Finitely Representable Databases†

Stéphane Grumbach*
I.N.R.I.A.
Rocquencourt BP 105
78153 Le Chesnay, France
stephane.grumbach@inria.fr

Jianwen Su**
Computer Science Department
University of California
Santa Barbara, CA 93106, USA
su@cs.ucsb.edu

February 6, 1995

Abstract: We study classes of infinite but finitely representable databases based on constraints, motivated by new database applications such as geographical databases. We formally define these notions and introduce the concept of query which generalizes queries over classical relational databases. We prove that in this context the basic properties of queries (satisfiability, containment, equivalence, etc.) are nonrecursive.

We investigate the theory of finitely representable models and prove that it differs strongly from both classical model theory and finite model theory. In particular, we show that most of the well known theorems of either one fail (compactness, completeness, locality, 0/1 laws, etc.). An immediate consequence is the lack of tools to consider the definability of queries in the relational calculus over finitely representable databases. We illustrate this very challenging problem through some classical examples.

We then mainly concentrate on dense order databases, and exhibit some new techniques to prove non first-order definability results. The techniques include complexity theoretical arguments and Ehrenfeucht-Fraïssé games. In particular, we show that queries over finite relations such as parity and graph connectivity, as well as topological queries such as region connectivity, existence of a hole, Eulerian traversal, homeomorphism, etc. are not first-order definable. We also show that inflationary Datalog with negation captures exactly all dense order queries computable in PTIME.

†An extended abstract [GS94] of this paper appeared in *Proc. ACM SIGACT-SIGMOD-SIGART Symp. on Principles of Database Systems, 1994*.

*Work supported in part by Esprit Project BRA AMUSING.

**Work supported in part by NSF grants IRI-9109520 and IRI-9117094 and NASA grant NAGW-3888. A part of work was done while visiting I.N.R.I.A.

1 Introduction

Until very recently, most data models and query languages have been designed for the modeling and the manipulation of finite collections of data items, for example, the classical network, hierarchical, and relational models, the nested relational or complex objects models, and object-oriented models. While these models fit well in the conventional database applications such as business applications, new applications, such as scientific or geographical databases, digital map and image libraries, etc. all have an urgent need for the support of the representations, the operations, and the management of infinite collections of data items. Typical examples are temporal and spatial data.

A new generation of data models for infinite collections of data items is starting to emerge in the literature (e.g., [KKR90, Rev90, Kup93a, BJM93]). We believe that in developing data models and languages for infinite data, there are two *fundamental principles* that should govern the choice of such a data model: (i) infinite databases should admit a finite representation, and (ii) a reasonable set of queries should be tractable. The first principle states that the new databases can be stored on a finite memory, while the second ensures that the data can be manipulated efficiently. In this paper, we consider classes of databases that satisfy the two principles above and that constitute a satisfactory answer to the trade-off between *classes of databases* and the *complexity of queries*. The focus of the paper is to investigate the theoretical foundation of databases for infinite data and to develop technical tools for studying the data models and query languages.

There have been only very few theoretical results on infinite databases. General results on the completeness of query languages for infinite recursive databases were reported by Hirst and Harel in [HH93], where two classes of recursive databases are considered, and it is shown that: (i) quantifier free first-order logic is complete on the class of all *recursive* databases, and (ii) a version of Chandra and Harel's *QL* [CH80] is complete on *highly symmetric* databases. The first result of [HH93] indicates that the class of recursive structures is much to general for database purposes, since quantifier free first-order logic is already complete on this class. Therefore, many basic queries are simply not computable (projection, connectivity, etc.). These results reveal the important trade-off between the class of structures taken as semantics and the class of admissible queries, which poses the challenging problem of exhibiting interesting classes that lie between the recursive and the highly symmetric ones. The trade-off between the class of structures and the complexity of queries has been further investigated in [GST94].

We study here infinite databases that are *finitely representable* by first order formulas in some logical language under the *context* of a first order structure (e.g., rational numbers with the natural order, the real closed field). Our framework follows and generalizes the pioneer work by Kanellakis, Kuper, and Revesz who introduced the *constraint query languages* in [KKR90]. The basic idea in [KKR90] is to generalize the relations of the relational model [Cod70] by defining generalized tuples as conjunctions of constraints. For instance, the formula $x^2 + y^2 = 1 \wedge x \leq 0$ defines a binary generalized tuple. A generalized, or finitely representable, relation is then a finite set of such tuples. In the real plane for instance, it results in an infinite set of points, or tuples over $\mathbb{R}^2$.

However, the finite representability in our context does not imply generally recursiveness of the databases. Indeed, we consider databases over general domains such as the set of real numbers, $\mathbb{R}$. We also consider countable domains (such as the integers, $\mathbb{N}$, or the rationals, $\mathbb{Q}$), and in this case, finitely representable databases are recursive.

The relational calculus over finitely representable relations constitutes a constraint query language which admits an efficient bottom-up and declarative semantics. It is suitable for expressing geometric queries, as shown by the examples illustrated in the present paper. Similar to the relational model, equivalent algebras were proposed [KG94, PVV94, GST94]. Deductive languages, such as Datalog, have also been studied in the context of constraint databases [KKR90]. Practical implementation issues, such as indexing of such databases have been considered [KRVV93], although this is probably one of the most fundamental research issues in this field.

Different types of constraints were considered in the literature, with dense or discrete order, and various arithmetic operations, such as dense linear order equations and inequalities, linear equations and inequalities, real polynomial equations and inequalities, etc. The relational calculus can be used as a query language as soon as the constraints are based on decidable theories which admit the elimination of quantifiers. The evaluation of a first-order query is then simply done by quantifier elimination. It is proved that, although the decision problem of the theories underlying the constraints used (e.g. dense order without endpoints, etc.) may be high, the data complexity of the query languages remains tractable. The data complexity is measured with respect to the size of a finitely representable relation, which is defined as the length of its finite representation. Upper bounds on the data complexity of both the relational calculus and inflationary Datalog with negation over constraint databases have been studied in [KKR90]. This shows the practical interest of this approach, which has been pursued in [Rev90, Kup93a, KG94, PVV94, GST94].

The goal of the present paper is to investigate the expressive power of query languages over finitely representable databases. For that purpose, we study the underlying logic of constraint query languages. In other words, we study first-order logic when the semantics is restricted to *finitely representable models*. We prove here that the model theory of finitely representable structures, differs strongly from the classical model theory of all structures. In particular, we prove that, like for finite model theory (see the survey by Fagin [Fag93]), most of the classical theorems of logic, such as the compactness and the completeness theorems, fail for finitely representable structures. The model theory of recursive structures has been studied in [HH94], and shown to differ strongly from classical model theory.

The failure of the main theorems of classical model theory, suggests the need to develop new tools applicable for finitely representable structures. Although powerful tools have been developed in the case of finite model theory, such as the locality by Gaifman [Gai81], or the 0/1 laws by Fagin [Fag76], the model theory of finitely representable structures differs from finite model theory. In particular, none of these tools are known to be applicable in the presence of some regular mathematical structure such as an order relation for instance.

Thus, the problem of the definability of a query over finitely representable databases is a very challenging issue. The Ehrenfeucht-Fraïssé games [Fra54, Ehr61] are among the few tools which carry over on finitely representable structures. They are quite intuitive when the language does not contain arithmetic operation, and is restricted to equality and an order relation. Once arithmetic is allowed, the combinatorics becomes rather tedious, though in some cases, proofs can still be carried out [ACGK94].

Complexity theoretic arguments can also be used to prove non-definability results. It has been shown that some rather low upper bounds, in terms of boolean circuits, of the relational algebra over finite structures carry over in the case of finitely representable structures. For example, it was first shown that the data complexity of first-order logic over dense order constraint databases in some normal form is in AC^0 [KG94]. The result was recently extended to classes of linear constraint databases, called *k-bounded*, where the number of occurrences of the addition symbol in each constraint is bounded [GST94]. Numerous examples of interesting queries, such as connectivity, are not in AC^0, and so are not expressible in first-order logic over dense order, or linear constraint databases.

We also investigate other criteria such as the invariance of the properties. There are interesting classes of queries that commute with groups of operations, such as permutations, rotations, translations, etc. This subject was explored in more detail in [PVV94]. We prove that the class of first-order queries definable by dense order constraints commute with the automorphisms of $\langle \mathbb{Q}, \leq \rangle$.

We illustrate these proof techniques on queries from computational geometry [Yao90], graph theory, and geographical databases. We investigate their definability in first-order logic over finitely representable databases. We show in particular the non first-order definability of the connectivity of an area in the context of a dense order. The proof can be made either with an Ehrenfeucht-Fraïssé game argument, or by a complexity argument. We illustrate both techniques.

Specifically, this paper makes the following contributions. First, the notion of a *finitely* *representable* database is defined and illustrated with examples in the contexts of the rational numbers with order and the real numbers with arithmetic operations. Second, we define the notions of a boolean and non boolean query and show that they generalize the classical notion of a query introduced by Chandra and Harel [CH80]. It is established that the existence of quantifier elimination procedure of the (complete) theory of a first order structure is a necessary and sufficient condition for a formula in the first order language to define a (non) boolean query. Third, we examine the theory of finitely representable models and prove that the compactness and completeness theorems of the classical first order logic fail for finitely representable models. We also show that validity over finitely representable models is in general co-r.e. but not r.e. while satisfiability over finitely representable models is not recursive. As a consequence, the basic properties of queries (satisfiability, containment, equivalence, and commutativity with bijections) are nonrecursive. Fourth, we study extensively dense order queries, which are queries over dense order databases or the databases defined using the rational numbers $\mathbb{Q}$ and its natural order. We focus on two query languages: the first order language and inflationary Datalog with negation. We show that graph queries such as parity and graph connectivity, topological queries such as region connectivity, existence of a hole, Eulerian traversal, homeomorphism are not definable in first order. However, all above queries except the homeomorphism query are expressible in the inflationary Datalog with negation. (If the homeomorphism query is expressible in the inflationary Datalog with negation then P = NP.) We further prove that inflationary Datalog with negation expresses exactly the set of all dense order queries computable in PTIME. This give a complete characterization of the complexity class. Finally, we briefly discuss first order queries over databases defined with real numbers and arithmetic operations and exhibit an "invariance" property which is then used to prove some non definability results.

In summary, the results provided here serve as a starting point for further study on query languages over finitely representable databases. Some of the results shown here have already been strengthened. In particular, it was shown in [GST94], that in the case of linear constraints (the language doesn't allow the multiplication symbol), then the parity and connectivity queries and the topological queries mentioned above are still not definable in first order. However, many problems remain open. For example, it is not known, when the multiplication operation is also used, if the parity query and the Euclidean spanning tree [KKR90] query are expressible.

The paper is organized as follows. In Section 2, we define finitely representable databases. In Section 3, we study the model theory of finitely representable structures and investigate the differences between finitely representable model theory and both finite model theory and classical model theory. In Section 4, we formally define the notion of queries for finitely representable databases. In Section 5, we examine in detail dense order queries and study definability in a first order language and inflationary datalog with negation. In Section 6, we briefly discuss queries over general polynomial constraint databases. Finally, open problems are presented in the last section.

2 Finitely Representable Databases

In this section, after reviewing basic notions of first-order languages, structures and their theories, we introduce the theory of *finitely representable models*. Throughout the whole paper, we consider only countable *first-order languages* with *equality*. We present the main definitions and illustrate them with important examples of finitely representable databases.

2.1 First-Order Languages, Structures and Theories

A *first order language* is a set $\mathcal{L}$ consisting of *predicate*, *function*, and *constant* symbols. Formulas are defined as usual with logical symbols and symbols from $\mathcal{L}$. The family of *terms* is recursively defined as follows. A constant or a variable is a term and $f(t_1, \ldots, t_n)$ is a term if f is an n-ary function symbol and $t_1, \ldots, t_n$ are terms. *Formulas* are constructed in the standard way. $P(t_1, \ldots, t_n)$ is an *atomic formula* where P is an n-ary predicate symbol and $t_1, \ldots, t_n$ are terms, $\neg\varphi$, $\varphi \vee \psi$, $\varphi \wedge \psi$, $\exists x\, \varphi$, and $\forall x\, \varphi$ are formulas where φ, ψ are formulas and x is a variable. A *sentence* is a formula with no free occurrences of variables. A formula is *quantifier-free* if no quantifier occurs in it.

A *structure* for a first-order language $\mathcal{L}$, or an $\mathcal{L}$-*structure*, is a pair $\mathcal{A} = \langle A, \rho \rangle$, where A is a non empty set, called the *universe*, and ρ is the interpretation function mapping predicate, function, and constant symbols in $\mathcal{L}$ to appropriate relations, functions, and elements in A. The notion of *satisfaction* is defined in the standard way. We say equivalently that a structure $\mathcal{A}$ is a *model* of a sentence φ, or that φ is *true* in $\mathcal{A}$, and we denote it by $\mathcal{A} \models \varphi$. The set of sentences true in a structure $\mathcal{A}$ is called the *theory of* $\mathcal{A}$. A theory $\mathcal{T}$ (set of sentences closed under logical implication) is complete, whenever for each sentence φ, we have $\varphi \in \mathcal{T}$ or $\neg\varphi \in \mathcal{T}$.

In this paper, we consider two examples of structures, that are of particular interest in the context of finitely representable models.

- The structure of the ordered rational numbers, $\mathcal{Q} = \langle \mathbb{Q}, =, \leq, (q)_{q \in \mathbb{Q}} \rangle$, over the language $\mathcal{L}_{\leq} = \{=, \leq\} \cup \mathbb{Q}$, and
- The structure of the ordered real numbers with arithmetic operations, $\mathcal{R} = \langle \mathbb{R}, =, \leq, +, \times, (q)_{q \in \mathbb{A}} \rangle$, over the language $\mathcal{L}_{\times} = \{=, \leq, \times, +\} \cup \mathbb{A}$, where $\mathbb{A}$ is the set of algebraic numbers.

Note that the structure $\mathcal{Q}$ satisfies the theory of dense order without endpoints, which is known to be complete [CK73]. Moreover, it admits the elimination of quantifiers, which as we will see in the following is a fundamental property. The structure $\mathcal{R}$ satisfies the theory of ordered real closed fields which, like the theory of dense order without endpoints, is also complete and admits the elimination of quantifiers [CK73]. Note that the languages considered contain a constant symbol for each rational number in the case of $\mathcal{Q}$, and for each algebraic number in the case of $\mathcal{R}$. This will be necessary in representing databases finitely (see Definition 2.2). Algebraic numbers are definable with $+, \times, 0,$ and 1, but this requires additional variables and quantifiers.

2.2 Finite Representability

Let $\mathcal{L}$ and $\mathcal{L}'$ be two first-order languages (with equality) such that $\mathcal{L}' \subseteq \mathcal{L}$, and let $\sigma = \{R_1, \ldots, R_n\}$ be a *signature* (or a *database schema*) such that $\mathcal{L} \cap \sigma = \emptyset$, where $R_1, \ldots, R_n$ are relation symbols. Throughout the paper, we always assume that the database schema is disjoint from the first-order language, and we distinguish between *logical predicates* (e.g., $=, \leq$) in $\mathcal{L}$ and *relations* in σ. Kanellakis, Kuper, and Revesz [KKR90] introduced the concept of a k-ary generalized tuple, which is a constraint expressed as a conjunction of atomic formulas in $\mathcal{L}$ over k variables. For instance,

$$(x \times x + y \times y = 1 \wedge x \leq 0)$$

is a binary generalized tuple in $\mathcal{L}_\times$ representing a half circle. A k-ary finitely representable relation (or generalized relation in [KKR90]) is then a finite set of k-ary generalized tuples. In this framework, a tuple $[a, b]$ in the context of classical relational databases [Cod70] is an abbreviation for the formula $(x = a \wedge y = b)$ represented using only the equality symbol, "$=$", and constants. A triangle $(x \leq y \wedge x \geq 0 \wedge y \leq 10)$ is also a generalized tuple represented using only "$\leq$" and constants.

Such a generalized database can be viewed as a finite representation of a structure, which is *expansion* of $\mathcal{R}$ to σ, i.e., a structure over the vocabulary $\{=, \leq, +, \times, (q)_{q \in \mathbb{A}}\} \cup \sigma$ that coincides with $\mathcal{R}$ on $\{=, \leq, +, \times, (q)_{q \in \mathbb{A}}\}$. The new relations of σ constitute the database (in the *context* of $\mathcal{R}$), and although they may be infinite, they have to be representable in a *finite* way with symbols in $\{=, \leq, +, \times, (q)_{q \in \mathbb{A}}\}$. We next define the fundamental concept of finitely representable relations and structures.

Definition 2.1 Let $\mathcal{A}$ be an $\mathcal{L}$-structure with universe A, $\varphi(x_1, \ldots, x_n)$ a quantifier-free formula in $\mathcal{L}$ with n distinct free variables $x_1, \ldots, x_n$, and $S \subseteq A^n$ be some n-ary relation. The relation S is *represented by* the formula φ *over* $\mathcal{A}$ if the following holds:

$$\forall a_1, \ldots, a_n \in A, \quad \mathcal{A} \models \varphi(a_1, \ldots, a_n) \;\Leftrightarrow\; (a_1, \ldots, a_n) \in S \; .$$

If S is represented by φ over $\mathcal{A}$, we also call φ a *finite representation of S over $\mathcal{A}$*.

Example 2.1 Let a, b, c, d be four rational numbers such that $a < c$ and $b < d$. In the rational plane, a rectangle with lower left and upper right corners being (a, b) and (c, d) (respectively) is represented by the following formula in $\mathcal{L}_{\leq}$ over $\mathcal{Q}$:

$$(a \leq x) \wedge (x \leq c) \wedge (b \leq y) \wedge (y \leq d).$$

In the 3-dimensional real space, the top half of the sphere of distance d from the point (a, b, c) is defined by the following formula in $\mathcal{L}_{\times}$:

$$(x-a)^2 + (y-b)^2 + (z-c)^2 = d. \qquad \square$$

Definition 2.2 Let $\mathcal{A}$ be an $\mathcal{L}$-structure with universe A and $S \subseteq A^n$ some n-ary relation. The relation S is *finitely representable in $\mathcal{L}' \subseteq \mathcal{L}$* (or *$\mathcal{L}'$-representable* for short) *over $\mathcal{A}$* ($\mathcal{A}$ is omitted when it is clear from the context) if it is represented by a quantifier-free formula $\varphi(x_1, \ldots, x_n)$ in $\mathcal{L}'$ with n distinct free variables.

Let $\mathcal{B}$ be an expansion of $\mathcal{A}$ to the schema σ. The structure $\mathcal{B}$ is said to be *$\mathcal{L}'$-representable (over $\mathcal{A}$)* if for every relation symbol R in σ, $R^{\mathcal{B}}$ is $\mathcal{L}'$-representable (over $\mathcal{A}$).

Consider the languages $\mathcal{L}_{\leq}$ and $\mathcal{L}_{\times}$ and the structure $\mathcal{R}$. It is easy to see that the half sphere in Example 2.1 is $\mathcal{L}_{\times}$-representable over $\mathcal{R}$. For $n = 2$, the following binary relations are $\mathcal{L}_{\leq}$-representable ($a, b, c, d \in \mathbb{Q}$):

- *Points* are represented by formulas of the form: "$(x = a) \wedge (y = b)$".
- *Segments* of vertical or horizontal lines or of the diagonal line $x = y$.
- *Rectangles* formed by vertical or horizontal lines are represented by formulas of the form: "$(a \leq x) \wedge (x \leq b) \wedge (c \leq y) \wedge (y \leq d)$".
- *Triangles* formed by a vertical line, a horizontal line, and the diagonal line ($x = y$) are represented by formulas of the form: "$(a \leq x) \wedge (x \leq y) \wedge (y \leq d)$".

Finitely representable relations are not arbitrary. In the case of the language $\mathcal{L}_{\leq}$, they are essentially finite unions of points, segments, rectangles and triangles (see Section 5). In particular it is easy to verify that the set of integers, $\mathbb{Z}$, is not finitely representable over $\mathcal{Q}$. Indeed, every $\mathcal{L}_{\times}$-representable set S over $\mathcal{Q}$ contains a finite number of segments of the reals [Tar51], and so its intersection with $\mathbb{Z}$ is either finite or co-finite.

We next define the fundamental notion of *database instance*.

Definition 2.3 Let $\mathcal{B}$ be an expansion of $\mathcal{A}$ to σ which is $\mathcal{L}'$-representable over $\mathcal{A}$. Then each mapping I such that, for each $R \in \sigma$, $I(R)$ is a finite representation of $R^{\mathcal{B}}$ in $\mathcal{L}'$ over $\mathcal{A}$, is called an *$\mathcal{L}'$-representable $(\mathcal{A}, \sigma)$-(database) instance*.

When everything is clear from the context, we will just say an $(\mathcal{A}, \sigma)$-instance or simply an instance. In the following, we fix $\mathcal{A}$ and σ, and consider the class $K(\mathcal{A}, \sigma, \mathcal{L}')$ of $\mathcal{L}'$-representable $(\mathcal{A}, \sigma)$-instances. If the cardinality of the language $\mathcal{L}'$ is countable, then $K(\mathcal{A}, \sigma, \mathcal{L}')$ is effectively enumerable. Moreover, if $\mathcal{A}$ has a countable universe such as the integers or the rationals, then every relation in each $(\mathcal{A}, \sigma)$-instance in $K(\mathcal{A}, \sigma, \mathcal{L}')$ is recursive. Note that $K(\mathcal{A}, \sigma, \mathcal{L}')$ has interesting closure properties. Indeed, it is closed under finite union and intersection and moreover under complementation. This differs from finite model theory (the complement of a finite model is not finite).

Definition 2.4 Let I and J be two $\mathcal{L}'$-representable $(\mathcal{A}, \sigma)$-instances of σ. I and J are *equivalent* if for each $R \in \sigma$ $I(R)$ and $J(R)$ are equivalent formulas under $\mathcal{A}$, that is:

$$\mathcal{A} \models \forall \overline{x}(I(R)(\overline{x}) \leftrightarrow J(R)(\overline{x})).$$

Note that for $\mathcal{L}_{\leq}$-representable $(\mathcal{Q}, \sigma)$-instances, equivalence of instances can be decided in AC^0, and for $\mathcal{L}_{\times}$-representable $(\mathcal{R}, \sigma)$-instances, equivalence can be decided in NC. This follows easily from results in [Col75, KKR90] (see Sections 5 and 6).

Finally, if φ is a sentence in $\mathcal{L} \cup \sigma$, we denote by $K_{\varphi}(\mathcal{A}, \sigma, \mathcal{L}')$ the collection of $\mathcal{L}'$-representable $(\mathcal{A}, \sigma)$-instances I satisfying φ, that is for each $\mathcal{L}'$-representable $(\mathcal{A}, \sigma)$-expansion $\mathcal{B}$, if $\mathcal{B} \models \varphi$, then the finite representation of $\mathcal{B}|_{\sigma}$ is in $K_{\varphi}(\mathcal{A}, \sigma, \mathcal{L}')$. (We may denote $K_{\varphi}(\mathcal{A}, \sigma, \mathcal{L}')$ simply by $K_{\varphi}^{\mathcal{A}}$ when σ is clear from the context and $\mathcal{L}' = \mathcal{L}$.)

In general, a finitely representable relation may be infinite (of arbitrary cardinality, not necessarily enumerable), but it admits a finite representation. A finite relation (database) is representable using only the equality predicate (and constants), and therefore is an $\{=\}$-representable relation (and the representing formula does not involve negation). The converse doesn't hold. Nevertheless, the following can be easily verified.

Proposition 2.1 A monadic relation is $\{=\}$-representable iff it is finite or co-finite.

Remark: For practical purposes, one need only to assume that the database contains the quantifier free formulas defining the relations. The rest is built-in. In the practical examples of the paper, we consider finite representability in $\mathcal{L}_{\leq}$ over $\mathcal{Q}$ and $\mathcal{R}$, and in $\mathcal{L}_{\times}$ over $\mathcal{R}$.

3 Finite Models, Finitely Representable Models, and all Models

In this section, we analyze the differences between finite model theory, finitely representable model theory, and classical model theory. We shall see that the theory of finitely representable models differs from classical model theory, and prove that, as for finite model theory, the compactness theorem fails, and the validity over finitely representable models is in general co-r.e. and not r.e.

We next define the *satisfiability over finitely representable models*.

Definition 3.1 A sentence φ in $\mathcal{L} \cup \sigma$ is *satisfiable* (respectively, *valid*) *over finitely representable models with respect to* $\mathcal{A}$ *and* σ if there exists a (respectively, for each) expansion $\mathcal{B}$ of $\mathcal{A}$ to σ which is $\mathcal{L}$-representable over $\mathcal{A}$, $\mathcal{B} \models \varphi$.

It is important to note that we always assume some fixed $\mathcal{A}$ and σ. The semantics is then restricted to expansions of $\mathcal{A}$ to σ. We next prove the failure of the compactness theorem.

Proposition 3.1 The *Compactness Theorem* fails for finitely representable models.

Proof: To illustrate the failure of the compactness theorem, we give the proof for two different contexts.

(i) $\mathcal{L} = \{=\}$: Consider the structure $\mathcal{N} = \langle \mathbb{N}, = \rangle$, where $\mathbb{N}$ is the set of natural numbers. We assume a relational schema σ with one monadic relation R. For each integer $k \geq 0$, consider the sentence τ_k which states that there are at least k distinct elements in R and at least k distinct elements not in R. Let $\Sigma = \{\tau_k \mid k \geq 0\}$. Every finite subset of Σ is obviously satisfiable over finitely representable models with respect to $\mathcal{N}$ and σ. However Σ is not satisfiable over the finitely representable models, since each $\{=\}$-representable monadic relation is either finite or co-finite (Proposition 2.1).

(ii) Consider the structure $\mathcal{R}$ and $\mathcal{L}_{\times}$. We assume again a relational schema σ with one monadic relation R. Let τ_k be the sentence which states that (1) the initial segment of k natural numbers are contained in R, $\{0, 1, \ldots, (k-1)\} \subseteq R$, and (2) if $x \in R$ and $0 \leq x < k$, then $x \in \{0, 1, \ldots, (k-1)\}$. Define $\Sigma = \{\tau_k \mid k \geq 0\}$. Obviously each finite subset of Σ has a model with respect to $\mathcal{R}$ and σ which is $\mathcal{L}_{\times}$-representable. On the other hand, Σ does not admit any $\mathcal{L}_{\times}$-representable model. Indeed, any monadic first-order formula over the reals define a finite set of intervals of $\mathbb{R}$ [Tar51, Col75]. $\square$

The failure of the compactness theorem for restricted classes of models of interested in computer science has been investigated in particular in [HH94, GST94]. The failure of the completeness theorem follows. In classical model theory, the completeness theorem implies, among other things, the fact that the set of valid first-order sentences is recursively enumerable (r.e.). It follows from Church's Theorem that, as soon as the language contains some relation symbol that is not unary, the set of valid first-order sentences is not co-r.e. The contrary holds for finite model theory. Indeed, the set of first-order sentences valid over finite structures is co-r.e., and Trakhtenbrot proved [Tra50] that it is not r.e. A similar phenomenon holds for finitely representable models.

Proposition 3.2 Let $\mathcal{A}$ be an $\mathcal{L}$-structure such that the theory of $\mathcal{A}$ is decidable. Then the set of sentences in $\mathcal{L} \cup \sigma$ valid over $\mathcal{L}$-representable expansions of $\mathcal{A}$ to σ is co-r.e.

Proof: We prove that the set of sentences in $\mathcal{L} \cup \sigma$ satisfiable over finitely representable models with respect to $\mathcal{A}$ and σ is r.e. Recall that $\mathcal{L}$ is assumed to be countable. The set of $\mathcal{L}$-representable $(\mathcal{A}, \sigma)$-instances can be effectively enumerated and each instance corresponds to an $\mathcal{L}$-representable expansion of $\mathcal{A}$ to σ. It suffices to enumerate these representations, which are the formulas without quantifiers representing the relations in σ. Now, let $\mathcal{A}'$ be an expansion whose restriction to σ is as above, and assume that $\sigma = \{R_1, \ldots, R_n\}$ and for each $i \in [1..n]$, $\psi_i(\bar{x}_i)$ is the finite representation of R_i. We have that

$$\mathcal{A}' \models \varphi \quad \text{if and only if} \quad \mathcal{A} \models \varphi|_{\psi_1}^{R_1} \cdots |_{\psi_n}^{R_n}$$

where $\phi|_{\psi}^{R}$ is the formula obtained from ϕ by replacing each occurrence of R by the formula ψ. Then the result follows from the decidability of the theory of $\mathcal{A}$. $\Box$

The theory of dense order without endpoints and the theory of real closed fields mentioned above are decidable [CK73]. Therefore, validity over finitely representable databases over, for instances, $\mathcal{Q}$ and $\mathcal{R}$ is co-r.e. We will see below that it is not r.e. We next prove that in general the set of first-order sentences satisfiable over finitely representable models is not recursive. Like for the compactness, we state the result in two different contexts.

Theorem 3.3 Let $\mathcal{L}$ be a first-order language with $=$ and $\mathcal{N}$ the structure of the natural numbers with equality only. Then the set of sentences in $\mathcal{L} \cup \sigma$ valid over $\{=\}$-representable models with respect to $\mathcal{N}$ and σ is not r.e.

Proof: The proof is based on Vaught's result [Vau60]. Let $V_{\text{r.e.}}$ (respectively V_{fin}) be the set of sentences true in all recursively enumerable (respectively finite) models. Vaught showed that for each set of sentences V, if $V_{\text{r.e.}} \subseteq V \subseteq V_{\text{fin}}$, then V is not r.e. Now, let $V_{\mathcal{N}}^{\sigma}$ be the set of sentences valid over finitely representable models with respect to $\mathcal{N}$ and σ. It is clear that $V_{\text{r.e.}} \subseteq V_{\mathcal{N}}^{\sigma} \subseteq V_{\text{fin}}$, and therefore $V_{\mathcal{N}}^{\sigma}$ is not r.e. $\Box$

Next, we consider languages with both $=$ and $\leq$, where $\leq$ satisfies the axioms of the theory of dense order without endpoints , and with possibly a countable set of constants and possibly $+$ and $\times$ with their usual interpretation (i.e., satisfying the axioms of the theory of real closed fields).

Theorem 3.4 Let $\mathcal{L}$ be a first-order language as described above. Assume that $\mathcal{A}$ is an $\mathcal{L}$-structure such that the theory of $\mathcal{A}$ admits a quantifier elimination procedure. Then the set of sentences in $\mathcal{L} \cup \sigma$ satisfiable over $\mathcal{L}$-representable models with respect to $\mathcal{A}$ and σ is not recursive.

Proof: The proof is done by a reduction from the satisfiability over finite structures. Let $\sigma = \{R\}$, where R is an n-ary relation symbol, $\mathcal{L}' = \{=\}$, and A be the universe of $\mathcal{A}$. Let ϕ be a sentence in $\mathcal{L}' \cup \sigma$. We construct a sentence ψ in $\mathcal{L} \cup \sigma$ such that ϕ has a finite model ($\mathcal{L}' \cup \sigma$-structure) iff ψ has an $\mathcal{L}$-representable model ($\mathcal{L} \cup \sigma$-structure). Let

$$\phi_i(x) = \exists x_1 \cdots x_{i-1} x_{i+1} \cdots x_n R(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n)$$

be the projection of R on the i-th column. Then we define $\psi = \phi \wedge \bigwedge_{i=1}^{n} \alpha_i$, where ($x < y$ denotes $x \leq y \wedge x \neq y$)

$$\alpha_i = \forall xy \ \Big((\phi_i(x) \wedge \phi_i(y) \wedge x < y) \to \exists z \left(x < z < y \wedge \neg \phi_i(z) \right) \Big) .$$

Suppose that ϕ admits a finite model $\mathcal{M} = (M, R^{\mathcal{M}})$ of cardinality ℓ. Let h be a 1-1 onto mapping from M to a finite subset of A. Let $\mathcal{A}'$ be an expansion of $\mathcal{A}$ to σ such that:

$$\mathcal{A}' \models \forall x_1 \cdots x_n \Big(R(x_1, \ldots, x_n) \leftrightarrow \bigvee_{(a_1,\ldots,a_n) \in R^{\mathcal{M}}} \ \bigwedge_{i=1}^{n} \ x_i = h(a_i) \Big).$$

Then, $\mathcal{A}'$ is an $\{=\}$-representable, hence $\mathcal{L}$-representable, expansion and a model of ψ. Indeed, since $\leq^{\mathcal{A}}$ is dense, α_i holds in $\mathcal{A}'$.

On the other hand, suppose that ψ has an $\mathcal{L}$-representable model $\mathcal{A}'$, which is an expansion of $\mathcal{A}$ to σ. We first prove that $R^{\mathcal{A}'}$ is finite. Since $R^{\mathcal{A}'}$ is an $\mathcal{L}$-representable relation, let ψ_R be the quantifier-free formula which is the representation of $R^{\mathcal{A}'}$.

By assumption, the theory of $\mathcal{A}$ admits a quantifier elimination procedure. Therefore, $R_i = \{x \mid \phi_i|^{R}_{\psi_R}(x)\}$ is also $\mathcal{L}$-representable. Let $\psi_{R_i}(x)$ be the quantifier-free monadic formula defining R_i. By [Tar51, Col75], $\psi_{R_i}(x)$ defines a finite set of segments (intervals) over the universe A. On the other hand, $\mathcal{A}'$ satisfies α_i, and so for every pair of distinct elements $a, b \in A$ such that $a, b \in R_i$ and $a < b$, there is some $c \in A$ such that $a < c < b$ and $c \notin R_i$. It follows that $\psi_{R_i}(x)$ defines a finite set of elements. Hence, $R_i^{\mathcal{A}'}$ is finite, and so is

$R^{\mathcal{A}'}$. Consider the $(\{=\} \cup \sigma)$-structure $\mathcal{M} = (M, R^{\mathcal{M}})$, where $M = \bigcup_{i=1}^{n} R_i$ and $R^{\mathcal{M}} = R^{\mathcal{A}'}$. $\mathcal{M}$ is a finite model of ϕ. $\square$

Note that finiteness can be expressed in the very specific case of finitely representable databases in the context of a dense order. This is a drastic difference with classical logic. The previous result admits the following corollaries.

Corollary 3.5 Under the same assumptions as in Theorem 3.4, the set of sentences in $\mathcal{L} \cup \sigma$ valid over $\mathcal{L}$-representable models with respect to $\mathcal{A}$ and σ is co-r.e. but not r.e.

We can therefore conclude with the following general result.

Corollary 3.6 The Completeness Theorem does not hold for finitely representable models.

The next theorem shows yet another similarity with finite model theory. The proof is easy and in the same spirit as the one for finite models.

Theorem 3.7 Consider an $\mathcal{L}$-representable $(\mathcal{A}, \sigma)$-expansion $\mathcal{B}$. There exists a sentence $\sigma_{\mathcal{B}}$ in $\mathcal{L} \cup \sigma$ such that for each $\mathcal{L}$-representable $(\mathcal{A}, \sigma)$-expansion $\mathcal{B}'$, $\mathcal{B}'$ is isomorphic to $\mathcal{B}$ iff $\mathcal{B}' \models \sigma_{\mathcal{B}}$.

On the other hand, the theory of finitely representable models differs from finite model theory. Gaifman [Gai81] proved that first-order logic was local in a topological sense. This fails in presence of an order relation. Indeed, every neighborhood based on Gaifman's distance contains the whole domain. Techniques based on asymptotic probabilities of the truth of sentences also fail in the presence of order or arithmetic. Some results were obtained for very specific languages (see [Com88]). They do not carry over in the general case.

4 Queries

The notion of a *database query* was introduced by Chandra and Harel [CH80] as a mapping Q from structures over a given signature σ to relations of a fixed arity n, which is *partial recursive* and satisfies the following *consistency criterion*: If two structures over σ, $\mathcal{A}$ and $\mathcal{B}$ are isomorphic by an isomorphism μ, then $Q(\mathcal{A})$ and $Q(\mathcal{B})$ are isomorphic by the same isomorphism μ. This criterion was then called "genericity" in the database literature.

Definition 4.1 Let $\mathcal{A}$ and $\mathcal{B}$ be two $\mathcal{L}$-structures with universe A and B (respectively) and μ be a mapping from A to B. If α is a term or a quantifier-free formula in $\mathcal{L}$, the *image of*

α *under* μ, denoted by $\mu(\alpha)$, is defined recursively as follows:

$$\mu(\alpha) = \begin{cases} x & \text{if } \alpha = x \text{ is a variable} \\ \mu(c) & \text{if } \alpha = c \text{ is a constant} \\ f(\mu(t_1), \ldots, \mu(t_n)) & \text{if } \alpha = f(t_1, \ldots, t_n) \text{ is a composed term and} \\ & \quad f \in \mathcal{L} \text{ is an } n\text{-ary function symbol} \\ p(\mu(t_1), \ldots, \mu(t_n)) & \text{if } \alpha = p(t_1, \ldots, t_n) \text{ is an atomic formula and} \\ & \quad p \in \mathcal{L} \text{ is an } n\text{-ary predicate symbol} \\ \neg\mu(\phi) & \text{if } \alpha = \neg\phi \\ \mu(\phi) \vee \mu(\psi) & \text{if } \alpha = \phi \vee \psi \\ \mu(\phi) \wedge \mu(\psi) & \text{if } \alpha = \phi \wedge \psi \end{cases}$$

If $S \subseteq A^n$ is an n-ary relation, the *image of* S *under* μ, denoted by $\mu(S)$, is defined as:

$$\mu(S) = \left\{ \ (\mu(a_1), \ldots, \mu(a_n)) \ \middle| \ (a_1, \ldots, a_n) \in S \ \right\}.$$

The *image of an* $(\mathcal{A}, \sigma)$*-instance* I *under* μ, denoted by $\mu(I)$, is defined analogously by: for each relation $R \in \sigma$, $\mu(I(R)) = I(\mu(R))$.

Let $\mathcal{A}$ be an $\mathcal{L}$-structure with universe A. A mapping μ from A to A is said to be $\mathcal{L}$*-preserving with respect to* a formula φ in $\mathcal{L}$ if $\mu(\varphi)$ is also a formula in $\mathcal{L}$.

Example 4.1 Consider the language $\mathcal{L}_\leq$ and the structure $\mathcal{Q}$. Every one-to-one onto mapping from $\mathbb{Q}$ to $\mathbb{Q}$ which preserves the order $\leq$ is an automorphism of $\mathcal{Q}$. Let $\varphi(x) \equiv x = 2$. Consider the two automorphisms:

$$\mu_1(x) = x^3 \qquad \text{and} \qquad \mu_2(x) = \frac{\pi}{2}\, x$$

Clearly, μ_1 is $\mathcal{L}_\leq$-preserving with respect to φ and μ_2 is not, since $\mu_1(\varphi) \equiv x = 8$ is a formula in $\mathcal{L}_\leq$, while $\mu_2(\varphi) \equiv x = \pi$ is not. □

In the remainder of the paper, we consider automorphisms which are $\mathcal{L}$-preserving with respect to some formulas defining a database instance. We will say that an automorphism is $\mathcal{L}$-preserving with respect to an instance, when it is $\mathcal{L}$-preserving with respect to the formulas defining the relations in the instance.

Proposition 4.1 Let $\mathcal{A}$ be an $\mathcal{L}$-structure with universe A, μ an automorphism of $\mathcal{A}$, $S \subseteq A^n$ an n-ary relation over A, and φ a quantifier-free formula in $\mathcal{L}$. We further assume that μ is $\mathcal{L}$-preserving with respect to φ. If the relation S is represented by φ, then $\mu(S)$ is represented by $\mu(\varphi)$.

Proof: Let $x_1, \ldots, x_n$ be the n distinct free variables in φ. It is sufficient to show

$$(a_1, \ldots, a_n) \in \mu(S) \quad \Leftrightarrow \quad \mathcal{A} \models \mu(\varphi)|_{a_1}^{x_1} \cdots |_{a_n}^{x_n} .$$

We prove the above equivalence by an induction on the formula φ. For the basis step, assume that $\varphi(x_1, \ldots, x_n) = p(t_1, \ldots, t_k)$ is an atomic formula, where p is a predicate in $\mathcal{L}$ and $t_1, \ldots, t_k$ are terms, we have:

$$\begin{array}{lll}
(a_1, \ldots, a_n) \in \mu(S) & & \\
\quad \Leftrightarrow & (\mu^{-1}(a_1), \ldots, \mu^{-1}(a_n)) \in S & \text{(by definition of } \mu(S)\text{)} \\
\quad \Leftrightarrow & \mathcal{A} \models p(t_1, \ldots, t_k)|_{\mu^{-1}(a_1)}^{x_1} \cdots |_{\mu^{-1}(a_n)}^{x_n} & \text{(basis case)} \\
\quad \Leftrightarrow & \mathcal{A} \models p(\mu(t_1), \ldots, \mu(t_k))|_{\mu^{-1}(a_1)}^{x_1} \cdots |_{\mu^{-1}(a_n)}^{x_n} & \text{(since } \mu \text{ is an automorphism)} \\
\quad \Leftrightarrow & \mathcal{A} \models p(t_1, \ldots, t_k)|_{a_1}^{x_1} \cdots |_{a_n}^{x_n} & \\
\quad \Leftrightarrow & \mathcal{A} \models p(\mu(t_1), \ldots, \mu(t_k))|_{a_1}^{x_1} \cdots |_{a_n}^{x_n} & \text{(since } \mu \text{ is an automorphism)} \\
\quad \Leftrightarrow & \mathcal{A} \models \mu(p(t_1, \ldots, t_k))|_{a_1}^{x_1} \cdots |_{a_n}^{x_n} & \\
\quad \Leftrightarrow & \mathcal{A} \models \mu(\varphi)|_{a_1}^{x_1} \cdots |_{a_n}^{x_n} . &
\end{array}$$

For the induction step, it is sufficient to consider the expressions $\varphi = \neg\phi$, $\varphi = \phi \wedge \psi$ and $\varphi = \phi \vee \psi$. Assume that S' is the relation represented by ϕ. Clearly,

$$\begin{array}{rcll}
(a_1, \ldots, a_n) \in \mu(S) & \Leftrightarrow & (\mu^{-1}(a_1), \ldots, \mu^{-1}(a_n)) \in S & \\
& \Leftrightarrow & (\mu^{-1}(a_1), \ldots, \mu^{-1}(a_n)) \notin S' & \\
& \Leftrightarrow & \mathcal{A} \models \neg\mu(\phi)|_{a_1}^{x_1} \cdots |_{a_n}^{x_n} & \text{(induction hypothesis)} \\
& \Leftrightarrow & \mathcal{A} \models \mu(\neg\phi)|_{a_1}^{x_1} \cdots |_{a_n}^{x_n} . &
\end{array}$$

The proof for $\varphi = \phi \wedge \psi$ and $\varphi = \phi \vee \psi$ is done analogously. □

Definition 4.2 A *boolean query over a structure* $\mathcal{A}$ *and a signature* σ is a partial recursive collection K of $(\mathcal{A}, \sigma)$-instances satisfying the following conditions:

1. If two $(\mathcal{A}, \sigma)$-instances I and J are equivalent, then $I \in K$ if and only if $J \in K$; and

2. If an $(\mathcal{A}, \sigma)$-instance $I \in K$ and μ is an automorphism of $\mathcal{A}$ preserving I, then $\mu(I) \in K$.

Intuitively, Condition (1) in Definition 4.2 states that either all equivalent representations of (finitely representable) "databases" are in a boolean query K or none of them is. Condition (2) is a natural generalization of the classical notion of genericity [CH80] where permutations of the universe U (i.e. automorphisms of the structure $\langle U, = \rangle$) are replaced by automorphisms of the underlying more complex structure $\mathcal{A}$. Note that we could also restricted to automorphisms that are identity on a fixed set C of constants, and this variation corresponds to the notion of C-genericity [Hul86].

Example 4.2 Consider a signature $\sigma = \{R\}$, where R is a unary relation symbol. Let K_{even} be the collection of $\mathcal{L}_{\leq}$-representable $(\mathcal{Q}, \sigma)$-instances I such that $I(R)$ represents a finite subset of $\mathcal{Q}$ of even cardinality. It is clear that K_{even} satisfies both conditions of Definition 4.2 and, in particular, is closed under automorphisms of $\mathcal{Q}$. Thus K_{even} is a boolean query over $\mathcal{Q}$. Consider now a signature σ' consisting of a single binary relation symbol. The collection K_{conn} of $\mathcal{L}_{\leq}$-representable $(\mathcal{Q}, \sigma')$-instances representing finite and connected graphs also defines a query over $\mathcal{Q}$. $\square$

More generally, classical graph and relational database queries are queries over $\mathcal{Q}$ in the sense defined above. This holds as well in the case of $\mathcal{R}$. The next example is devoted to properties of infinite relations.

Example 4.3 Consider the structure $\mathcal{R}$ and a binary relation $S \subseteq \mathbb{R}^2$. S is an (infinite) set of points in the real plane $\mathbb{R}^2$. S is *region connected* if every pair of points in S is connected by a curve lying entirely in S. Let σ be a signature consisting of a single binary relation R, the collection K of $\mathcal{L}_{\times}$-representable $(\mathcal{R}, \sigma)$-instances representing connected regions is a boolean query over $\mathcal{R}$, called the (2-dimensional) *region connectivity* query. $\square$

Boolean queries might enjoy various interesting closure properties. Consider a boolean query K over an $\mathcal{L}$-structure $\mathcal{A}$ with universe A and let $\mathcal{L}' \subseteq \mathcal{L}$. We define an $\mathcal{L}'$-*automorphism* to be a permutation of A which preserves constants, functions, and relations in $\mathcal{L}'$. The *parity* and the *graph connectivity* queries in Example 4.2 are closed under $\emptyset$-automorphisms (i.e., closed under permutations of the universe); the region connectivity query in Example 4.3 is closed under $\{\leq\}$-automorphisms. In Section 6, we shall exhibit a technique based on closure under $\mathcal{L}'$-automorphism for $\mathcal{L}' \subset \mathcal{L}$, for proving inexpressibility results. More geometry-based properties of queries have been exhibited by Paredaens, Van den Bussche, and Van Gucht [PVV94].

We next focus on queries that are defined using first-order logic. We first consider boolean queries. The next proposition, shows that under the restriction to finitely representable models, every first-order sentence defines a boolean query.

Proposition 4.2 Let $\mathcal{A}$ be an $\mathcal{L}$-structure. For each constant-free sentence φ in $\mathcal{L} \cup \sigma$, $K_{\varphi}^{\mathcal{A}}$ defines a boolean query over $\mathcal{A}$.

Proof: We assume for simplicity that σ contains only one relation symbol R. Condition (1) of Definition 4.2 easily follows from the definition of $K_{\varphi}^{\mathcal{A}}$. The proof of Condition (2) is accomplished by an induction on the structure of the formulas representing the $\mathcal{L}$-representable relations ($(\mathcal{A},\sigma)$-instances). Consider an $\mathcal{L}$-representable expansion $\mathcal{M}$ of $\mathcal{A}$ to σ such that a representation I of $\mathcal{M}|_\sigma$ is in $K_\varphi^{\mathcal{A}}$. Then:

$$\mathcal{M} \models \forall \bar{x}\ (R(\bar{x}) \leftrightarrow \bigvee_i \bigwedge_j Atom_{i,j}(\bar{x})),$$

where $Atom_{i,j}(\bar{x})$ are (negation of) atomic formulas in $\mathcal{L}$. Define $\phi = \varphi|_\psi^R$, where $\psi = \bigvee_i \bigwedge_j Atom_{i,j}(\bar{x})$ and $\varphi|_\psi^R$ is the formula obtained from φ by replacing each occurrence of R with the formula ψ. It follows that $\mathcal{M} \models \phi$. Now, let μ be an automorphism of $\mathcal{A}$ which is $\mathcal{L}$-preserving with respect to I, and let $\mathcal{N} = \mu(\mathcal{M})$ and $J = \mu(I)$. By Proposition 4.1, J is an $(\mathcal{A},\sigma)$-instance and a representation of $\mathcal{N}|_\sigma$. Since μ is an automorphism of $\mathcal{A}$ and ϕ a sentence in $\mathcal{L}$, it is straightforward that $\mathcal{N} \models \mu(\phi)$. Because φ is constant-free, $\mu(\varphi) = \varphi$. Thus, $\mu(\phi) = \varphi|_{\mu(\psi)}^R$, which implies that $J \in K_\varphi^{\mathcal{A}}$. □

The above proposition can be generalized to sentences having constants by considering only automorphisms that are identity on the constants in φ.

We now consider non-boolean queries. In general, not every formula with free variables defines a query. Let σ and τ be two signatures. Intuitively, a "query" $\sigma \rightarrow \tau$ is a total mapping from $\mathcal{L}$-representable $(\mathcal{A},\sigma)$-instances to $\mathcal{L}$-representable $(\mathcal{A},\tau)$-instances. Formally, it is defined as a collection of $\mathcal{L}$-representable $(\mathcal{A},\sigma\cup\tau)$-instances.

Let $\sigma = \{R_1,\ldots,R_n\}$ be a database schema and $\tau = \{R_o\}$ where R_o is a k-ary relation. A first-order formula $\varphi(x_1,\ldots,x_k)$ in $\mathcal{L}\cup\sigma$, with free variables $\{x_1,\ldots,x_k\}$, defines a class of structures K_φ such that $\mathcal{B} \in K_\varphi$ if and only if:

1. $\mathcal{B}$ is an $\mathcal{L}$-representable expansion of $\mathcal{A}$ to $\sigma\cup\tau$, and

2. $R_o^{\mathcal{B}} = \{(a_1,\ldots,a_k) \mid \mathcal{B}|_{\mathcal{L}\cup\sigma} \models \varphi(a_1,\ldots,a_k)\}$.

For an arbitrary formula φ, K_φ does not necessarily define a query. Indeed, $R_o^{\mathcal{B}}$ may well not be $\mathcal{L}$-representable, and then φ only defines a *partial* mapping from $\mathcal{L}$-representable $(\mathcal{A},\sigma)$-instances to $\mathcal{L}$-representable $(\mathcal{A},\tau)$-instances. The next proposition exhibits a sufficient and necessary condition on the structure $\mathcal{A}$, such that every formula in $\mathcal{L}$ defines a (total) *query*.

Proposition 4.3 Let $\mathcal{A}$ be some $\mathcal{L}$-structure, and $\mathcal{T}$ be the complete theory such that $\mathcal{A} \models \mathcal{T}$. Then, the following conditions are equivalent.

1. $\mathcal{T}$ admits a quantifier elimination procedure.

2. Every formula φ in $\mathcal{L}\cup\sigma$ with k free variables defines a query $\sigma \rightarrow \tau$ over $\mathcal{L}$-representable $(\mathcal{A},\sigma)$-instances.

Proof: The direction $(1 \Rightarrow 2)$ is obvious. Conversely, it suffices to consider a formula φ in $\mathcal{L}$. Since it defines a query, there exists a formula without quantifiers in $\mathcal{L}$ which is $\mathcal{T}$-equivalent to φ. Therefore $\mathcal{T}$ admits a quantifier elimination procedure. □

The direction $(1 \Rightarrow 2)$ was used in [KKR90] to see that the query languages were well-defined for finitely representable databases. The previous proposition shows that this is a necessary condition, and therefore finitely representable databases can only be defined over models of theories admitting a quantifier elimination procedure.

A theory admitting a quantifier elimination procedure, that is of great use in the context of finitely representable databases, is the theory of real closed fields over the language $\mathcal{L}_\times$ [Tar51]. It should be noted that the functions allowed in the language play a fundamental role in the quantifier elimination procedure. In particular, van den Dries showed that the extension of the previous theory to a language with an exponential function, $\{=, \leq, +, \times, e^x, 0, 1\}$, doesn't admit a quantifier elimination procedure [Dri82]. The theory of dense order without endpoints is another example of a theory admitting a quantifier elimination procedure that is of great use here.

We next briefly discuss the case of finite databases. They can be viewed as finitely representable databases over the integers $\mathcal{N} = \langle \mathbb{N}, = \rangle$ (with or without order). The representation is based on the equality predicate only, and no negation is allowed in tuples. The concepts defined above carry over, and corresponds to well known concepts of (finite) database theory.

In [Kup90], finite databases over the structure $\mathcal{R}$ of real numbers were considered. It was claimed that if a query in $\mathcal{L}_\times$ is closed under $\emptyset$-automorphisms (permutations of real numbers), then it is definable using only equality. Unfortunately, this result doesn't hold [Via93, Kup93b]. A simple counterexample can be found in ([AHV94], Exercise 17.27, page 162). It has been conjectured more recently that a query in $\mathcal{L}_\times$ closed under $\emptyset$-automorphisms is definable with the equality and the order relation only [VdB95]. Such a result would have important consequences on the expressive power of the languages considered for classical relational queries (see Section 5). It remains open if the transitive closure of a graph, or of the parity of the cardinality of a relation are definable in $\mathcal{L}_\times$. In the next section, we give the proof that they are not definable in $\mathcal{L}_\leq$.

Finally, we consider some general problems concerning satisfiability, containment, and equivalence of queries on finitely representable databases.

Let $\mathcal{A}$ be an $\mathcal{L}$-structure, and φ and ψ be two sentences in $\mathcal{L} \cup \sigma$. The query ψ is said to be *contained in* φ *with respect to* $\mathcal{L}$*-representable* $(\mathcal{A}, \sigma)$*-instances*, denoted by $\psi \sqsubseteq_{\mathcal{A},\sigma} \varphi$, if $K_\psi^\mathcal{A} \subseteq K_\varphi^\mathcal{A}$. Queries φ and ψ are *equivalent with respect to* $\mathcal{L}$*-representable* $(\mathcal{A}, \sigma)$*-instances*, denoted $\varphi \equiv_{\mathcal{A},\sigma} \psi$, if $\psi \sqsubseteq_{\mathcal{A},\sigma} \varphi$ and $\varphi \sqsubseteq_{\mathcal{A},\sigma} \psi$. Finally, the query ψ is *satisfiable* if $K_\psi^\mathcal{A}$ is not empty.

We consider again the languages $\mathcal{L}_{\leq}$, and $\mathcal{L}_{\times}$ and the corresponding structures $\mathcal{Q}$ and $\mathcal{R}$ (respectively). The following results easily follow from Theorem 3.4.

Theorem 4.4 Under the same assumptions as in Theorem 3.4, the following problems are undecidable: the satisfiability, the equivalence, the containment, and whether a query is closed under $\{\leq\}$-automorphism.

In particular, Theorem 4.4 indicates that the above problems are undecidable in the first-order constraint query language of [KKR90] over the real field.

The "data complexity" of queries is defined as usual based on computational devices and standard encoding of the input/output. Intuitively, the time (space) data complexity of a query is the time (space) needed in evaluating the query with respect to the size of the input database instance. Since the primary focus of the paper is on dense order databases and queries, we formally introduce the notion of data complexity for dense order queries. For other kinds of queries, it suffices to extend the tape alphabet to the relation and function symbols in $\mathcal{L}$, and define a standard binary encoding of the constants.

We first introduce the *standard encoding* of a database instance, which is obtained by encoding the quantifier-free formulas representing it. Formulas are encoded in the alphabet:

$$\{\#, [,], (,), \wedge, \vee, \neg, =, \leq, 0, 1, \div\} \cup \sigma \cup \{\mathtt{x_1}, \ldots, \mathtt{x_k}\}$$

where σ is a signature, and k the maximal arity of relation symbols in σ. Natural numbers are encoded in binary notation, and rationals as pairs (fractions) of natural numbers $(n \div m)$. Variables are encoded by themselves. In the following we first illustrate the encoding of a relation defined in $\mathcal{L}_{\leq}$ with an example and then describe the encoding in the general case for relations and database instances.

Example 4.4 Suppose that R is a binary relation defined by:

$$R(x, y) \equiv ((2.75 \leq x) \wedge (x \leq 7) \wedge (x > y)) \vee (x \leq y) \ .$$

The encoding of R is done as follows:

$$\boxed{\mathtt{R}[\leq ((\div(\mathtt{1011}, \mathtt{100}), \mathtt{x1}) \wedge \leq (\mathtt{x1}, \mathtt{111}) \wedge \neg(\leq (\mathtt{x1}, \mathtt{x2}))] \vee [\leq (\mathtt{x1}, \mathtt{x2})]\#} \qquad \square$$

Let *enc* denote the encoding function. For atomic formulas of form "$p(t_1, \ldots, t_k)$", the encoding is defined as:

$$enc(p(t_1, \ldots, t_k)) = \boxed{p(enc(t_1), \ldots, enc(t_k))} \ .$$

The encoding of terms is made in the same way as for atomic formulas. If $\varphi \equiv \psi_1 \wedge \cdots \wedge \psi_\ell$ is a conjunction of atomic formulas, then:

$$enc(\varphi) = \boxed{enc(\psi_1) \wedge \cdots \wedge enc(\psi_\ell)}\ .$$

Suppose I is a database and a k-ary relation $I(R)$ is represented by a formula in *disjunctive normal form:* $R(x_1, \ldots, x_k) \equiv \phi_1 \vee \cdots \vee \phi_\ell$, where each ϕ_i is a conjunction of atomic formulas. Then

$$enc(I(R)) = \boxed{R[enc(\phi_1)] \vee\ \cdots\ \vee [enc(\phi_\ell)]\#}\ .$$

Finally, Let σ be a signature with n relation symbols and $R_1, \ldots, R_n$ be an enumeration of them. Then the encoding of an instance I is defined as:

$$enc(I) = \boxed{enc(I(R_1))\#\ \cdots\ \#enc(I(R_n))\#\#}\ .$$

The *size* of a database instance is defined as the size of the encoding of its representation (assuming an enumeration of the relations of the database schema).

Let $\mathcal{C}$ be a complexity class, such as PTIME, PSPACE, etc. A query Q is computable in $\mathcal{C}$ if there is a Turing machine[1] M, such that M starting on a standard encoding of the input database produces a standard encoding of the output database, in a bounded amount of resources such as polynomial time, polynomial space, etc. in the size of the input.

5 Dense-Order Databases and Queries

In this section, we restrict our attention to dense order constraints over the rational numbers $\mathbb{Q}$, and to two query languages: the corresponding first-order language $\mathcal{L}_\leq$ ($\cup\, \sigma$ for some signature σ), and inflationary Datalog with negation, $DATALOG^\neg$. We study the definability and non-definability of queries in $\mathcal{L}_\leq$ and $DATALOG^\neg$ with respect to finitely representable databases (using $\leq$ and rational numbers) over $\mathcal{Q}$. We show that queries expressible in $\mathcal{L}_\leq$ are closed under homeomorphisms of $\mathcal{Q}$, and therefore express essentially topological properties. We also prove that many interesting topological queries such as region connectivity, "existence of a hole", "Eulerian traversal", "k-dimensional homeomorphisms" ($k \geq 2$) etc. and well-known graph queries such as connectivity and parity are not expressible in first-order language while most of them are expressible in $DATALOG^\neg$. The proof techniques for non-definability results include Ehrenfeucht-Fraïssé games and data complexity arguments.

$\mathcal{Q}$ is a model of the theory of dense order without endpoints. The theory is decidable and admits quantifier elimination procedures [CK73]. Let $\mathcal{R}_\leq = (\mathbb{R}, \leq)$ be the structure of

[1] Or any other device such as a family of boolean circuits in the case of parallel complexity classes.

the real dense order. It is also a model of the theory of dense order without endpoints and therefore, the results concerning the language $\mathcal{L}_{\leq}$ and finitely representable databases over $\mathcal{Q}$ are also applicable to $\mathcal{L}_{\leq}$ and finitely representable databases over $\mathcal{R}_{\leq}$.

5.1 Queries

When focusing on queries over $\mathcal{L}_{\leq}$-representable $(\mathcal{Q}, \sigma)$-instances (in the remainder of this section, we will simply call an "$\mathcal{L}_{\leq}$-representable $(\mathcal{Q}, \sigma)$-instance" a "database instance" or an "instance"), one easily verifies that each formula in $\mathcal{L}_{\leq} \cup \sigma$ defines a query over these instances. Indeed, first-order mappings commute with (preserving[2]) automorphisms of $\mathcal{Q}$. In particular, each first-order boolean query defines a recursive collection of database instances over σ closed under (preserving) automorphisms of $\mathcal{Q}$.

Example 5.1 There are very simple mappings in the rational plane which are not queries.

- There is a line with empty intersection with the input, which separates the input, i.e., there are points in the input relation on both sides of the line.
- R defines a *grid* structure if it has a finite set of points and for each point (x, y) in R, there exist integers i_x, i_y such that $x = x_0 + i_x\Delta_x$ and $y = y_0 + i_y\Delta_y$ for some fixed $x_0, y_0, \Delta_x, \Delta_y$. The *grid query* checks if an input is a grid.
- Two disconnected areas have the same surface.

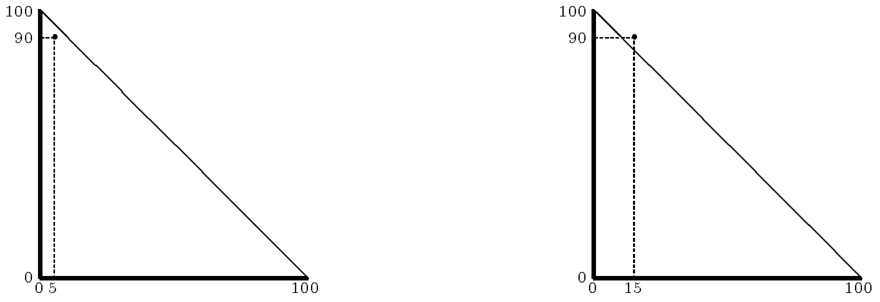
We next prove that the first mapping above is not a query. Consider indeed the input binary relation R defined by:

$$R(x, y) \equiv (y = 0 \wedge 0 \leq x \leq 100) \bigvee (0 \leq y \leq 100 \wedge x = 0) \bigvee (x = 5 \wedge y = 90).$$

R contains an isolated point $(5, 90)$ and two connected line segments (thick lines in Figure 1(a). It can be easily verified that the isolated point cannot be separated by any straight line not intersecting the input (Figure 1(a)). Consider now the following automorphism of $\mathcal{Q}$:

$$\mu(x) = \begin{cases} x & \text{if } -\infty \leq x \leq 0 \\ 3x & \text{if } \quad 0 \leq x \leq 10 \\ \dfrac{x+80}{3} & \text{if } \quad 10 \leq x \leq 40 \\ x & \text{if } \quad 40 \leq x \leq \infty \end{cases}$$

[2]Note that an automorphism is preserving with respect to a given formula.



(a) (b)

Figure 1: *Line separation* is not a query

The image of R under μ is:

$$\mu(R)(x,y) \equiv (y = 0 \wedge 0 \leq x \leq 100) \bigvee (0 \leq y \leq 100 \wedge x = 0) \bigvee (x = 15 \wedge y = 90)$$

and the isolated point $(15, 90)$ in the image can now be separated from the other two segments of R by a straight line, namely $y = -x + 101$ (Figure 1(b)). □

Consider the usual topology on the set of rationals, $\mathbb{Q}$. A *homeomorphism* of $\mathbb{Q}$ is a bicontinuous bijection from $\mathbb{Q}$ onto $\mathbb{Q}$. It maps open sets to open sets and closed sets to closed sets. The following can be easily verified.

Proposition 5.1 Every boolean query is closed under (preserving) homeomorphisms of $\mathbb{Q}$.

Example 5.1 illustrates the fact that there are very simple mappings which are not queries over dense order instances. The concept of query introduced in Section 4 is therefore quite restrictive. Nevertheless it includes all first-order definable mappings, and so constitutes a tool to prove non first-order definability results. Proposition 5.1 shows that queries over dense order constraints are insensitive to homeomorphic transformations on the axis. This constitutes an intuitive tool for non-definability. Other notions of queries were introduced in [PVV94], based on closure under various groups of transformations, such as rotations, translations, etc. They have strong mathematical and practical motivations. The concept of query used in the present paper, is related to the logical definability, but as shown by Proposition 5.1 also corresponds to a natural closure, namely the closure under homeomorphisms of $\mathbb{Q}$.

We next consider the complexity of dense order queries. For each complexity class $\mathcal{C}$, we denote by $\mathcal{C}_{\mathbb{Q}}$ the set of queries over dense-order constraint databases computable in $\mathcal{C}$. The data complexity of queries over $\mathcal{L}_{\leq}$-representable $(\mathcal{R}_{\leq}, \sigma)$-instances has been studied in [KKR90, KG94]. Kanellakis and Goldin have shown recently that the data complexity of queries in $\mathcal{L}_{\leq} \cup \sigma$ is in AC^0, while Kanellakis, Kuper, and Revesz proved that the data complexity of $DATALOG^{\neg}$ is in PTIME.

Theorem 5.2 [KG94, KKR90] Each query in $\mathcal{L}_{\leq} \cup \sigma$ (respectively, $DATALOG^{\neg}$) over $\mathcal{L}_{\leq}$-representable $(\mathcal{R}_{\leq}, \sigma)$-instances is in $\mathrm{AC}^0{}_{\mathcal{Q}}$ (respectively, $\mathrm{PTIME}_{\mathcal{Q}}$).

The proof of the AC^0 upper bound [KG94] is made by an encoding of the dense-order constraint databases into finite models. This encoding itself is not in AC^0. An algebra working on the representation in terms of finite relations, simulates first-order operations on the dense-order constraint databases. This algebra is easily shown to be in AC^0. An alternative proof is given in [GST94], which doesn't require an encoding in terms of finite relations. The AC^0 upper bound also applies to a wider class.

On the other hand, the expressive power of first-order queries in the language $\mathcal{L}_{\leq}$ has not been seriously investigated. In [KKR90], it was shown that inflationary Datalog with negation can express all PTIME *relational* queries (mappings from finite relational databases to finite relations).

5.2 First-Order Queries

We now study definability and non-definability of numerous queries in the first-order query language $\mathcal{L}_{\leq}$ for dense order databases. The queries studied include geometry-based queries such as region connectivity, existence of a hole, Eulerian traversal, k-dimensional homeomorphism, etc. and graph queries such as parity and graph connectivity. These queries are shown to be non-definable in first order. We prove the results mainly using a complexity-theoretic approach but also illustrate the main ideas of the Ehrenfeucht-Fraïssé games-based approach.

We first consider *k-dimensional region connectivity* (cf Example 4.3 with $k = 2$). Assume that $\sigma = \{R\}$, where R is a k-ary relation symbol. Intuitively, R defines some area (infinite set of points) in the rational space $\mathbb{Q}^k$. Recall that R defines a connected region if every pair of points in R can be linked by a continuous curve entirely in R. Note that when $k = 1$, the 1-dimensional region connectivity can be easily expressed using the order $\leq$. We now present the following non-definability result with a detailed complexity-theoretic proof and an outline of a proof using Ehrenfeucht-Fraïssé games.

Theorem 5.3 For each positive integer $k \geq 2$ and each k-ary relation symbol R, the k-dimensional region connectivity query over $\mathcal{L}_{\leq}$-representable $(\mathcal{Q}, \{R\})$-instances is not definable in $\mathcal{L}_{\leq} \cup \{R\}$.

Proof: The proof is accomplished by a first order reduction [Imm87] from the function MAJORITY to region connectivity. Since the boolean function MAJORITY is not in AC^0 [FSS84] and queries in $\mathcal{L}_{\leq} \cup \{R\}$ are in AC^0 (Theorem 5.2), it follows that region connectivity is not definable in $\mathcal{L}_{\leq} \cup \{R\}$. The boolean function MAJORITY takes as input n boolean variables $x_1, \ldots, x_n$ and outputs 1 if more than half of the variables have the value 1, and 0 otherwise. We fix k and consider a two dimensional plane in the k-dimensional space. In the following, we use (i, j) to refer to a point in the plane. Let $(x_1, \ldots, x_n)$ be the inputs for MAJORITY. We construct the following line segments in the plane:

- From $(n, \frac{n+1}{2})$ to (n, n), from $(0, n)$ to (n, n), and from $(0, 0)$ to $(n, 0)$;
- For each $1 \leq i \leq n$ and each $0 \leq j < n$, from $(i-1, j)$ to $(i, j + x_i)$ (corresponding to x_i).

An example of the resulting line segments is shown in Figure 2(a), where $n = 6$ and

$$x_1\, x_2\, x_3\, x_4\, x_5\, x_6 = 1\ 0\ 1\ 0\ 1\ 1.$$

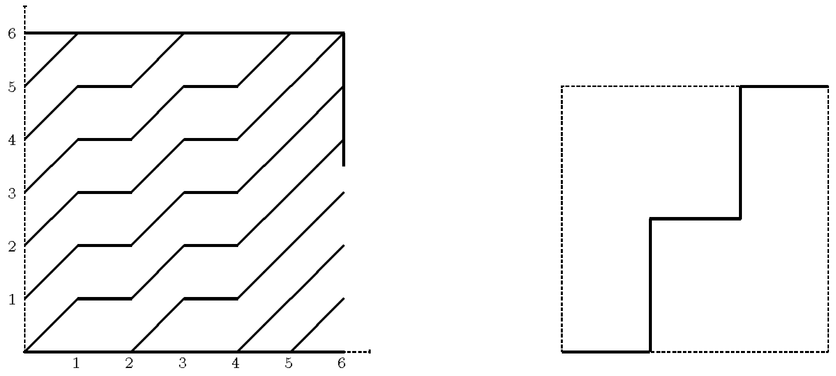
It is quite easy to see that the set of line segments is connected if and only if

$$\text{MAJORITY}(x_1, \ldots, x_n) = 1,$$

since the line starting from the point $(0, 0)$ reaches the vertical line at the right of the figure (line segment from $(n, \frac{n+1}{2})$ to (n, n)). To complete the proof, it is noted that dense order constraints do not allow diagonal lines. Hence in constructing the dense order database, we have to replace each diagonal line by 3 horizontal and 2 vertical line segments as shown in Figure 2(b). Clearly, the reduction is first order. $\Box$

The above data complexity approach for proving non definability results compares the complexity upper bound of the language and the lower bound of the problem being considered. On the other hand, Theorem 5.3 can also be proved using Ehrenfeucht-Fraïssé games. We sketch the proof ideas below.

Let $\mathcal{A}$ and $\mathcal{B}$ be two $\mathcal{L}$-structures with universes A and B (respectively). Suppose further that r is a positive integer. The *game of length r associated with $\mathcal{A}$ and $\mathcal{B}$* is played by two players, I (the spoiler), and II (the duplicator), making r moves each. Player I starts by picking an element in A or B and player II picks an element in the opposite structure. This is repeated for r times. At each move, player I has the choice of the structure, and player II must respond in the opposite structure. Let a_i (respectively b_i) be the i^{th} element picked in A (respectively B). Player II wins the round $\{(a_1, b_1), \ldots, (a_r, b_r)\}$ iff the mapping $a_i \mapsto b_i$ (for $1 \leq i \leq r$) is an isomorphism of the substructures of $\mathcal{A}$ and $\mathcal{B}$ generated by $\{a_1, \ldots, a_r\}$ and $\{b_1, \ldots, b_r\}$, respectively, $\mathcal{A}/\{a_1, \ldots, a_r\}$ and $\mathcal{B}/\{b_1, \ldots, b_r\}$.



(a) (b)

Figure 2: Illustration of reductions

Player II *wins the game of length* r associated with $\mathcal{A}$ and $\mathcal{B}$ if he has a winning strategy, i.e., player II can always win any game of length r on $\mathcal{A}$ and $\mathcal{B}$, no matter how player I plays. This is denoted by $\mathcal{A} \equiv_r \mathcal{B}$. Note that the relation $\equiv_r$ is an equivalence relation on structures.

Intuitively, the equivalence $\mathcal{A} \equiv_r \mathcal{B}$ says that $\mathcal{A}$ and $\mathcal{B}$ cannot be distinguished by looking at just r constants at a time in the two structures. The main result concerning Ehrenfeucht-Fraïssé games states that the ability to distinguish among structures using games of length r is equivalent to the ability to distinguish among structures using some first-order sentence of quantifier depth r. Ehrenfeucht-Fraïssé's general result implies in particular the following proposition.

Proposition 5.4 Let ϕ be a sentence in $\mathcal{L} \cup \sigma$ with quantifier depth r, and let $\mathcal{A}$ and $\mathcal{B}$ be two $(\mathcal{L} \cup \sigma)$-structures. If $\mathcal{A} \models \phi$ and $\mathcal{A} \equiv_r \mathcal{B}$, then $\mathcal{B} \models \phi$.

Since $\mathcal{L}$-representable $(\mathcal{A}, \sigma)$-instances are $(\mathcal{L} \cup \sigma)$-structures, Proposition 5.4 can be applied in the context of finitely representable databases. This provides a technique for proving that a boolean query K is not definable by any first-order sentence. Indeed, it is sufficient to exhibit, for each r, two $\mathcal{L}$-representable $(\mathcal{A}, \sigma)$-instances $\mathcal{A}_r$ and $\mathcal{B}_r$ such that $\mathcal{A}_r$ belongs to K, $\mathcal{B}_r$ does not, and $\mathcal{A}_r \equiv_r \mathcal{B}_r$.

We next use Ehrenfeucht-Fraïssé games to sketch a proof of Theorem 5.3 for a binary relation R. The players play on two instances. They choose alternatively, rational numbers. These numbers are the coordinates of points in the relation R. We assume that they have both r pebbles that can be placed on coordinates.

We describe the main ideas of the two instances on which the game is played and the strategy of the duplicator player. The two instances, say $\mathcal{A}_r$ and $\mathcal{B}_r$, are built with segments. They look like graphs which differ only by one edge (i.e. segment). Each of them looks like two imbricate combs sharing one tooth in $\mathcal{A}_r$ and none in $\mathcal{B}_r$ (Figure 3). The teeth are actually not straight lines but complex zigzag. The number of teeth and the number of angles in a tooth is exponential in r (the number of steps of the game). Moreover, the "distinguished points", such as the root of a teeth, or its extremity, have no coordinates in common. In particular, the root and the extremity of a tooth are not on a vertical line, and teeth all have different length. (We assume moreover that no rational coordinates can be defined explicitly. That corresponds to the case, where a sentence contains some rational constants. Otherwise, it suffices to define the structures away from these numbers.)

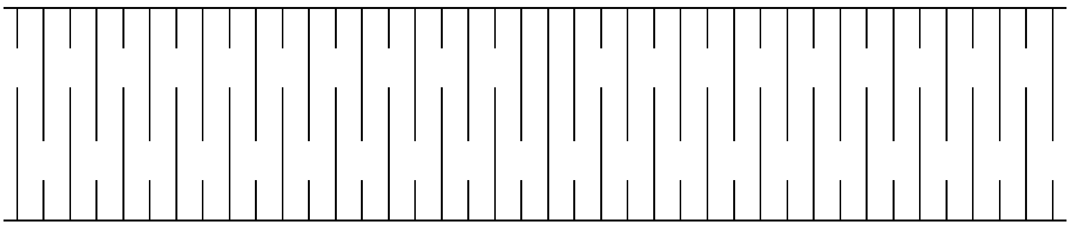


Figure 3: A skeleton of the instance $\mathcal{A}_r$

For simplicity reasons, we explain the strategy of the duplicator as if the two players were picking points in $\mathbb{Q}^2$ instead of rational numbers in $\mathbb{Q}$. In fact, they are choosing coordinates, and do not even necessarily alternate first and second coordinates. Nevertheless, this simplifies the description of the strategy, and the way this strategy can be implemented on the real game follows. The winning strategy of the duplicator goes as follows. Problems arise only when the spoiler plays on the tooth on $\mathcal{A}_r$ linking the two combs. If the spoiler plays on its two roots, the duplicator, plays on the roots of two opposite teeth in $\mathcal{B}_r$. Now, if the spoiler plays on their extremities, the duplicator plays on the extremities of other teeth in the neighborhood in $\mathcal{A}_r$. Since the number of teeth and the number of angles in a tooth are exponential in r, this can go on for r moves. The duplicator plays on points such that the coordinates of the marked points in the two instances satisfy identically $=$ and $\leq$. What makes this strategy possible is that the distances do not have to be respected. This is due to the fact that the language is restricted to the predicates $=$ and $\leq$. $\square$

Ehrenfeucht-Fraïssé games can be used to prove other results. In particular, it can be seen that the *parity* of the number (if finite) of points, segments, connected components, etc. cannot be expressed in $\mathcal{L}_{\leq}$ nor can the transitive closure of finite graphs in this language. The technique of Ehrenfeucht-Fraïssé games, however, cannot be easily extended to languages including arithmetic operations. Indeed, any two instances such as $\mathcal{A}_r$ and $\mathcal{B}_r$ in the previous proof can be distinguished using a sentence using $+$ and $\times$. Any algebraic number can be

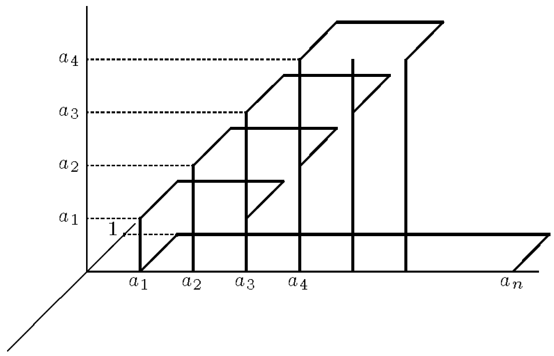


Figure 4: Reduction from PARITY

also defined. So, the following can be defined: there is a point (a, a') in $R^{\mathcal{A}_r}$ not in $R^{\mathcal{B}_r}$, where a and a' are algebraic.

We next present an alternative complexity theoretic proof of the non definability of the region connectivity query in the case $k \geq 3$. Essentially, it is a reduction from the PARITY query that is known to be outside of AC^0 [FSS84, Aj83]. The reduction is first-order [Imm87], and so in AC^0. Let $\{a_1, \ldots, a_n\}$ be the set of elements. For technical convenience and without loss of generality, we assume that for each $i \in [2..n]$, $a_{i-1} < a_i$. Now the relation $R(x, y, z)$ is obtained by adding the following elements. For each $i \in [1..n]$, we represent a_i by the point $(a_i, 0, 0)$ in R and for each $i \in [1..(n-2)]$, we construct (straight) line segments from $(a_i, 0, 0)$ via points $(a_i, 0, a_i)$, $(a_i, 1, a_i)$, $(a_{i+2}, 1, a_i)$, $(a_{i+2}, 0, a_i)$, and eventually to $(a_{i+2}, 0, 0)$. Finally, R also includes a set of lines starting from $(a_n, 0, 0)$ via points $(a_n, 1, 0)$, $(a_1, 1, 0)$, and then to $(a_1, 0, 0)$. Figure 4 shows some of the lines in R. It is easy to verify that R is region connected iff n is even. Hence, region connectivity in the 3-dimensional space is not definable. Note that this reduction does not naturally apply to the case when $k = 2$, since the lines linking odd nodes and even nodes would overlap. Furthermore, the reduction becomes non first order (nor in AC^0) if it lays the lines of even (or odd) nodes in the other side of the y axis. This alternative proof exhibits the difference between the boolean function MAJORITY and the query PARITY in proving non definability results.

Using similar first order reductions from the boolean functions MAJORITY and PARITY, one can verify that the following queries are also not definable in $\mathcal{L}_{\leq}$.

Corollary 5.5 The transitive closure query and the parity query over finite databases are not expressible in $\mathcal{L}_{\leq}$ (under appropriate signatures).

We next consider several topological queries. Let R be an input relation of arity k. The k-dimensional *at least one hole* (*exactly one hole*) query answers "yes" if the region R

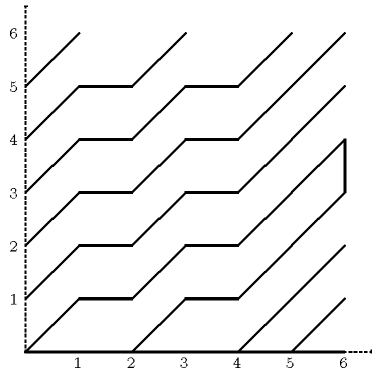


Figure 5: Reduction to at least and exactly one hole queries

has at least (respectively exactly) one hole. Similar to the region connectivity query, these queries are easily definable for 1 dimensional inputs.

Theorem 5.6 For each positive integer $k \geq 2$ and each k-ary relation symbol R, the k-dimensional at least and exactly one hole queries over $\mathcal{L}_{\leq}$-representable $(\mathcal{Q}, \{R\})$-instances are not expressible in $\mathcal{L}_{\leq} \cup \{R\}$.

Proof: The proof is based on reductions from MAJORITY. We now describe the reduction for at least one hole query. Let $x_1, \ldots, x_n$ be the input variables for MAJORITY. Assume again we consider a two dimensional subplane in the k dimensional space and use (i, j) to refer to a point in the plane. We construct the following line segments in the plane for the relation R (Figure 5):

- From $(0,0)$ to $(n,0)$ and from $(n, \left\lfloor \frac{n}{2} \right\rfloor)$ to $(n, \left\lfloor \frac{n}{2} \right\rfloor + 1)$, and
- For each $1 \leq i \leq n$ and each $0 \leq j < n$ from $(i-1, j)$ to $(i, j + x_i)$ (corresponding to x_i).

Now, MAJORITY$(x_1, \ldots, x_n) = 1$ if and only if the point $(0,0)$ eventually reaches a point (n, i) for some $i > \left\lfloor \frac{n}{2} \right\rfloor$ if and only if the line segment $(n, \left\lfloor \frac{n}{2} \right\rfloor)$ to $(n, \left\lfloor \frac{n}{2} \right\rfloor + 1)$ forms a hole with the segment $(0,0)$ to $(n,0)$, i.e., the at least and exactly one hole queries will answer "yes". □

In the next example, we consider the query *Eulerian traversal* over dense order constraint databases. The input is a set of line segments in the rational plane and the query returns "yes" if there is a traversal which goes through each line segment exactly once continuously. In other words, if we view a line as a set of points, a traversal goes continuously through each point exactly once except for a finite set of points (the crossings of lines). It can be verified that it is indeed a query.

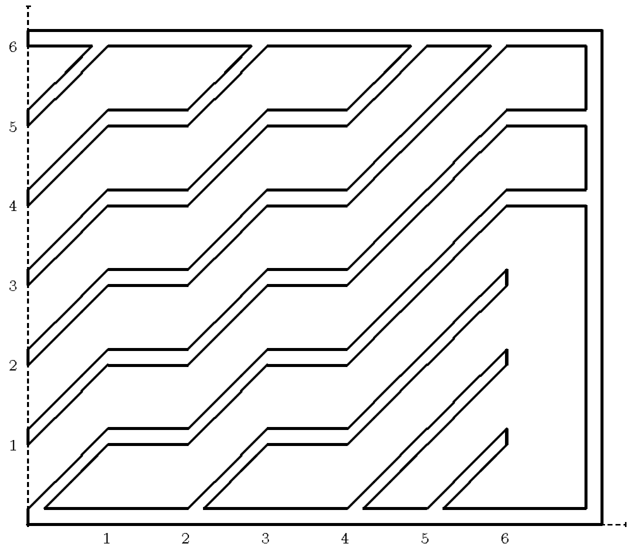


Figure 6: Reduction to Euler Traversal

Theorem 5.7 Let R be a binary relation symbol. The Eulerian traversal query over $\mathcal{L}_{\leq}$-representable $(\mathcal{Q}, \{R\})$-instances is not expressible in $\mathcal{L}_{\leq} \cup \{R\}$.

Proof: The proof of non-definability is based on a reduction from the boolean function HALF which outputs 1 if exactly half of the inputs have the value 1 and 0 otherwise. It can be verified that using HALF gates, one can compute MAJORITY by circuits of constant depth. So HALF is not in AC^0. The basic construction of the reduction is to use pairs of parallel line segments in the reduction which is similar to the one used in Theorem 5.3. An example of the reduction for $x_1x_2x_3x_4x_5x_6 = 1\ 0\ 1\ 0\ 1\ 1$ is shown in Figure 6. Now if HALF outputs 1, the parallel lines originating from $(0,0)$ go to just below the lowest teeth on the right. Hence a Eulerian traversal exists. Otherwise, if the lines from $(0,0)$ go too high (like the ones in Figure 6) or too low, the lines will be broken into at least two connected parts and Eulerian traversal is impossible. $\square$

Note that the Eulerian traversal query is however computable in PTIME.

In the next example, we consider a topological property. We say that two point sets in the k-dimensional rational space $\mathbb{Q}^k$ are *k-homeomorphic* if there is a *bi-continuous* bijection on $\mathbb{Q}^k$ which maps one to the other. The *k-dimensional homeomorphism* query is defined over database instances with two k-ary relations. The query returns "yes" if the two relations are k-homeomorphic and "no" otherwise.

Theorem 5.8 For each positive integer $k \geq 2$ and two arbitrary k-ary relation symbols R and S, the k-dimensional homeomorphism query over $\mathcal{L}_{\leq}$-representable $(\mathcal{Q}, \{R, S\})$-instances

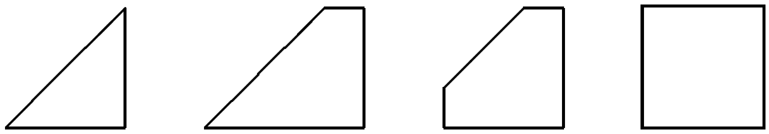


Figure 7: Some Possible Convex Regions

is not expressible in $\mathcal{L}_{\leq} \cup \{R, S\}$.

Proof: The proof uses a similar reduction as in the Eulerian traversal query. The primary differences are that (i) here we use closed areas instead of parallel line segments, and (ii) we need to construct the second relation which is a connected and closed area. $\square$

We end the discussion on $\mathcal{L}_{\leq}$ with a couple of definability results. Consider $\sigma = \{R\}$ where R is a binary relation symbol. The relation R defines a *convex* if for each pair of points in R, the straight line between them is entirely in R. Consider the function *convex*, mapping each $(\mathcal{Q}, \{R\})$-instance I to "yes" if $I(R)$ defines a convex and "no" otherwise. To see that the mapping *convex* is a query over dense order database instances, we note that for each instance I, the formula defining the relation R may contain only atomic formulas of the following forms (with free variables x, y):

$$x \leq c, \quad x \leq y, \quad y \leq c$$

and their negations, where c is a rational number. Thus, only horizontal, vertical lines and the line $x = y$ are representable and these lines are preserved under each automorphism of $\mathcal{Q}$ (applied to both dimensions simultaneously). It is easy to verify that the collection of convex instances is closed under automorphisms of $\mathcal{Q}$.

We now show that the convexity query is expressible in $\mathcal{L}_{\leq}$. Let I be a database instance of σ. Then, $I(R)$ is equivalent to a formula $\bigvee_i \bigwedge_j \psi_{i,j}$, where $\psi_{i,j}$ is either $s = t, s \leq t$, or their negations, and s, t are either constants in $\mathbb{Q}$ or variables. Hence each atomic formula represents either a line parallel to the horizontal or vertical axis or the diagonal line $x = y$, or the complement, or a half plane defined by a line of the above three types. It is easy to see that for $I(R)$ to define a single convex, the region it defines must be of one of a fixed, finite set of shapes. Several possible shapes in dimension 2 are shown in Figure 7. Now the query in $\mathcal{L}_{\leq} \cup \{R\}$ expressing the convexity query checks if the input corresponds to one of these known shapes and the construction is straightforward.

Finally, we consider the *k-convex covering* query which takes as input a binary relation representing possibly a set of regions and answers *true* if the input regions can be covered by k convexes, each of which is contained in the input regions, i.e., the input regions are covered exactly by the union of at most k convexes.

Theorem 5.9 Let R be a binary relation symbol. Then the following queries over $\mathcal{L}_\leq$-representable $(\mathcal{Q}, \{R\})$-instances are expressible in $\mathcal{L}_\leq \cup \{R\}$:

1. The convexity query.

2. For each $k \geq 0$, the k-convex covering query.

Proof: The proof for (1) follows from the above discussion. For (2), note that k is fixed. We can simply "try" all possible combinations of covering by k largest possible shapes from the set of fixed shapes from lower left corner to the upper right corner of the input. Due to the fact that there are only a fixed number of possible convex shapes representable in the databases, the query can pick a corner which violates convexity and then construct largest possible convexes to cover it. The process need to be repeated for at most k times. □

5.3 Inflationary DATALOG$^\neg$

In this section we study inflationary $DATALOG^\neg$, which was originally introduced in [KKR90] for dense order constraint databases. We prove that some of the queries not expressible in $\mathcal{L}_\leq$ discussed earlier are in fact definable in $DATALOG^\neg$. Recall that PTIME$_\mathcal{Q}$ is the set of all dense order queries computable in PTIME. We show that $DATALOG^\neg$ expresses exactly the set of all queries in PTIME$_\mathcal{Q}$. This extends the result that inflationary $DATALOG^\neg$ captures all relational queries in the context of finite databases [AV91].

Datalog with constraints is defined as follows. Constraints are allowed in the bodies of rules. For $DATALOG^\neg$, negations are allowed in the bodies of rules. The inflationary semantics is computed by adding after iteration the set of facts just derived to the set previously obtained. It has been shown in [KKR90] that the language can be evaluated bottom-up and in closed form, that is instances are mapped to instances. It is easy to see that under the restriction to $\mathcal{L}_\leq$-representable $(\mathcal{Q}, \sigma)$-instances, inflationary $DATALOG^\neg$ programs define queries over $\mathcal{Q}$, in a way analogous to Proposition 4.2.

Example 5.2 We now consider expressing the (2 dimensional) region connectivity query in $DATALOG^\neg$. The basic idea is to perform alternatively horizontal and vertical "sweeps". We first pick an arbitrary point (e.g., lowest and leftmost) within the input region and store it in a temporary relation S at the initial iteration step. For odd (respectively even) iterations, we extend S horizontally (respectively vertically) by adding points of R into S which are horizontally (respectively vertically) connected to S. The process ends when S stops growing. Finally, if S is the same as R, the algorithm stops and answers "the region is connected"; otherwise, it is "not connected". It is clear that this is expressible by a fixpoint query, and by a known translation technique (cf [AHV94]), it can be expressed in $DATALOG^{\neg}$. $\square$

Note that in the above example, the number of sweeps needed in the $DATALOG^{\neg}$ program is roughly the number of necessary "turns" the input relation contains, which depends on the number of constraints in the input.

Example 5.3 The Eulerian traversal query can be expressed in $DATALOG^{\neg}$ with several steps. First, all intersection points and end points are "detected" and stored in a temporary relation. Then we examine the direct connectivity among the points in the temporary relation. This gives us a set of "edges" when viewing the points in the temporary relation as "nodes". Finally, the existence of a Eulerian traversal can be determined by connectivity and the number of lines segments at each crossing, as if to test the existence of a Euler path in the context of a (finite) graph. $\square$

We now present an exact characterization of the expressive power of $DATALOG^{\neg}$.

Theorem 5.10 Inflationary $DATALOG^{\neg} = \mathrm{PTIME}_{\mathcal{Q}}$.

Remark: A similar statement can be found in [KKR90] and also in [KG94], but its meaning is different. What was proved there was that "inflationary $DATALOG^{\neg}$ can express any <u>relational</u> database query computable in PTIME" (Theorem 3.15 of [KKR90]). In other words, it was shown that

$$\begin{aligned} \mathrm{PTIME}_{\text{relational input/output}} &\subseteq \text{inflationary } DATALOG^{\neg} \\ &\subseteq \mathrm{PTIME}_{\text{dense-order input/input}} \\ &= \mathrm{PTIME}_{\mathcal{Q}}. \end{aligned}$$

In our result, $\mathrm{PTIME}_{\mathcal{Q}}$ denotes a set of queries over dense-order constraint databases, and not over relational databases as defined in [CH80]. $\square$

The proof of Theorem 5.10 relies on a syntactic normal form of the instances, called *maximal cover*. For simplicity, we consider only relations in dimension 2 and it will be clear how to generalize the following to relations of arbitrary arity. Suppose that R is some binary relation. The relation R defines a point set over $\mathbb{Q}^2$. It can be represented by a finite set of regions of one of a finite set of atomic shapes. In dimension 2, there are four types of atomic shapes: (i) *isolated point*, (ii) *line segment*, (iii) *triangle*, and (iv) *rectangle* (see Figure 8). Note that atomic shapes can also have a border at infinite (bi-infinite line, half-plan, etc.).

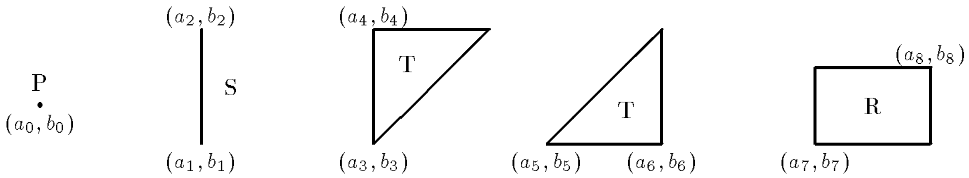


Figure 8: Atomic shapes in dimension 2

Definition 5.1 Let R be a k-ary finitely representable relation. An atomic shape S of dimension less or equal to k is *covered by* R if each point in S is also in R. S is *maximal in* R if it is covered by R but not by any other atomic shape covered by R. Finally, the set of all maximal atomic shapes in R is called the *maximal cover of* R.

It is easy to verify that the maximal cover of each finitely representable relation is unique and finite. Also, the cardinality of the maximal cover is bounded by the number of disjuncts (generalized tuples) in the finitely representable relation. The finite representation of a relation is in *normal form*, when each tuple denotes a maximal atomic shape. It is important to note that the objects of these particular shapes may be represented by four constants (in $\mathbb{Q} \cup \{-\infty, +\infty\}$) along with a flag indicating the shape. This lead to efficient encoding of dense-order constraint databases. An alternative encoding was proposed in [KG94].

Proof of Theorem 5.10: The inclusion of inflationary $DATALOG^{\neg}$ in PTIME$_{\mathbb{Q}}$ has been shown in [KKR90]: each query in $DATALOG^{\neg}$ expresses a query over dense order databases and can be computed in PTIME. We only prove the converse inclusion. For the sake of simplicity, we detail the proof in the case of a binary query over a database schema containing a single relation of arity 2. In other words, we work in a space of dimension 2. The technique carries over easily to higher dimensions.

Let Q be a query over dense order constraints computable in polynomial time and M be a Turing machine that computes Q. By definition, M operates on the standard encoding of each input database and produces a standard encoding of the answer to Q. The proof consists of the following steps:

1. Given a quantifier-free formula ϕ_I for an input instance I, there is a $DATALOG^{\neg}$ program, *Encode*, which "computes" a relational representation of the maximal cover of I, and its encoding on the input tape of the machine.

2. There is a $DATALOG^{\neg}$ program, *Compute*, which computes an encoding of the output starting from an encoding of I, i.e., the program simulates the computation of M.

3. There is a *DATALOG*$^\neg$ program, *Decode*, which computes a relational representation of the output O, from its encoding on the output tape of the machine.

We first define the *relational representation* of a database, which is a finite 5-ary relation representing the maximal cover of a quantifier-free formula (instance). Let I be a database instance. Recall that the maximal cover of the instance I consists of a set of triangles, rectangles, and possibly some degenerated shapes. The *relational representation* of I is a (classical) relation of fixed arity 5, where each tuple denotes an atomic shape as follows. Consider the rectangle between the points (a_7, b_7) and (a_8, b_8) (see Figure 8) defined by the conjunct: "$(a_7 < x < a_8) \wedge (b_7 < y < b_8)$". It is represented by a tuple: $[a_7, b_7, a_8, b_8, \text{R}]$, where R denotes the type of the object. Points, segments and triangles are represented similarly as follows for the shapes shown in Figure 8: $[a_0, b_0, a_0, b_0, \text{P}]$, $[a_1, b_1, a_2, b_2, \text{S}]$, $[a_3, b_3, a_3, b_4, \text{T}]$, $[a_5, b_5, a_6, b_5, \text{T}]$, since $a_3 = a_4$ and $b_5 = b_6$. The following can be easily verified.

Lemma 5.11 There is a *DATALOG*$^\neg$ program which, given a quantifier-free formula ϕ_I representing the input instance I, produces a finite 5-ary relation, $R_I \subset (\mathbb{Q} \cup \{-\infty, +\infty\})^5$, containing the relational representation of the maximal cover of I.

Proof: We describe the algorithm and give evidence that it can be done in *DATALOG*$^\neg$. Assume that the plane is oriented with North, South, West and East. Moreover, we assume the existence of some symbols representing $+\infty$ and $-\infty$. Rationals are represented by pairs of natural numbers as before. The algorithm uses an idea similar to that for expressing the k-convex covering query in $\mathcal{L}_\leq$. Essentially, obtaining the maximal cover is to find the least k such that the k-convex covering query answers "yes". Using the power of recursion or iteration provided by *DATALOG*$^\neg$, the modified algorithm is as follows:

1. Repeat (2)–(5) until $\phi_I - R_I = \emptyset$ (where $\phi_I - R_I$ denotes the infinite relation ϕ_I after removal of the triangles, rectangles, etc./ encoded in R_I):

2. Find the point (a_0, b_0) the most (i) in the south and (ii) in the west such that $\phi_I(a_0, b_0)$.

3. Find each point (a_1, b_1) the most (i) in the north and (ii) in the east, such that the rectangle defined by the two points is in ϕ_I.

4. Add the tuple $[a_0, b_o, a_1, b_1, \text{R}]$ to R_I.

5. Apply the previous two rules to each type of shape.

6. Delete from R_I the tuples that are covered by other tuples in R_I.

It is clear that $\phi_I - R_I$ as well as each individual rule, can be expressed in *DATALOG*$^\neg$. $\square$

The rest of the proof of the theorem involves techniques which have been used already in the literature, following the results of [Var82, Imm86]. Note that in I and its relational representation, the constants are rational numbers. These constants occurring in the relational representation of the input or in the query itself, are encoded into consecutive integers by respecting their order. Zero is zero. The smallest positive (respectively largest negative) constant occurring in the relation is 1 (respectively -1), and so on. The integers are translated in binary notation, and the whole relation is finally encoded on the Turing tape (i.e. a relation representing the tape, whose first attributes denote the indices over the alphabet of the initial relation, and whose last attribute contains the value of the cell over the alphabet $\sigma \cup \{\#, [,], 0, 1, \text{P}, \text{S}, \text{T}, \text{R}\}$, where σ is the database schema. $\square$

Theorem 5.12 The following queries are expressible in *DATALOG*$^\neg$:

1. For each $k \geq 2$, k-dimensional region connectivity;
2. For each $k \geq 2$, k-dimensional at least one-hole and exactly one-hole ;
3. Eulerian traversal.

It should be noted that it is open if the k-dimensional homeomorphism query (for $k \geq 2$) is expressible in *DATALOG*$^\neg$. However, there is an easy reduction from the graph isomorphism problem, and so if the k-dimensional homeomorphism query ($k \geq 2$) is expressible in *DATALOG*$^\neg$, then the graph isomorphism is in PTIME. The definability and non-definability of queries discussed in this section are summarized in Figure 9.

6 Polynomial Databases and Queries

In this section we briefly discuss polynomial databases and corresponding queries over $\mathcal{L}_\times$-representable $(\mathcal{R}, \sigma)$-instances. $\mathcal{R}$ is a model of the theory of real closed fields. The theory also admit quantifier elimination procedures [CK73]. The data complexity of queries over $(\mathcal{R}, \sigma)$-instances has been studied in [KKR90]. Queries queries in $\mathcal{L}_\times \cup \sigma$ over $\mathcal{L}_\times$-representable $(\mathcal{R}, \sigma)$-instances are computable in NC data complexity [BKR86, KKR90].

On the other hand, the only automorphism of $\mathcal{R}$ is the identity mapping. Hence, under Definition 4.2, if σ is a signature, then each partial recursive collection of $\mathcal{L}_\times$-representable $(\mathcal{R}, \sigma)$-instances is a boolean query over $\mathcal{R}$. In particular, the mappings line separation, grid checking, and same surface discussed in Example 5.1, which are not queries over $\mathcal{Q}$, are queries over $\mathcal{R}$.

Queries	*First-Order*	$DATALOG^{\neg}$
Convexity	Yes	Yes
Convex Covering	Yes	Yes
Region connectivity	No	Yes
At least one hole	No	Yes
Exactly one hole	No	Yes
Eulerian Traversal	No	Yes
Homeomorphism	No	? *
Parity	No	Yes
Graph connectivity	No	Yes

* A "Yes" answer implies that graph isomorphism is in PTIME.

Figure 9: Definability and non-definability of some queries

Example 6.1 Let $\sigma = \{R\}$ be a schema with a single binary relation. Recall that R defines a *grid* structure if it has a finite set of points and for each point (x, y) in R, there exist integers i_x, i_y such that $x = x_0 + i_x\Delta_x$ and $y = y_0 + i_y\Delta_y$ for some fixed $x_0, y_0, \Delta_x, \Delta_y$. It is easy to see that the query to check if an input is a grid structure is expressible in $\mathcal{L}_\times$. The finiteness can be expressed by stating that each point is "isolated". For example, the following formula φ states that whenever two numbers occur in R there is another number in-between them which does not occur in R, assuming $\delta(x)$ denotes x occurs in R:

$$\varphi \equiv \forall x \forall y \exists z \left(\delta(x) \wedge \delta(y) \wedge x < y \rightarrow x < z < y \wedge \neg\delta(z) \right) .$$

The formula φ can be used to ensure finiteness of the input relation. □

Obviously, the line separation query and the convexity query are also expressible in $\mathcal{L}_\times$. And it is easy to see that the *convex hull*, *Voronoi diagram*, and *Delaunay triangle* queries are expressible in $\mathcal{L}_\times$. For example, to express the Voronoi diagram query, suppose a binary relation S holds a finite set of points. The following formula constructs a 4-ary relation $V(x, y, u, v)$ such that the point (x, y) is in S and is the closest point in S to a point (u, v).

$$V(x, y, u, v) \equiv S(x, y) \wedge \forall x' \forall y' \left(S(x', y') \rightarrow \mathit{dist}(x, y, u, v) \leq \mathit{dist}(x', y', u, v) \right) ,$$

where *dist* is a term in $\mathcal{L}_\times$ which computes the distance between two points.

Now consider the *Euclidean travel salesman* query. It was shown to be NPTIME-complete [Yao90]. Since each query in $\mathcal{L}_\times$ can be evaluated in NC, it follows that it is not expressible in $\mathcal{L}_\times$ unless NC = NPTIME.

In the remainder of the section we study the definability of queries over $\mathcal{R}$ in $\mathcal{L}_{\leq}$, in other words, whether a polynomial query is definable using only the order $\leq$. In proving non-definability results of this sort, we present a technique based on an "invariance" argument: queries definable in $\mathcal{L}_{\leq}$ are closed under bijections preserving only the order relation.

Though the specific results concern only queries over $\mathcal{R}$, our presentation is in a more general setting. To begin with, we assume that $\mathcal{L}' \subseteq \mathcal{L}$ are two first-order languages and that $\mathcal{A}$ is an $\mathcal{L}$-structure with universe A. We are interested in automorphisms of $\mathcal{A}$ viewed as an $\mathcal{L}'$-structure and images of $\mathcal{L}$-representable databases instances under these automorphisms. However, we have to ensure that the automorphisms do not alter the finite representability of the databases. Again, we will only consider mappings that are $\mathcal{L}$-preserving with respect to the instances in question.

We next define the notion of *invariance* of queries. A one-to-one mapping from A onto A is an *$\mathcal{L}'$-automorphism* of $\mathcal{A}$ if it is an automorphism of the substructure $\mathcal{A}|_{\mathcal{L}'}$ of $\mathcal{A}$ over the language $\mathcal{L}'$. Thus, an $\emptyset$-automorphism is a bijection over the universe. A class K of $\mathcal{L}$-representable $(\mathcal{A}, \sigma)$-instances is said to be *weakly closed under $\mathcal{L}'$-automorphisms of $\mathcal{A}$*, if for each $\mathcal{L}$-representable $(\mathcal{A}, \sigma)$-instances I in K, and for each $\mathcal{L}'$-automorphism μ of $\mathcal{A}$ which is $\mathcal{L}$-preserving with respect to I, if $\mathcal{B}$ is an expansion of $\mathcal{A}$ such that I is a finite representation of $\mathcal{B}|_{\sigma}$, then each representation of $\mu(\mathcal{B}|_{\sigma})$ is also in K.

The weak closure under automorphisms may seem to be a complicated condition. The motivations are that (i) we cannot consider the full closure, since automorphisms of $\mathcal{A}$ may map finitely representable instances to non finitely representable instances, and (ii) we want to consider as good a closure as possible. This is needed to prove invariance properties (see below).

Definition 6.1 Let $\mathcal{A}$ be an $\mathcal{L}$-structure. A query K over $\mathcal{A}$ and schema σ is $(\mathcal{L} - \mathcal{L}')$-*invariant* if K is weakly closed under $\mathcal{L}'$-automorphisms of $\mathcal{A}$.

If φ is some sentence in $\mathcal{L}$, then $K_{\varphi}^{\mathcal{A}}$ is weakly closed under automorphisms of $\mathcal{A}$ and hence $\emptyset$-invariant, as shown in Proposition 4.2. More generally, there are very natural examples of queries definable in some language $\mathcal{L}$, which are $(\mathcal{L} - \mathcal{L}')$-invariant for some strict subset $\mathcal{L}' \subset \mathcal{L}$. This is the case of the query presented in ([AHV94], Exercise 17.27, page 162), which is expressed with order, but is order invariant although it cannot be expressed without the order relation, but only equality.

Proposition 6.1 Let $\mathcal{L}' \subseteq \mathcal{L}$ be two first-order languages, $\mathcal{A}$ an $\mathcal{L}$-structure. For each sentence φ in $\mathcal{L}' \cup \sigma$, $K_{\varphi}^{\mathcal{A}}$ is $(\mathcal{L} - \mathcal{L}')$-invariant.

The proof of the above proposition is based on the same technique as the proof of Proposition 4.2 and is omitted here.

We now consider the following geometric queries and prove their non definability using an argument based on invariance. Consider again a signature $\sigma = \{R\}$ where R is binary and the convexity query. Obviously, the convexity query is expressible in $\mathcal{L}_\times$. In Example 5.9, we show that the query is also expressible in $\mathcal{L}_\leq$ over dense order databases, We now show that the query is not expressible in $\mathcal{L}_\leq$ over $\mathcal{L}_\times$-representable $(\mathcal{R}, \sigma)$-instances, i.e., polynomial databases.

Proposition 6.2 The convexity query over $\mathcal{L}_\times$-representable $(\mathcal{R}, \sigma)$-instances is not expressible in $\mathcal{L}_\leq \cup \sigma$.

Proof: Let $\sigma = \{R\}$, where R is a binary relation symbol. Suppose that there is a sentence φ in $\mathcal{L}_\leq \cup \sigma$ that expresses the convexity query for $(\mathcal{R}, \sigma)$-instances. Let

$$a \geq \max \left\{ c \;\middle|\; c \text{ is a constant appearing in } \varphi \right\}.$$

Without loss of generality, we assume that $a \geq 1$. We construct an instance I such that

$$I(R) \equiv \Big((y + a = x) \wedge (a \leq x \leq a + 2) \Big).$$

Obviously, $I(R)$ consists of a single line segment which by definition is a convex. Consider the function f defined as:

$$f(z) = \begin{cases} z & \text{if } \; z \leq a + 1 \\ \dfrac{z + 1 - a}{2} + a & \text{if } \; z \geq a + 1 \end{cases}$$

Then f is $\mathcal{L}_\times$-preserving with respect to I. Note that $f(I(R))$ contains two connected line segments and is not a convex. On the other hand, f is an $\mathcal{L}_\leq$-automorphism, i.e., it preserves $\leq$. By Proposition 6.1, K_φ is closed under f and $f(I(R))$ should be in K_φ and also a convex, which leads to a contradiction. $\square$

Next we consider several non-boolean queries over finite relations as input. Let $\sigma = \{S\}$, where S is a binary relation (defining a finite set of points in $\mathbb{R}^2$) and $\tau = \{T\}$ where T is of arity 4. The *Euclidean spanning tree* query returns the relation T which defines the Euclidean spanning tree of the points in S. Since sentences in $\mathcal{L}_\leq$ cannot "check" distances, it can be shown that this query is not expressible in $\mathcal{L}_\leq$ over $(\mathcal{R}, \sigma)$-instances, nor over $(\mathcal{R}_\leq, \sigma)$-instances, by an argument similar to Proposition 6.2.

The *Voronoi diagram* query returns the relation V of arity 4, which associates to each point in S, its Voronoi region. Obviously, the last query may generate an $(\mathcal{R}, \tau)$-instance, when the input is an $(\mathcal{R}_\leq, \sigma)$-instance. The query is not closed under $\mathcal{R}_\leq$-instances. However, formulas in $\mathcal{L}_\leq$ only output $(\mathcal{R}_\leq, \tau)$-instances on $(\mathcal{R}_\leq, \sigma)$-instance inputs. Again, using a similar argument one can show that the query is not expressible for $(\mathcal{R}_\leq, \sigma)$-instances.

7 Conclusions

We introduced the theory of finitely representable models, and analyzed the differences with finite model theory and classical model theory. Most of the fundamental classical theorems (completeness and compactness) of model theory were shown to fail in the context of finitely representable structures. There is a lack of tools to investigate the properties of first-order logic over finitely representable models. Ehrenfeucht-Fraïssé games are among the few tools that still hold, but they result in complex and heavy combinatorics in presence of arithmetic functions. We believe that various proof techniques could be adapted to the case of finitely representable structures. This includes in particular, the locality in the context of dense orders, and results based on the asymptotic probabilities of sentences.

In the case of dense order constraint databases, we achieved a detailed study of the expressive power of first-order logic. We also gave a characterization of PTIME, in terms of $DATALOG^{\neg}$ over dense order constraint databases.

The case of dense order constraint databases is interesting for it resembles the paradigm of finite relations. Indeed, the results are quite similar, in particular in the complex theoretic sense. Nevertheless, dense order constraint databases offer a limited modeling power.

We presented just a few results in the case of polynomial constraint databases, for the sake of completeness and to compare the expressive power of polynomial constraints and dense order constraints. Polynomial constraints have a higher complexity upper bound (NC) which is rather tractable. On the other hand, there are still numerous open questions on polynomial constraint query languages. In particular, it is still open if the connectivity of a graph, or the parity of a relation, to take classical examples, are expressible. Partial results were already obtained in this direction. It was shown in [GST94], that in the case of linear constraints, that is when the language doesn't allow the multiplication symbol, then parity and connectivity are not definable. We conjecture that they are not expressible with polynomial constraints.

An other question of theoretical importance, is the capture of complexity classes. We claim that in the case of constraint databases, the class of polynomial time queries can easily be captured by a bounded fixpoint [Su93], to avoid the arbitrary explosion due to the iteration over addition for instance.

Acknowledgments

The authors wish to thanks Richard Lassaigne and Victor Vianu for helpful discussions.

References

[ACGK94] F. Afrati, S. Cosmadakis, S. Grumbach, and G. Kuper. Expressiveness of linear vs. polynomial constraints in database query languages. In *Proc. Workshop on the Principles and Practice of Constraint Programming*, LNCS, 1994.

[AHV94] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1994.

[Aj83] M. Ajtai. Σ_1^1 formulae on finite structures. Ann. of Pure and Applied Logic, Vol 24, pages 1–48, 1983.

[AV91] S. Abiteboul and V. Vianu. Datalog extensions for database queries and updates. *Journal of Computer and System Sciences*, 43(1):62–124, August 1991.

[BJM93] A. Brodsky, J. Jaffar, and M. J. Maher. Towards practical constraint databases. In *Proc. Int. Conf. on Very Large Data Bases*, pages 567–580, 1993.

[BKR86] M. Ben-Or, D. Kozen, and J. Reif. The complexity of elementary algebra and geometry. *Journal of Computer and System Sciences*, 32(2):251–264, April 1986.

[CH80] A. K. Chandra and D. Harel. Computable queries for relational data bases. *Journal of Computer and System Sciences*, 21(2):156–78, 1980.

[CK73] C.C. Chang and H.J. Keisler. *Model Theory*, volume 73 of *Studies in Logic*. North-Holland, 1973.

[Cod70] E.F. Codd. A relational model of data for large shared data banks. *Communications of ACM*, 13:6:377–387, 1970.

[Col75] G. E. Collins. Quantifier elimination for real closed fields by cylindric decompositions. In *Proc. 2nd GI Conf. Automata Theory and Formal Languages*, volume 35 of *Lecture Notes in Computer Science*, pages 134–83. Springer-Verlag, 1975.

[Com88] K. J. Compton. 0-1 laws in logic and combinatorics. In I. Rival, editor, *NATO Adv. Study Inst. on Algorithms and Order*, pages 353–383. D. Reidel, 1988.

[Dri82] L. Van den Dries. Remarks on Tarski's problem concerning $(\mathbb{R}, +, \times, exp)$. In *Logic Colloquium*, North-Holland, 1982. Elsevier.

[Ehr61] A. Ehrenfeucht. An application of games to the completeness problem for formalized theories. *Fund. Math*, 49, 1961.

[Fag76] R. Fagin. Probabilities on finite models. *Journal of Symbolic Logic*, 41(1):50–58, 1976.

[Fag93] R. Fagin. Finite model theory — a personal perspective. *Theoretical Computer Science*, 116(1):3–31, August 1993.

[Fra54] R. Fraïssé. Sur les classifications des systèmes de relations. *Publ. Sci. Univ Alger*, I:1, 1954.

[FSS84] M. Furst, J. B. Saxe, and M. Sipser. Parity, circuits, and the polynomial-time hierarchy. *Math. Syst. Theory*, 17:13–27, 1984.

[Gai81] H. Gaifman. On local and non local properties. In J. Stern, editor, *Proc. Herbrand Symposium Logic Colloquium*, pages 105–135. North Holland, 1981.

[GS94] S. Grumbach, J. Su. Finitely representable databases. In *Proc. 13th ACM Symp. on Principles of Database Systems*, pages 289–300, Minneapolis, 1994.

[GST94] S. Grumbach, J. Su, and C. Tollu. Linear constraint databases. In *Proc. LCC*, 1994. To appear in LNCS Spring-Verlag volume.

[HH93] T. Hirst and D. Harel. Completeness results of recursive data bases. In *Proc. 12th ACM Symp. on Principles of Database Systems*, pages 244–252, 1993.

[HH94] T. Hirst and D. Harel. Recursive model theory, 1994. Draft.

[Hul86] R. Hull. Relative information capacity of simple relational schemata. *SIAM Journal on Computing*, 15(3):856–886, 1986.

[Imm86] N. Immerman. Relational queries computable in polynomial time. *Inf. and Control*, 68:86–104, 1986.

[Imm87] N. Immerman. Languages that capture complexity classes. *SIAM Journal on Computing*, 16(4):760–778, 1987.

[KG94] P. C. Kanellakis and D. Q. Goldin. Constraint programming and database query languages. In *Proc. 2nd Conference on Theoretical Aspects of Computer Software (TACS)*, April 1994. (To appear in LNCS Spring-Verlag volume).

[KKR90] P. Kanellakis, G Kuper, and P. Revesz. Constraint query languages. In *Proc. 9th ACM Symp. on Principles of Database Systems*, pages 299–313, Nashville, 1990.

[KRVV93] P. Kanellakis, S. Ramaswamy, D. Vengroff, and J. Vitter. Indexing for data models with constraints and classes. In *Proc. ACM Symp. on Principles of Database Systems*, pages 233–243, 1993.

[Kup90] G. M. Kuper. On the expressive power of the relational calculus with arithmetic constraints. In *Proc. Int. Conf. on Database Theory*, pages 202–211, 1990.

[Kup93a] G. M. Kuper. Aggregation in constraint databases. In *Proc. Workshop on the Principles and Practice of Constraint Programming*, pages 176–183, April 1993.

[Kup93b] G. M. Kuper. Personal communications, 1993.

[PVV94] J. Paredaens, J. Van den Bussche, and D. Van Gucht. Towards a theory of spatial database queries. In *Proc. 13th ACM Symp. on Principles of Database Systems*, pages 279–88, 1994.

[Rev90] P. Revesz. A closed form for datalog queries with integer order. In *Proc. Int. Conf. on Database Theory*, pages 187–201, 1990.

[Su93] D. Suciu. Bounded fixpoints for complex objects. In *4th Int. Workshop on Database Programming Languages*, Morgan Kaufmann, 1993.

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

[Tra50] B. A. Trakhtenbrot. The impossibilty of an algorithm for the decision problem for finite models. *Doklady Akademii Nauk SSR*, 70:569–572, 1950.

[Var82] M. Vardi. Relational queries computable in polynomial time. In *Proc. 14th ACM Symp. on Theory of Computing*, pages 137–146, 1982.

[Vau60] R. L. Vaught. Sentences true in all constructive models. *Journal of Symbolic Logic*, 25(1):39–53, March 1960.

[VdB95] J. Van den Bussche. Personal communications, 1995.

[Via93] V. Vianu. Personal communications, 1993.

[Yao90] F. F. Yao. Computational geometry. In J. van Leeuwen, editor, *Handbook of Theorical Computer Science*, volume A, chapter 7, pages 343–389. North Holland, 1990.