Reachability and Connectivity Queries in Constraint Databases*

Michael Benedikt† Bell Laboratories
Martin Grohe‡ Univ. Freiburg
Leonid Libkin§ Bell Laboratories
Luc Segoufin¶ INRIA

Abstract

It is known that standard query languages for constraint databases lack the power to express connectivity properties. Such properties are important in the context of geographical databases, where one naturally wishes to ask queries about connectivity (what are the connected components of a given set?) or reachability (is there a path from A to B that lies entirely in a given region?). No existing constraint query languages that allow closed form evaluation can express these properties.

In the first part of the paper, we show that in principle there is no obstacle to getting closed languages that can express connectivity and reachability queries. In fact, we show that adding any topological property to standard languages like FO+LIN and FO+POLY results in a closed language. In the second part of the paper, we look for tractable closed languages for expressing reachability and connectivity queries. We introduce *path logic*, which allows one to state properties of paths with respect to given regions. We show that it is closed, has polynomial time data complexity for linear and polynomial constraints, and can express a large number of reachability properties beyond simple connectivity. Query evaluation in the logic involves obtaining a discrete abstraction of a continuous path, and model-checking of temporal formulae on the discrete structure.

1 Introduction

Several recent data models generalize the relational model by allowing direct modeling of structured database objects beyond the traditional flat tuple. Examples of the additional structure that can be modeled include nesting of tuples within other tuples, and the modeling of pointers and other datatypes in the object-oriented database model. We will deal in this paper with another such extension, the *constraint database model* [19, 23], in which database relations need not be simple finite collections of tuples, but can instead be constraint-definable collections, finite or infinite. The constraint model is appropriate for a variety of domains in which application data is naturally represented as solutions to constraints, such as geographic data and temporal data. Constraint databases allow queries to symbolically manipulate infinite collections of data, using both relational operators and the algebraic operations appropriate to the application domain.

We refer to the constraint model as the *polynomial constraint model* or the *linear constraint model*, depending on whether database objects are represented using general polynomial constraints or only linear constraints. More generally, we can parameterize the constraint model by any first-order structure $\mathcal{M}$ admitting a quantifier-elimination algorithm. In this general setting the 'constraint sets' that define database relations are simply the quantifier-free definable sets of $\mathcal{M}$.

Relational Calculus generalizes in an elegant and simple way to the constraint model. The analogous query languages use first-order logic (FO) over the vocabulary consisting of the schema relations supplemented with the operations of $\mathcal{M}$ (e.g., addition, multiplication). The implementation of the calculus reduces to constraint-solving, or in the general case, quantifier-elimination. Many of the results and techniques of classical relational calculus, including complexity and expressivity bounds, carry over to these first-order constraint query languages [23]. However, first-order constraint queries are limited in much the same way that Relational Calculus is limited. Natural recursive queries, such as the transitive closure of a graph, remain inexpressible in first-order constraint query lan-

*Part of this work was done while M. Benedikt, M. Grohe and L. Libkin visited INRIA-Rocquencourt.

†Bell Laboratories, 263 Shuman Blvd, Naperville, IL 60566, USA. E-mail: benedikt@research.bell-labs.com.

‡Institut für Mathematische Logik, Eckerstr. 1, 79104 Freiburg, Germany. E-mail: grohe@logik.mathematik.uni-freiburg.de.

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

¶INRIA-Rocquencourt, B.P. 105, Le Chesnay Cedex 78153, France. E-mail: Luc.Segoufin@inria.fr.

guages. Even more importantly, geometric analogs of these queries that are crucial for spatial database applications are inexpressible as well. The query

$$\texttt{CONNECTED}(S)$$

that tells whether a database relation S is topologically connected, is inexpressible. The query

$$\texttt{CONNECTS-TO}(x, y, S)$$

that tells whether there is a path from point x to point y within relation S is also inexpressible. Both of these results follow from [4]. The inability to express connectedness is a crucial obstacle in applying these languages to geographic databases. Although connectivity is perhaps the most natural geometric property that is absent from first-order constraint query languages, there are many other geometric properties which are conceptually (and even algorithmically) simple that are inexpressible as well: the query asking whether a planar region is simply connected, to take just one example. In fact, results of [20, 22] show, roughly, that the only purely topological facts of a single region expressible in first-order queries are 'local' – they merely assert the existence or nonexistence of points with a given topological type.

In this paper, we consider extensions of first-order constraint query languages that can express the reachability queries mentioned above, as well as other important non-local topological properties. Of course, in the context of the *relational* model, it is fairly well-understood how to add graph reachability as well as other tractable recursive queries to a first-order language; there are numerous results mapping out the query languages that result from such extensions, cf. [1]. What are the difficulties in extending beyond first-order logic in the constraint context?

When designing a query language one faces two major difficulties: achieving *closure* and *tractability*. A key idea behind constraint query languages is *closed-form evaluation*: if we start with databases definable over some structure $\mathcal{M}$ and apply an FO query, the result is again definable. In other words, the solution of a set of inequalities can be represented again as a set of inequalities. This closure property enables the use of a variety of inductive query-evaluation algorithms. We also want languages that are computationally *tractable*. The usual first-order constraint query languages have polynomial data complexity, as do standard languages for recursive querying of traditional relational databases.

Adding a traditional relational recursion mechanisms such as fixed points or while loops to first-order constraint query languages does *not* give this closure property, see [19]. In fact, the interaction of arithmetic with recursion can produce output databases that are not even computable, much less definable with constraints. In fact, queries for topological connectivity have been handled only through query languages that are both non-closed and computationally intractable [21, 13].

In contrast, our first main result shows that there is a way to add reachability and a vast number of other topological queries while retaining closure. In fact, for any set $\mathbb{T}$ of topological properties, we present a language, $\mathrm{FO} + \mathbb{T}$ that can define $\mathbb{T}$, and that is still closed. Moreover we can prove that the complexity of evaluating queries in $\mathrm{FO} + \mathbb{T}$ is polynomial in the complexity of checking the properties from $\mathbb{T}$. This implies a PTIME complexity for the extension of CONNECTED. The language $\mathrm{FO} + \mathbb{T}$ demonstrates that the closure problem by itself is no obstacle to admitting spatial reachability queries into constraint query languages.

Our second main result identifies a powerful logic, *Path Logic* – denoted L_{PATH} – which can define CONNECTED, CONNECTS-TO and other reachability queries, and which also admits effective, tractable query evaluation over linear and polynomial constraint databases. Path Logic has syntax and semantics that are reminiscent of traditional temporal logics used in the verification of reactive programs. Not only can query evaluation be done in polynomial data complexity, but we show that query evaluation reduces to a combination of *cell-decomposition algorithms* from real analytic geometry, followed by *model-checking* of a discrete system. In the model-checking phase, techniques from verification of discrete transition systems can be applied.

We will show positive results about both the tractability and expressivity of Path Logic, making the case that it is sufficiently expressive to capture the recursive queries that are most important to spatial database applications. Path Logic can thus be seen as a constraint database language that generalizes many recursive extensions of first-order logic, as well as a spatial analog of temporal logics such as CTL and CTL*.

Related work Most work on Datalog extensions for the constraint model deal with highly restrictive classes of constraints over integers [29], as over linear and polynomial constraints Datalog is not closed. Topological connectivity for 2-dimensional polynomial constraint databases of degree 2 was shown to be definable in a language called Spatial Datalog [21]; later [13] extended this to arbitrary polynomial constraint databases. However, Spatial Datalog is likely to be computationally complete, and it certainly does not admit efficient evaluation. Results in computational algebraic geometry show that many connectivity and reachability queries (e.g., find the connected components of a set) are in PTIME if the dimension is fixed (see, e.g., [16]). However, languages capturing complexity classes over constraint databases are scarce (natural languages exist for databases definable with order constraints only [14]; also, a rather complicated language was given for linear constraints [15]). Besides, this approach can only work for those queries applied as top level operators (i.e., outputs cannot be reused by other queries). There exists extensive literature on first-order definable topological properties of constraint databases [26, 20, 22, 31] which connectivity and reachability are not first-order. Our results on application to hybrid systems are directly inspired by [24, 25].

Organization We introduce notations in Section 2. In Section 3, we deal with closure under topological properties. We first prove a number of general decomposition results for sets definable in various structures. We then use them to show closure of FO+LIN and FO+POLY under adding tests for topological properties, and tractability, assuming topological properties can be tested in polynomial time. We also study the special case of finding connected components.

In Section 4, we introduce Path Logic L_{PATH} for expressing reachability and connectivity queries in a unified framework. We give examples, analyze expressive power, and prove closure. For linear and polynomial constraints we show tractability as well. We conclude by an application to verification of hybrid systems. All proofs are in the appendix.

2 Notations

Structures, databases, queries Most notations are fairly standard in the literature on constraint databases, cf. [4, 5, 23, 27]. Let $\mathcal{M} = \langle \mathcal{U}, \Omega \rangle$ be an infinite structure, where $\mathcal{U}$ is an infinite set, called a universe (in the database literature often called the domain), and Ω is a set of interpreted functions, constants, and predicates. In the field of constraint databases, most examples have $\mathcal{U} = \mathbb{R}$, the set of real numbers. Examples of signatures (and corresponding classes of constraints) that have been considered are:

Dense Order Constraints:	$\langle \mathbb{R}, < \rangle$;
Linear Constraints:	$\mathbf{R}_{\text{lin}} = \langle \mathbb{R}, +, -, 0, 1, < \rangle$;
Polynomial Constraints:	$\mathbf{R} = \langle \mathbb{R}, +, \cdot, 0, 1, < \rangle$.

A (relational) *database schema SC* is a nonempty collection of relation names $\{S_1, \ldots, S_l\}$ with associated arities $p_1, \ldots, p_l > 0$. We shall consider *finitely representable*, or *definable* instances. A definable database instance of SC over $\mathcal{M}$ is a family of sets $\{X_1, \ldots, X_l\}$, with $X_i \subseteq \mathcal{U}^{p_i}$, such that for each X_i there exists a formula $\alpha_i(x_1, \ldots, x_{p_i})$ in the language of $\mathcal{M}$ with $X_i = \{\vec{a} \in \mathcal{U}^{p_i} \mid \mathcal{M} \models \alpha_i(\vec{a})\}$. Most applications of constraint databases consider definable instances over $\mathbf{R}_{\text{lin}}$ (called *semi-linear sets*) or over $\mathbf{R}$ (called *semi-algebraic sets*). These are sets definable by Boolean combinations of linear (resp., polynomial) inequalities.

As our basic query language, we consider relational calculus, or *first-order logic*, FO, over the underlying structure and the database schema. We use te notation FO + Ω to denote the class of all first-order formulae built up from the atomic SC and Ω formulae by using Boolean connectives $\vee, \wedge, \neg$ and quantifiers $\forall, \exists$. When Ω is $(+, -, 0, 1, <)$, we use the notation FO + LIN (*first-order with linear constraints*), and when Ω is $(+, \cdot, 0, 1, <)$, we denote the language by FO + POLY (*first-order with polynomial constraints*).

Given $\varphi(\vec{x}, \vec{y})$ and $\vec{a}$, we write $\varphi(\vec{a}, D)$ for $\{\vec{b} \mid D \models \varphi(\vec{a}, \vec{b})\}$; in the absence of $\vec{x}$ we just write $\varphi(D)$ for the output of φ on D. We say that a language FO + Ω is *closed*, if for any schema SC, any definable SC-database (over $\langle \mathcal{U}, \Omega \rangle$) and every FO + Ω query $\varphi(\vec{y})$ on SC-databases, the output $\varphi(D)$ is a definable set.

Languages FO+LIN, FO+POLY, as well as FO with dense order constraints are closed; this is a consequence of quantifier-elimination for $\mathbf{R}_{\text{lin}}$, $\mathbf{R}$ and $\langle \mathbb{R}, < \rangle$.

O-minimality, cell decomposition Many results that we prove extend beyond linear and polynomial constraints. To state them in greater generality, we use *o-minimality* [33], which plays an important role in the study of constraint query languages (cf. [4, 5, 23]).

A structure $\mathcal{M} = \langle \mathcal{U}, \Omega \rangle$ is o-minimal, if every definable subset of $\mathcal{U}$ is a finite union of points and open intervals $(a, b) = \{x \mid a < x < b\}$, $(-\infty, a) = \{x \mid x < a\}$, and $(a, \infty) = \{x \mid x > a\}$ (we assume that $<$ is in Ω). All the structures on the reals we mentioned so far – $\mathbf{R}_{\text{lin}}$, $\mathbf{R}$, $\langle \mathbb{R}, < \rangle$ – are o-minimal (this is implied by quantifier elimination and the fundamental theorem of algebra, for the case of $\mathbf{R}$.) There are a number of known o-minimal expansions of $\mathbf{R}$, most notably, the exponential field $\langle \mathbb{R}, +, \cdot, e^x \rangle$ [35].

A key property of o-minimal structures is *cell decomposition*. A *cell* in $\mathbb{R}^k$ is a subset homeomorphic to $\mathbb{R}^{k'}$, $k' \leq k$ (by convention, $\mathbb{R}^0$ is a point). We now fix a structure $\mathcal{M} = \langle \mathbb{R}, \Omega \rangle$ and define $\mathcal{M}$-cells by induction on dimension. An $\mathcal{M}$-cell in $\mathbb{R}^0$ is just $\mathbb{R}^0$. $\mathcal{M}$-cells in $\mathbb{R}$ are singletons $\{a\}$, or open intervals $(a, b), (-\infty, a), (a, \infty)$, where a, b are definable constants. Assume that $C \subseteq \mathbb{R}^{n-1}$ is a cell, and

$f, g : C \to \mathbb{R}$ are continuous definable function on C, with $f(x) < g(x)$ for all $x \in C$. Then the sets $\{(x, f(x)) \mid x \in C\}$ and $\{(x, r) \mid x \in C, f(x) < r < g(x)\}$ are cells in $\mathbb{R}^n$. In the latter case, we allow f to be $-\infty$ and/or g to be ∞.

A cell decomposition of $\mathbb{R}^n$ (with respect to $\mathcal{M}$) is a partition of $\mathbb{R}^n$ into a finite union of $\mathcal{M}$-cells. Again, it is defined inductively on n. A decomposition of $\mathbb{R}$ is the collection of $\mathcal{M}$-cells of the form $\{(-\infty, a_1), \{a_1\}, (a_1, a_2), \{a_2\}, \ldots, \{a_k\}, (a_k, \infty)\}$. A decomposition of $\mathbb{R}^n$ is a family $\mathcal{C} = \{C_1, \ldots, C_l\}$ of $\mathcal{M}$-cells that partition $\mathbb{R}^n$ such that for the natural projection $\pi : \mathbb{R}^n \to \mathbb{R}^{n-1}$ given by $\pi(\vec{x}, t) = \vec{x}$ for $\vec{x} \in \mathbb{R}^n, t \in \mathbb{R}$, the collection $\pi(\mathcal{C}) = \{\pi(C_1), \ldots, \pi(C_k)\}$ is a decomposition of $\mathbb{R}^{n-1}$.

In particular, if $\mathcal{C}$ is a decomposition of $\mathbb{R}^{n+m}$ and $\pi : \mathbb{R}^{n+m} \to \mathbb{R}^n$ is the projection on the first n coordinates, then $\pi(\mathcal{C})$ is a decomposition of $\mathbb{R}^n$, and for every cell $C \in \mathcal{C}$ there exists a unique cell $C_0 \in \pi(\mathcal{C})$ such that $C \subseteq C_0 \times \mathbb{R}^m$. This property, especially in the context of the real field, is referred to as being a *cylindric* decomposition.

Fact 1 [33] *Let $\mathcal{M} = \langle \mathbb{R}, \Omega \rangle$ be o-minimal. Assume that $S_1, \ldots, S_m$ are definable sets in $\mathbb{R}^n$. Then there exists a cell decomposition $\mathcal{C}$ of $\mathbb{R}^n$ such that each S_i is a union of some cells of $\mathcal{C}$.* $\square$

Let $X \subseteq \mathbb{R}^{n+m}$ and $\vec{a} \in \mathbb{R}^n$. Then $X_{\vec{a}}$ denotes the *fiber* $\{\vec{b} \in \mathbb{R}^m \mid (\vec{a}, \vec{b}) \in X\}$. Let $\mathcal{C}$ be a decomposition of $\mathbb{R}^{n+m}$ and $\vec{a} \in \mathbb{R}^n$. By $\mathcal{C}_{\vec{a}}$ we denote the collection of all nonempty sets $\{C_{\vec{a}} \mid C \in \mathcal{C}\}$. It is known [33] that $\mathcal{C}_{\vec{a}}$ is a cell decomposition of $\mathbb{R}^m$.

In the case when $\mathcal{M}$ is $\mathbf{R}_{\text{lin}}$ or $\mathbf{R}$, we can get more information about cell decompositions. Namely, given any finite collection $f_1(\vec{x}), \ldots, f_k(\vec{x})$ of polynomials (resp., linear functions) in n variables with coefficients from $\mathbb{Q}$, one can find a cell decomposition $\mathcal{C}$ of $\mathbb{R}^n$ such that on each cell C_i, none of the functions f_j changes its sign. Furthermore, this decomposition can be found in time $O((kd)^{h(n)})$, where d is the maximal degree of a polynomial among f_js, and h is some function (typically, $h(n) = O(2^n)$) [11, 9]. It is important to notice that for a fixed dimension, the cell decomposition algorithm is thus in PTIME (in fact, in NC [6]).

3 Topological properties and closure

The goal of this section is to show that adding topological properties to languages like FO + Lin and FO+Poly results in *closed* query languages. In particular, one can add the connectivity test, or an operator computing connected components of a set, and still remain within semi-linear or semi-algebraic databases.

In Subsection 3.1, we prove a general decomposition result for definable sets that is key to the closure theorems of this section and next. Our starting point is the Local Triviality Theorem in real algebraic geometry, which implies, for example, that for a semi-algebraic set $S \subseteq \mathbb{R}^{n+m}$, the number of topological types of sets $S_{\vec{a}} \subseteq \mathbb{R}^m$ is finite, as $\vec{a}$ ranges over $\mathbb{R}^n$ [3, 7]. For $\mathbf{R}$, the known Local Triviality theorem gives the decompositions necessary for our results. In $\mathbf{R}_{\text{lin}}$, however, the Local Triviality Theorem fails, so we prove a weakening of it that is sufficient for the needs of the paper.

Once the decomposition lemma is proved, the general closure result for adding tests of topological properties follows easily. We treat it in Subsection 3.2, and then in Subsection 3.3 analyze more general topological operators, in particular, one for computing connected components of a set. We derive closure for FO+Poly and FO+Lin, although in very different ways: for FO+Poly it is an easy consequence of Local Triviality for semi-algebraic functions, while for FO+Lin the proof relies on the special form of decompositions.

3.1 Decomposition Lemma

We need the notion of decomposition into cells that is more general than the notion introduced in Section 2. Recall that a cell is a definable set homeomorphic to $\mathbb{R}^p$ for some $p \geq 0$. By a *decomposition* of $\mathbb{R}^k$ we mean a finite partition of $\mathbb{R}^k$ into cells $A_1, \ldots, A_l$. We consider special kinds of decompositions *of* $\mathbb{R}^{n+m}$ *over* $\mathbb{R}^n$. A decomposition $\mathcal{C}$ is *cylindric* over $\mathbb{R}^n$ if for every $A_i, A_j \in \mathcal{C}$, $\pi(A_i)$ and $\pi(A_j)$ are cells and either $\pi(A_i) = \pi(A_j)$, or $\pi(A_i) \cap \pi(A_j) = \emptyset$. Here π is the projection $\mathbb{R}^{n+m} \to \mathbb{R}^n$. Note that the nonempty sets among $\pi(A_i)$s induce a decomposition on $\mathbb{R}^n$ which we denote by $\pi(\mathcal{C})$.

A decomposition $\mathcal{C}$ of $\mathbb{R}^{n+m}$ is called *trivial* over $\mathbb{R}^n$ if it is cylindric over $\mathbb{R}^n$ and for any cell C' of the induced decomposition $\pi(\mathcal{C})$ of $\mathbb{R}^n$, and for any $\vec{a}, \vec{b} \in C'$, there exists a homeomorphism $h : \mathbb{R}^m \to \mathbb{R}^m$ such that $h(C_{\vec{a}}) = C_{\vec{b}}$ for every cell $C \in \mathcal{C}$.

Let $\text{cl}(\cdot)$ denote the closure of a set (in the usual topology of $\mathbb{R}$). A decomposition $\mathcal{C}$ is called *adjacency preserving* over $\mathbb{R}^n$ if it is cylindric over $\mathbb{R}^n$ and for any cell C' of the induced decomposition $\pi(\mathcal{C})$, for any $\vec{a}, \vec{b} \in C'$, and for all cells $A^1, A^2, A^3 \in \mathcal{C}$, $\text{cl}(A^1_{\vec{a}}) \cap A^2_{\vec{a}} \neq \emptyset$ iff $\text{cl}(A^1_{\vec{b}}) \cap A^2_{\vec{b}} \neq \emptyset$ and $\text{cl}(A^1_{\vec{a}}) \cap \text{cl}(A^2_{\vec{a}}) \cap A^3_{\vec{a}} \neq \emptyset$ iff

$\mathrm{cl}(A^1_{\vec{b}}) \cap \mathrm{cl}(A^2_{\vec{b}}) \cap A^3_{\vec{b}} \neq \emptyset$. Note that a trivial decomposition is adjacency preserving.

Lemma 1 *a) Let $S_1, \ldots, S_k$ be a collection of semi-linear sets in $\mathbb{R}^{n+m}$ and let $f_1, \ldots, f_p$ be all the (degree 1) polynomials used in the representation of $S_1, \ldots, S_k$. Then there exists a decomposition $\mathcal{C}$ of $\mathbb{R}^{n+m}$ into semi-linear sets which is trivial over $\mathbb{R}^n$, and such that the sign of each f_i is constant on every cell of $\mathcal{C}$. In particular, every S_j is a union of cells of $\mathcal{C}$. Moreover, for n and m fixed, $\mathcal{C}$ can be found in time polynomial in the size of the descriptions of $f_1, \ldots, f_p$.*

b) The statement a) holds if one replaces semi-linear with semi-algebraic.

c) Let $\mathcal{M} = \langle \mathbb{R}, \Omega \rangle$ be o-minimal, and let $S_1, \ldots, S_k$ be a collection of definable sets in $\mathbb{R}^{n+m}$. Then there exists a decomposition $\mathcal{C}$ of $\mathbb{R}^{n+m}$ which is adjacency preserving over $\mathbb{R}^n$ such that each set S_i is a union of cells of $\mathcal{C}$. Moreover, if $\mathcal{M}$ is decidable, then $\mathcal{C}$ can be effectively computed.

d) If $\mathcal{M} = \langle \mathbb{R}, \Omega \rangle$ is an o-minimal expansion of the real field $\mathbf{R}$, and $S_1, \ldots, S_k$ is a collection of definable sets in $\mathbb{R}^{n+m}$, then there exists a decomposition $\mathcal{C}$ of $\mathbb{R}^{n+m}$ which is trivial and such that each set S_i is a union of cells of $\mathcal{C}$.

Proof sketch. The proof of b) and d) is by Local Triviality for the real field [3, 7] and o-minimal expansions [33]. As it fails over $\mathbf{R}_{\mathrm{lin}}$, the proof of a) is by combining standard cell decomposition techniques with results from convex analysis on the topology of polyhedra. The proof of c) is by cell decomposition and uniform bounds of [28]. See the appendix for details. $\square$

3.2 Closure theorem

We now prove the closure result for topological properties. Formally, a *topological property* Top is a collection $\{\mathcal{T}_1, \ldots, \mathcal{T}_n, \ldots\}$ where $\mathcal{T}_n$ is a family of sets in $\mathbb{R}^n$ such that if $X \in \mathcal{T}_n$, then for each homeomorphism h of $\mathbb{R}^n$, $h(X) \in \mathcal{T}_n$. For example, Top could express the property of being connected, being closed, being of dimension $n-1$, containing exactly one hole, etc. When the dimension n is clear from the context, we write $X \in \mathsf{Top}$ instead of $X \in \mathcal{T}_n$.

For a set $\mathbb{T}$ of topological properties, we define the language $\mathrm{FO}(\Omega) + \mathbb{T}$ by extending the definition of $\mathrm{FO}(\Omega)$ with the following rule: if $\varphi(\vec{x}, \vec{y})$ is a query, then $\psi(\vec{x}) \equiv \mathsf{Top}\,\vec{y}.\ \varphi(\vec{x}, \vec{y})$ is a query. The semantic is as follows: $D \models \psi(\vec{a})$ iff $\varphi(\vec{a}, D) \in \mathsf{Top}$. Recall that $\varphi(\vec{a}, D) = \{\vec{b} \mid D \models \varphi(\vec{a}, \vec{b})\}$. For Ω being $(+, -, 0, 1, <)$ or $(+, \cdot, 0, 1, <)$ we use the notation $\mathrm{FO} + \mathrm{LIN} + \mathbb{T}$ and $\mathrm{FO} + \mathrm{POLY} + \mathbb{T}$.

For instance, the query "is the intersection of regions R and S connected" could be written as $\mathsf{C}\vec{x}.\ R(\vec{x}) \wedge S(\vec{x})$ (where we denote the property of being connected by C). To illustrate the use of free variables, consider a set $S \subseteq \mathbb{R}^3$. Then the query $\varphi(x) \equiv \mathsf{C}(y, z).\ S(x, y, z)$ returns the set of all $c \in \mathbb{R}$ for which the intersection of S with the plane $x = c$ is a connected set.

We say that the data complexity of $\mathrm{FO}(\Omega) + \mathbb{T}$ is $\mathrm{PTIME}^{\mathbb{T}}$ if $\mathrm{FO}(\Omega) + \mathbb{T}$ queries can be evaluated in polynomial time in the size of the database, assuming an oracle that can test each $\mathsf{Top} \in \mathbb{T}$ in constant time.

Theorem 1 *Let $\mathbb{T}$ be any set of topological properties. Then $\mathrm{FO} + \mathrm{LIN} + \mathbb{T}, \mathrm{FO} + \mathrm{POLY} + \mathbb{T}$ and $\mathrm{FO}(\Omega) + \mathbb{T}$ are closed, for $\langle \mathbb{R}, \Omega \rangle$ an o-minimal expansion of the real field. Furthermore, the data complexity of $\mathrm{FO} + \mathrm{LIN} + \mathbb{T}$ and $\mathrm{FO} + \mathrm{POLY} + \mathbb{T}$ is $\mathrm{PTIME}^{\mathbb{T}}$.*

Proof sketch. Closure is proved by induction on the formulae. Consider $\psi(\vec{x}) \equiv \mathsf{Top}\,\vec{y}.\ \varphi(\vec{x}, \vec{y})$ for $\mathsf{Top} \in \mathbb{T}$. Let $\vec{x}$ and $\vec{y}$ be of length n and m, resp., and let D be a definable database. By induction, $\varphi(D)$ is a definable set $S \subseteq \mathbb{R}^n \times \mathbb{R}^m$. By Lemma 1, there is a decomposition $\mathcal{C}$ of $\mathbb{R}^{n+m}$ trivial over $\mathbb{R}^n$ such that S is a union of cells of $\mathcal{C}$. Let $\mathcal{C}'$ be the projection of $\mathcal{C}$ onto $\mathbb{R}^n$, and C a cell in $\mathcal{C}'$. By triviality, for every $\vec{a}, \vec{b} \in C$, it is the case that $S_{\vec{a}}$ and $S_{\vec{b}}$ are homeomorphic, and thus they agree on Top. Therefore, the output of ψ on D is a union of some cells in $\mathcal{C}'$ and thus definable. $\square$

Now consider the case when $\mathbb{T}$ consists of just the property C (being a connected set). As connectivity of semi-algebraic sets can be tested in polynomial time (for a fixed dimension) [16], the proof of the complexity bounds in Theorem 1 implies the following.

Corollary 1 $\mathrm{FO} + \mathrm{LIN} + \mathsf{C}$ *and* $\mathrm{FO} + \mathrm{POLY} + \mathsf{C}$ *are closed, and the queries they define have* PTIME *data complexity.* $\square$

3.3 Topological queries and connected components

So far we have seen closure and tractability for languages which add the `CONNECTED` operator mentioned in the introduction. We now deal with the `CONNECTS-TO` operator; that is, with computing connected components. In fact, we treat a more general case of non-boolean topological queries in the context of polynomial constraints.

Let T be a map from subsets of $\mathbb{R}^m$ to subsets of in $\mathbb{R}^{mk}$. We call T *topological* if for any homeomorphism $h : \mathbb{R}^m \to \mathbb{R}^m$ and any $(\vec{x}_1, \ldots, \vec{x}_k) \in T(S)$, for $S \subseteq \mathbb{R}^m$, we have $(h(\vec{x}_1), \ldots, h(\vec{x}_k)) \in T(h(S))$. For example, the mapping $\mathsf{Conn} : \mathbb{R}^m \to \mathbb{R}^m \times \mathbb{R}^m$ such that $(\vec{x}_1, \vec{x}_2) \in \mathsf{Conn}(S)$ iff $\vec{x}_1, \vec{x}_2$ are in the same connected component of S, is topological. We say that T is *definable* if $T(S)$ is a definable set for every definable set S. As connected components of a set definable in an o-minimal structure on $\mathbb{R}$ are definable [33], the topological query Conn is definable over such structures.

Let $T : 2^{\mathbb{R}^m} \to 2^{\mathbb{R}^{mk}}$ be a topological query. We define the language $\mathrm{FO}(\Omega) + T$ by extending the definition of $\mathrm{FO}(\Omega)$ with the following rule: if $\varphi(\vec{x}, \vec{y})$ is a query with $\vec{y}$ having length m, then we get a new query $\psi(\vec{x}, \vec{y}_1, \ldots, \vec{y}_k) \equiv T\vec{y}.\ \varphi(\vec{x}, \vec{y})$, with all $\vec{y}_i$s having length m. The semantic is as follows:

$$D \models \psi(\vec{a}, \vec{b}_1, \ldots, \vec{b}_k) \text{ iff } (\vec{b}_1, \ldots, \vec{b}_k) \in T(\varphi(\vec{a}, D)).$$

For example, if $\varphi(D)$, for $\varphi(\vec{x}, \vec{y})$, is a set $S \subseteq \mathbb{R}^{n+m}$, and $\psi(\vec{x}, \vec{y}_1, \vec{y}_2) \equiv \mathsf{Conn}\vec{y}.\ \varphi$, then $D \models \psi(\vec{a}, \vec{b}_1, \vec{b}_2)$ iff $\vec{b}_1$ and $\vec{b}_2$ are in the same connected component of $S_{\vec{a}}$.

In the semi-algebraic case or in any o-minimal expansion, the Local Triviality Theorem used in the proof of Lemma 1 can also be used to prove the following:

Theorem 2 *For every definable topological map T,* $\mathrm{FO} + \mathrm{POLY} + T$ *is closed. Moreover,* $\mathrm{FO}(\Omega) + T$ *is closed, whenever $\langle \mathbb{R}, \Omega \rangle$ is an o-minimal expansion of the real field.* □

We note in passing that the fact that T produces a definable output on a definable input by no means implies closure. For example, the convex hull operator preserves semi-linearity, but when added to FO+LIN, gives it the full power of FO+POLY [2]. One can find definable operators that, when added to FO+POLY, define non-semi-algebraic sets (e.g., given two sets X and Y in $\mathbb{R}^n$, return the one with the larger volume).

Since Conn is definable and topological (over the real field and its o-minimal expansions), we conclude from Theorem 2 that FO + POLY + Conn is closed. However, the proof above cannot possibly be extended to FO + LIN. Indeed, we used not only the triviality of the partition which is guaranteed by Lemma 1, but also the fact that homeomorphisms between fibers are definable. This latter condition fails over $\langle \mathbb{R}, +, -, 0, 1, < \rangle$. Nevertheless, we can show that FO + LIN + Conn is closed.

Proposition 1 FO + LIN + Conn *is closed; that is, it defines a semi-linear output on a semi-linear input. Furthermore,* FO + LIN + Conn *queries have* PTIME *data complexity.*

Proof is by using the special structure of decompositions produced by Lemma 1 for semi-linear sets. □

We note that the results on closure under topological operators and Conn leave something to be desired. First, the proof of Theorem 2 does not produce a complexity bound, as it is not immediately clear how hard it is to compute definable homeomorphisms between fibers. The proof for FO+LIN, although giving us tractability, is rather ad-hoc, and slight modification of a query may require an entirely different proof of the closure.

We now want to find a single language that captures the properties that are of interest for applications, which has a small number of constructors, and which has a uniform evaluation method over all queries. Such a language is presented in the next section.

4 Path Logic

Our goal is to present a unifying query language for expressing reachability and connectivity queries. The language is based on the concept of a path. One can then state properties of path with respect to given sets in the Euclidean space $\mathbb{R}^n$. For example, to test if two points in a given set S belong to the same connected component, we write a query $\varphi(\vec{x}, \vec{y})$ (simplifying the syntax a bit) as the conjunction of $S(\vec{x}) \wedge S(\vec{y})$ and

$$\exists \mathbf{P}[\exists p_1 \exists p_2\ (p_1 = \vec{x} \wedge p_2 = \vec{y} \wedge (\forall p\ S(p)))].$$

It says that there exists a path $\mathbf{P}$ and two points p_1, p_2 on this path such that p_1 is $\vec{x}$, p_2 is $\vec{y}$ and every point p on the path belongs to S. Similarly, one writes $\forall \vec{x} \forall \vec{y}\ (S(\vec{x}) \wedge S(\vec{y}) \to \exists \mathbf{P}[\exists p_1 \exists p_2\ (p_1 = \vec{x} \wedge p_2 = \vec{y} \wedge (\forall p\ S(p)))])$ to test if S is connected.

In Subsection 4.1, we formally define the path logic L_{PATH}. In Subsection 4.2, we give more examples and analyze the expressive power. In Subsection 4.3, we show that L_{PATH} is closed and tractable over polynomial and linear constraints (more generally, the closure is shown for o-minimal structures). Finally, Subsection 4.4 gives applications to hybrid systems.

4.1 Definition of the path logic

The logic will be two-sorted, having element variables $x, y, \ldots$ and *path variables* $p, q \ldots$. Each path variable p has an associated arity, $arity(p) > 0$. There are also two kinds of formulae: *state formulae* and *path formulae.* A state formula can only have free element variables.

We now define the syntax, together with three predicates: $\mathrm{FV_s}(\cdot)$ for free state variables, $\mathrm{FV_p}(\cdot)$ for free path variables, and $\mathrm{OCC}(\cdot)$ for the set of currently used path variables (those could be either free or bound). For every state formula φ, $\mathrm{OCC}(\varphi) = \emptyset$. Given a database schema SC, and a structure $\mathcal{M} = \langle \mathbb{R}, \Omega \rangle$, formulae of $L_{\mathrm{PATH}}(\Omega)$ are defined inductively as follows:

1. Every $\mathrm{FO} + \Omega$ formula φ is a state formula, with $\mathrm{FV_p}(\varphi) = \mathrm{OCC}(\varphi) = \emptyset$ and $\mathrm{FV_s}(\varphi)$ defined in the usual way.
2. State formulae are closed under the Boolean connectives $\vee, \wedge, \neg$, and quantification $\exists, \forall$, with the usual rules for free variables.
3. If $\varphi(\vec{x}, \vec{y})$ is a state formula with $|\vec{x}| = n$, and p is a path variable of $arity(p) = n$, then $\psi(p, \vec{y}) \equiv \varphi(p \leftarrow \vec{x}, \vec{y})$ is a path formula with $\mathrm{FV_s}(\psi) = \vec{y}, \mathrm{FV_p}(\psi) = \mathrm{OCC}(\psi) = \{p\}$.
4. If $arity(p_1) = arity(p_2)$, then $p_1 = p_2, p_1 < p_2$ and $p_1 > p_2$ are path formulae. For all these formulae ψ, we have $\mathrm{FV_p}(\psi) = \mathrm{OCC}(\psi) = \{p_1, p_2\}$ and $\mathrm{FV_s}(\psi) = \emptyset$.
5. If φ_1 and φ_2 are two path formulae such that for every $p_1, p_2 \in \mathrm{OCC}(\varphi_1) \cup \mathrm{OCC}(\varphi_2)$ it is the case that $arity(p_1) = arity(p_2)$, then $\varphi_1 \vee \varphi_2$ and $\varphi_1 \wedge \varphi_2$ are path formulae. Furthermore, for ψ either $\varphi_1 \vee \varphi_2$ or $\varphi_1 \wedge \varphi_2$, we have: $P(\psi) = P(\varphi_1) \cup P(\varphi_2)$, where P is any of $\mathrm{FV_s}$, $\mathrm{FV_p}$ or OCC.
6. If φ is a path formula, then $\neg\varphi$ is a path formula, with all the predicates $\mathrm{FV_s}$, $\mathrm{FV_p}$ and OCC unchanged.
7. If φ is a path formula, with $arity(q) = n$ for all $q \in \mathrm{OCC}(\varphi)$ and p is a path variable with $arity(p) = n$, then $\psi \equiv \exists p\, \varphi$ is a path formula with $\mathrm{FV_s}(\psi) = \mathrm{FV_s}(\varphi), \mathrm{FV_p}(\psi) = \mathrm{FV_p}(\varphi) - \{p\}$ and $\mathrm{OCC}(\psi) = \mathrm{OCC}(\varphi) \cup \{p\}$.
8. If φ is a path formula with $\mathrm{FV_p}(\varphi) = \emptyset$, then $\psi \equiv \exists \mathbf{P} \varphi$ is a state formula, with $\mathrm{FV_s}(\psi) = \mathrm{FV_s}(\varphi)$ and $\mathrm{FV_p}(\psi) = \mathrm{OCC}(\psi) = \emptyset$.

When $\Omega = (+, -, 0, 1, <)$, we use the notation $L_{\mathrm{PATH}}(\mathrm{LIN})$ for $L_{\mathrm{PATH}}(\Omega)$; for $\Omega = (+, \cdot, 0, 1, <)$, we use the notation $L_{\mathrm{PATH}}(\mathrm{POLY})$.

We shall sometimes write $\varphi(\vec{x}, \vec{y})[p \leftarrow \vec{x}]$ instead of $\varphi(p \leftarrow \vec{x}, \vec{y})$. This will often be convenient if φ is a boolean combination of comparisons between x_is and y_js, as in this case the official syntax will be quite awkward. If $\varphi(\vec{x}, \vec{y})$ is $\vec{x} = \vec{y}$, then we write $p = \vec{y}$ for $\varphi(p \leftarrow \vec{x}, \vec{y})$.

Let us make some simple observations about the syntax. In every path formula φ, any two variables in $\mathrm{OCC}(\varphi)$ must have the same arity. If we have a formula $\psi(\vec{y}) \equiv \exists \mathbf{P} \varphi$, then there are state formulae $\varphi_1(\vec{x}, \vec{y}), \ldots, \varphi_k(\vec{x}, \vec{y})$ such that φ is built from the formulae of the form $\varphi_i(p \leftarrow \vec{x}, \vec{y})$ and $p_1 \theta p_2$, $\theta \in \{<, =, >\}$, by using Boolean connectives and quantification over path variables only.

Note that in formulae $\varphi(p \leftarrow \vec{x}, \vec{y})$, only one path variable can get instantiated. Intuitively, this correspond to coloring a path with previously defined regions. To see why one cannot bind two path variables at the same time, consider $\gamma(\vec{x}, \vec{y}) \equiv (x_2 = 0) \wedge (y_2 = 0) \wedge (y_1 = x_1 + 1)$, and $\beta(\vec{x}) \equiv (x_2 = 0)$, where $\vec{x} = (x_1, x_2)$ and $\vec{y} = (y_1, y_2)$. Now consider the formula $\varphi(x) \equiv \exists \mathbf{P} (\exists p_1, p_2 (p_1 = (x, 0) \wedge p_2 = (0,0) \wedge p_2 < p_1 \wedge [\forall p_3 \forall p_4 (\beta(p_2 \leftarrow \vec{x}) \wedge \beta(p_4 \leftarrow \vec{x}) \wedge p_3 < p_4 \wedge (\neg \exists p_5 (\beta(p_5 \leftarrow \vec{x}) \wedge p_3 < p_5 < p_4))) \rightarrow \gamma(p_3 \leftarrow \vec{x}, p_4 \leftarrow \vec{y})]))$. It says that there is a path from $(0,0)$ to $(x, 0)$ in $\mathbb{R}^2$ such that two consecutive points of the form $(x_1, 0), (x_2, 0)$ on this path must satisfy $x_2 = x_1 + 1$. Hence, $\varphi(x)$ holds iff x is an integer, while $\mathbb{Z}$ is *not* a semi-algebraic set. This explains why we had to put a restriction that only one path variable can get instantiated at a time – without this restriction, the logic cannot possibly be closed and decidable.

Semantics A *path* in $\mathbb{R}^n$ is a continuous map P from $\mathbb{R}_+ = \{r \in \mathbb{R} \mid r \geq 0\}$ to $\mathbb{R}^n$. Let $\mathcal{P}$ be a collection of paths, $P : \mathbb{R}_+ \rightarrow \mathbb{R}^n$ for dimensions $n > 0$. We define the semantics $D \models \varphi$ for each such collection. More precisely, for a state formula $\varphi(x_1, \ldots, x_k)$, we define the relation $D \models \varphi(\vec{a})$, where $\vec{a} \in \mathbb{R}^k$. For a path formula $\varphi(p_1, \ldots, p_n, x_1, \ldots, x_k)$, we define the relation $(D, P) \models \varphi(t_1, \ldots, t_n, a_1, \ldots, a_k)$ where $P : \mathbb{R}_+ \rightarrow \mathbb{R}^n$ is a path from $\mathcal{P}$, $a_i \in \mathbb{R}, t_j \in \mathbb{R}_+$. The intention is that instantiations of path variables are their "time-stamps" t_js, and path variables themselves are interpreted as $P(t_j) \in \mathbb{R}^n$.

We now give the formal definition. The cases of $\mathrm{FO} + \Omega$ formulae, as well as first-order quantification and Boolean connectives are standard. For $\psi(p, \vec{y}) \equiv \varphi(p \leftarrow \vec{x}, \vec{y})$, we have $(D, P) \models \psi(t, \vec{a})$ iff $D \models \varphi(P(t), \vec{a})$. For $\varphi(p, q) \equiv (p \theta q)$, with $\theta \in \{<, =, >\}$, we define $(D, P) \models \varphi(t, t')$ iff $t \theta t'$. Finally, $D \models \exists \mathbf{P} \varphi(\vec{a})$ iff there exists a path $P \in \mathcal{P}$ such that $(D, P) \models \varphi(\vec{a})$.

We now define the class of paths that we shall deal with. One possibility would be to use unrestricted semantics, by allowing any continuous path. However, continuous functions can oscillate very wildly, and including pathological curves may lead to counterintuitive truth values for sentences. Instead, we chose $\mathcal{P}$ to be the set of *non-zeno* paths, which are reasonably smooth and can only oscillate mildly. Formally, given an o-minimal structure $\mathcal{M}$, we say that a path $P : \mathbb{R}_+ \to \mathbb{R}^n$ is non-zeno with respect to $\mathcal{M}$ if for any set $X \subseteq \mathbb{R}^n$ definable in any o-minimal expansion of $\mathcal{M}$, the set $P^{-1}(X) = \{t \mid P(t) \in X\}$ is a union (not necessarily finite) of intervals, and the set of endpoints of those intervals is discrete.

For example, let $\mathbb{R}_+$ be partitioned into intervals $I_1, I_2, \ldots$ (which could be open or closed on each side) such that for some $\epsilon > 0$, the length of each I_j is at least ϵ. Let $P : \mathbb{R}_+ \to \mathbb{R}^n$ be piece-wise semi-algebraic with respect to this partition; that is, P is continuous and its restriction to each I_j is semi-algebraic. Then P is non-zeno with respect to $\mathbf{R}_{\text{lin}}$ and $\mathbf{R}$.

In what follows, we assume the non-zeno semantics; that is, the validity of $L_{\text{PATH}}(\Omega)$ formulae is with respect to paths non-zeno over $\langle \mathbb{R}, \Omega \rangle$. Intuitively, one can think of these paths as piece-wise semi-algebraic, or even piece-wise analytic.

We finally note that this semantics gives us a lot more paths than just semi-algebraic or semi-linear continuous functions. Consider, for example, a function $P : \mathbb{R}_+ \to \mathbb{R}^2$ given by $P(t) = (t, t - \lfloor t \rfloor)$ if $\lfloor t \rfloor$ is even, and $P(t) = (t, \lceil t \rceil - t)$ if $\lfloor t \rfloor$ is odd. This is a non-zeno path, as it is a semi-linear function of every interval $[n, n+1]$, $n \in \mathbb{N}$. However, $\{t \mid P(t) = (t, 0)\}$ is the set of natural numbers, which is not semi-algebraic.

4.2 Expressive power

In this section we collect a few facts we know about the expressive power of L_{PATH}. We have indicated that `CONNECTED` and `CONNECT-TO` can be expressed: for example, to test $\vec{x}, \vec{y}$ are in the same connected component of R, one writes $\big(R(\vec{x}) \wedge R(\vec{y})\big) \wedge \exists \mathbf{P} \big(\forall p\ R(p) \wedge \exists p_1 (p_1 = \vec{x}) \wedge \exists p_2 (p_2 = \vec{y})\big)$.

As another example, we show how to test if a region $R \subseteq \mathbb{R}^2$ is *simply connected*. Intuitively, R being simply connected means there are no holes in it (formally, a connected region R is simply connected if every closed curve in it is homotopic to a single point). Note that it is easy to check if a connected region R has a hole: either $\mathbb{R}^2 \setminus R$ is bounded, or this is not the case, and there are points $\vec{x}, \vec{y}$ such that every path from $\vec{x}$ to $\vec{y}$ intersects R. Clearly, this can be expressed in L_{PATH}.

The example above is an instance of a general result, saying that the language L_{PATH} is quite expressive when it comes to topological queries in 2-dimensional space. Proposition 2 below can be used to express many more 2-dimensional topological queries. Note that this is particularly relevant in geographical information systems, which most often deal with 2-dimensional data.

With every 2-dimensional spatial database instance D, one can associate a finite structure $I(D)$, its *topological invariant* [26]. $I(D)$ captures the topological information about D, which means that two instances D and D' are homeomorphic if, and only if, $I(D)$ and $I(D')$ are isomorphic. We now show the definability of $I(D)$ in L_{PATH}.

Proposition 2 *The topological invariant of semi-linear or semi-algebraic 2-dimensional spatial database instances is definable in* L_{PATH}. *More precisely, the topological invariant of semi-linear instances is definable in* $L_{\text{PATH}}(\text{LIN})$, *and the topological invariant of semi-algebraic instances is definable in* $L_{\text{PATH}}(\text{POLY})$. □

In the following section, we will see that every L_{PATH}-query is computable in polynomial time. We do not believe that the converse is true, although proving this remains open. L_{PATH} seems to be closer to NL (nondeterministic logarithmic space) than to PTIME. The following proposition may illustrate this:

Proposition 3 *Every query on finite constraint databases that is computable in* NL *is expressible in* L_{PATH}.

Proof sketch. By well-known results from descriptive complexity theory [18], to prove this proposition it suffices to define *directed reachability* in L_{PATH}. This can be done by first defining a topological embedding of the input graph in $\mathbb{R}^3$ and then using suitable paths in this embedded graph. □

Of course this proposition is only a small step towards an understanding of the expressive power of L_{PATH}. We would like to prove the converse statement that every L_{PATH}-query on finite instances is computable in NL. Unfortunately, this seems to be quite difficult. One approach would be to prove a collapse result saying that generic L_{PATH}-queries on finite structures are all expressible in some finitary path logic, and then use results from finite model theory. It is not clear how to extend first-order collapse results to L_{PATH}, however.

Another difficult problem is to compare the expressive power of L_{PATH} with that of the various extensions of FO by the connectivity quantifiers. We believe that directed reachability in finite graphs is not expressible in

$\mathrm{FO} + \mathrm{POLY} + \mathsf{Conn}$, but this appears to be very hard to prove, even under the complexity theoretic assumption that undirected reachability is not NL-complete.

4.3 Query evaluation: Closure and complexity

While L_{PATH} can express a great deal of reachability queries in constraint databases, it is not immediately clear whether it is either closed or tractable. Indeed, we saw earlier that if one extends L_{PATH} by allowing new binary predicates on path variables, the resulting logic is neither closed nor decidable. We now show that L_{PATH} has very desirable closure and tractability properties for domains relevant in spatial applications. More precisely, we say that $L_{\mathrm{PATH}}(\Omega)$

- is *decidable* if for every definable (in $\langle \mathbb{R}, \Omega \rangle$) database D, and every state sentence Φ, it is decidable whether $D \models \Phi$;
- is *closed* if for every definable database and every state formula $\varphi(\vec{x})$, the set $\{\vec{a} \mid D \models \varphi(\vec{a})\}$ is definable;
- *admits effective query evaluation* (of data complexity $\mathcal{C}$) if in addition a formula defining the set $\{\vec{a} \mid D \models \varphi(\vec{a})\}$ can be effectively obtained (and the complexity of obtaining such a formula is $\mathcal{C}$ in terms of the size of the database D).

Our goal is to prove the following result.

Theorem 3 *Let $\mathcal{M} = \langle \mathbb{R}, \Omega \rangle$ be o-minimal and decidable. Then $L_{\mathrm{PATH}}(\Omega)$ is closed and decidable. Furthermore, if $\mathcal{M}$ admits effective quantifier-elimination, then $L_{\mathrm{PATH}}(\Omega)$ admits effective query evaluation, and for $L_{\mathrm{PATH}}(\mathrm{LIN})$ and $L_{\mathrm{PATH}}(\mathrm{POLY})$ the data complexity is* PTIME.

Proof sketch. The proof is rather involved and consists of two main steps. We first show that an L_{PATH} query can be transformed into a "discrete path query" in which instead of quantifying over any time in a path P, we quantify only over components of the path with respect to some discretization of the path. In the second step we show that these discrete path queries can be evaluated on a database by model-checking the adjacency structure of the appropriate cell-decomposition provided by Lemma 1. □

Note that the proofs in Subsection 3.3 established tractability of $\mathrm{FO} + \mathrm{LIN} + \mathsf{Conn}$ but not $\mathrm{FO} + \mathrm{POLY} + \mathsf{Conn}$. Since connected components are definable in $L_{\mathrm{PATH}}(\mathrm{POLY})$, we conclude now:

Corollary 2 $\mathrm{FO} + \mathrm{POLY} + \mathsf{Conn}$ *queries have* PTIME *data complexity.* □

4.4 Model-checking for hybrid systems

We mentioned that our logic L_{PATH} has been inspired by temporal logics used in the model-checking approach to automated verification of finite-state reactive systems. Recent work on real-time and hybrid systems includes a number of formalisms for expressing reachability properties of infinite state systems defined from real parameters [17]. We now show that our results can be applied in this area. More precisely, we show that the *model-checking problem* for linear time temporal logic LTL for one class of hybrid systems, the o-minimal hybrid systems of [24, 25] is decidable and, furthermore, tractable, if the dimension of the hybrid systems is fixed. It is straightforward to extend our approach to the branching time temporal logics CTL and CTL*.

A *hybrid system* (cf. [17, 24]) of dimension n is a tuple $H = (S, S_0, S_F, F, E, I, G, R)$, where

- $S = Q \times \mathbb{R}^n$, where Q is a finite set, is the *state space*,
- $S_0 \subseteq S$ is the set of *initial states*,
- $S_F \subseteq S$ is the set of *final states*,
- $F : S \to \mathbb{R}^n$ assigns to each $q \in Q$ a vector field $F(q, \cdot)$,
- $E \subseteq Q \times Q$ is a set of *discrete transitions*,
- $I : Q \to 2^{\mathbb{R}^n}$ assigns to each $q \in Q$ a set $I(q)$ called the *invariant of* q,
- $G : E \to 2^{\mathbb{R}^n}$ assigns to each discrete transition $e = (q_1, q_2) \in E$ a set $G(e) \subseteq I(q_1)$ called the *guard of* e,
- $R : E \to 2^{\mathbb{R}^n}$ assigns to each discrete transition $e = (q_1, q_2) \in E$ a set $R(e) \subseteq I(q_2)$ called the *reset of* e.

Associated with the hybrid system H is a ternary *transition relation* $\to \subseteq S \times (E \cup \{c\}) \times S$, where c is a new symbol not contained in E. We write $s \xrightarrow{e} s'$ instead of $(s, e, s') \in \to$. We have two kinds of transitions:

- *Discrete Transitions:* $(q, \vec{x}) \xrightarrow{e} (q', \vec{x}')$ iff $e = (q, q') \in E$ and $\vec{x} \in G(e)$, $\vec{x}' \in R(e)$.

- *Continuous Transitions:* $(q,\vec{x}) \xrightarrow{c} (q',\vec{x}')$ iff $q = q'$ and there exists a $\delta \geq 0$ and a curve $x : [0,\delta] \to \mathbb{R}^n$ such that $x(0) = \vec{x}$, $x(\delta) = \vec{x}'$ and for every $t \in [0,\delta]$ it satisfies $\dot{x}(t) = F(q, x(t))$ and $x(t) \in I(q)$.

We assume that our hybrid systems are *non-blocking*, that is, for every state $s \in S$ there is an $e \in E \cup \{c\}$ and a state s' such that $s \xrightarrow{e} s'$.

A *trajectory* of H is a sequence $s_1 e_i s_2 e_2 \ldots$ such that for all $i \geq 1$ we have $s_i \xrightarrow{e_i} s_{i+1}$.

An *interpreted hybrid system* of signature $\Sigma = \{\pi_1, \ldots, \pi_m\}$ consists of a hybrid system H and a mapping Π that assigns to each state $s \in S$ a subset of Σ. Then Π associates with each trajectory $\tau = s_1 e_1 s_2 e_2 \ldots$ of H an ω-word $\Pi(\tau) := \Pi(s_1)\Pi(s_2)\ldots$ over the alphabet 2^{Σ}. We assume that the reader is familiar with the linear time temporal logic LTL (interpreted over ω-words), see [12]. The LTL-*model checking problem for hybrid systems* is defined as follows:

Input: An interpreted hybrid system (H,Π) and an LTL-formula φ.

Problem: Decide if for every trajectory τ of H the word $\Pi(\tau)$ satisfies φ.

Let $\mathcal{M}$ be an o-minimal structure over the reals. A hybrid system $H = (S, S_0, S_F, F, E, I, G, R)$ is $\mathcal{M}$-*definable* if $Q \subseteq \mathbb{R}$ is a definable set[1] and the sets S_0, S_F, the mappings I, G, R, and the relation $T := \{(q,\vec{x},\vec{y}) \mid (q,\vec{x}) \xrightarrow{c} (q,\vec{y})\}$ are definable in $\mathcal{M}$. A hybrid system is *o-minimal* if it is definable in some o-minimal structure over $\mathbb{R}$. An interpreted hybrid system (H,Π) is $\mathcal{M}$-*definable* if H is $\mathcal{M}$-definable, and for every π in the signature, the set $\Pi^{-1}(\pi)$ is definable.

Theorem 4 *Let $\mathcal{M} = \langle \mathbb{R}, \Omega \rangle$ be such that its expansion with $+, \cdot, 0, 1$ is a decidable o-minimal structure. Then the restriction of the LTL-model-checking problem for hybrid systems to $\mathcal{M}$-definable interpreted hybrid systems is decidable.*

Furthermore, if $\mathcal{M} = \mathbf{R}_{\text{lin}}$ or $\mathbf{R}$, for every fixed LTL-formula φ and $n \geq 1$, the restriction of the LTL-model-checking problem problem to $\mathcal{M}$-definable interpreted hybrid systems of dimension n can be solved in PTIME.

Proof sketch. We code hybrid systems as definable databases, and show that for every LTL-formula φ, there exists and can be effectively found an L_{PATH} formula φ^* such that, for every interpreted hybrid system (H,Π), we have $(H,\Pi) \models \varphi^*$ iff for every trajectory τ of H, $\Pi(\tau)$ satisfies φ. Then the result follows from Theorem 3. □

[1] We assume that there exists a finite set of $card(Q)$ definable constants. This is certainly true for all structures of practical interest like $\mathbf{R}_{\text{lin}}$ and $\mathbf{R}$; otherwise one can restate the definition by talking about definability of restrictions of sets like S, S_F, T, etc for each $q \in Q$.

5 Conclusion

Reachability between points in a region is a fundamental notion in spatial reasoning. A tool for analyzing spatial information – a spatial database, or a verification tool for systems that manipulate spatial data – has to somehow take reachability and connectivity into account. From previous work it appeared that incorporating reachability into a spatial language might be fundamentally incompatible with the use of constraint-based representations. Our first results here showed that this is not the case. Instead of attempting to approach connectivity through the use of some discrete recursion mechanism, we added reachability and other topological operators *directly*, and showed that this leads to closed languages. We then tackled the question of getting tractable, closed languages that can express the reachability queries of interest. The language L_{PATH} has a lot of what one wants in a spatial query language. In addition to the positive results on the data complexity, expressiveness, and closure, we think L_{PATH} is interesting as a synthesis of the temporal languages for verification of discrete systems with first-order constraint query languages.

Although we approached L_{PATH} from the point of view of spatial databases, it could also be seen as a general language for stating path properties of systems that are defined from semi-algebraic or semi-linear objects. Because of this, it is possible to compare it with languages for specifying properties of real-time or hybrid systems. We gave an example of how to model one specification formalism for hybrid systems within L_{PATH}.

We do not claim that L_{PATH} is a practical query language for connectivity queries in spatial databases – it still remains to find a more natural syntax, and to get query evaluation algorithms that run tractably on real applications. In addition, we do not have many results on the expressivity of L_{PATH} (beyond the PTIME complexity bound), and we know even less about languages FO + $\mathbb{T}$. We conjecture that FO + $\mathsf{C} \subset$ FO + $\mathsf{Conn} \subset L_{\text{PATH}}$, but we know of no techniques for proving separation results of this kind.

Acknowledgement We thank Suagata Basu, Mihalis Yannakakis, and Thomas Wilke for helpful discussions.

References

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

[2] F. Afrati, S. Cosmadakis, S. Grumbach and G. Kuper. Linear vs. polynomial constraints in database query languages. In *PPCP'94*, pages 181–192.

[3] R. Benedetti and J.-J. Risler. *Real Algebraic and Semi-algebraic Sets.* Hermann, Paris, 1990.

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

[5] M. Benedikt and L. Libkin. Relational queries over interpreted structures. *J. ACM*, to appear. Extended abstract in *PODS'97*, pages 87–98.

[6] M. Ben-Or, D. Kozen and J. Reif. The complexity of elementary algebra and geometry. *JCSS* 32 (1986), 251–264.

[7] J. Bochnak, M. Coste, M.-F. Roy. *Real Algebraic Geometry.* Springer Verlag, 1998.

[8] J.R. Büchi. On a decision method in restricted second-order arithmetic. In *Proc. 1960 Intl. Congress for Logic, Methodology and Philosophy of Science.* Stanford University Press, Stanford, 1962, pages 1-11.

[9] B.F. Caviness and J.R. Johnson, Eds. *Quantifier Elimination and Cylindrical Algebraic Decomposition.* Springer Verlag, 1998.

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

[11] G. E. Collins. Quantifier elimination for real closed fields by cylindrical algebraic decomposition. In *Autom. Theor. Form. Lang*, Springer LNCS 33, Springer Verlag, Berlin, 1975, 134–183.

[12] E. A. Emerson. Temporal and modal logic. Chapter 16 of Volume B of *Handbook of Theoretical Computer Science*, Elsevier, 1990.

[13] F. Geerts and B. Kuijpers. Expressing topological connectivity of spatial databases. In *DBPL'99*.

[14] E. Grädel, S. Kreutzer. Descriptive complexity theory for constraint databases. In *CSL'99*, pages 67–81.

[15] S. Grumbach, G. Kuper. Tractable recursion over geometric data. In *CP'97*, pages 450–462.

[16] J. Heintz, M.-F. Roy and P. Solernó. Description of connected components of a semialgebraic set in single exponential time. *Discrete and Computational Geometry*, 11 (1994), 121–140.

[17] T. A. Henzinger. The theory of hybrid automata. In *LICS'96*, pages 278–292.

[18] N. Immerman. Languages that capture complexity classes. *SIAM J. Comput.* 16 (1987), 760–778.

[19] P. Kanellakis, G. Kuper, and P. Revesz. Constraint query languages. *JCSS*, 51 (1995), 26–52.

[20] B. Kuijpers, J. Paredaens and J. Van den Bussche. On topological elementary equivalence of spatial databases. In *ICDT'97*, pages 432–446.

[21] B. Kuijpers and M. Smits. On expressing topological connectivity in spatial datalog. In *CDB'97*, pages 116–133.

[22] B. Kuijpers and J. Van den Bussche. On capturing first-order topological properties of planar spatial databases. In *ICDT'99*, pages 187–198.

[23] G. Kuper, L. Libkin and J. Paredaens, eds. *Constraint Databases.* Springer Verlag, 1999.

[24] G. Lafferriere, G. Pappas and S. Sastry. A new class of decidable hybrid systems. In *Hybrid Systems 1999*, LNCS 1569, Springer, 1999, pages 137–151.

[25] G. Lafferriere, G. Pappas and S. Sastry. O-minimal hybrid systems. *Mathematics of Control, Signals, and Systems*, to appear.

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

[27] J. Paredaens, J. Van den Bussche, and D. Van Gucht. First-order queries on finite structures over the reals. *SIAM J. Comput.* 27 (1998), 1747–1763.

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

[29] P. Revesz. Datalog and constraints. In [23], pages 155–170.

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

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

[32] J. Stoer and C. Witzgall. *Convexity and Optimization in Finite Dimensions.* Springer, 1970.

[33] L. van den Dries. *Tame Topology and O-minimal Structures.* Cambridge, 1998.

[34] M. Y. Vardi and P. Wolper. An automata-theoretic approach to automatic program verification. In *LICS'86*, pages 322–331.

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

APPENDIX

Proofs

Proof of Lemma 1

In the proofs of a) and b), we will use the definition of *stratification* [3, 7]. A stratification of $\mathbb{R}^n$ is a decomposition $\{A_1, \ldots, A_k\}$ of $\mathbb{R}^n$ such that $A_i \cap \mathrm{cl}(A_j)$ iff $A_i \subseteq \mathrm{cl}(A_j)$ for all $i \neq j$, and the following property holds. There exist a family of polynomials (of degree 1, for the linear case) $\{p_1(\vec{x}), \ldots, p_m(\vec{x})\}$ in n variables such that $A_1, \ldots, A_k$ are exactly the nonempty sets among

$$\bigcap_{i=1}^{m} \{\vec{a} \in \mathbb{R}^n \mid p(\vec{a})\ \sigma(i)\ 0\}$$

where σ ranges over the functions from $\{1, \ldots, n\}$ to $\{<, =, >\}$, and the closure of each A_j is obtained by relaxing the inequalities involved in the above representation; that is, changing $<$ to $\leq$ and $>$ to $\geq$.

We start the proof with b), as it is an easy consequence of the Local Triviality Theorem in algebraic geometry [7, 3]. Let $f_1, \ldots, f_p$ be all the polynomials used in a given representation of the S_is, and let $X_1, \ldots, X_s$ be a cylindric decomposition of $\mathbb{R}^{n+m}$ with respect to $f_1, \ldots, f_p$. That is, for each i, X_i is a semi-algebraic set homeomorphic to $\mathbb{R}^{k'}$ for some $k' \leq n+m$, and the polynomials f_js do not change sign on X_i. Let $\Pi : \mathbb{R}^{n+m} \to \mathbb{R}^n$ be the natural projection on the first n coordinates. The Local Triviality Theorem, applied to Π and $X_1, \ldots, X_s$ states that there exists a stratification $Z_1, \ldots, Z_p$ of $\mathbb{R}^n$ and, for each $1 \leq i \leq p$, a semi-algebraic set F_i, a semi-algebraic partition $\{F_{i1}, \ldots, F_{is}\}$ of F_i and a semi-algebraic homeomorphism $h_i : Z_i \times \mathbb{R}^m \to Z_i \times F_i$ such that

- $\Pi = \Pi \circ h_i$ on $Z_i \times \mathbb{R}^m$, and
- $h_i((Z_i \times \mathbb{R}^m) \cap X_j) = Z_i \times F_{ij}$.

In particular, for any $\vec{a} \in Z_i$ and any $\vec{y} \in \mathbb{R}^m$, we have $h_i((\vec{a}, \vec{y})) = (\vec{a}, \vec{z})$ for some $\vec{z} \in F_i$. Thus, for every $\vec{a} \in Z_i$, $h_i^{\vec{a}} : \mathbb{R}^m \to F_i$ that sends $\vec{y}$ to $\vec{z}$ is a homeomorphism, and is onto (since X_js partition $\mathbb{R}^{n+m}$ and F_{ij}s partition F_i).

We now fix $\vec{a}, \vec{b}$ in Z_i. Our goal is to find a homeomorphism $h_{\vec{a},\vec{b}} : \mathbb{R}^m \to \mathbb{R}^m$ such that $h_{\vec{a},\vec{b}}((X_j)_{\vec{a}}) = (X_j)_{\vec{b}}$ for each XX_j. First, define $\mathrm{id}_{\vec{a},\vec{b}} : \{\vec{a}\} \times F_i \to \{\vec{b}\} \times F_i$ to be the natural mapping that is the identity on F_i. Then we let $h_{\vec{a},\vec{b}} : \mathbb{R}^m \to \mathbb{R}^m$ be defined as

$$\pi \circ h_i^{-1} \circ \mathrm{id}_{\vec{a},\vec{b}} \circ h_i \circ (\{\vec{a}\} \times \mathrm{id}).$$

Here π is the projection the last m coordinates. That is, given $\vec{y}_1 \in \mathbb{R}^m$, apply h_i to $(\vec{a}, \vec{y}_1)$ to obtain $(\vec{a}, \vec{z}) \in Z_i \times F_i$. Then $h_{\vec{a},\vec{b}}(\vec{y}_1) = \vec{y}_2$ such that $h_i(\vec{b}, \vec{y}_2) = (\vec{b}, \vec{z})$. It is clear that $h_{\vec{a},\vec{b}}$ is a homeomorphism. We next note that $h_i((\{\vec{a}\} \times \mathbb{R}^m) \cap X_j) = \{(\vec{a}, \vec{z}) \mid \vec{z} \in F_{ij}\}$ and $h_i((\{\vec{b}\} \times \mathbb{R}^m) \cap X_j) = \{(\vec{b}, \vec{z}) \mid \vec{z} \in F_{ij}\}$, which therefore implies $h_{\vec{a},\vec{b}}((X_j)_{\vec{a}}) = (X_j)_{\vec{b}}$.

We now look at the decomposition of $\mathbb{R}^n$ given by the nonempty sets among $\Pi(X_1), \ldots, \Pi(X_s)$ and find a decomposition $V_1, \ldots, V_l$ such that each V_i is a subset of a unique $\Pi(X_k)$ and a unique Z_r. Given such a decomposition, we construct a decomposition $\mathcal{C}$ consisting of nonempty sets among $(V_i \times \mathbb{R}^m) \cap X_j$. Since we took X_js to be a cylindric decomposition, and each V_i is contained in a projection of some cell from that decomposition, we obtain that $\mathcal{C}$ itself is a cylindric decomposition over $\mathbb{R}^n$. Furthermore, since each V_i is subset of some Z_l, we obtain from the paragraph above that $\mathcal{C}$ is trivial over $\mathbb{R}^n$.

It remains to show polynomial time complexity, assuming that n and m are fixed. First, the cylindric decomposition $X_1, \ldots, X_s$ can be found in polynomial time [9, 11]. To see that the stratification $Z_1, \ldots, Z_p$ can be found in polynomial time, one analyzes the proof of the Local Triviality Theorem in [3, 7] to see that it is essentially constructing a decomposition except that at each inductive step, one may have to make a linear change of coordinates. Again, this can be done in polynomial time if the dimension is fixed. Finally, to find the V_is, one computes all possible intersections $Z_l \cap \Pi(X_j)$, and this is again polynomial for a fixed dimension. This completes the proof of b).

The proof of d) is identical, expect for the last step, as the Local Triviality Theorem is known to hold in any o-minimal expansion of the real filed [33]; clearly, nothing can be said about the complexity in this case.

We now move to the proof of a). First note that we cannot apply the proof above as the Local Triviality Theorem does not hold over $\mathbf{R}_{\text{lin}}$ (see, for example, in [33]). However, we can recover enough of it to prove a).

Let $f_1(\vec{x}, \vec{y}), \ldots, f_p(\vec{x}, \vec{y})$ be all the linear functions involved in the representation of $S_1, \ldots, S_k$. We assume that included in this collection are the $n+m$ functions x_i (for each variable x_i; this is done to ensure that none of the cells contains a line). Let us use the standard cylindric cell decomposition algorithm for linear functions, thus obtaining a cell decomposition $\mathcal{C}$ of $\mathbb{R}^{n+m}$ such that on every cell of $\mathcal{C}$, the sign of each f_i remains constant. In particular, each S_j is then a union of cells. We claim that $\mathcal{C}$ satisfies the condition of the theorem.

First, the fact that it can be computed in time polynomial in the representation of all the f_is (for n and m fixed) is derived from the standard bounds on cell decomposition [11]. Second, analyzing the proof of the existence of stratifications for semi-algebraic sets (see, for example, [3, 7]), one obtains that $\mathcal{C}$ is actually a stratification. This is because the only step in the proof of the existence of stratifications in the semi-algebraic case that deviates from the standard cell decomposition is a linear change of coordinates to ensure that certain products of variables do not appear in polynomials. However, since we deal with linear functions, and multiplication is not allowed, no linear change of variables is necessary.

We now fix a cell $C' \in \pi(\mathcal{C})$ and $\vec{a}, \vec{b} \in C' \subseteq \mathbb{R}^n$. Let $C \in \mathcal{C}$ be a cell in $C' \times \mathbb{R}^m$. We first note that since $\dim(C) = \dim(C') + \dim(C_{\vec{c}})$ for an arbitrary $\vec{c} \in C'$ [33], we obtain that $\dim(C_{\vec{a}}) = \dim(C_{\vec{b}})$.

We next fix two cells $B, C \in \mathcal{C}$ in $C' \times \mathbb{R}^m$ and show that the following four conditions are equivalent:

1. $C_{\vec{a}} \cap \mathrm{cl}(B_{\vec{a}}) \neq \emptyset$;
2. $C \cap \mathrm{cl}(B) \neq \emptyset$;
3. $C \subseteq \mathrm{cl}(B)$;
4. $C_{\vec{a}} \subseteq \mathrm{cl}(B_{\vec{a}})$.

Note that $4 \to 1$ is immediate, and $2 \to 3$ follows from the fact that $\mathcal{C}$ is a stratification. Both $1 \to 2$ and $3 \to 4$ follow from the fact that for any cell C in $C' \times \mathbb{R}^m$, and any $\vec{a} \in C'$, we have $\{\vec{a}\} \times \mathrm{cl}(C_{\vec{a}}) = \mathrm{cl}(C) \cap (\{\vec{a}\} \times \mathbb{R}^m)$. To prove this, assume that $C \neq \emptyset$ is given by a conjunction of strict inequalities $g_1(\vec{x}, \vec{y}) > 0, \ldots, g_p(\vec{x}, \vec{y}) > 0$ and equalities $v_1(\vec{x}, \vec{y}) = 0, \ldots, v_s(\vec{x}, \vec{y}) = 0$, where either s or p can be zero, and g_is and v_is are linear functions. The case of $p = 0$ is immediate, so we assume $p \neq 0$. Since $\mathcal{C}$ defines a stratification of $\mathbb{R}^{n+m}$, we obtain that $\mathrm{cl}(C)$ is given by the conjunction of $g_i(\vec{x}, \vec{y}) \geq 0$, $i = 1, \ldots, p$, and $v_j(\vec{x}, \vec{y}) = 0$, $j = 1, \ldots, s$. Let $G_i^{\vec{a}} = \{\vec{c} \mid g_i(\vec{a}, \vec{c}) > 0\}$, $\overline{G}_i^{\vec{a}} = \{\vec{c} \mid g_i(\vec{a}, \vec{c}) \geq 0\}$, and $V_j^{\vec{a}} = \{\vec{c} \mid v_j(\vec{a}, \vec{c}) = 0\}$. We have $\mathrm{cl}(G_i^{\vec{a}}) = \overline{G}_i^{\vec{a}}$ and $\mathrm{ri}(\overline{G}_i^{\vec{a}}) = G_i^{\vec{a}}$, where $\mathrm{ri}(\cdot)$ is the relative interior. Since $\bigcap_i G_i^{\vec{a}} \neq \emptyset$, we have $\mathrm{cl}(\bigcap_i G_i^{\vec{a}}) = \mathrm{cl}(\bigcap_i \mathrm{ri}(\overline{G}_i^{\vec{a}})) = \bigcap_i \mathrm{cl}(\mathrm{ri}(\overline{G}_i^{\vec{a}})) = \bigcap_i \overline{G}_i^{\vec{a}}$ (see, e.g., [30]). Let $V = \bigcap_j V_j^{\vec{a}}$. Then V is a closed set, and it intersects the open set $\bigcap_i G_i^{\vec{a}}$. Hence, $\mathrm{ri}(V) \cap \bigcap_i G_i^{\vec{a}} \neq \emptyset$, and thus $\mathrm{cl}(V \cap \bigcap_i G_i^{\vec{a}}) = \mathrm{cl}(V) \cap \mathrm{cl}(\bigcap_i G_i^{\vec{a}}) = V \cap \bigcap_i \overline{G}_i^{\vec{a}}$. Thus, $\mathrm{cl}(C_{\vec{a}}) = \{\vec{c} \mid g_i(\vec{a}, \vec{c}) \geq 0, v_j(\vec{a}, \vec{c}) = 0, i = 1, \ldots, p, j = 1, \ldots, s\}$, which proves $\{\vec{a}\} \times \mathrm{cl}(C_{\vec{a}}) = \mathrm{cl}(C) \cap (\{\vec{a}\} \times \mathbb{R}^m)$, and hence the equivalences above.

The equivalences above show that the adjacency structures of $\mathcal{C}_{\vec{a}}$ and $\mathcal{C}_{\vec{b}}$ are the same for any $\vec{a}$ and $\vec{b}$ in C', and, moreover, the boundary of $C_{\vec{a}}$ in $\mathbb{R}^m$ is the union of cells of the form $B_{\vec{a}}$, $B \in \mathcal{C}$, and likewise for $\vec{b}$. Furthermore, the proof shows that $\mathrm{cl}(C_{\vec{a}})$ is a convex polyhedron, which does not contain a line in $\mathbb{R}^m$ (since all variables have the same sign in every cell, by inclusion of all the functions x_is before computing the decomposition).

Furthermore, since $\mathrm{cl}(C_{\vec{a}}) = \mathrm{cl}(C)_{\vec{a}}$, we obtain from convex analysis that each face of $\mathrm{cl}(C_{\vec{a}})$ is a union of cells of the form $B_{\vec{a}}$, $B \in \mathcal{C}$. Thus, each vertex of $\mathrm{cl}(C_{\vec{a}})$ is a cell of the above form, each segment face is a union of a bounded one-dimensional cell of the form $B_{\vec{a}}$ and two vertices, and each ray face is a union of an unbounded one-dimensional cell and a vertex. The same statements hold for $\mathrm{cl}(C_{\vec{b}})$ in view of the above equivalences.

We next show that for any cell $B \in \mathcal{C}$, $B_{\vec{a}}$ is bounded iff $B_{\vec{b}}$ is bounded. Indeed, $B_{\vec{a}}$ is bounded iff $\mathrm{cl}(B_{\vec{a}})$ is bounded. By a simple induction on dimension, $\mathrm{cl}(B_{\vec{a}})$ of dimension > 0 is bounded iff all its one dimensional faces are segments, which in turn happens iff every one dimensional cell of the form $A_{\vec{a}}$ contained in $\mathrm{cl}(B_{\vec{a}})$ is adjacent to two vertices (0-dimensional cells). Since the adjacency structures of $\mathcal{C}_{\vec{a}}$ and $\mathcal{C}_{\vec{b}}$, we obtain from here that in $\mathrm{cl}(B_{\vec{b}})$ every 1-dimensional face is bounded, and thus $B_{\vec{b}}$ is bounded.

Next, we need the following observation. Let P_1 and P_2 be to convex polyhedra in $\mathbb{R}^k$, with $\dim(P_1) = \dim(P_2)$. Assume that P_1 and P_2 are homeomorphic, and none contains a line. Since boundary is a topological invariant of a convex set [32], this in particular implies that $\mathrm{bd}(P_1)$ is homeomorphic to $\mathrm{bd}(P_2)$. Fix any homeomorphism $g : \mathrm{bd}(P_1) \to \mathrm{bd}(P_2)$. We claim that g can be extended to a homeomorphism $G : P_1 \to P_2$.

To prove this claim, assume without loss of generality that $\dim(P_1) = k$ (if not, one works in its affine hull). It also suffices to show that the statement above is true for some convex set X such that both P_1 and P_2 are homeomorphic to X. Indeed, let $h_1 : P_1 \to X$ and $h_2 : P_2 \to X$ be homeomorphisms (in particular, $\mathrm{bd}(X) = h_1(\mathrm{bd}(P_1)) = h_2(\mathrm{bd}(P_2)))$. Consider a map v from $\mathrm{bd}(X)$ to itself given by $h_2 \circ g \circ h_1^{-1}$. Clearly, it is a homeomorphism, so by assumption we can extend it to a homeomorphism $V : X \to X$. But now $G = h_2^{-1} \circ V \circ h_1$ is a homeomorphism $P_1 \to P_2$ that extends g.

Now the claim about the extension of a homeomorphism from the boundary to the whole polyhedron follows from the fact that a polyhedron not containing a line in $\mathbb{R}^k$ is homeomorphic to either the unit ball $\mathbf{B}^k = \{\vec{x} \mid \parallel \vec{x} \parallel \leq 1\}$ (if it is bounded) or to $\mathbf{D}^k = [0,1)^k$ (if it unbounded), see [32]. In the first case, a homeomorphism $g : \mathrm{bd}(\mathbf{B}^k) \to \mathrm{bd}(\mathbf{B})^k$ is extended as follows. The origin is mapped to itself. Given $\vec{x} \in \mathbf{B}^k$, let the ray from the origin in the direction of $\vec{x}$ intersect $\mathrm{bd}(\mathbf{B}^k)$ at $\vec{y}$. Then $G(\vec{x})$ is the point $\vec{x}_0$ on the segment between the origin and $g(\vec{y})$ such that $\parallel \vec{x} \parallel = \parallel \vec{x_0} \parallel$. In the second case, consider any $\vec{x}$ in the interior of $\mathbf{D}^k$. Let $\mathbf{1}$ stand for $(1, \ldots, 1)$. Let the ray originating in $\mathbf{1}$ and passing through $\vec{x}$ intersect $\mathrm{bd}(\mathbf{D}^k)$ at $\vec{y}$. Consider a point $\vec{x}_0$ on the segment between $\mathbf{1}$ and $g(\vec{y})$ such that

$$\frac{d(\vec{x}_0, \mathbf{1})}{d(g(\vec{y}), \mathbf{1})} = \frac{d(\vec{x}, \mathbf{1})}{d(\vec{y}, \mathbf{1})}$$

(where $d(\cdot, \cdot)$ is the usual euclidean distance), and let $G(\vec{x}) = x_0$. It is routine to verify that in both cases G is a homeomorphism extending g.

Let now $\mathcal{C}^k_{\vec{a}}$ be the union of cells $C_{\vec{a}}$ whose dimension is at most k, and likewise for $\mathcal{C}^k_{\vec{b}}$. Since for each cell $C_{\vec{a}}$, $\mathrm{cl}(C_{\vec{a}})$ is the union of lower-dimensional cells in $\mathcal{C}_{\vec{a}}$, we obtain that $\mathcal{C}^k_{\vec{a}}$ and $\mathcal{C}^k_{\vec{b}}$ are closed. We now conclude the proof of a) by induction, by showing that for every k, there is a homeomorphism $h_k : \mathcal{C}^k_{\vec{a}} \to \mathcal{C}^k_{\vec{b}}$ such that $h_k(B_{\vec{a}}) = B_{\vec{b}}$ for any cell B with $\dim(B_{\vec{a}}) \leq k$. For $k = 0$, the statement follows from the fact that $\dim(B_{\vec{a}}) = \dim(B_{\vec{b}})$ for every $B \in \mathcal{C}$; thus, h_0 maps every 0-dimensional cell (point) of the form $B_{\vec{a}}$ to the point $B_{\vec{b}}$.

For the induction step, assume we have already constructed h_k. Consider any cell $B \in \mathcal{C}$ such that $\dim(B_{\vec{a}}) = \dim(B_{\vec{b}}) = k+1$. Consider $\mathrm{cl}(B_{\vec{a}}) = B_{\vec{a}} \cup \mathrm{bd}(B_{\vec{a}})$. We know that $\mathrm{bd}(B_{\vec{a}})$ is a subset of $\mathcal{C}^k_{\vec{a}}$ and moreover a union of cells. We thus have a mapping g_B which is a restriction of h_k on $\mathrm{bd}(B_{\vec{a}})$ and therefore a homeomorphism $\mathrm{bd}(B_{\vec{a}}) \to \mathrm{bd}(B_{\vec{b}})$ (because $\mathrm{bd}(B_{\vec{b}})$ is a union of cells too, and the adjacency structures of $\mathcal{C}_{\vec{a}}$ and $\mathcal{C}_{\vec{b}}$ are the same). Note that $\mathrm{cl}(B_{\vec{a}})$ and $\mathrm{cl}(B_{\vec{b}})$ are both $k+1$-dimensional polyhedra, none containing a line, and $\mathrm{cl}(B_{\vec{a}})$ is bounded iff $\mathrm{cl}(B_{\vec{b}})$ is. Thus, they are homeomorphic, as by the above they can be only be homeomorphic to $\mathbf{B}^{k+1}$ or $\mathbf{D}^{k+1}$. Applying the claim, we extend g_B to a homeomorphism $G_B : \mathrm{cl}(B_{\vec{a}}) \to \mathrm{cl}(B_{\vec{b}})$; note that $G_B(B_{\vec{a}}) = B_{\vec{b}}$ as $G_B(\mathrm{bd}(B_{\vec{a}})) = \mathrm{bd}(B_{\vec{b}})$.

Let $B^1, \ldots, B^s$ be all the cells with $\dim(B^i_{\vec{a}}) = k+1$. Then $\mathcal{C}^{k+1}_{\vec{a}} = \bigcup_i \mathrm{cl}(B^i_{\vec{a}})$ and $\mathcal{C}^{k+1}_{\vec{b}} = \bigcup_i \mathrm{cl}(B^i_{\vec{b}})$. We have homeomorphisms $G_{B_i} : \mathrm{cl}(B^i_{\vec{a}}) \to \mathrm{cl}(B^i_{\vec{b}})$ for each i. Note that B^is are pairwise disjoint, and for any $x \in \mathrm{cl}(B^i_{\vec{a}}) \cap \mathrm{cl}(B^j_{\vec{a}})$ it is the case that $x \in \mathcal{C}^k_{\vec{a}}$ and $G_{B_i}(x) = G_{B_j}(x) = h_k(x)$. We thus can define h_{k+1} as the union of all G_{B_i}. Clearly, it extends h_k and $h_{k+1}(B^i_{\vec{a}}) = B^i_{\vec{b}}$ for all i. It is known that if one has a 1-1 function $f : X \to Y$ on two topological spaces such that $X = X_1 \cup \ldots \cup X_t$ and $Y = Y_1 \cup \ldots \cup Y_t$, where all X_is and Y_is are closed and the restriction of f to each X_i is a homeomorphism between X_i and Y_i, then f is a homeomorphism between X and Y. This implies that h_{k+1} is a homeomorphism $\mathcal{C}_{\vec{a}}^{k+1} \to \mathcal{C}_{\vec{b}}^{k+1}$, thus completing the induction case.

We now finally take h to be h_m; it is a homeomorphism $\mathbb{R}^m \to \mathbb{R}^m$ (since every element is in some cell) with the property that $h(C_{\vec{a}}) = C_{\vec{b}}$ for every cell $C \in \mathcal{C}$. This completes the proof of a).

We finally prove c). Start with a cell decomposition $\mathcal{C}$ of $\mathbb{R}^{n+m}$ such that each S_i is a union of cells, and $\pi(\mathcal{C})$ is a cell decomposition of $\mathbb{R}^n$ (recall that π here is the natural projection onto the first n coordinates). It is known that each cell is definable in the structure $\langle \mathbb{R}, <, S_1, \ldots, S_k \rangle$ [33]. To see what is needed in order to obtain formulae defining each cell, one can check all the steps in the proof of cell decomposition for o-minimal structures (see, e.g., [28, 33]) and observe that the only step that is needed to ensure effectiveness is the calculation of uniform bounds. That is, for a formula $\alpha(x, \vec{y})$, one should be able to calculate a number K such that for each $\vec{a}$, the set $\{c \mid \mathcal{M} \models \alpha(c, \vec{a})\}$ is composed of fewer than K intervals. This can be done using the decidability of $\mathcal{M}$. For each number i, we can write a sentence Φ_α^i stating that the set $\{c \mid \mathcal{M} \models \alpha(c, \vec{a})\}$ is composed of fewer than i intervals for all i, and then check if $\mathcal{M} \models \Phi_\alpha^i$. The uniform bounds theorem [28] says that there is a number K such that $\mathcal{M} \models \Phi_\alpha^K$, and thus it can be found since $\mathcal{M}$ is decidable.

Now we have a decomposition of $\mathbb{R}^{n+m}$ into, say, s cells. We consider a cell $C' \in \pi(\mathcal{C})$ (note that the decomposition $\pi(\mathcal{C})$ is also computable). Let $t \leq s$ be the number of cells in the cylinder $C' \times \mathbb{R}^m$; denote them by $A^1, \ldots, A^t$. For two mappings

$$\sigma : \{1, \ldots, t\} \times \{1, \ldots, t\} \to \{=, \neq\},$$

$$\theta : \{1, \ldots, t\} \times \{1, \ldots, t\} \times \{1, \ldots, t\} \to \{=, \neq\},$$

let $C'_{\sigma,\theta}$ be the set of all $\vec{a} \in C'$ such that for every $1 \leq i, j, k \leq t$,

$$(A_{\vec{a}}^i \cap \mathrm{cl}(A_{\vec{a}}^j)) \;\; \sigma(i,j) \;\; \emptyset$$

and

$$(A_{\vec{a}}^i \cap \mathrm{cl}(A_{\vec{a}}^j) \cap \mathrm{cl}(A_{\vec{a}}^k)) \;\; \theta(i,j,k) \;\; \emptyset.$$

Since the closure of a definable set is definable in any o-minimal structure on $\mathbb{R}$ [33], we obtain that $C'_{\sigma,\theta}$ is definable by a formula $\beta_{\sigma,\theta}(\vec{x})$. We now consider the collection F of all the $2^{t^2+t^3}$ formulae $\beta_{\sigma,\theta}$, as σ and θ range over the maps as above. Note that F can be effectively found from the representation of $S_1, \ldots, S_k$. We next do a cell decomposition $\hat{\mathcal{C}}$ of $\mathbb{R}^n$ so that every cell in $\pi(\mathcal{C})$ and every set definable by $\beta_{\sigma,\theta}$ is a union of cells. By the same argument as in the previous paragraph, we can effectively construct formulae defining the cells of $\hat{\mathcal{C}}$.

Let $\tilde{\mathcal{C}}$ be the collection of all nonempty subsets of $\mathbb{R}^{n+m}$ of the form $C \cap (A \times \mathbb{R}^m)$ where C ranges over $\mathcal{C}$ and A ranges over $\hat{\mathcal{C}}$. Clearly, $\tilde{\mathcal{C}}$ is a decomposition of $\mathbb{R}^{n+m}$ which is cylindric over $\mathbb{R}^n$. Furthermore, every cell in the projection $\pi(\tilde{\mathcal{C}})$ is a cell of $\hat{\mathcal{C}}$. Next, fix a cell A in $\pi(\tilde{\mathcal{C}})$. Let $\vec{a}, \vec{b} \in A$. Assume that for two cells C^1, C^2 of $\tilde{\mathcal{C}}$, we have $C_{\vec{a}}^1 \cap \mathrm{cl}(C_{\vec{a}}^2) \neq \emptyset$. Since $C^i = C_0^i \cap (A \times \mathbb{R}^m)$ for some cell C_0^i of $\mathcal{C}$ in the same cylinder over $\mathbb{R}^n$, we have $C_{\vec{b}}^1 \cap \mathrm{cl}(C_{\vec{b}}^2) \neq \emptyset$ as $\vec{a}$ and $\vec{b}$ satisfy all the same formulae $\beta_{\sigma,\theta}$. Thus, $C_{\vec{a}}^1 \cap \mathrm{cl}(C_{\vec{a}}^2) \neq \emptyset$ iff $C_{\vec{b}}^1 \cap \mathrm{cl}(C_{\vec{b}}^2) \neq \emptyset$. The proof that $C_{\vec{a}}^1 \cap \mathrm{cl}(C_{\vec{a}}^2) \cap \mathrm{cl}(C_{\vec{a}}^3) \neq \emptyset$ iff $C_{\vec{b}}^1 \cap \mathrm{cl}(C_{\vec{b}}^2) \cap \mathrm{cl}(C_{\vec{b}}^3) \neq \emptyset$ for any $C^1, C^2, C^3 \in \tilde{\mathcal{C}}$ is identical. This shows that $\tilde{\mathcal{C}}$ is adjacency preserving over $\mathbb{R}^n$. It is immediate from its definition and the previous paragraph that first-order descriptions of its cells can be effectively found. This completes the proof of d), and the lemma. $\square$

Proof of Theorem 1

The result is by a simple induction on the formulae. The only case to prove is $\psi(\vec{x}) \equiv \mathsf{Top}\,\vec{y}.\ \varphi(\vec{x}, \vec{y})$ for $\mathsf{Top} \in \mathbb{T}$. Let $\vec{x}$ and $\vec{y}$ be of length n and m, resp. On a definable database D, by induction, $\varphi(\vec{x}, \vec{y})$ gives us a definable set $S \subseteq \mathbb{R}^n \times \mathbb{R}^m$. By Lemma 1, there exists a decomposition $\mathcal{C}$ of $\mathbb{R}^{n+m}$ into finitely many definable cells which is trivial over $\mathbb{R}^n$ and such that S is a union of cells of $\mathcal{C}$. Let $\mathcal{C}'$ be the projection of $\mathcal{C}$ onto $\mathbb{R}^n$, and C a cell in $\mathcal{C}'$. By triviality, for every $\vec{a}, \vec{b} \in C$, it is the case that $S_{\vec{a}}$ and $S_{\vec{b}}$ are homeomorphic, and thus they agree on

Top. Therefore, the output of ψ on D is a union of (finitely) many cells in $\mathcal{C}'$; since each cell is definable, the output is definable, too.

To get the complexity bound for $\mathrm{FO} + \mathrm{LIN} + \mathbb{T}$ and $\mathrm{FO} + \mathrm{POLY} + \mathbb{T}$, we show by induction that for each query φ, there exists a number k such that the complexity of evaluating φ on D is $O(N^k)$, where N is the size of a given representation of D (assuming Top can tested in constant time). Again, the only case to consider is that of $\psi(\vec{x}) \equiv \mathsf{Top}\,\vec{y}.\ \varphi(\vec{x},\vec{y})$, as others follow from the standard bounds on quantifier-elimination. Given $S = \varphi(D)$ computed in $O(N^k)$, we can find, by Lemma 1, a trivial decomposition $\mathcal{C}$ in time polynomial in N^k. Since the projection operation is polynomial for a fixed dimension, we get that for some k_1 that depends only on φ, we can construct both $\mathcal{C}$ and $\mathcal{C}'$ in time $O(N^{k_1})$. We next select a point $\vec{a}$ in each cell of $\mathcal{C}'$ and construct the fiber $S_{\vec{a}}$. This can be done in polynomial time, too (indeed, cell decomposition algorithms already return a point from each cell when they produce a decomposition [11, 9], and then one substitutes those representative points for $\vec{x}$ in the definition of S). Finally, for each cell we test in constant time if the fiber $S_{\vec{a}}$ is in Top. Thus, the total complexity is polynomial in N, with the exponent depending on ψ only. This completes the proof. $\square$

Proof of Theorem 2

As usual this is proved by induction on the structure of the formulae. We only need to prove the case of $\psi(\vec{x},\vec{y}_1,\ldots,\vec{y}_k) \equiv T\vec{y}.\ \varphi(\vec{x},\vec{y})$. Assume φ defines $S \subseteq \mathbb{R}^n \times \mathbb{R}^m$: $S = \{(\vec{a},\vec{b}) \mid D \models \varphi(\vec{a},\vec{b})\}$. By Lemma 1, there exists a decomposition $\mathcal{C}$ trivial over $\mathbb{R}^n$. Let $C_1,\ldots,C_p$ be all the cells in the projection of $\mathcal{C}$ onto $\mathbb{R}^n$. The proof of b) in Lemma 1 (which is just an application of Local Triviality) shows that for every i, and every $\vec{a},\vec{b} \in C_i$, there is a definable homeomorphism $h^i_{\vec{a},\vec{b}} : \mathbb{R}^m \to \mathbb{R}^m$ such that $h^i_{\vec{a},\vec{b}}(S_{\vec{a}}) = S_{\vec{b}}$. Since T is topological, it implies that $(\vec{e}_1,\ldots,\vec{e}_k) \in T(S_{\vec{a}})$ if and only if $(h^i_{\vec{a},\vec{b}}(\vec{e}_1),\ldots,h^i_{\vec{a},\vec{b}}(\vec{e}_k)) \in T(S_{\vec{b}})$.

Since each cell C_i is a definable set, it has a definable representative $\vec{c}_i \in C_i$ [33]. Thus, a tuple $(\vec{a},\vec{b}_1,\ldots,\vec{b}_k)$ is in $\psi(D)$ iff for $i \in 1,\ldots,p$ such that $\vec{a}$ is in C_i, the following holds:

$$\exists(\vec{e}_1,\ldots,\vec{e}_k) \in T(S_{\vec{c}_i}) \bigwedge_{j=1}^{p} \vec{b}_j = h^i_{\vec{c}_i,\vec{a}}(\vec{e}_j)$$

Since $S_{\vec{c}_i}$ is definable, and $h^i_{\vec{c}_i,\vec{a}}$ is a definable homeomorphism, $T(S_{\vec{c}_i})$ is definable, which implies that $\psi(D)$ is a definable set, and proves closure. $\square$

Proof of Proposition 1

The proof is by induction on the formulae, with only the case of applying the Conn operator being nontrivial. Suppose we are given $\varphi(\vec{x},\vec{y})$ which defines, on a semi-linear database D, a semi-linear set $S \subseteq \mathbb{R}^n \times \mathbb{R}^m$. We must show that the set $S' \subseteq \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^m$ of triples $(\vec{a},\vec{b}_2,\vec{b}_2)$ such that $\vec{b}_1$ and $\vec{b}_2$ are in the same connected component of $S_{\vec{a}}$, is semi-linear, and can be computed in time polynomial in the size of a given representation of S (assuming n and m fixed). For this, we use Lemma 1. We compute in PTIME a decomposition $\mathcal{C}$ of $\mathbb{R}^{n+m}$, which is trivial over $\mathbb{R}^n$, such that the signs of all the functions used in the representation of S remain constant on each cell.

Let C be a cell in $\pi(\mathcal{C})$, where $\pi : \mathbb{R}^{n+m} \to \mathbb{R}^n$ is the natural projection. Let $C^1,\ldots,C^k$ be all the cells in the cylinder $C \times \mathbb{R}^m$. We know from the proof of Lemma 1 that for any $\vec{a},\vec{b}$, $C^i_{\vec{a}}$ adj $C^j_{\vec{a}}$ iff $C^i_{\vec{b}}$ adj $C^j_{\vec{b}}$ iff C^i adj C^j where X adj Y means $X \cap \mathrm{cl}(Y) \neq \emptyset$ or $Y \cap \mathrm{cl}(X) \neq \emptyset$. We let G_C be a graph with the set of nodes being the indices of the cells among $C^1,\ldots,C^k$ that belong to S, and edges (i,j) for every C^i adj C^j. Let $K_1,\ldots,K_p$ be the connected components of G_C. Define $P^C_{\vec{a}}$ as

$$\bigcup_{l=1}^{p}((\bigcup_{i\in K_l} C^i_{\vec{a}}) \times (\bigcup_{i\in K_l} C^i_{\vec{a}})) \ \subseteq\ \mathbb{R}^m \times \mathbb{R}^m$$

for $\vec{a} \in C$, and let

$$S' = \bigcup_{C \in \pi(\mathcal{C})} \bigcup_{\vec{a} \in C} \{\vec{a}\} \times P^C_{\vec{a}} \subseteq \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^m.$$

It is known [33] that the sets of the form $(\bigcup_{i \in K_l} C^i)$ are exactly the connected components of $S \cap (C \times \mathbb{R}^m)$. It thus follows from the above that the sets of the form $(\bigcup_{i \in K_l} C^i_{\vec{a}})$ are exactly the connected components of $S_{\vec{a}}$, and hence S' is the result of $\mathsf{Conn}\vec{x}.\ \varphi$. By converting the above into a FO definition, we obtain PTIME data complexity as the number of quantifiers only depends on n and m. $\square$

Proof of Proposition 2

We briefly recall the definition of topological invariant. The reader is referred to [26, 31] for a more precise description. Given a spatial instance D over a database schema *Reg* containing only binary relations, a *cell partition* of D is a partition of $\mathbb{R}^2$ into finitely many connected subsets called *cells* such that each relation of D is a (finite) union of cells. The topological invariant $I(D)$ is roughly a finite description of the maximal cell partition of D. It is a finite structure consisting of the following relations (their meaning is explained intuitively):

1. A unary relation C, providing the cells of dimension 0, 1, 2, and a distinguished cell of dimension 0 called the *exterior-cell*.
2. A unary function *Dim*, which associates a dimension to each cell.
3. A binary relation *Adjacent* providing the topological adjacency relationship between the cells.
4. For each region name $p \in$ *Reg*, a unary relation p providing the set of cells contained in region p.
5. *Orientation* is a 5-ary relation providing the clockwise and counterclockwise orientation of edges incident to each cell of dimension 0. More precisely, $(+, v, e_1, e_2, e_3) \in$ *Orientation* iff v is a cell of dimension 0, e_1, e_2, e_3 are cells of dimension 1 incident to v, and e_2 lies between e_1 and e_3 in the clockwise order on the incident cells of v, and $(-, v, e_1, e_2, e_3) \in$ *Orientation* iff v is a cell of dimension 0, e_1, e_2, e_3 are cells incident to v, and e_2 lies between e_1 and e_3 in the counterclockwise order on the incident cells of v (one can use 0 and 1 to code + and $-$).

Let *inv(Reg)* denote the above schema. We want to prove that, given a semi-algebraic or semi-linear instance D over *Reg*, there exists a formula in L_{PATH} that gives $I(D)$, the topological invariant over D.

We start by giving the definition of cells as an equivalence relation E over points of $\mathbb{R}^2$: two points p and p' are in E iff they are in the same cell of the topological invariant. (Cells themselves can be defined from E, as any semi-linear or semi-algebraic equivalence relation has a FO-definable set of representatives [33].)

The cells of dimension 0 are non-regular points and can be defined in FO(<).

A cell of dimension 1 is a connected set of points with the same boundary cone type (cone type as defined in [20, 31], a boundary cone type is the cone type of a point that lies on the common boundary of several region of *Reg*). The fact that two points p and p' have the same boundary cone type is expressible in FO(<); this follows from [20, 22, 31]. From the definition of topological invariant, two points p and p' are in the same 1-dimensional cell iff they have the same boundary cone type c and if there exists a path P from p to p' such that all the points q in P have the same boundary cone type c (in particular, there is no non-regular point in P). This can be expressed in L_{PATH}.

A 2-dimensional cell is a connected set of points having the same trivial *full* cone type. The set of points having the same full cone type is definable in FO(<) and therefore its connected components are definable in L_{PATH}.

The function *Dim* which associates to each point the dimension of the cell it belongs to is easily derived from the cone type of each cell (non-regular, boundary or full) and this is definable in FO($<$).

The adjacency relationship is given as a binary relationship A over points in $\mathbb{R}^2$: $A(p, p')$ iff the cell of p is topologically adjacent to the cell of p'. As we have seen before, the cells of p and p' are definable in L_{PATH} (the cell of p is the set of points q such that $R(p,q)$), and testing for adjacency is FO($<$). Therefore A is L_{PATH} definable.

We now show how to define the relation *Orientation* in L_{PATH}. From the above we know how to check in L_{PATH} that v is a cell of dimension 0, e_1, e_2, e_3 are cells of dimension 1 and that e_1, e_2, e_3 are adjacent to v. If this is the case, it can be checked in L_{PATH} whether e_2 lies between e_1 and e_3 in the clockwise (counterclockwise) order on the incidents cells of v in the following way. For each sufficiently small square S with center v, let v_1, v_2, v_3 be the intersection of the edges e_1, e_2, e_3 with S. It suffices to check that, in S, v_2 lies between v_1 and v_3 in the clockwise (counterclockwise) order. This can be done in $FO(<)$ by considering all the possible cases corresponding to which sides of S v_is lie on. □

Proof of Proposition 3

Directed reachability is the following query: given a directed graph G, compute the set of all pairs (v, w) of vertices of G such that there is a path from v to w. It was proved in [18] that directed reachability is complete for NL under first-order reductions. Since it can easily be seen that the class of finite database queries definable in L_{PATH} is closed under first-order reductions, it suffices to prove that directed reachability is expressible in L_{PATH}.

Finite directed graphs are represented by a finite subset $V \subseteq \mathbb{R}$ and a finite set $E \subseteq V^2 \subseteq \mathbb{R}^2$. Note first that the following formula, intuitively saying that there exists a path from x to y such that for all successive $z_1, z_2 \in V$ appearing on this path we have $E(z_1, z_2)$ is *not* an L_{PATH}-formula:

$$\begin{aligned}\exists \mathbf{P}\Big(&\exists p_1 \exists p_2 (p_1 = x \wedge p_2 = y)\\ &\wedge \forall q_1 \forall q_2 \big((V(q_1) \wedge V(q_2) \wedge \neg \exists q_3 (V(q_3) \wedge q_1 < q_3 \wedge q_3 < q_2)) \to E(z_1, z_2)[z_1 \leftarrow q_1, z_2 \leftarrow q_2]\big)\Big).\end{aligned}$$

(The reason that this is not an L_{PATH}-formula is that we are not allowed to substitute both z_1 and z_2 in $E(z_1, z_2)$ by path variables.)

To find an L_{PATH}-formula expressing directed reachability, we first define a topological representation of the input graph in $\mathbb{R}^3$. Then undirected reachability can be defined in the same way as topological connectivity. In a second step, we use two additional predicates to encode the direction of the edges.

For $a \in \mathbb{R}$ we let $\bar{a} := (a, 0, 0)$. For a pair $(a, b) \in \mathbb{R}^2$ we let $\overline{(a,b)}$ be the curve in $\mathbb{R}^3$ connecting the following points by straight line segments: $\bar{a}$, $(a, b, 0)$, (a, b, a), (b, b, a), $(b, 0, a)$, $\bar{b}$. It is easy to define a formula $\gamma(x, y, \vec{z}) \in$ FO($<$) such that for all $a, b \in \mathbb{R}$, $\vec{c} \in \mathbb{R}^3$ we have $\langle \mathbb{R}, < \rangle \models \gamma(a, b, \vec{c})$ if, and only if, $\vec{c}$ appears on the curve $\overline{(a,b)}$. Observe that for all $a, b, a', b' \in \mathbb{R}$ the curves $\overline{(a,b)}$ and $\overline{(a',b')}$ intersect if, and only if, either $a = a'$ or $b = b'$.

For a directed graph $G = (V, E)$ with $V \subseteq \mathbb{R}$ we let $\bar{V} := \{\bar{a} \mid a \in V\}$ and $\bar{E} := \bigcup_{(a,b) \in E} \overline{(a,b)}$. Clearly, there are formulae $\varphi_V(\vec{z}), \varphi_E(\vec{z})$ in FO over $<$ and E, V defining the sets $\bar{V}, \bar{E}$, respectively. Note that for all $a, b \in V$ there is an path from a to b in the undirected graph underlying G if, and only if, there is a path in $\mathbb{R}^3$ from $\bar{a}$ to $\bar{b}$ that is contained in $\bar{E}$.

To encode the direction of the edges we define two new sets Tail $:= \{(a, b, 0) \in \mathbb{R}^3 \mid (a, b) \in E\}$ and Head $:= \{(b, 0, a) \in \mathbb{R}^3 \mid (a, b) \in E\}$. Let $\varphi_{\text{Tail}}(\vec{z}), \varphi_{\text{Head}}(\vec{z})$ in FO over $<, E, V$ be formulas defining these sets.

To express that there is a directed path from a to b we say that there is a path in $\mathbb{R}^3$ such that a occurs before b on this path, and every point of $\bar{V}$ on this path that is not the final point is followed by a point in Tail, every point in Tail is followed by a point in Head, and every point in Head is followed by a point in $\bar{V}$. To formalize this in L_{PATH}, we let

$$\varphi_{\text{Empty}}(q_1, q_2) = \forall q_3 \Big((q_1 < q_3 \wedge q_3 < q_2)) \rightarrow \neg(\varphi_V(q_3) \vee \varphi_{\text{Tail}}(q_3) \vee \varphi_{\text{Head}}(q_3)) \Big).$$

Then the following L_{PATH}-formula defines directed reachability:

$$\begin{aligned} V(x) \wedge V(y) \wedge \exists \mathbf{P} \Big(& \exists p_1 \exists p_2 \ (p_1 = (x, 0, 0) \wedge p_2 = (y, 0, 0) \wedge p_1 \leq p_2) \\ & \wedge \forall q_1 \big((\varphi_V(q_1) \wedge \exists q_2 \ q_1 < q_2) \rightarrow \exists q_2 (\varphi_{\text{Tail}}(q_2) \wedge \varphi_{\text{Empty}}(q_1, q_2)) \big) \\ & \wedge \forall q_1 (\varphi_{\text{Tail}}(q_1) \rightarrow \exists q_2 (\varphi_{\text{Head}}(q_2) \wedge \varphi_{\text{Empty}}(q_1, q_2))) \\ & \wedge \forall q_1 (\varphi_{\text{Head}}(q_1) \rightarrow \exists q_2 (\varphi_V(q_2) \wedge \varphi_{\text{Empty}}(q_1, q_2))) \Big). \end{aligned}$$

□

Proof of Theorem 3

We prove the theorem in two steps: we first show that an L_{PATH} query can be transformed into a "discrete path query" in which instead of quantifying over any time in a path P, we quantify only over components of the path with respect to some discretization of the path. In the second step we show that these discrete path queries can be evaluated on a database by model-checking the adjacency structure of the appropriate cell-decomposition.

We start by making the first step more precise, by introducing a restricted logic that will be used as a normal form.

Let $\mathcal{A} = \{A_1 \dots A_k\}$ be some finite collection of Ω-definable sets in $\mathbb{R}^n$. Fix a non-zeno path $P : \mathbb{R}_+ \rightarrow \mathbb{R}^n$. For $r, s \in R_+$, we say that r and s agree on $\mathcal{A}$ if $P(r) \in A_i \Leftrightarrow P(s) \in A_i$ for all $i = 1, \dots, k$. We then write $r <_{\mathcal{A}} s$ if there is t such that $r < t < s$ and t does not agree with r on $\mathcal{A}$. Let $=_{\mathcal{A}}$ be the equivalence class associated with $<_{\mathcal{A}}$ (that is, $\neg(r <_{\mathcal{A}} s \vee s <_{\mathcal{A}} r)$). Note that the relation $r <_{\mathcal{A}} s$ is actually definable by a path formula over $\mathcal{A}$ in $L_{\text{PATH}}(\Omega)$, as is the relation $=_{\mathcal{A}}$. We also note that, since P was assumed to be non-zeno, we have that the ordering $<_{\mathcal{A}}$ is actually discrete. It is also clear from the definition of $=_{\mathcal{A}}$ that each equivalence class of $=_{\mathcal{A}}$ is an interval.

Now suppose we have a finite collection of $L_{\text{PATH}}(\Omega)$ state formulae $\mathcal{A} = \{A_1(\vec{x}, \vec{y}) \dots A_k(\vec{x}, \vec{y})\}$, each of arity $n + m$.

We now introduce a new path formula $p <_{\mathcal{A}(\vec{y})} q$ with free path variables p, q and free element variables $\vec{y} = (y_1, \dots, y_m)$, which holds in a database interpreting the predicates in $\mathcal{A}$ by Ω-definable sets $X^1 \dots X^k$, for non-zeno path P in $\mathbb{R}^n$, path elements p_0, q_0 and $\vec{c} \in \mathbb{R}^m$ exactly when $p_0 <_{\mathcal{X}} q_0$ where $\mathcal{X} = \{X^i_{\vec{c}}\}$. Recall that $X^i_{\vec{c}} = \{\vec{b} \mid (\vec{b}, \vec{c}) \in X^i\} \subseteq \mathbb{R}^n$. Similarly we introduce the formula $=_{\mathcal{A}(\vec{y})}$ saying two path variables are in the same equivalence class modulo $\mathcal{A}(\vec{y})$. Clearly, this can be expressed as queries in $L_{\text{PATH}}(\Omega)$ as well.

We consider a language $L^-_{\text{PATH}}(\Omega)$ that is built up as follows. For a finite collection $\mathcal{A} = \{A_i(\vec{x}, \vec{y})\}$ of first-order formulae (over Ω and the schema predicates) we have new atomic formulae:

- For every path variable p of arity the same as that of $\vec{x}$, $A_i(\vec{y})([p]_{\mathcal{A}})$ is an atomic formula with p and $\vec{y}$ free, with meaning the same as $A_i(p \leftarrow \vec{x}, \vec{y})$.
- For every path variable p with arity the same as that of $\vec{x}$, we have a formula $\mathbf{O}(p, \mathcal{A}(\vec{x}, \vec{y}), \vec{y})$, with free variables p and $\vec{y}$ meaning that the equivalence class of p under $=_{\mathcal{A}}$ is an open interval. If free element variables are clear from the context, we use the abbreviation $\mathbf{O}([p]_A)$.

  Similarly, we have formulae $\mathbf{C}([p]_A), \mathbf{OC}([p]_A), \mathbf{CO}([p]_A)$, meaning that the equivalence class of p under $=_{\mathcal{A}(\vec{y})}$ is closed, open on right and closed on left, closed on right open on left, respectively.

  We also have a formula $\mathbf{S}(p, A(\vec{x}, \vec{y}), \vec{y})$ (abbreviated as $\mathbf{S}([p]_A)$ if $\vec{y}$ is understood), meaning that the equivalence class of p under $=_{\mathcal{A}(\vec{y})}$ consists of a single element.
- For path variables p and q of the appropriate arity, we have formulae $[p]_{\mathcal{A}} <_{\mathcal{A}(\vec{y})} [q]_{\mathcal{A}}$ with p, q and $\vec{y}$ free, and with the meaning the same as $p <_{\mathcal{A}(\vec{y})} q$. We also have the formula $[p]_{\mathcal{A}} = [q]_{\mathcal{A}}$.

We choose the abbreviations to underline the key point about these new atomic formulae, which is that they all talk only about the equivalence class of a path variable p.

As constructors of $L^{-}_{\mathrm{PATH}}(\Omega)$, we have only boolean operators (where the set A must be the same in each operand) and path-variable quantification. Neither path quantification $\exists\mathbf{P}$ nor element quantification is present.

It is clear that every query in $L^{-}_{\mathrm{PATH}}(\Omega)$ can be expressed in $L_{\mathrm{PATH}}(\Omega)$. We now reduce the expressivity of L_{PATH} to that of L^{-}_{PATH}.

With $\mathcal{A}(\vec{x}, \vec{y})$ a collection of first-order formulae as above, and p a path variable, let $\mathit{First}_{\mathcal{A}}(p)$ be a new formula that says p is the first element in the path in its $\mathcal{A}$ equivalence class. Similarly let $\mathit{Last}_{\mathcal{A}}(p)$ mean that p is the last element in the path in the $\mathcal{A}$ equivalence class of p. The next lemma explains exactly what an open formula of L_{PATH} can say about a set of path variables.

Lemma 2 *Suppose $\varphi(\vec{p}, \vec{w})$ is an $L_{\mathrm{PATH}}(\Omega)$ path formula (where $\vec{p}$ are path variables and $\vec{w}$ are ordinary variables) that has no occurrences of $\exists\mathbf{P}$. Then from φ we can effectively find a collection $\mathcal{A}$ of first-order formulae and a formula φ' that is equivalent to φ (on every database and every non-zeno path), and such that φ' is a boolean combination of formulae of L^{-}_{PATH} and atomic formulae $p_i <_{\mathcal{A}} p_j$, $\mathit{First}_{\mathcal{A}}(p_i)$, and $\mathit{Last}_{\mathcal{A}}(p_i)$. In particular, if φ has no free path-variables, then it is equivalent to a formula of $L^{-}_{\mathrm{PATH}}(\Omega)$.*

Proof of Lemma 2. We show this by induction on formula complexity. For atomic formulae of the form $p_1 < p_2$, this is clear. We consider the induction step. The steps for conjunction and disjunction simply involves combining statements about $\mathcal{A}$ and $\mathcal{B}$ equivalence classes for formula sets $\mathcal{A}$ and $\mathcal{B}$ to statements about equivalence classes modulo $\mathcal{AB} = \mathcal{A} \cup \mathcal{B}$. For example the induction step for conjunction reduces to taking a formula $\varphi(p_1, \ldots p_n, \vec{y})$ that is a boolean combination of $\varphi_i([p_1]_{\mathcal{A}}, \ldots [p_k]_{\mathcal{A}})$, $\mathit{First}_{\mathcal{A}}(p_i)$, $\mathit{Last}_{\mathcal{A}}(p_i)$, and a formula γ that is a boolean combination of $\gamma_i([p_1]_{\mathcal{B}}, \ldots [p_k]_{\mathcal{B}})$, $\mathit{First}_{\mathcal{B}}(p_i)$, $\mathit{Last}_{\mathcal{B}}(p_i)$, and writing it in terms of formulae mentioning only equivalence modulo $\mathcal{AB}$. Here $\mathcal{A}$ and $\mathcal{B}$ are collections of formulae with free variables $\vec{x}, \vec{y}$, and the formulae φ_i may have path variable quantifications in them.

We can do this simply by transforming the atomic formulae. For example, $p =_{\mathcal{A}} q$ is transformed to

$$\forall r\ ([p] \leq_{\mathcal{AB}} [r] \leq_{\mathcal{AB}} [q]\ \vee\ [q] \leq_{\mathcal{AB}} [r] \leq_{\mathcal{AB}} [p])\ \rightarrow\ \bigwedge_i A_i([r]_{\mathcal{AB}}) \leftrightarrow A_i([q]_{\mathcal{AB}})\ \wedge\ A_i([r]_{\mathcal{AB}}) \leftrightarrow A_i([p]_{\mathcal{AB}}).$$

Here we use abbreviation $[p] \leq_{\mathcal{AB}} [r]$ for $[p] <_{\mathcal{AB}} [r]\ \vee\ [p] =_{\mathcal{AB}} [r]$. The transformations for $[p] <_{\mathcal{A}} [q]$ and $A_i([p])$ are also straightforward.

The formula $\mathbf{O}([p]_{\mathcal{A}})$ is equivalent to

$$\exists q, r \left(\begin{array}{ll} & (\mathbf{O}([q]_{\mathcal{AB}}) \vee \mathbf{OC}([q]_{\mathcal{AB}})) \\ \wedge & (\mathbf{O}([r]_{\mathcal{AB}}) \vee \mathbf{CO}([r]_{\mathcal{AB}})) \\ \wedge & (q =_{\mathcal{A}} p =_{\mathcal{A}} r) \\ \wedge & \forall q'\ (q' <_{\mathcal{AB}} q \rightarrow (\neg(q' =_{\mathcal{A}} p))) \\ \wedge & \forall r'\ (r' >_{\mathcal{AB}} r \rightarrow (\neg(r' =_{\mathcal{A}} p))) \end{array} \right),$$

which can then be converted to the proper form, since we have seen above how to convert formulae using the relation $=_{\mathcal{A}}$ on path variables. The other interval types are similar, as are $\mathit{First}_{\mathcal{A}}$ and $\mathit{Last}_{\mathcal{A}}$.

For path variable substitution, suppose we have $\chi(p, \vec{y}) \equiv \varphi(p \leftarrow \vec{x}, \vec{y})$, where $\varphi(\vec{x}, \vec{y})$ is a state formula. Since we assume φ to have no quantifications of the form $\exists\mathbf{P}$, φ must be first-order, and hence χ is certainly in L^{-}_{PATH}, since it is $\chi([p]_{\mathcal{A}})$, where $\mathcal{A}$ consists of only $\chi(\vec{x}, \vec{y})$.

The last step is existential path variable quantification. Suppose we have a formula $\exists p_1 \varphi(p_1, p_2, \ldots p_k, \vec{y})$. By induction, φ is a Boolean combination of L^{-}_{PATH} formulae, inequalities among the p_i, and $\mathit{First}_{\mathcal{A}}(p_i)$ and $\mathit{Last}_{\mathcal{A}}(p_i)$ statements. By combining sets, we can assume that these last statements all refer to the same set of formulas $\mathcal{A}_i(\vec{x}, \vec{y})$. Without loss of generality, φ is

$$\bigvee_i (\varphi_i([p_1]_{\mathcal{A}}, \ldots, [p_k]_{\mathcal{A}}) \wedge t_i(p_1, \ldots, p_k),$$

where $\mathcal{A} = \{A_i(\vec{x}, \vec{y})\}$ is a collection of first-order formulae, φ_i is a statement about the ordering and interval types of equivalence classes, and the A_i labels of equivalence classes, while t_i is a simple ordering statement about $\vec{p}$, giving inequalites between them and which ones are first in their equivalence classes. We may assume w.l.o.g. that φ_i completely specifies the ordering relations $<_{\mathcal{A}}$ that hold among the (equivalence classes of) p_j's. Note that since $<$ refines $<_A$, we can assume that the $<$ ordering given in t_i is consistent with the ordering $<_{\mathcal{A}}$ given in φ_i (otherwise, we can eliminate this disjunct).

Let S be the collection of i such that φ_i specifies p_1 to be equivalent to some other p_j with $j > 1$. For $i \in S$, let $\delta(i)$ be any $1 < j \leq k$ such that p_1 is specified to be equivalent to p_j.

For $i \notin S$, the formula $\exists p_1 \varphi_i([p_1]_{\mathcal{A}}, \ldots, [p_k]_{\mathcal{A}}) \wedge t_i(p_1, \ldots, p_k)$ is equivalent to $\exists p_1 \varphi_i([p_1]_{\mathcal{A}}, \ldots, [p_k]_{\mathcal{A}}) \wedge t'_i(p_2, \ldots, p_k)$, where t'_i is obtained from t_i by eliminating all inequalities involving p_1. This is true because the $<_{\mathcal{A}}$ inequalities involving p_1 in φ_i already imply the inequalities involving p_1 in t_i.

For i in S, the formula $\exists p_1 \varphi_i([p_1]_{\mathcal{A}}, \ldots, [p_k]_{\mathcal{A}}) \wedge t_i(p_1, \ldots, p_k)$ is equivalent to

$$\varphi_i([p_{\delta(i)}]_{\mathcal{A}}, \ldots, [p_k]_{\mathcal{A}}) \ \wedge \ \exists p_1 \ (p_1 =_{\mathcal{A}} p_{\delta(i)} \wedge t_i(p_1, \ldots p_k)).$$

Furthermore, $\exists p_1 \ (p_1 =_{\mathcal{A}} p_{\delta(i)} \wedge t_i(p_1, \ldots p_k))$ can be converted into an atomic formula $t'_i(p_2 \ldots p_n)$, due to the fact that φ_i specifies the interval type (closed, open, etc.) of $[p_{\delta(i)}]_{\mathcal{A}}$, and each interval type has quantifier elimination in the language $<$, *First*, *Last* (although for some of the interval types, *First* or *Last* may be equivalent to false). This is true because Dense Linear Order without endpoints is known to have effective quantifier elimination in the language of order, and Dense Linear Order with endpoints has quantifier elimination in the language of order with constants for endpoints [10].

Thus, $\exists p_1 \varphi(p_1, \ldots, p_k)$ is a disjunct of formulae, each of which is equivalent to a boolean combination of L^-_{PATH} formulae and inequalities of the form $p_1 < p_2$, we have this holds for the entire formula, completing the proof of Lemma 2. $\square$

Our next lemma shows that L^-_{PATH} formulae can be evaluated effectively.

Lemma 3 *For every $L^-_{\mathrm{PATH}}(\Omega)$ query $\varphi(\vec{w})$, where the $\vec{w}$ are free element variables, and there are no free path variables, and for every Ω-definable database D, there is a first-order Ω-formula $\varphi_D(\vec{w})$ such that for every $\vec{a}$, $D \models \exists \mathbf{P} \varphi(\vec{a})$ iff $\mathcal{M} \models \varphi_D(\vec{a})$.*

If $\mathcal{M}$ is decidable, then φ_D can be found effectively from φ and D, and if $\mathcal{M}$ is either $\mathbf{R}_{\mathrm{lin}}$ or $\mathbf{R}$, then for every fixed φ, φ_D can be found in polynomial time in the representation of D.

Proof of Lemma 3. Let $\mathcal{A} = \{A_i(\vec{x}, \vec{y}) \mid i \leq K\}$ be the set of formulae over Ω (unioned with the schema) which are used in φ. For any Ω-definable database D, we construct a family of finite discrete structures parameterized by $\vec{y}$.

We do it as follows. Let A_i^D be $\{(\vec{a}, \vec{b}) \mid D \models A_i(\vec{a}, \vec{v})\}$. Applying Lemma 1, we find an adjacency-preserving cell-decomposition $\mathcal{B} = \{B^1, \ldots, B^N\}$, which is cylindric over the $\vec{y}$ coordinates, such that each A_i^D is a union of cells. Define an equivalence relation $\equiv$ on $\mathbb{R}^{|\vec{y}|}$ by letting $\vec{b} \equiv \vec{c}$ iff $\vec{b}$ and $\vec{c}$ are in the same cell of the projection $\mathcal{B}$ onto the $\vec{y}$ coordinates. It follows from the definition of adjacency preservation that the following are true for $\vec{b} \equiv \vec{c}$.

- For all $i \leq N$, $B^i_{\vec{b}} \neq \emptyset$ iff $B^i_{\vec{c}} \neq \emptyset$.
- For all $i, j \leq N$, $B^i_{\vec{b}}$ Adj $B^j_{\vec{b}}$ iff $B^i_{\vec{c}}$ Adj $B^j_{\vec{c}}$, where E Adj F means $\mathrm{cl}(E) \cap F \neq \emptyset$.
- For all $i, j, k \leq N$, we have $T(B^i_{\vec{b}}, B^j_{\vec{b}}, B^k_{\vec{b}})$ iff $T(B^i_{\vec{c}}, B^j_{\vec{c}}, B^k_{\vec{c}})$, where $T(E, F, G)$ means $E \cap \mathrm{cl}(F) \cap \mathrm{cl}(G) \neq \emptyset$.

For every $\vec{b}$, we form the *adjacency structure of* $\vec{b}$, which is a labeled multi-graph whose nodes are all of the nonempty sets of the form $B^i_{\vec{b}}$, with two binary relations, C and O, where C is the adjacency relation Adj defined above, and O is the inverse of the adjacency relation, and one ternary relation T, where T is the ternary adjacency relation defined above. Nodes are labeled according to which $A_i(\vec{x}, \vec{b})$ are satisfied in the node (necessarily by all elements of the node or by none).

Note that the equivalence relation above partitions the $\vec{y}$ plane according to the isomorphism type of the adjacency structure. Hence the set of $\vec{y}$s corresponding to any collection of isomorphism types is definable.

Given an adjacency structure, an *adjacency path* is a sequence of pairs $\langle\langle N(i), E(i)\rangle \mid i \in \mathbb{N}\rangle$, where $N(i)$ is a node and $E(i)$ is an edge (either an O edge or a C edge) out of node $N(i)$. We first show the following.

Claim 1 *If* $\vec{c} \equiv \vec{d}$, *then they agree on* $\exists \mathbf{P}\varphi$; *that is,* $\exists \mathbf{P}\varphi(\vec{c})$ *holds iff* $\exists \mathbf{P}\varphi(\vec{d})$ *holds.*

Proof of Claim 1. Suppose we have $\vec{c} \equiv \vec{d}$. Then the adjacency structures of $\vec{c}$ and $\vec{d}$ are isomorphic (with the natural isomorphism that sends $B^i_{\vec{c}}$ to $B^i_{\vec{d}}$). Suppose we are given a non-zeno path P that witnesses that $\exists \mathbf{P}\varphi(\vec{c})$. Then (by non-zeno-ness) P is the union of P_i on intervals I_i, where I_i is a maximal subinterval on which P is contained in a particular component $M(i)$ of the cell decomposition. P maps into a path through the adjacency graph of $\vec{c}$, by taking $M(i)$s to be the sequence of components hit by P, and with the edge associated to $M(i)$ being an O if I_i is closed on the right and C if I_i is open on the right. Call this path P'; it runs through the adjacency graph of $\vec{c}$. Since the adjacency graph of $\vec{d}$ is isomorphic to that of $\vec{c}$, there is a corresponding path $Q' = \langle\langle N(i), X(i)\rangle \mid i \in \mathbb{N}\rangle$ through the adjacency graph of $\vec{d}$ with the same edges $X(i) = O$ or C as in P' and the same node labels, and also such that for every i, j, k, $T(M(i), M(j), M(k))$ holds iff $T(N(i), N(j), N(k))$ holds.

We now have to build a non-zeno path Q in euclidean space corresponding to Q'. We will define Q as $\bigcup_n Q_n$ where the partial functions Q_n with domain J_n will be defined inductively below. We will preserve the following properties in the construction:

- If $P'_n = \langle M(n), X\rangle$, where $X = C$ or O, then Q_n takes all its values in $N(n)$.
- If $P'_n = \langle M(n), O\rangle$, then the domain J_n of Q_n is closed on the right and the value of Q_n at the right endpoint is in the closure of $N(n+1)$.
- If $P'_n = \langle N(n), C\rangle$, then the domain J_n is open on the right and the limit of the path Q_n as we approach the right endpoint is in $N(n+1)$.
- Suppose I_n is open on the right, and the right-hand limit of P_n is the same as the left-hand limit of P_{n+2} (that is, I_{n+1} is a singleton). Then J_n is open on the right, and the right hand-limit of Q_n is in $N(n+2)$.
- I_n, the domain of the n-th component of the path P, is a singleton, iff J_n, the domain of Q_n, is a singleton.

Inductive construction. Suppose we have constructed $Q_1 \ldots Q_{n-1}$, and now want to construct J_n and Q_n. There are several cases to consider, depending on whether the edge from Q'_{n-1} to Q'_n was O or C, and depending on whether I_n and I_{n+1} are singletons or not.

Case 1. Suppose we have $P'_{n-1} = \langle M(n-1), C\rangle$, $P'_n = \langle M(n), O\rangle$, I_n is not a singleton, and I_{n+1} is not a singleton.

Then by construction J_{n-1} is open on the right with some right endpoint l, and the path Q_{n-1} converges to some point x in $N(n)$ as the domain element goes to l. Since there is an O edge from $N(n)$ to $N(n+1)$ (which follows since this is preserved from P' to Q' by assumption), there must be some point y in $N(n)$ that is in the closure of $N(n+1)$. Let Q_n be any smooth path on a nondegenerate closed interval going from x to y that remains in $N(n)$. Such a path exists since $N(n)$ is connected. In fact, by the curve selection lemma [33], it can be chosen to be definable in $\mathcal{M}$. The domain can be made nondegenerate even if $x = y$, by making the path constant.

Case 2. Suppose we have $P'_{n-1} = \langle M(n-1), C\rangle$, $Q'_n = \langle M(n), O\rangle$, I_n is a singleton, and (hence) I_{n+1} is not a singleton.

Then by construction J_{n-1} is open on the right with some right endpoint l, and the path Q_{n-1} converges to some point x in $N(n)$ as the domain element goes to l. Since I_n is a singleton it must further be true (by the inductive assumption on this construction) that x is in the closure of $N(n+1)$. Let Q_n map the single point l to x.

Case 3. Suppose we have $P'_{n-1} = \langle M(n-1), C\rangle$, $P'_n = \langle M(n), C\rangle$, and I_{n+1} is a singleton.

Then by construction J_{n-1} is open on the right with some right endpoint l, and the path Q_{n-1} converges to some point x in $N(n)$ as the domain element goes to l. Since there is a C edge from $M(n)$ to $M(n+1)$, the same must be true for $N(n)$ and $N(n+1)$. Since I_{n+1} is a singleton, it must be that the value of P_{n+1} on I_{n+1} is in $M(n+1)$ and in the closure of both $M(n)$ and $M(n+2)$. Hence there must be a point y in $N(n+1)$ that is in the closure of both $N(n)$ and $N(n+2)$ (since the paths P' and Q' had the same type with respect to the ternary relation T). Let Q_n be any path in $N(n)$ from a half-open interval that begins at x and converges toward y. Such a path exists since $N(n)$ is connected and y is in the closure of $N(n)$ (and again can be taken to be definable).

Case 4. Suppose we have $P'_{n-1} = \langle M(n-1), C\rangle$, $P'_n = \langle M(n), C\rangle$, and I_{n+1} is not a singleton.

This is similar to Case 3, but simpler, since we do not have to ensure that Q_n converges to a point in the closure of $N(n+2)$.

Case 5. Suppose we have $P'_{n-1} = \langle M(n-1), O\rangle$, $P'_n = \langle M(n), C\rangle$, and I_{n+1} is a singleton.

We have that J_{n-1} is closed on the right, and the value x of $Q(n-1)$ at the right endpoint is in the closure of $N(n)$. Since I_{n+1} is a singleton, we must have T holding of $M(n), M(n+1), M(n+2)$, hence also of $N(n), N(n+1), N(n+2)$, so there is a point y in $N(n+1)$ which is in $N(n)$ closure and $N(n+2)$ closure. Let Q_n be a path converging to x on the left and to y on the right.

Case 6. Suppose we have $P'_{n-1} = \langle M(n-1), O\rangle$, $P'_n = \langle M(n), C\rangle$, and I_{n+1} is not a singleton.

This is similar to Case 5, but simpler, since we have no obligations to fulfill on the right endpoint y.

Case 7. Suppose we have $P'_{n-1} = \langle M(n-1), O\rangle$, $P'_n = \langle M(n), O\rangle$ (hence I_n is not a singleton), and I_{n+1} is a singleton.

In this case we know inductively that the domain of Q_{n-1} is closed on the right, and that the value of Q_{n-1} at the right endpoint is in the closure of $N(n)$. Furthermore, we know, as in the previous arguments, that there is some point y in $N(n+1)$ that is in the closure of both $N(n)$ and $N(n+2)$. Take Q_n to be a path that converges to x on the right and y on the left.

Case 8. Suppose we have $P'_{n-1} = \langle M(n-1), O\rangle$, $P'_n = \langle M(n), O\rangle$ (hence I_n is not a singleton), and I_{n+1} is not a singleton.

This is similar to Case 7.

In the above, it is clear that by making the intervals appropriately wide, and making each path component Q_n definable (which can be done, since connected definable sets are definably connected [33]), the resulting path Q can be made to be non-zeno. We now want to verify that Q also witnesses that $\exists \mathbf{P}\varphi(\vec{d})$, using that P witnesses $\exists \mathbf{P}\varphi(\vec{c})$. This follows easily from the fact that L^-_{PATH} can only describe properties of equivalence classes of elements (Lemma 2). This completes the proof of the claim. ☐

Given Claim 1, and the fact that each adjacency type is described by a first-order formula, we conclude that that on D, the formula $\exists \mathbf{P}\varphi$ is equivalent to a first-order formula. Hence we have completed the proof of the first part of Lemma 3.

It remains to show how we can find out effectively (in polynomial time in the linear and polynomial constraint cases) which adjacency structures correspond to vectors $\vec{y}$ realizing $\exists \mathbf{P}\varphi$ on D. Since in the linear or polynomial cases the adjacency structures are of polynomial size in the complexity of D (see Lemma 1), and can be produced from D in polynomial time, the problem reduces to the following. Given φ and an adjacency structure AS, determine efficiently in the size of AS whether $\exists\mathbf{P}\varphi$ holds on all $\vec{y}$ with adjacency type AS or none of the $\vec{y}$ with that type.

We now show how to do this. From AS, we will form a Kripke structure $K(AS)$ composed of all five-tuples (B, X, E, X', F) where B, E, F are cells in AS and X, X' are transitions (O or C) from AS where $(B, E) \in X$ and $(E, F) \in X'$. The binary relation G of the Kripke structure relates any two tuples (B, X, E, X', F) and (E, X', F, X'', H). We add to this an extra copy τ' of each tuple τ of the form (B, C, E, O, F) where $E \cap \mathrm{cl}(B) \cap \mathrm{cl}(F) \neq \emptyset$. This tuple τ' has the same transitions in and out of it as τ does. We also add nodes of the form $(START, H, X, I)$, where H and I are cells and X is a transition in AS. For nodes τ of the form $(START, H, O, I)$, we add a copy τ'. Both τ and τ' transition to those nodes of the form (I, X', J, Y', K), and have no nodes transitioning into them.

We next define a finite alphabet Σ that consists of symbols L_{ij}, with $i \in \{1, 2, 3\}$ and $1 \leq j \leq K = \mid \mathcal{A} \mid$, $L_{CO}, L_{OC}, L_{CC}, L_{OO}$, $L_{sng}, L_{nsng}, L_{START}, L_{START,O}$ and $L_{START,C}$.

We now show how to label nodes in $K(AS)$ by symbols in Σ. In a tuple (B, X, E, X', F), we refer to B as the first, E as the second, and F as the third cell in it. We start by labeling a tuple depending on what elements of the original partition are satisfied by its cells. That is, label a tuple with L_{ij}, $i \leq 3, j \leq K$ exactly when the ith cell in the tuple is in A_j. Label elements of the form $(START, H, O, I)$ with L_{2j} exactly when H is in A_j and with L_{3j} when I is in A_j. We have a label L_{CO} for each tuple of the form (B, C, E, O, F), and similarly for labels L_{OC}, L_{CC}, and L_{OO}. For each tuple τ of the form (B, C, E, O, F) where $E \cap \mathrm{cl}(B) \cap \mathrm{cl}(F) \neq \emptyset$, we label τ' with L_{sng} and τ with L_{nsng}. For each tuple τ of the form $(START, H, O, I)$, we label τ' with L_{sng} and τ with L_{nsng}. Finally, we label each tuple of the form $(START, H, X, I)$ with L_{START}. We label $(START, H, O, I)$ with $L_{START,O}$ and $(START, H, C, I)$ with $L_{START,C}$.

We convert φ into a first-order formula φ' over ω-words in Σ^* with the language $<$ unioned with the label alphabet Σ. We can assume that there are no occurrences of $\equiv_{\mathcal{A}}$, since they can be reduced to $<_{\mathcal{A}}$. We translate atomic formulae $[p] <_{\mathcal{A}} [p']$ as

$$(p < p') \ \wedge \ \bigvee_i (\neg L_{2i}(p) \leftrightarrow L_{2i}(p')).$$

We translate the formula $\mathbf{OC}([p]_{\mathcal{A}})$ by

$$\exists p' \exists p'' \left(p' < p < p'' \wedge (L_{OC}(p') \vee L_{OO}(p')) \wedge (L_{OC}(p'') \vee L_{CC}(p'')) \wedge \forall q \ (p' \leq q \leq p'' \rightarrow \bigwedge_i (L_{2i}(q) \leftrightarrow L_{2i}(p))) \right)$$

and similarly for other interval types (analogously to the proof of the conjunction step in Lemma 2).

We translate the formula $\mathbf{S}([p]_{\mathcal{A}})$ into the disjunction of the formula

$$L_{sng}(p) \wedge \bigvee_j \neg(L_{1j}(p) \leftrightarrow L_{2j}(p)) \ \wedge \ \bigvee_j \neg(L_{3j}(p) \leftrightarrow L_{2j}(p))$$

with

$$L_{START}(p) \ \wedge \ \bigvee_j \neg(L_{2j}(p) \leftrightarrow L_{3j}(p)).$$

Finally, we translate statements $A_j([p]_A)$ by the corresponding labels $L_{2j}(p)$. Now a straightforward modification of the proof of Claim 1 shows the following.

Claim 2 *For every vector $\vec{c}$, if AS is the adjacency structure formed from $\vec{c}$, then there is a path through $K(AS)$ from some initial point of the form $(START, B, X, E)$, such that the ω-word corresponding to the path satisfies φ' iff $\exists\mathbf{P}\varphi(\vec{c})$ is satisfied.*

Proof sketch of Claim 2. Given a path P that witnesses $\exists\mathbf{P}\varphi(\vec{c})$, take the quotient path through AS. Then form a path P' through $K(AS)$ as follows: take the sequence of tuples hit on the quotient path through AS, with the caveat that if an $\mathcal{A}$-equivalence class E in the quotient is a singleton, we choose the tuple $(B, C, E, O, F$ that is labeled with L_{sng}. Otherwise we always choose the path through tuples not labeled with L_{sng}. We then modify this path by taking the initial tuple (B, X, E, Y, F) encountered on the path through AS and adding before it one of the two tuples in $K(AS)$ corresponding to $(START, B, X, E)$: we take τ' if the initial equivalence class in the quotient path though AS is a singleton (in which case, that singleton must have been contained in cell B), and take τ otherwise.

For $t \in R^+$, let $\delta(t)$ be the element in P' corresponding to $P(t)$. More precisely, $\delta(t)$ is the appropriate element (B, X, E, Y, F), where E is the cell containing the $P(t)$, or the appropriate element $(START, E, X, F)$ in the case that $P(t)$ is in the initial component E. Using the definition of the translation above, we see that atomic formulae on path elements $P(t)$ are satisfied exactly when the translation of those formulae holds of $\delta(t)$, which proves one direction of the claim.

Now suppose conversely that we have a path P' through $K(AS)$ that begins with an element of the required form. Clearly, this path corresponds to a path through AS. Now, from this path we construct a non-zeno path P just as in the proof of Claim 1. The only modification is this: we decide whether or not to make a particular path element P_n a singleton based on whether in the original path through $K(AS)$, the corresponding element was labeled with L_{sng}. This completes the proof of Claim 2.

The formula φ' can now be translated into a Büchi Automata BA, and one now needs to check whether the product of $K(AS)$ and BA has an accepting path, which can be done in polynomial time (see, e.g. [34]) in the size of $K(AS)$ with the size of BA being fixed. Since the size of $K(AS)$ is polynomial in the size of the cell decomposition that gives rise to AS, Lemma 1 implies decidability for decidable o-minimal structures, and polynomial time data complexity for $\mathbf{R}_{\text{lin}}$ and $\mathbf{R}$. This completes the proof of Lemma 3

Proof of Theorem 3. We finally prove closure and effective query evaluation by induction on the complexity of the formula φ. Clearly, atomic state formulae can be evaluated effectively if Ω is decidable, and in polynomial time in the polynomial and linear constraint cases. The induction step for boolean connectives is clear. In the inductive step for existential element quantification, closure and effectivity are immediate, and polynomial time data complexity follows from known results in constraint databases [23]. The interesting case is where we have a query of the form $\exists\mathbf{P}\psi(\vec{y})$. By the induction hypothesis, we can assume that every proper state subformula ψ_i of ψ can be evaluated (effectively, if $\mathcal{M}$ is decidable, and in polynomial time, in the polynomial or linear case). By replacing all maximal proper state subformulae with predicate symbols, we can consider ψ as a query over the outputs of these subformulae. and thus can assume that ψ has no existential path quantifiers within it. By Lemma 2, ρ can be transformed into an L^-_{PATH} query ψ' over these predicates, and by Lemma 3, $\exists\mathbf{P}\psi'$ can be converted into a first-order formula (effectively, for decidable $\mathcal{M}$, and in polynomial time, for polynomial and linear constraints). This completes the inductive proof. □

Remark. The proof can be simplified in the semi-linear case, where one does not need to consider the T relation among cells. Indeed, the proof of Lemma 1, a), implies that for fibers of three cells, $C^1_{\vec{a}}, C^2_{\vec{a}}, C^3_{\vec{a}}$, one has $C^1_{\vec{a}} \cap \text{cl}(C^2_{\vec{a}}) \cap \text{cl}(C^3_{\vec{a}}) \neq \emptyset$ iff $C^1_{\vec{a}} \cap \text{cl}(C^2_{\vec{a}}) \neq \emptyset$ and $C^1_{\vec{a}} \cap \text{cl}(C^3_{\vec{a}}) \neq \emptyset$. Thus, T can be reconstructed from C and O edges of the adjacency structure, which simplifies the construction of the Kripke structure $K(AS)$.

Proof of Theorem 4

Without loss of generality we can assume that $\mathcal{M}$ is an expansion of the real field. We fix a dimension n and a signature $\Sigma := \{\pi_1, \ldots, \pi_m\}$. Let $\sigma := \{<, P_1, \ldots, P_m\}$, where $P_1, \ldots, P_m$ are unary relation symbols. When we speak of a hybrid system in the following, we always assume it to be $\mathcal{M}$-definable and n-dimensional. When we speak of an interpreted hybrid system (H, Π), we also assume it to be $\mathcal{M}$-definable and of signature Σ.

We consider interpreted hybrid systems as database instances over the schema $SC := \{Q, S_0, S_F, T, E, I, G, R, R_1, \ldots, R_m\}$, where Q is unary, S_0, S_F are $(n+1)$-ary, T is $(2n+1)$-ary, E is binary, I is $(n+1)$-ary, G, R are $(n+2)$-ary, and $R_1, \ldots, R_m$ are $(n+1)$-ary.

We shall prove that for every LTL-formula φ there is an L_{PATH}-formula φ^* such that for every interpreted hybrid system (H,Π) we have $(H,\Pi) \models \varphi^*$ if, and only if, for every trajectory τ of H the ω-word $\Pi(\tau)$ satisfies φ. Furthermore, the translation $\varphi \mapsto \varphi^*$ is effective, uniformly over all n and all vocabularies.

It is well-known that every LTL-formula φ of signature Σ can effectively be transformed into an equivalent FO$[\sigma]$-sentence φ'. What we actually show in the following is how to translate an FO$[\sigma]$-sentence φ into an L_{PATH}-formula φ^* such that for every interpreted hybrid system (H,Π) we have $(H,\Pi) \models \varphi$ if, and only if, there *exists* a trajectory τ of H such that the word $\Pi(\tau)$ satisfies $\varphi(x)$. Clearly, this is sufficient.

At first sight it seems very simple: We just let φ^* be a formula of the form $\exists \mathbf{P}\varphi'$, where φ' is more or less the same as our original φ. The path whose existence we state by $\exists \mathbf{P}$ is supposed to be the state-sequence of a "run" of the hybrid system. Of course this simple-minded approach does not work, mainly for the following reasons:

1. Because of the discrete transitions, a run of a hybrid system is not a continuous curve.
2. In a continuous transition the hybrid system moves along an integral curve of some vector field $F(q,\cdot)$ that originated at some point $\vec{x} \in \mathbb{R}^n$. However, in a path-formula we cannot say that our path is on such a curve (unless we treat $\vec{x}$ as a parameter).
3. A trajectory is only a discrete abstraction of a run of the hybrid system.

Despite these problems, we will follow the basic idea.

We start by formally defining a *run* of a hybrid system $H = (S, S_0, S_F, F, E, I, G, R)$: It is a sequence $(r_i, x_i, r_i')_{i\geq 1}$ of triples such that for all $i \geq 1$ either $x_i = \emptyset$ and $r_i = r_i'$ or $r_i = (q,\vec{x})$, $r_i' = (q,\vec{x}')$ and $x_i : [0,\delta] \to \mathbb{R}^n$ is a curve such that $x_i(0) = \vec{x}$, $x_i(\delta) = \vec{x}'$ and for every $t \in [0,\delta]$ it satisfies $\dot{x}_i(t) = F(q, x_i(t))$ and $x_i(t) \in I(q)$. Furthermore, for all $i \geq 1$ there is a *discrete* transition $r_i' \xrightarrow{e} r_{i+1}$ (for an $e \in E$).

Let us forget about hybrid systems for a moment and just talk about arbitrary runs and trajectories. Let a *run* be a sequence $(r_i, x_i, r_i')_{i\geq 1}$, where $r_i, r_i' \in \mathbb{R}^{n+1}$ and either $x_i = \emptyset$ and $r_i = r_{i'}$ or $r_i = (q,\vec{x})$, $r_i' = (q,\vec{x}')$ and $x_i : [0,\delta] \to \mathbb{R}^n$ for some $\delta > 0$ and $x_i(0) = \vec{x}$, $x_i(\delta) = \vec{x}'$. Similarly, let a *trajectory* be a sequence $(s_i, e_i)_{i\geq 1}$ where $s_i \in \mathbb{R}^{n+1}$ and $e_i \in \{e, c\}$.

We say that a trajectory $(s_i, e_i)_{i\geq 1}$ is *consistent* with a run $(r_i, x_i, r_i')_{i\geq 1}$ if there is a mapping $f : \mathbb{N} \to \mathbb{N}$ such that $f(1) = 1$, $s_1 = r_1$, and for all $i \geq 1$ we have:

- If $e_i = e$ then $f(i+1) = f(i) + 1$, $s_i = r'_{f(i)}$ and $s_{i+1} = r_{f(i+1)}$.
- If $e_i = c$ then $f(i+1) = f(i)$ and, assuming that $r_{f(i)} = (q,\vec{x})$ for some $q \in \mathbb{R}$, $\vec{x} \in \mathbb{R}^n$ and $x_{f(i)} : [0,\delta] \to \mathbb{R}$, there are $t < t' \in [0,\delta]$ such that $s_i = (q, x_{f(i)}(t))$ and $s_i' = (q, x_{f(i)}(t'))$.

Let $r \geq 1$; r is going to be the quantifier rank of the input-formula $\varphi(x)$. In the following, let *words* be structures W of some vocabulary ν consisting of the binary relation symbol $<$ that is always interpreted as a linear order of the universe and finitely many unary relation symbols. In particular, let an ω-word be a word whose universe is $\mathbb{N}$. A *subword* of a word W is a substructure V of W such that if W has a first-element then V has the same first element and if W has a last element then V has the same last element.

The *r-type* of a word W of vocabulary ν is the set of all FO$[\nu]$-sentences of quantifier-rank at most r it satisfies. Note that there are only finitely many r-types of a fixed vocabulary.

Lemma 4 *There is a computable function f such that the following holds: Let $r \geq 1$ and W, W' be words of vocabulary σ whose universes are closed intervals in $\mathbb{R}$ such that W and W' have the same $f(r)$-type. Then for every finite subword V of W there is a finite subword V' of W' such that V and V' have the same r-type (and vice versa).*

Moreover, every FO$[\sigma]$*-sentence ψ of quantifier-rank at most r can be effectively transformed into a set $\Theta(\psi)$ of $f(r)$-types such that for every word W whose universe is a closed interval in $\mathbb{R}$ we have: The $f(r)$-type of W is contained in $\Theta(\psi)$ if, and only if, there is a finite subword V of W that satisfies ψ.*

Proof Sketch: Choose $f(r)$ such that every finite word V of length greater than $f(r)$ has a subword V^- of length at most $f(r)$ such that V and V^- have the same r-type. The existence of such an f follows from the fact that first-order definable languages are regular. The rest is easy.

Now suppose we are given a mapping $\Pi : \mathbb{R}^{n+1} \to 2^\Sigma$. For a triple (r, x, r'), where $r = (q, \vec{x})$, $r' = (q, \vec{x}')$ with $q \in \mathbb{R}$, $\vec{x}, \vec{x}' \in \mathbb{R}^n$ and $x : [0, \delta] \to \mathbb{R}^n$ we let $W(r, x_i, r')$ be the word of vocabulary σ with universe $[0, \delta]$ and $P_i := \{t \in [0, \delta] \mid \pi_i \in \Pi(q, x(t))\}$. The *$r$-abstraction* of a run $\rho = (r_i, x_i, r'_i)_{i \geq 1}$ is the sequence $\rho_r = (r_i, \Theta_i, r'_i)_{i \geq 1}$, where for all $i \geq 1$ we have:

- If $x_i = \emptyset$ then $\Theta_i = \emptyset$.
- If $x_i : [0, \delta] \to \mathbb{R}^n$, then Θ_i is the $f(r)$-type of the word $W(r_i, x_i, r'_i)$. Here f is taken from Lemma 4.

Let σ^+ be the extension of the vocabulary σ that contains a new unary relation symbol P_Θ for every $f(r)$-type Θ of vocabulary σ. We let $\Pi(\rho_r)$ be the ω-word W with $P_i^W := \{j \mid \pi_i \in \Pi(r_j) \text{ and } \Theta_j = \emptyset\}$ (for $1 \leq i \leq m$) and $P_\Theta := \{j \mid \Theta_j = \Theta\}$ (for every $f(r)$-type Θ). For a trajectory $\tau = (s_i, e_i)_{i \geq 1}$ we let $\Pi(\tau)$ be the ω-word $(\Pi(s_i))_{i \geq 1}$.

The following lemma is a generalization of Lemma 4 that is proved using "Feferman-Vaught"-type arguments and the fact that for every $f(r)$-type Θ there is a FO-sentence θ of quantifier-rank $f(r)$ such that a word satisfies θ if, and only if, its r-type is Θ.

Lemma 5 *Let f be as in Lemma 4. Let ρ, ρ' be runs such that $\Pi(\rho_r)$ and $\Pi(\rho'_r)$ have the same r-type. Then for every trajectory τ consistent with ρ there is a trajectory τ' consistent with ρ' such that $\Pi(\tau)$ and $\Pi(\tau')$ have the same r-type.*

Moreover, every $\mathrm{FO}[\sigma]$-sentence ψ of quantifier-rank at most r can be effectively transformed into an $\mathrm{FO}[\sigma^+]$-sentence ψ^+ such that for every run ρ we have: The word $\Pi(\rho_r)$ satisfies ψ^+ if, and only if, there is a trajectory τ consistent with ρ such that $\Pi(\tau)$ satisfies ψ.

We now return to hybrid systems. Remember that we wanted to translate a given $\mathrm{FO}[\sigma]$-sentence φ to an L_{PATH}-sentence φ^* such that for every interpreted hybrid system (H, Π) we have $(H, \Pi) \models \varphi^*$ if, and only if, there is a trajectory τ of H such that $\Pi(\tau)$ satisfies φ.

Applying Lemma 5, we first translate φ to a sentence φ^+. In the remaining proof we show how to translate φ^+ to an L_{PATH}-sentence φ^* such that for every interpreted hybrid system (H, Π) we have $(H, \Pi) \models \varphi^*$ if, and only if, there is a run ρ of H such that $\Pi(\rho_r)$ satisfies φ^+. What we have gained is that we have "filtered out" the continuous transitions and now only have to deal with a sequence of discrete transitions.

Let $\Theta_1, \ldots, \Theta_l$ be an enumeration of all $f(r)$-types of vocabulary σ. We can interpret the r-abstraction of a run as a sequence of tuples $(s, t, s') \in \mathbb{R}^{2n+3}$, where we replace the Θ either by $t := 0$ if $\Theta = \emptyset$ or by $t := i$ if $\Theta = \Theta_i$. How can we express that such a sequence is the r-abstraction of a run?

We call a tuple $(s, t, s') \in \mathbb{R}^{2n+3}$ *good* (with respect to an interpreted hybrid system (H, Π)) if $s, s' \in S$ and either $i = 0$ and $s = s'$ or $s \xrightarrow{c} s'$ via a curve $x : [0, \delta] \to \mathbb{R}^n$ such that the $f(r)$-type of the word $W(s, x, s')$ is Θ_t. The first thing we do is define an L_{PATH}-formula χ such that for every interpreted hybrid system (H, Π) and $\vec{z} \in \mathbb{R}^{2n+3}$ we have $(H, \Pi) \models \chi(\vec{z})$ if, and only if, $\vec{z}$ is good. Let θ_t be an $\mathrm{FO}[\sigma]$-formula defining the type Θ_t. Recall that $T := \{(q, \vec{x}, \vec{y}) \mid (q, \vec{x}) \xrightarrow{c} (q, \vec{y})\}$ is definable. Then the following formula says that $(q, \vec{x}) \xrightarrow{c} (q', \vec{x}')$ via a curve $x : [0, \delta] \to \mathbb{R}^n$ such that the $f(r)$-type of $W((q, \vec{x}), x, (q', \vec{x}'))$ is Θ_t:

$$\begin{aligned}
\xi(q, \vec{x}, q'\vec{x}') := (q = q') \wedge \exists \mathbf{P} \Big(& \exists p (p = \vec{x} \wedge \forall p''\, p \leq p'') \\
& \wedge \exists p' (p' = \vec{x}' \wedge \forall p''\, p'' \leq p') \\
& \wedge \forall p'' \big(T(q, \vec{x}, \vec{y}) \wedge I(q, \vec{y})\big)[p'' \leftarrow \vec{y}] \\
& \wedge \theta'_t \Big),
\end{aligned}$$

where θ'_t is the formula obtained from θ_t by replacing all variables by path variables and every atomic subformula $P_i(p)$ by $R_i(q,\vec{y})[p \leftarrow \vec{y}]$. Given the formula ξ, it is easy to define the desired χ defining the good tuples.

It will be convenient to make the following assumption for every hybrid system H:

> **(*)** All $e, e' \in E \subseteq \mathbb{R}^2$ are linearly independent (i.e. we do not have $\lambda e = e'$ for any $\lambda \in \mathbb{R}$).

(If this assumption does not hold, we can code the discrete transitions by elements of $\mathbb{R}^3$ in such a way that the additional component guarantees that they are pairwise linearly independent. This causes the dimension of our path to be $\mathbb{R}^{2n+7}$, but otherwise the proof goes through.)

Let H be a hybrid system. We model r-abstractions of runs of H by paths in $\mathbb{R}^{2n+6}$.

For $s_1, s'_1, s_2, s'_2 \in S$, $t, t' \in \mathbb{R}$, and $e \in E$ we write $(s_1, t, s'_1) \xrightarrow{e} (s_2, t, \vec{s}'_2)$ if (s_1, t, s'_1), $(s_2, t, \vec{s}'_2)$ are good and $s'_1 \xrightarrow{e} s_2$ in H. We write $\vec{z} \rightarrow^* \vec{z}'$ if there is a sequence $\vec{z}_1 = \vec{z}, \vec{z}_2, \ldots, \vec{z}_k = \vec{z}' \in \mathbb{R}^{2n+1}$ and a sequence $e_1, \ldots, e_{k-1} \in E$ such that $\vec{t}_i \xrightarrow{e_i} \vec{t}_{i+1}$ for $1 \le i \le k$.

For all $\vec{z}, \vec{z}' \in \mathbb{R}^{2n+3}$ and $e \in E$ such that $\vec{z} \xrightarrow{e} \vec{z}'$ we let $\gamma(e, \vec{z}, \vec{z}')$ be the curve connecting the following points in $\mathbb{R}^{2n+6}$ by straight-line segments:

$$(0,0,\vec{z}), \quad (e,1,\vec{z}), \quad (e,2,\vec{z}), \quad (e,2,\vec{z}'), \quad (e,3,\vec{z}'), \quad (0,0,\vec{z}').$$

(Recall that $E \subseteq \mathbb{R}^2$, so these points are indeed $(2n+6)$-tuples. The third place in these tuples will be used to encode the direction of the transitions). Let $\gamma^\circ(e, \vec{z}, \vec{z}')$ denote the interior of $\gamma(e, \vec{z}, \vec{z}')$, i.e. the curve obtained from $\gamma(e, \vec{z}, \vec{z}')$ by removing its endpoints.

For every transition $e \in E$ we let

$$\Gamma^\circ(e) := \bigcup_{\vec{z} \xrightarrow{e} \vec{z}'} \gamma^\circ(e, \vec{z}, \vec{z}'),$$

and we let $\Gamma^\circ := \bigcup_{e \in E} \Gamma^\circ(e)$. By our assumption **(*)**, for $e \neq f \in E$ we have $\Gamma^\circ(e) \cap \Gamma^\circ(f) = \emptyset$, and every path from a point in $\Gamma^\circ(e)$ to a point in $\Gamma^\circ(f)$ intersects $\mathbb{R}^{2n+6} \setminus (\Gamma^\circ(e) \cup \Gamma^\circ(f))$. (In other words, there is no connected component C of Γ° such that there are $e \neq f \in E$ with $C \cap \Gamma^\circ(e) \neq \emptyset$ and $C \cap \Gamma^\circ(f) \neq \emptyset$.)

Note that $\Gamma^\circ = \Gamma^\circ(H, \Pi)$ is definable in L_{PATH}, that is, there is an L_{PATH}-formula η such that for every interpreted hybrid system (H, Π) and $\vec{v} \in \mathbb{R}^{2n+6}$ we have $(H, \Pi) \models \eta(\vec{v})$ if, an only if, $\vec{v} \in \Gamma^\circ$.

Suppose for a moment that $\xrightarrow{e}$ is symmetric and reflexive for all $e \in E$. Then for all $\vec{z}, \vec{z}' \in T$ we have $\vec{z} \xrightarrow{e} \vec{z}'$ if, and only if, there is a path β from $(0,0,0,\vec{z})$ to $(0,0,0,\vec{z}')$ such that the interior β° of β is contained in $\Gamma^\circ(e)$. (To prove this we use that fact that there is no coupling between the initial and final state of a discrete transition, that is, that $(q, \vec{x}_1) \xrightarrow{e} (q', \vec{x}'_1)$ and $(q, \vec{x}_2) \xrightarrow{e} (q', \vec{x}'_2)$ implies $(q, \vec{x}_1) \xrightarrow{e} (q', \vec{x}'_2)$.) It follows that there exists an $e \in E$ such that $\vec{z} \xrightarrow{e} \vec{z}'$ iff there is a path β from $(0,0,\vec{t})$ to $(0,0,\vec{t}')$ such that $\beta^\circ \subseteq \Gamma^\circ$.

Since in general $\xrightarrow{e}$ is not symmetric and reflexive, we have to encode the direction of the transitions. To do this, we define three sets

$$\begin{aligned}
\mathrm{GOOD} &:= \{0\} \times \{0\} \times \{0\} \times \{\vec{z} \in \mathbb{R}^{2n+3} \mid \vec{z} \text{ good}\},\\
\mathrm{TAIL} &:= (\mathbb{R}^2 \times \{1\} \times \mathbb{R}^{2n+3}) \cap \Gamma^\circ,\\
\mathrm{HEAD} &:= (\mathbb{R}^2 \times \{3\} \times \mathbb{R}^{2n+3}) \cap \Gamma^\circ,\\
\mathrm{REST} &:= \Gamma^\circ \setminus (\mathrm{TAIL} \cup \mathrm{HEAD}).
\end{aligned}$$

Since the good tuples and the set Γ° are definable in L_{PATH}, these sets are also definable.

To say that there is a run ρ such that $\Pi(\rho_r)$ satisfies φ^+ we say that there exists a path $\alpha \subseteq \mathbb{R}^{2n+6}$ with the following properties:

- α starts in a point in GOOD.

- Whenever a point in GOOD appears on α, it is followed by an interval in REST and then by a point in TAIL.
- Whenever a point in TAIL appears on α, it is followed by an interval in REST and then by a point in HEAD.
- Whenever a point in HEAD appears on α, it is followed by an interval in REST and then by a point in GOOD.
- The ω-word of vocabulary σ^+ with universe $\alpha \cap \mathrm{GOOD}$ and

  $$\begin{array}{rll} P_i := \{(0,0,0,s,t,s') \in \mathrm{GOOD} \mid t = 0 \text{ and } \pi_i \in \Pi(s)\} & (\text{for } 1 \leq i \leq m), \\ P_{\Theta_i} := \{(0,0,0,s,t,s') \in \mathrm{GOOD} \mid t = i\} & (\text{for } 1 \leq i \leq l) \end{array}$$

  satisfies φ^+.

The existence of such a path can be expressed in L_{PATH}.

There is one case that we have missed so far: It could be that after some point a trajectory does no more discrete transitions. Then the corresponding run would be a finite sequence. But this case can easily be included, using an analogue of Lemma 4 for ω-words. We omit the details. □