection of ϵ-approximation operators VOL_I^ϵ, $\epsilon > 0$, for $\mathbf{R}$.*

Proof. Replace each occurrence of a *SC* predicate by its definition (a FO formula over $\mathbf{R}$) and apply the approximating formulae of [24, 25, 26] (see below). □

The addition of the operators VOL_I^ϵ, $\epsilon > 0$, to $\mathrm{FO} + \mathrm{POLY}$ allows the calculation of approximate volumes, while retaining the property of FO+POLY that the output of a query is representable as a constraint database.

We give a rough sketch of the formulae approximating the volume that are constructed in [24, 25, 26]. Assume that we are given a first-order formula $\varphi(\vec{x}, \vec{y})$ over the real field $\mathbf{R}$, with $|\vec{x}| = n$ and $|\vec{y}| = m$, and $\epsilon > 0$. We want to find a formula $\psi(\vec{x}, z)$ approximating $\mathrm{VOL}_I(\varphi(\vec{x}, \mathbf{R})) = \mathrm{VOL}(\{\vec{y} \mid \varphi(\vec{x}, \vec{y})\} \cap I^m)$. [1]

The construction of ψ is based on two key observations. First, one can find a small random sample that gives a good volume approximation uniformly for all $\vec{x}$. Second, the sampling procedure can be derandomized.

The first observation follows from the finiteness of VC dimension of definable families. As we noted before, o-minimal structures, and the real field in particular, are structures with finite VC dimension. That is, each definable family $F_\varphi(\mathbf{R}) = \{\varphi(\vec{a}, \mathbf{R}) \mid \vec{a} \in \mathbb{R}^n\} \subseteq 2^{\mathbb{R}^m}$ has a finite VC dimension d (we assume from now on that all sets are restricted to $[0,1]$). Then, the classical result relating VC dimension and learnability [3, 10] states the following. Fix $\epsilon, \delta > 0$, and let $M > \max(\frac{4}{\epsilon} \log \frac{2}{\delta}, \frac{8d}{\epsilon} \log \frac{13}{\epsilon})$. Assume that an M-point sample $X = \{\vec{x}_1, \ldots, \vec{x}_M\}$ is randomly chosen in I^m. For each $\vec{a}$, let $v(\vec{a}, X)$ be the fraction of X that falls into $\varphi(\vec{a}, \mathbf{R})$. Then $| v(\vec{a}, X) - \mathrm{VOL}_I(\varphi(\vec{a}, \mathbf{R})) | < \epsilon$ for all $\vec{a} \in \mathbb{R}^n$, with probability at least $1 - \delta$.

Since the VC dimension of $F_\varphi(\mathbf{R})$ is a finite number d, one can write a formula that takes as input an M-sample and calculates the number of elements in the sample that fall into sets $\varphi(\vec{a}, \mathbf{R})$. Note that the M given in the paragraph above depends only on d, ϵ and δ, not $\vec{a}$. Hence with probability $> 1 - \delta$ a sample X plugged into the formula gives us a good approximation to the volume over $\vec{a}$ for every $\vec{a}$. The construction in [24, 25, 26] then derandomizes this sampling, along the lines of the classical proof of BPP $\subseteq$ PH. Namely, one gets a formula $\gamma(\vec{x}, v)$ that determines whether the set $\{X \mid X$ is an M-sample from I^m whose portion falling into $\varphi(\vec{x}, \mathbf{R})$ is within $\epsilon/2$ from $v\}$ has a certain number of translates covering the entire unit cube in the appropriate dimension. Then [24, 25, 26] prove that a v for which these translates cover the cube must be an ϵ-approximation to the volume of $\varphi(\vec{x}, \mathbf{R})$.

Shortcomings of the approximation technique We note here some shortcomings of the technique of Lemma 1 in the context of constraint databases. In the technique, one has to put the definition of a constraint database D into a query φ, and then apply the method of [24, 25, 26] to the result. That method produces an output formula whose size is a polynomial in the input formula and $\frac{1}{\epsilon}$: theoretically, a nice bound. In attempting to apply this technique in practice, however, we find that the bounds obtained are rather unpleasant, even for modest values of ϵ, as the size of the quantifier prefix is quite large. In the constraint database setting, those will have to be eliminated, via a quantifier-elimination

[1] The notion of approximation in [24, 25, 26] is slightly more specific: it requires that $\models \psi(\vec{a}, v)$ imply $| v - \mathrm{VOL}_I(\varphi(\vec{a}, \mathbf{R})) | < \epsilon$, and $| v - \mathrm{VOL}_I(\varphi(\vec{a}, \mathbf{R})) | < \epsilon/4$ imply $\models \psi(\vec{a}, v)$.

procedure, which will be very costly. Let us illustrate this by a simple example.

Example: Let the schema contain one unary predicate U interpreted as a subset of $[0,1]$. The query $\varphi(x_1, x_2; y_1, y_2)$ is given by

$$U(x_1) \wedge U(x_2) \wedge x_1 < y_1 \wedge y_1 < x_2 \wedge 0 \leq y_2 \wedge y_2 \leq y_1$$

Let $\epsilon = 1/10$. We want to evaluate the query $(\mathrm{VOL}_I^\epsilon \vec{y}.\varphi(x_1, x_2; y_1, y_2))(x_1, x_2, z)$ saying that z is an ϵ-approximation to the volume of $\varphi(x_1, x_2, U) = \{(y_1, y_2) \mid U \models \varphi(x_1, x_2; y_1, y_2)\}$, where VOL_I^ϵ is the operator obtained through the method above. Note that $\mathrm{VOL}_I(\varphi(a, b, U)) = (b^2 - a^2)/2$. To evaluate this query on a database where U consists of n elements, by applying Lemma 1, we would first plug U in φ to obtain a formula with $> 2n$ atomic subformulae that does not mention U. Using the bounds from [25], we obtain a formula for ϵ-approximation of the volume that has at least 10^9 atomic subformulae, and at least 10^{11} quantifiers. Thus, applying the method of [24, 25, 26] 'as is' appears to be infeasible in the context of constraint databases.

The technique of Lemma 1 also tells us nothing about the definability of the operators VOL_I^ϵ, nor the power of the language that results from adding them to a standard language, like $\mathrm{FO}+\mathrm{POLY}$, since the approximating formula φ_D^ϵ varies with the input database.

4 Uniformly definable volume operators and expansion of the signature

We saw in the last section that the main shortcoming of all known examples of approximate volume operators was the blow-up in the size of the representation. It was also left open whether some volume approximation operators can be defined in standard languages, like $\mathrm{FO}+\mathrm{POLY}$, uniformly for all database instances. We now investigate whether we can find other approximation methods that can be expressed in nicely-behaved languages and that admit low complexity evaluation techniques. The main result is that one *cannot* capture approximate volume operators in a nice constraint language such as $\mathrm{FO}+\mathrm{POLY}$. That is,

Inexpressibility of Approximate Operators $\mathrm{FO}+\mathrm{LIN}$, $\mathrm{FO}+\mathrm{POLY}$ *and* $\mathrm{FO}+\mathrm{EXP}$ *cannot express* VOL_I^ϵ *for any* $\epsilon < 1/2$. □

In fact, we prove a stronger result. Theorem 2 shows that even if one extends the constraint signature to include functions beyond $\mathrm{FO}+\mathrm{EXP}$, as long as we stay within a well-behaved structure, we cannot capture approximate volume. Furthermore, we show that in languages like $\mathrm{FO}+\mathrm{POLY}$, only *trivial* approximations are possible. An example of a trivial approximation is returning $1/2$ for every subset of I^n – in this case we know that the difference between the real volume and its approximation is $\leq 1/2$.

Proving expressivity bounds such as Theorem 2 and Corollary 1 is not very simple. Almost all, if not all, existing expressivity bounds for constraint query languages either involve generic queries (e.g., the parity test, see [5, 6, 32, 4]) or are proved by reduction to generic queries (e.g., [20]). However, queries involving approximation defined as in Section 2 are extremely nongeneric. We introduce the main ideas for the proof in several steps. We first consider an easier case of the aggregate AVG for finite instances and prove that it can be neither defined nor approximated in languages like $\mathrm{FO}+\mathrm{POLY}$. The proof introduces the idea of reduction to what we call a (c_1, c_2)-separating sentence, with c_1, c_2 being constant real numbers. We then show how the same reduction easily proves that $\mathrm{FO}+\mathrm{POLY}$ and the likes cannot produce *relative* approximations of VOL. For the absolute approximation VOL_I^ϵ, the reduction only works under very special assumptions on the input, and to conclude the proof we need to use results from circuit complexity.

This section gives further evidence that if one wants to stay within a reasonable (for spatial applications) class of constraints, one must give up uniform closure under any nontrivial approximation to the volume.

Separating sentences We shall consider a relational database schema SC that consists of two unary relations, U_1 and U_2. Let $c_1, c_2 > 1$ be two real numbers. We say that Φ is a (c_1, c_2)*-separating sentence* if for any finite instance D of SC, it is the case that $card(U_1) > c_1 \cdot card(U_2)$ implies $D \models \Phi$ and $card(U_2) > c_2 \cdot card(U_1)$ implies $D \models \neg\Phi$. Note that this definition says nothing about the case when $\frac{1}{c_2} \cdot card(U_2) \leq card(U_1) \leq c_1 \cdot card(U_2)$, and thus direct application of bounds on expressiveness of generic queries is impossible. Still, we can show:

Proposition 1 *Let* $\mathcal{M} = \langle \mathcal{U}, \Omega \rangle$ *be o-minimal,* $c_1, c_2 > 1$*, and* SC *as above. Then no* (c_1, c_2)*-separating sentence is definable in* $\mathrm{FO}(SC, \Omega)$.

Proof sketch. Assume such a sentence Φ is definable. By [6], there exists a definable extension $\mathcal{M}'$ of $\mathcal{M}$ such that over $\mathcal{M}'$, Φ is equivalent to an active semantics sentence Ψ. By [5], there is an infinite subset of $\mathcal{U}$ over which Ψ is definable in the language of U_1, U_2 and $<$. Then one uses Ehrenfeucht-Fraïssé games to show that this is impossible. □

4.1 Dealing with AVG

We assume that instances store elements of a numerical domain, for example $\mathbb{R}$. Given a query $\varphi(\vec{x}, z)$, we define $\text{AVG}_\varphi(\vec{x}, y)$ by letting $D \models \text{AVG}_\varphi(\vec{a}, v)$ iff $\mathit{card}(\varphi(\vec{a}, D)) < \infty$ and $v = \text{AVG}(\varphi(\vec{a}, D))$, where $\text{AVG}(C) = (\sum_{c \in C} c)/\mathit{card}(C)$. Note that the aggregate AVG is typically defined using the bag semantics; however, as we show *in*expressibility results, dealing with this simplified version will suffice. [2]

It is easily shown (by reduction to equal cardinality) that AVG_φ is not definable in FO + POLY, even if $D \models \varphi(\vec{a}, c)$ implies $0 \leq c \leq 1$. We now define ϵ-approximation of AVG just as we did it for VOL. Assume a query $\varphi(\vec{x}, z)$ is given. An operator AVG_I^ϵ, when applied to φ, produces a query $\psi_\epsilon(\vec{x}, z)$ such that, for any instance D and any $\vec{a}$, $D \models \exists z.\varphi(\vec{a}, z)$, and if $D \models \varphi(\vec{a}, v)$, then $| v - \text{AVG}(\varphi(\vec{a}, D) \cap I) | < \epsilon$ and $0 \leq v \leq 1$. For convenience, we let $\text{AVG}(C) = 0$ for C infinite.

For $\epsilon \geq 1/2$, AVG_I^ϵ is definable in $\text{FO}(SC, \Omega)$ if the input is finite or f.r. over Ω, as long as the constants $0, 1/2$ and 1 are definable. However,

Theorem 1 *Let $M = \langle \mathbb{R}, \Omega \rangle$, and let $\mathcal{M}_{+,*}$ be o-minimal. Let $\epsilon < 1/2$. Then AVG_I^ϵ is not definable in* FO + Ω*, even over finite instances. In particular, AVG_I^ϵ is not definable in* FO + POLY.

Proof sketch. Given $0 < \epsilon < 1/2$, it is possible to find a number $0 < \Delta < 1/2$ and two FO + POLY queries that translate two finite sets U_1 and U_2 into intervals $(0, \Delta)$ and $(1 - \Delta, 1)$ respectively, in such a way that for the results of translation, U_1' and U_2', $\text{AVG}(U_1' \cup U_2')$ can be written as a function of $\frac{\mathit{card}(U_1)}{\mathit{card}(U_2)}$. Using this, and assuming that AVG_I^ϵ is definable, one obtains a (c_1, c_2)-separating sentence in $\text{FO} + (\Omega \cup \{+, *\})$ for appropriate $c_1, c_2 > 1$ that depend on ϵ. This contradicts Proposition 1. □

4.2 Dealing with volume

We start with two easy results. First, for unbounded measures (no restriction to I^n) volume cannot be approximated in languages like FO + POLY.

Proposition 2 *Let $\mathcal{M} = \langle \mathbb{R}, \Omega \rangle$, and let $\mathcal{M}_{+,*}$ be o-minimal. Then no ϵ-approximation operator VOL^ϵ is definable in* FO + Ω. □

[2] We shall come back to the multiset semantics later.

The proof is by reduction to equal cardinality, for sparse finite sets. It relies on the fact that there is no a priori bound on the outputs of queries. Thus, a different approach is needed to show inexpressibility of VOL_I^ϵ.

For a query $\varphi(\vec{x}, \vec{y})$ and two constants $0 < c_1 < c_2$, we say that $\psi(\vec{x}, z)$ gives a (c_1, c_2)*-relative approximation* of the volume if for any $\vec{a}$, $\psi(\vec{a}, \cdot)$ is satisfiable, and

$$D \models \psi(\vec{a}, v) \quad \Rightarrow \quad c_1 \; < \; (v/\text{VOL}(\varphi(\vec{a}, D))) \; < \; c_2$$

An easy reduction to separating sentences shows:

Proposition 3 *Assume that $\langle \mathbb{R}, \Omega \rangle$ is such that $\langle \mathbb{R}, \Omega, +, * \rangle$ is o-minimal. Then for any $0 < c_1 < c_2$, the (c_1, c_2)-relative approximation of the volume is not definable in* FO + Ω*, for any dimension $k > 0$, even for queries restricted to $[0, 1]^k$.* □

One often is interested in an ϵ-relative approximation V_{app} or volume $V \neq 0$ defined such that $| V_{\text{app}} - V | / V < \epsilon$, for $0 \leq \epsilon < 1$. Since the existence of an ϵ-relative approximation means the existence of a $(1 - \epsilon, 1 + \epsilon)$-relative approximation to the volume in the sense defined above, we get that languages like FO + LIN, FO + POLY and FO + EXP cannot express relative approximations of the volume, even for subsets of $[0, 1]$.

4.3 Absolute approximation

We shall now prove the strongest of the inexpressibility results: that VOL_I^ϵ, for $\epsilon < 1/2$, cannot be defined in languages like FO + LIN and FO + POLY. First note:

Proposition 4 FO + LIN *defines VOL_I^ϵ for $\epsilon \geq 1/2$.*

Proof sketch. If the volume is not 0 or 1, then $1/2$ is the ϵ-approximation. □

This trivial approximation is the bext one can hope for in languages like FO + LIN and FO + POLY.

Theorem 2 *Let $\mathcal{M} = \langle \mathbb{R}, \Omega \rangle$, and let $\mathcal{M}_{+,*}$ be o-minimal. Assume that $\epsilon < 1/2$. Then VOL_I^ϵ is not definable in* FO + Ω.

Proof sketch. Let SC consist of two unary relations A and B. Call a finite instance good if A is an initial fragment of natural numbers, and B is a nonempty proper subset of A. Let $c_1 = (1 - 2\epsilon)/3$ and $c_2 = (2 + 2\epsilon)/3$. A sentence Φ in the language of SC and Ω is called a (c_1, c_2)-good sentence if $\mathit{card}(B) < c_1 \cdot \mathit{card}(A)$ implies $D \models \neg\Phi$ and $\mathit{card}(B) > c_2 \cdot \mathit{card}(A)$ implies $D \models \Phi$ for any good instance D. Note that this is the same as having a separating sentence for B and $A-B$; however, here we only require that the above conditions hold for a good instance. The theorem follows from two lemmas.

Lemma 2 *Assume* VOL_I^ϵ *is definable in* $\mathrm{FO}+\Omega$. *Then for* c_1, c_2 *as above there exists a signature* Ω' *extending* Ω *and a* (c_1,c_2)*-good sentence in* $\mathrm{FO}_{\mathrm{act}}(SC,\Omega')$.

Proof sketch of Lemma 2. With an FO+POLY query, we can map the active domain of a good instance into $[0,1]$ so that the distance between two consecutive elements is the same. Next, consider the union X of all intervals that start with an element of B and span to the next (in the order $<$) element of $A-B$, or to 1 if there is no such element. Let Y be obtained in the same way by changing the roles of B and $A-B$. Then, using ϵ-approximations of $\mathrm{VOL}(X)$ and $\mathrm{VOL}(Y)$, one can construct a (c_1,c_2)-good sentence in $\mathrm{FO}+\Omega$. The result now follows from the natural-active collapse [6].

Lemma 3 *Let* Θ *be an arbitrary signature on* $\mathbb{R}$. *Then* $\mathrm{FO}_{\mathrm{act}}(SC,\Theta)$ *cannot define a* (c_1,c_2)*-good sentence.*

Proof sketch of Lemma 3. Suppose Φ defines such a sentence. With each $n>0$ and each $B\subseteq\{0,\ldots,n-1\}$, associate a structure $S(B,n)$ whose universe is $\{0,\ldots,n-1\}$, one unary symbol U is interpreted as B, and the remaining signature operators correspond to atomic Θ-subformulae of Φ, which naturally inherit their interpretation from Θ. We then show that there is a sentence Ψ such that for $card(B)<c_1n$ we have $S(B,n)\models\neg\Psi$ and for $card(B)>c_2n$ we have $S(B,n)\models\Psi$. Next, using standard techniques (see, e.g., [13]) we convert Ψ into a family of non-uniform AC^0 circuits (nonuniformity comes from the interpretation of Θ-predicates). Thus, this family of circuits can distinguish cardinalities $>c_2n$ from those $<c_1n$; in particular, from large enough n, it can distinguish some cardinalities in $[\sqrt{n}, n-\sqrt{n}]$. However, AC^0 circuits are not capable of doing this, cf. [13]. This proves Lemma 3 and thus the theorem. □

Corollary 1 FO + LIN, FO + POLY *and* FO + EXP *cannot express* VOL_I^ϵ *for any* $\epsilon<1/2$. □

Theorem 2 shows that one cannot possibly adjust the method of [24, 25, 26] to get the approximation operators uniformly definable. This is somewhat surprising, for the following reasons. It is possible that there exists an o-minimal structure which is closed under taking integrals. That is, for every $\varphi(\vec{x},\vec{y})$ in the language of the structure, there is a formula $\psi(\vec{x},z)$ such that $\models\psi(\vec{a},v)$ iff $v=\int\ldots\int\chi_{\varphi(\vec{a},\mathbb{R}^n)\cap I^n}d\vec{y}=\mathrm{VOL}(\varphi(\vec{a},\mathbb{R}^n)\cap I^n)$. The existence of such a structure is conjectured in [25]. By Theorem 2, even if such a structure $\mathcal{M}=\langle\mathbb{R},\Omega\rangle$ existed, the volume of outputs of very simple queries on finite instances could not be approximated in $\mathrm{FO}+\Omega$!

Is it possible that one can express the approximate volume computation over outputs of some particularly simple queries? We now show that for two very simple classes, this remains impossible in FO + POLY and similar languages.

Corollary 2 *In languages* FO + LIN, FO + POLY, FO + EXP, *it is impossible to express* VOL_I^ϵ *even restricted to a) outputs of conjunctive* $<$*-queries over finite instances, or b) schema predicates, interpreted as f.r. instances definable with dense-order constraints.* □

Remarks One may ask where the procedure of [24, 25, 26] fails if we try to apply it, in a uniform way, to, say, FO + POLY queries. Note that the method of [24, 25, 26] produces a formula whose quantifier prefix is proportional to the VC dimension of the family of sets defined by the input formula. However, for relational calculus queries, this may depend on the size of the database, thus making it impossible to quantify uniformly over random samples. For a query $\varphi(\vec{x},\vec{y})$ with and a database D, the *definable family given by* φ *and* D is $F_\varphi(D)=\{\varphi(\vec{a},D)\mid\vec{a}\in\mathcal{U}^n\}$ where $\varphi(\vec{a},D)=\{\vec{b}\mid D\models\varphi(\vec{a},\vec{b})\}$. The size of a finite database D, $|D|$, is defined to be $card(adom(D))$.

Proposition 5 *There exists a (quantifier-free) relational calculus query* $\varphi(x,y)$, *and a sequence of databases* $D_1, D_2, \ldots$ *of increasing size such that* $\mathrm{VCdim}(F_\varphi(D_n))\geq\log|D_n|$. □

We also remark that under some special assumptions on the outputs of the queries, their volumes can be approximated. We can show, using Löwner-John ellipsoids [18], that for a FO + POLY query $\varphi(\vec{x},\vec{y})$ with $|\vec{y}|=k$, under the assumption that $\varphi(\vec{a},D)$ is convex, a relative (c_1,c_2) approximation of its volume can be found with $c_1=\frac{k^k+1}{2\cdot k^k}-\epsilon$ and $c_2=\frac{k^k+1}{2}+\epsilon$ for an arbitrarily small $\epsilon>0$.

5 FO + POLY + SUM: An aggregate language for constraint databases

We now introduce a language for extending FO + POLY with a summation operator. The main difficulty is to make sure that when summation is done over all elements in some query output, we are guaranteed that the query output is *finite*. To do this, we use techniques from [7] for guaranteeing that a query is *safe* (that is, that a query yields finite output).

Let Q be a non-boolean query over a database schema SC. We say that Q is a *semi-algebraic query* if it gives semi-algebraic output on semi-algebraic inputs. We say Q is *semi-algebraic-to-finite* and write $Q \in \mathrm{SAF}$ if Q produces finite output on semi-algebraic input databases. If Q is expressed as $\varphi(y, \vec{x})$, we say that Q is *$\vec{x}$-semi-algebraic-to-finite* if for every $\vec{a}$ the query $\varphi(y, \vec{a})$, with one free variable y, is in SAF. In the language $\mathrm{FO} + \mathrm{POLY} + \mathrm{SUM}$, *all* queries are semi-algebraic queries, but in the construction we will have to ensure that certain subqueries are in the smaller class SAF.

A first-order formula $\gamma(x, \vec{w})$ with distinguished variable x in the language of the real field is said to be *deterministic* if it produces at most one output x for every vector of real numbers $\vec{w}$. Deterministic formulae are the building blocks from which safe queries can be formed. Given a deterministic formula $\gamma(x, \vec{w})$ and a finite set of tuples of reals A (having the same length as $\vec{w}$), we let $\gamma(A)$ refer to the *bag* $\uplus_{\vec{a} \in A} f_\gamma(\vec{a})$, where f_γ is the corresponding partial function taking $\vec{w}$ to the unique x such that $\gamma(x, \vec{w})$ holds. Note that it is decidable if a formula is deterministic.

Definition of $\mathrm{FO} + \mathrm{POLY} + \mathrm{SUM}$ The query language $\mathrm{FO} + \mathrm{POLY} + \mathrm{SUM}$ is defined inductively as follows. Atomic queries are the same as for $\mathrm{FO} + \mathrm{POLY}$. The formulae of $\mathrm{FO} + \mathrm{POLY} + \mathrm{SUM}$ are closed under boolean connectives and quantification $\forall$ and $\exists$ (over the reals).

Next, we define the *summation term-former*. Given any $\mathrm{FO} + \mathrm{POLY} + \mathrm{SUM}$ formula $\varphi(y, \vec{z})$, we let $\mathrm{END}[y, \varphi(y, \vec{z})](u, \vec{z})$ be the query that holds for a tuple $(b, \vec{a})$ on an input database D iff b is an endpoint of the intervals that compose $\varphi(D, \vec{a})$. Note that if φ is a semi-algebraic query (which is guaranteed by Lemma 4 below), then $\mathrm{END}[y, \varphi(y, \vec{z})]$ is $\vec{z}$-SAF.

A *range-restricted* $\mathrm{FO} + \mathrm{POLY} + \mathrm{SUM}$ expression is an expression of the form $\rho(\vec{w}, \vec{z}) \equiv (\varphi_1(\vec{w}, \vec{z}) | \mathrm{END}[y, \varphi_2(y, \vec{z})])$ where $\varphi_1(y, \vec{z})$ and $\varphi_2(\vec{w}, \vec{z})$ are $\mathrm{FO} + \mathrm{POLY} + \mathrm{SUM}$ queries. It binds y, that is, the free variables are $\vec{z}, \vec{w}$. We have $D \models \rho(\vec{a}, \vec{b})$ for $\vec{a} = (a_1, \ldots, a_n)$ iff $D \models \varphi_1(\vec{a}, \vec{b})$ and

$$D \models (\mathrm{END}[y, \varphi_2(y, \vec{z})])(a_i, \vec{b}),\ i = 1, \ldots, n.$$

It then follows from the closure property (Lemma 4) that for any D and any $\vec{b}$, the set $\rho(D, \vec{b}) = \{\vec{a} \mid D \models \rho(\vec{a}, \vec{b})\}$ is finite.

For any deterministic formula $\gamma(x, \vec{w})$ in the language of the real field and any range-restricted expression $\rho(\vec{w}, \vec{z})$ as above we now define a term $t(\vec{z})$ by

$$[\sum_{\rho(\vec{w}, \vec{z})} \gamma](\vec{z})$$

Given D and $\vec{b}$, the value of $t(\vec{b})$ in D is the sum of all the members of the finite bag $\gamma(A)$, where $A = \rho(D, \vec{b})$.

Finally, new terms in $\mathrm{FO} + \mathrm{POLY} + \mathrm{SUM}$ can be built by applying composition with the real functions $+, *$. If t_is are terms and φ is a formula, then $t_1 = t_2, t_1 < t_2$ and $\varphi(t_1, \ldots, t_k)$ are $\mathrm{FO} + \mathrm{POLY} + \mathrm{SUM}$ formulae.

Examples of $\mathrm{FO} + \mathrm{POLY} + \mathrm{SUM}$ **queries** Let $\varphi(w)$ be an $\mathrm{FO}+\mathrm{POLY}$ query. Let $\gamma(x, w) \equiv (x = w)$ and $\rho(w) = (w = w) | \mathrm{END}[w, \varphi(w)]$. Then the $\mathrm{FO} + \mathrm{POLY} + \mathrm{SUM}$ term (without free variables) $\sum_{\rho(w)} \gamma$ gives the sum of all the endpoints of the intervals that compose $\varphi(D)$.

The area of a convex polygon in $\mathbb{R}^2$ can be defined in $\mathrm{FO} + \mathrm{POLY} + \mathrm{SUM}$. Assume that the polygon is given by a predicate $P(x, y)$ (it could be an input relation or the output of a query). There is a $\mathrm{FO} + \mathrm{POLY}$ query $\varphi_P(x, y)$ that computes all the vertices of P – this is because $\vec{a}$ is vertex iff $\vec{a} \notin \mathrm{conv}(P - \{\vec{a}\})$. Since one can compute the boundary of P by a $\mathrm{FO} + \mathrm{POLY}$ query, it follows that there is a $\mathrm{FO} + \mathrm{POLY}$ query $\nu_P(\vec{x}, \vec{y})$ that tests if $\vec{x}, \vec{y}$ are two adjacent vertices of P.

We now form two $\mathrm{FO} + \mathrm{POLY}$ queries. The query $\psi_2(u)$ tests if u is a coordinate of a vertex of P. The query $\psi_1(\vec{x}, \vec{y}, \vec{z})$ tests the following conditions: (1) $\varphi_P(\vec{x}) \wedge \varphi_P(\vec{y}) \wedge \varphi_P(\vec{z})$ holds; (2) $\vec{x}$ is a lexicographically minimal vertex of P; (3) either $\nu_P(\vec{y}, \vec{z})$ holds and $\vec{y}$ is lexicographically less than $\vec{z}$ and $\neg\nu_P(\vec{x}, \vec{y}) \wedge \neg\nu_P(\vec{x}, \vec{z})$, or $\nu_P(\vec{x}, \vec{y}) \wedge \nu_P(\vec{y}, \vec{z}) \wedge \neg\nu_P(\vec{x}, \vec{z})$.

We now let $\rho(\vec{x}, \vec{y}, \vec{z})$ be the range-restricted expression $(\psi_1(\vec{x}, \vec{y}, \vec{z}) | \mathrm{END}[u, \psi_2(u)])$. It can be easily seen that for P convex, the output of ρ is finite and produces a triangulation of P.

Since for each triangle with vertices (a_1, a_2), (b_1, b_2), (c_1, c_2), its area is computable as $(a_1 b_2 - a_2 b_1 + a_2 c_1 - a_1 c_2 + b_1 c_2 - c_2 b_1)/2$, we obtain a deterministic formula $\gamma(v, \vec{x}, \vec{y}, \vec{z})$ saying that v is the area of the triangle with vertices $\vec{x}, \vec{y}, \vec{z}$. We then conclude that the term $\sum_{\rho(\vec{x}, \vec{y}, \vec{z})} \gamma$ defines the area of P.

Note that the above method codes a standard computation of area used in computational geometry [34] which generalizes to nonconvex polygons, and is in fact used in GISs for area computation [40].

Properties of $\mathrm{FO}+\mathrm{POLY}+\mathrm{SUM}$ The language $\mathrm{FO} + \mathrm{POLY} + \mathrm{SUM}$ has a number of attractive features. It extends both $\mathrm{FO} + \mathrm{POLY}$ and the relational calculus with summation and other standard aggregates. It is also related to aggregate languages for statistical databases studied recently in [21]. Furthermore, we have the following properties.

Lemma 4 • *Every* FO + POLY + SUM *query returns semi-algebraic output on a semi-algebraic input.*

- *For any* SAF FO + POLY + SUM *query* $\varphi(\vec{z})$*, we can express in* FO + POLY + SUM *the cardinality of the output of* φ*.*
- *For any* SAF FO + POLY *query* $\varphi(\vec{z})$ *and any deterministic formula* $\chi(x, \vec{w})$ *we can express in* FO + POLY + SUM *the sum of the* x *values of* χ *for* $\vec{w}$ *ranging over the output of* φ *and the average of the* x *values of* χ *over the output of* φ*.*

The most important of these properties – closure – is obtained from the fact that there is a uniform bound on the number of intervals composing definable sets $\alpha(\vec{x}, \mathbf{R})$ for any formula $\alpha(\vec{x}, y)$ in the language of the real field.

6 Computing and approximating the volume

In this section we show how to use the aggregate language FO + POLY + SUM for volume computation and approximation. We first show it can precisely compute volumes of semi-linear sets. We then show how it can be used to uniformly approximate volumes of semi-algebraic sets.

6.1 Computing the volume of semi-linear sets in FO + POLY + SUM

Our first goal is to prove that FO + POLY + SUM can compute the volume of semi-linear sets. We start by noting that taking volumes of semi-linear sets does not take us out of the semi-algebraic setting. This fact is easily derived from known results in the literature (and may have been published before, see, for example, [9] for a closely related result).

Lemma 5 *For any formula* $\varphi(\vec{x}, \vec{y})$ *over the real ordered group* $\mathbf{R}_{\text{lin}}$*, the volume of* φ *is semi-algebraic. That is,* $\{\vec{a}, v \mid [\text{VOL}\ \vec{y}.\varphi(\vec{x}, \vec{y})](\vec{a}, v)\}$ *is a semi-algebraic set.* □

We now prove that the language FO + POLY + SUM can express volumes of semi-linear sets.

Theorem 3 • *For every schema predicate* $S \in SC$ *there is an* FO + POLY + SUM *term* τ *which, for any semi-linear database* D*, computes the volume of* S *in* D*.*

- *For every* FO + LIN *query* φ *there is an* FO + POLY + SUM *term* τ_φ *such that for any semi-linear database* D*,* $\tau_\varphi(D)$ *returns the volume of* $\varphi(D)$*.*

Proof is by induction on dimension. We sketch it in dimensions 1 and 2, assuming S is bounded. If $S \subseteq \mathbb{R}$ is semilinear, it is a finite union of intervals, and hence volume is definable with summation. If $S \subseteq \mathbb{R}^2$, then $\text{VOL}(S) = \int\int \chi_S(x, y) dy\, dx$, where χ_S is the characteristic function. The innermost integral is $[\sum_{\rho_1(l,u,x)} \gamma](x)$, where ρ_1 is the query saying that l and u are the lower and the upper endpoints of a maximal interval from the set $\{y \mid S(x, y)\}$, and $\gamma(w, l, u) \equiv (w = u - l)$. The function $g(x) = [\sum_{\rho_1(l,u,x)} \gamma](x)$ is a piecewise linear function of x – this follows from the proof of Lemma 5. We can define in FO + POLY + SUM the set of points x where it is not smooth. Let T be the sum of all values $(mu^2 - ml^2)/2 + b(u - l)$, where the quadruples (u, l, m, b) vary over all quadruples of points such that (l, u) are consecutive points of nonsmoothness of g, and $g(x) = mx + b$ on the interval (l, u). Since g is piecewise linear, there are only finitely many pairs of consecutive points of nonsmoothness. Therefore there are only finitely many quadruples (u, l, m, b) as above. Also note that the formula $\gamma(w, l, u, m, b)$ given by $w = (mu^2 - ml^2)/2 + b(u - l)$ is a deterministic formula. Hence there is an FO + POLY + SUM query returning the sum of all γ output values w as (l, u, m, b) vary. Therefore, T is FO + POLY + SUM definable. That $T = \text{VOL}(S)$ follows from Fubini's theorem. □

6.2 Approximating volumes of semi-algebraic sets and FO + POLY + SUM

We now discuss a possible extension of FO+POLY+SUM to approximate volumes of semi-algebraic sets. The idea is to get a random sample and use it to approximate volume, since we can compute the number of points in the sample that fall into a given set. The sampling idea was used previously for approximating traditional relational aggregates (see [16, 22]). We extend this to the spatial context, and also obtain uniform dependence on parameters: for a query $\varphi(\vec{x}, \vec{y})$, one can find one sample that will provide a good approximation for *all* $\text{VOL}(\varphi(\vec{x}, D))$, with high probability.

The addition to the language that we propose is the witness, or choice, operator W of [2]. Given a query $\varphi(\vec{x}, \vec{y})$, $\text{W}\vec{y}.\varphi$ is a new query, with the same free variables, that randomly selects for each $\vec{a}$ one tuple from $\varphi(\vec{a}, D)$, if it is nonempty. $\text{W}\vec{x}.\varphi(\vec{x})$ selects randomly one tuple from $\varphi(D)$. For the use of the witness operator in query languages, see [2, 29].

We first deal with the case of finite instances D.

Theorem 4 *Let $\varphi(\vec{x}, \vec{y})$ be a* FO + POLY *query, with* $|\vec{x}| = n, |\vec{y}| = m$. *Let* $\epsilon, \delta > 0$. *Then there exists a* FO + POLY + SUM + W *query* $\psi_{\epsilon,\delta}(\vec{x}, z)$ *such that for every* $\vec{a}$, *there is a unique element* $v_{\vec{a}}$ *satisfying* $\psi_{\epsilon,\delta}(\vec{a}, \cdot)$, *and*

$$\sup_{\vec{a} \in I^n} |v_{\vec{a}} - \mathrm{VOL}_I(\varphi(\vec{a}, D))| < \epsilon$$

with probability at least $1 - \delta$. *Moreover, this query has at most* $\max(\frac{4}{\epsilon} \log \frac{2}{\delta}, \frac{C \log |D|}{\epsilon} \log \frac{13}{\epsilon})$ *calls to the witness operator* W, *where* C *is a constant that depends only on* φ.

Proof sketch. The proof follows from the classical results in learning theory on the size of a sample [10] and a $C \log |D|$ bound on the VC dimension. The latter is established in the Proposition below.

Proposition 6 *Let* $\mathcal{M} = \langle \mathcal{U}, \Omega \rangle$ *be o-minimal. Let SC be a relational schema, and* $\varphi(\vec{x}, \vec{y})$ *a* $\mathrm{FO}(SC, \Omega)$ *query. Then there is a number* C *that depends on* φ *only such that for the family* $F_\varphi(D) = \{\varphi(\vec{x}, D) \mid \vec{x} \in \mathcal{U}^{|\vec{x}|}\}$ *we have:* $\mathrm{VCdim}(F_\varphi(D)) < C \log(|D|)$. *If* $\mathcal{M}$ *is a structure with finite VC dimension, the same is true for all active-semantics* φ. □

With this result, [10] gives us the bound on the size of a single sample that tests multiple volumes; the sample is then generated using counting abilities of FO + POLY + SUM and the W operator. □

In some cases, it is possible to determine the constant C. For example, let $\varphi(\vec{x}, \vec{y})$ be an active semantics FO + POLY query, with $|\vec{y}| = k$. Let q be the quantifier rank of φ, and let p be the maximal arity of a relation in the schema. Let d be the maximal degree of a polynomial constraint used in φ (1, if none is used), and let s be the total number of atomic subformulae of φ. Then the bounds of [17] can be used to show that C can be taken to be $16k(p + q)(\log(8edps) + 1)$.

Remark Note that the bound of Theorem 4 holds for f.r. instances if querying is done via finite codings whose size is at most polynomial in the size of the finite representation. Such codings are known; see, e.g., [7, 30]; several papers studied querying via such finite codings [35, 7, 38]. Note also that the method of Theorem 4 can only be applied as top-level aggregation, as the result is not guaranteed to be semi-algebraic.

7 Conclusions

This paper has dealt with the key question of how to add aggregation to constraint query languages. The first fundamental question is whether there can be a language that is closed under the natural spatial aggregation operators, and which also retains the basic closure property that is fundamental to a constraint-based approach: namely, that every query output can be again represented as a constraint solution set. Our results give indication that this is impossible: these two closure properties are fundamentally incompatible. Perhaps more surprisingly, we show that the problem is not particular to the polynomial or linear constraint model; even going to a larger well-behaved constraint set does not remedy the problem.

The results above motivated us to look for languages that are not closed under volume operators, but which are closed under natural discrete aggregations and which permit the computation of volumes for semi-linear sets. The language FO + POLY + SUM defined here gives a natural approach to the addition of discrete aggregation operators to a constraint language. The key idea is the notion of range-restricted querying: allowing aggregation to be formed only on sets that are guaranteed to be finite. We show not only that FO + POLY + SUM has some attractive closure properties analogous to classical aggregate languages, but it allows one to do a significant amount of spatial aggregation, e.g., volumes of semi-linear sets.

The approach given here based on classical summation over range-restricted sets is natural, and allows one to re-use many of the evaluation strategies for classical aggregation operators; it is clear, however, that the syntax given here for FO + POLY + SUM is quite awkward. We hope to find more streamlined and natural syntax for FO + POLY + SUM, and we are looking at subsets of FO + POLY + SUM that can be more efficiently evaluated than the full language. It remains to discover how one could best provide support for directly expressing volumes in some language built 'on top of' FO + POLY + SUM, and how to add grouping constructs to the language.

Acknowledgements Part of this work was done while the second author was visiting INRIA. We thank Serge Abiteboul, Stéphane Grumbach, Michel Scholl and Luc Segoufin for helpful discussions. Libkin thanks all the members of the Verso team for their hospitality.

References

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

[2] S. Abiteboul, V. Vianu. Datalog extensions for database queries and updates. *JCSS* 43 (1991), 62–124.

[3] M. Anthony and N. Biggs. *Computational Learning Theory.* Cambridge Univ. Press, 1992.

[4] O. Belagradek, A. Stolboushkin, M.Tsaitlin. Extended order-generic queries. *APAL*, to appear.

[5] M. Benedikt, G. Dong, L. Libkin and L. Wong. Relational expressive power of constraint query languages. *Journal of the ACM* 45 (1998), 1–34.

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

[7] M. Benedikt and L. Libkin. Safe constraint queries. In *PODS'98*, pages 99–108.

[8] M. Benedikt and L. Libkin. Exact and approximate aggregation in constraint query languages. Technical Memo, Bell Labs, 1998.

[9] H. Bieri and W. Nef. A sweep-plane algorithm for computing the volume of polyhedra represented in Boolean form. *Linear Algebra and Its Applications* 52/53 (1983), 69–97.

[10] A. Blumer, A. Ehrenfeucht, D. Haussler, M. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM* 36 (1989), 929–965.

[11] J. Chomicki, D. Goldin and G. Kuper. Variable independence and aggregation closure. In *PODS'96*, pages 40–48.

[12] J. Chomicki and G. Kuper. Measuring infinite relations. In *PODS'95*, pages 78–85.

[13] L. Denenberg, Y. Gurevich and S. Shelah. Definability by constant-depth polynomial-size circuits. *Information and Control* 70 (1986), 216–240.

[14] M. Dyer and A. Frieze. On the complexity of computing the volume of a polytope. *SIAM J. Comput.*, 17 (1988), 967–974.

[15] M. Dyer, A. Frieze and R. Kannan. A random polynomial-time algorithm for approximating the volume of convex bodies. *Journal of the ACM*, 38 (1991), 1–17.

[16] P. Gibbons and Y. Matias. New sampling-based summary statistics for improving approximate query answers. In *SIGMOD'98*, pages 331–342.

[17] P. Goldberg and M. Jerrum. Bounding the Vapnik Chervonenkis dimension of concept classes parameterized by real numbers. *Machine Learning* 18 (1995), 131–148.

[18] M. Grötschel, L. Lovász and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization.* Springer, 1993.

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

[20] S. Grumbach and J. Su. Queries with arithmetical constraints. *Theoretical Computer Science* 173 (1997), 151–181.

[21] S. Grumbach, M. Rafanelli and L. Tininini. Querying aggregate data. This volume.

[22] J. Hellerstein, P. Haas and H. Wang. Online aggregation. In *SIGMOD'97*, pages 171-182.

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

[24] M. Karpinski and A. Macintyre. Approximating the volume of general Pfaffian bodies. In *Structures in Logic and Computer Science: A Selection of Essays in Honor of A. Ehrenfeucht*, Springer LNCS 1261, 1997, pages 162–173.

[25] M. Karpinski and A. Macintyre. Approximating volume and integrals in o-minimal and p-minimal theories. Technical Report, University of Bonn, 1997.

[26] P. Koiran. Approximating the volume of definable sets. In *FOCS'95*, pages 134–141.

[27] G. Kuper. Aggregation in constraint databases. In *PPCP'93*, 166–173.

[28] M. C. Laskowski. Vapnik-Chervonenkis classes of definable sets. *J. London Math. Soc.*, 45:377–384, 1992.

[29] S. Naqvi and S. Tsur. *A Logical Language for Data and Knowledge Bases.* Computer Science Press, 1989.

[30] C. Papadimitriou, D. Suciu and V. Vianu. Topological queries in spatial databases. In *PODS'96*, pages 81–92.

[31] C. Papadimitriou and M. Yannakakis. On limited nondeterminism and the complexity of the V-C dimension. *JCSS* 53 (1996), 161–170.

[32] J. Paredaens, J. Van den Bussche, and D. Van Gucht. First-order queries on finite structures over the reals. In *LICS'95*, pages 79–89.

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

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

[35] L. Segoufin and V. Vianu. Querying spatial databases via topological invariants. In *PODS'98*, pages 89–98.

[36] S. Shelah. Stability, the f.c.p., and superstability. *Ann. of Math. Logic* 3 (1971), 271–362.

[37] L. van den Dries. *Tame Topology and o-minimal Structures.* Cambridge Univ. Press, 1998.

[38] L. Vandeurzen, M. Gyssens and D. Van Gucht. An expressive language for linear spatial database queries. In *PODS'98*, pages 109–118.

[39] 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.

[40] M. Worboys. *GIS: A Computing Perspective.* Taylor & Francis, 1995.