Reachability and Connectivity Queries in Constraint Databases

Michael Benedikt†  Martin Grohe‡  Leonid Libkin§  Luc Segoufin⁸
Bell Laboratories  University of Edinburgh  University of Toronto  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+LIR 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 [20, 25], 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

†Part of this work was done while M. Benedikt, M. Grohe and L. Libkin visited INRIA-Rocquencourt.
‡Bell Laboratories, 2701 Lucent Lane, Lisle, IL 60532, USA. E-mail: benedikt@research.bell-labs.com.
§Division of Informatics, University of Edinburgh, Edinburgh EH9 3JZ, Scotland, UK. Email: grohe@dcs.ed.ac.uk
¶Department of Computer Science, University of Toronto, Toronto, Ontario M5S 3H5, Canada. E-mail: libkin@cs.toronto.edu. Research affiliation: Bell Laboratories.
⁸INRIA-Rocquencourt, B.P. 105, Le Chesnay Cedex 78153, France. E-mail: Luc.Segoufin@inria.fr.
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 over the reals. 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 $\mathcal{M}$. In this paper we consider only structures over the real field $\mathbb{R}$.

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 [25]. However, first-order constraint queries are limited in much the same way as that Relational Calculus is limited. Natural recursive queries, such as the transitive closure of a graph, remain inexpressible in first-order constraint query languages. Even more importantly, geometric analogs of these queries that are crucial for spatial database applications are inexpressible as well. The query

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

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

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

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 [22, 24] 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 over $\mathcal{M}$. 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 [20]. 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 [23, 12, 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 $T$ of topological properties, we present a language, FO + $T$ that can define $T$, and that is still closed. Moreover for the polynomial and linear constraint model we can prove that the complexity of evaluating queries in FO + $T$ is polynomial in the complexity of checking the properties from $T$. This implies a PTIME complexity for the extension of CONNECTED. The language FO + $T$ 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 [31], as over linear and polynomial constraints Datalog is not closed. Topological connectivity for 2-dimensional polygonal constraint databases of degree 2 was shown to be definable in a language called Spatial Datalog [23]; later [12] extended this to arbitrary polynomial constraint databases. However, it is likely that Spatial Datalog is computationally complete, and thus does not admit efficient evaluation [13]. 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., [17]). However, languages capturing complexity classes over constraint databases are scarce (natural languages exist for databases definable with order constraints only [15]; also, rather complicated languages were given for linear constraints [16, 21]). 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 [28, 22, 24, 33] and it is well known that connectivity and reachability are not first-order [4]. Our results on application to hybrid systems are directly inspired by [26, 27].
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 in Section 5 by giving an application to verification of hybrid systems.

2 Notations

Structures, databases, queries Most notations are fairly standard in the literature on constraint databases, cf. [4, 5, 25, 29]. Let $\mathcal{M} = (\mathcal{U}, \Omega)$ be an infinite structure, where $\mathcal{U}$ is an infinite set, called a universe (in the database literature often called the domain), and $\Omega$ is a set of interpreted functions, constants, and predicates. A set $X \subseteq \mathcal{U}^n$ is definable in $\mathcal{M}$ if there is a formula $\alpha(x_1, \ldots, x_n)$ in the language of $\mathcal{M}$ such that $X = \{ \bar{a} \in \mathcal{U}^n \mid \mathcal{M} \models \alpha(\bar{a}) \}$.

In this paper, we will always 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: $(\mathbb{R}, <)$;
- Linear Constraints: $R_{lin} = (\mathbb{R}, +, -, 0, 1, <)$;
- Polynomial Constraints: $R = (\mathbb{R}, +, -, 0, 1, <)$.

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 definable 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 $X_i = \{ \bar{a} \in \mathcal{U}^{p_i} \mid \mathcal{M} \models \alpha_i(\bar{a}) \}$. Most applications of constraint databases consider definable instances over $R_{lin}$ (called semi-linear sets) or over $R$ (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 the notation $FO + \Omega$ to denote the class of all first-order formulæ built up from the atomic $SC$ and $\Omega$ formulæ by using Boolean connectives $\forall, \wedge, \rightarrow$ and quantifiers $\forall, \exists$. When $\Omega$ is $(+, -, 0, 1, <)$, we use the notation $FO + LIN$ (first-order with linear constraints), and when $\Omega$ is $(+, -, 0, 1, <)$, we denote the language by $FO + POLY$ (first-order with polynomial constraints).

Given $\varphi(\bar{x}, \bar{y})$ and $\bar{a}$, we write $\varphi(\bar{a}, D)$ for $\{ \bar{b} \mid D \models \varphi(\bar{a}, \bar{b}) \}$; in the absence of $\bar{x}$ we just write $\varphi(D)$ for the output of $\varphi$ on $D$. We say that a language $FO + \Omega$ is closed, if for any schema $SC$, any definable $SC$-database (over $\langle \mathbb{R}, \Omega \rangle$) and every $FO + \Omega$ query $\varphi(\bar{y})$ 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 \( \mathbb{R}_{\text{lin}}, \mathbb{R} \) and \( (\mathbb{R}, <) \) [25].

**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* [35], which plays an important role in the study of constraint query languages (cf. [4, 5, 25]).

A structure \( \mathcal{M} = (\mathbb{R}, \Omega) \) is o-minimal, if every definable subset of \( \mathbb{R} \) 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 \( \Omega \)). All the structures on the reals we mentioned so far - \( \mathbb{R}_{\text{lin}}, \mathbb{R}, (\mathbb{R}, <) \) - are o-minimal (this is implied by quantifier elimination and the fundamental theorem of algebra, for the case of \( \mathbb{R} \)). There are a number of known o-minimal expansions of \( \mathbb{R} \), most notably, the exponential field \( (\mathbb{R}, +, \cdot, e^x) \) [37].

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 \( \mathcal{M} = (\mathbb{R}, \Omega) \) and define \( \mathcal{M} \)-cells by induction on dimension. An \( \mathcal{M} \)-cell in \( \mathbb{R}^n \) is just \( \mathbb{R}^n \). \( \mathcal{M} \)-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 functions 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 \( \infty \).

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 \( \{(-\infty, a_1), \{a_1, a_2\}, \{a_2, \ldots, a_k\}, (a_k, \infty)\} \). A decomposition of \( \mathbb{R}^n \) is a family \( C = \{C_1, \ldots, C_l\} \) 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 \( \pi(x,t) = x \) for all \( t \), the collection \( \pi(C) = \{\pi(C_1), \ldots, \pi(C_l)\} \) is a decomposition of \( \mathbb{R}^{n-1} \).

In particular, if \( C \) 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 \( \pi(C) \) is a decomposition of \( \mathbb{R}^n \), and for every cell \( C \in \mathbb{R}^n \) there exists a unique cell \( C_0 \in \pi(C) \) 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** [35] Let \( \mathcal{M} = (\mathbb{R}, \Omega) \) be o-minimal. Assume that \( S_1, \ldots, S_m \) are definable sets in \( \mathbb{R}^n \). Then there exists a cell decomposition \( C \) of \( \mathbb{R}^n \) such that each \( S_i \) is a union of some cells of \( C \). \( \Box \)

Let \( X \subseteq \mathbb{R}^{n+m} \) and \( \bar{a} \in \mathbb{R}^n \). Then \( X_{\bar{a}} \) denotes the fiber \( \{ \bar{b} \in \mathbb{R}^m \mid \langle \bar{a}, \bar{b} \rangle \in X \} \). Let \( C \) be a decomposition of \( \mathbb{R}^{n+m} \) and \( \bar{a} \in \mathbb{R}^n \). By \( C_{\bar{a}} \) we denote the collection of all nonempty sets \( \{C_{\bar{a}} \mid C \in \mathcal{C}\} \). It is known [35] that \( C_{\bar{a}} \) is a cell decomposition of \( \mathbb{R}^m \).

In the case when \( \mathcal{M} \) is \( \mathbb{R}_{\text{lin}} \) or \( \mathbb{R} \), we can get more information about cell decompositions. Namely, given any finite collection \( f_1(x), \ldots, f_k(x) \) of polynomials (resp., linear functions) in \( n \) variables with coefficients from \( \mathbb{Q} \), one can find a cell decomposition \( C \) 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 \( O(\max\{h|k|\}) \), where \( d \) is the maximal degree of a polynomial among \( f_j \)'s, and \( h \) is some function (typically, \( h(n) = O(2^n) \)) [10, 8]. It is important to notice that for a fixed dimension, the cell decomposition algorithm is thus in PTIME (in fact, in NC [6]).
We will need stronger notions of decomposition into cells. A decomposition $C$ of $\mathbb{R}^{n+m}$ is called *trivial* over $\mathbb{R}^{n}$ if it is cylindrical over $\mathbb{R}^{n}$ and for any cell $C'$ of the induced decomposition $\pi(C)$ of $\mathbb{R}^{n}$, and for any $\bar{a}, \bar{b} \in C'$, there exists a homeomorphism $h : \mathbb{R}^{m} \rightarrow \mathbb{R}^{m}$ such that $h(C_{\bar{a}}) = C_{\bar{b}}$ for every cell $C \in C$.

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

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 $S_{\bar{a}} \subseteq \mathbb{R}^{m}$ is finite, as $\bar{a}$ ranges over $\mathbb{R}^{n}$ [3, 7]. For $\mathbb{R}$, the known Local Triviality theorem gives the decompositions necessary for our results. In $\mathbb{R}_{lin}$, 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. Notice that the Local Triviality Theorem does not hold in the semi-linear case as the homeomorphisms defined by this theorem could be non semi-linear.

3.1 Decomposition Lemma

The key lemma for our results is the following:

**Lemma 2**  

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 $C$ 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 $C$. In particular, every $S_{i}$ is a union of cells of $C$. Moreover, for $n$ and $m$ fixed, $C$ 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 $\mathcal{M} = (\mathbb{R}, \Omega)$ be $\sigma$-minimal, and let $S_1, \ldots, S_k$ be a collection of definable sets in $\mathbb{R}^{n+m}$. Then there exists a decomposition $C$ 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 $C$. Moreover, if $\mathcal{M}$ is decidable, then $C$ can be effectively computed.

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

Proof: In the proofs of a) and b) and d), 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 $A_i \cap \text{cl}(A_j) = \emptyset$ iff $A_i \subset \text{cl}(A_j)$ 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

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

where $\sigma$ ranges over the functions from $\{1, \ldots, m\}$ 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 $\geq$. Notice that a stratification is not necessarily a cylindrical decomposition.

We start the proof with b), as it is an easy consequence of the Local Triviality Theorem in algebraic geometry [3, 7]. Let $f_1, \ldots, f_p$ be all the polynomials used in a given representation of the $S_i$s, 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 $\mathbb{R}^{k_i}$ for some $k_i \leq n + m$, and the polynomials $f_j$s 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 $\Pi$ and $X_1, \ldots, X_s$, states that there exists a stratification $Z_1, \ldots, Z_\alpha$ of $\mathbb{R}^n$ and, for each $1 \leq i \leq \alpha$, a semi-algebraic set $F_i \subset \mathbb{R}^m$, a semi-algebraic partition $\{F_{ij} \mid 1 \leq j \leq \beta_i\}$ 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 $\vec{a} \in Z_i$, $h_i^\alpha : \mathbb{R}^m \to F_i$ that sends $\vec{y}$ to $\vec{z}$ is a homeomorphism, and is onto (since $X_i$s 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 $X_j$. First, define $\text{id}_{\vec{a}, \vec{b}} : \{\vec{a}\} \times F_i \to \{\vec{b}\} \times F_i$ 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

$$\pi \circ h_i^{-1} \circ \text{id}_{\vec{a}, \vec{b}} \circ h_i \circ \{\vec{a}\} \times \text{id}.$$
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 \( V_1, \ldots, V_t \) 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 \( C \) consisting of nonempty sets among \( (V_i \times \mathbb{R}^m) \cap X_j \). Since we took \( X_j \)'s to be a cylindric decomposition, and each \( V_i \) is contained in a projection of some cell from that decomposition, we obtain that \( C \) itself is a cylindric decomposition over \( \mathbb{R}^n \). Furthermore, since each \( V_i \) is subset of some \( Z_r \), we obtain from the paragraph above that \( C \) 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 \([8, 10]\). To see that the stratification \( Z_1, \ldots, Z_n \) 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 \( V_i \)'s, one computes all possible intersections \( Z_i \cap \Pi(X_j) \), and this is again polynomial for a fixed dimension. This completes the proof of b).

The proof of d) is identical, except for the last step, as the Local Triviality Theorem is known to hold in any \( \alpha \)-minimal expansion of the real field \([35]\); 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 \( \mathbb{R}_{\text{fin}} \) (see, for example, in \([35]\)). However, we can recover enough of it to prove a).

Let \( f_1(\bar{x}, \bar{y}), \ldots, f_p(\bar{x}, \bar{y}) \) 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 \( x_j \) (for each variable \( x_j \); this is done to ensure that none of the cells contains a line). Let us use the standard cylindric decomposition algorithm for linear functions, thus obtaining a cell decomposition \( C \) of \( \mathbb{R}^{n+m} \) such that on every cell of \( C \), the sign of each \( f_i \) remains constant. In particular, each \( S_j \) is then a union of cells. We claim that \( C \) satisfies the condition of the theorem.

First, the fact that it can be computed in time polynomial in the representation of all the \( f_i \)'s (for \( n \) and \( m \) fixed) is derived from the standard bounds on cell decomposition \([10]\). Second, analyzing the proof of the existence of stratifications for semi-algebraic sets (see, for example, \([3, 7]\)), one obtains that \( C \) 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 \( C' \in \pi(C) \) and \( \bar{a}, \bar{b} \in C' \subseteq \mathbb{R}^n \). Let \( C \in C \) be a cell in \( C' \times \mathbb{R}^m \). We first note that since \( \dim(C) = \dim(C') + \dim(C_L) \) for an arbitrary \( \tilde{c} \in C' \) \([35]\), we obtain that \( \dim(C_{\bar{a}}) = \dim(C_{\bar{b}}) \).

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

1. \( C_{\bar{a}} \cap \text{cl}(B_{\bar{a}}) \neq \emptyset \);
2. \( C \cap \text{cl}(B) \neq \emptyset \);
3. \( C \subseteq \text{cl}(B) \);
4. \( C_\bar{a} \subseteq \text{cl}(B_{\bar{b}}) \).

Note that \( 4 \rightarrow 1 \) is immediate, and \( 2 \rightarrow 3 \) follows from the fact that \( C \) is a stratification. Both \( 1 \rightarrow 2 \) and \( 3 \rightarrow 4 \) follow from the fact that for any cell \( C \subseteq C' \times \mathbb{R}^m \), and any \( \bar{a} \in C' \), we have \( \{ \bar{a} \} \times \text{cl}(C_{\bar{a}}) = \text{cl}(C) \cap (\{ \bar{a} \} \times \mathbb{R}^m) \). To prove this, assume that \( C \neq \emptyset \) is given by a conjunction of strict inequalities \( g_i(x, y) > 0 \), \( i = 1, \ldots, p \), and \( v_j(x, y) = 0 \), \( j = 1, \ldots, s \). Let \( G_i = \{ \bar{c} \mid g_i(\bar{a}, \bar{c}) > 0 \} \), \( \bar{c} \in \mathbb{R}^m \), and \( V_j = \{ \bar{c} \mid v_j(\bar{a}, \bar{c}) = 0 \} \). We have \( \text{cl}(G_i) = \{ \bar{c} \mid g_i(\bar{a}, \bar{c}) > 0 \} \), \( \text{cl}(G_i) = \{ \bar{c} \mid g_i(\bar{a}, \bar{c}) \geq 0 \} \). Let \( V_j = \cap_{i=1}^{\infty} V_j \). Then \( V_j \) is a closed set, and it intersects the open set \( \cap_{i=1}^{\infty} G_i \). Hence, \( \text{cl}(V_j) \cap \cap_{i=1}^{\infty} G_i \neq \emptyset \) and thus \( \text{cl}(C_{\bar{a}}) \cap \cap_{i=1}^{\infty} G_i = \text{cl}(V_j) \cap \cap_{i=1}^{\infty} G_i = V_j \). Thus, \( \text{cl}(C_{\bar{a}}) = \{ \bar{c} \mid g_i(\bar{a}, \bar{c}) \geq 0, v_j(\bar{a}, \bar{c}) = 0, i = 1, \ldots, p, j = 1, \ldots, s \} \), which proves \( \{ \bar{a} \} \times \text{cl}(C_{\bar{a}}) = \text{cl}(C) \cap (\{ \bar{a} \} \times \mathbb{R}^m) \), and hence the equivalences above.

The equivalences above show that the adjacency structures of \( C_{\bar{a}} \) and \( C_{\bar{b}} \) are the same for any \( \bar{a} \) and \( \bar{b} \) in \( C' \), and, moreover, the boundary of \( C_{\bar{a}} \) in \( \mathbb{R}^m \) is the union of cells of the form \( B_{\bar{b}} \), \( B \in C \), and likewise for \( \bar{b} \). Furthermore, the proof shows that \( \text{cl}(C_{\bar{a}}) \) 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 \( x_i \)s before computing the decomposition). Furthermore, since \( \text{cl}(C_{\bar{a}}) = \text{cl}(C_{\bar{b}}) \), we obtain from convex analysis (see [32]) that each face of \( \text{cl}(C_{\bar{a}}) \) is a union of cells of the form \( B_{\bar{b}} \), \( B \in C \). Thus, each vertex of \( \text{cl}(C_{\bar{a}}) \) is a cell of the above form, each segment face is a union of a bounded one-dimensional cell of the form \( B_{\bar{b}} \) and two vertices, and each ray face is a union of an unbounded one-dimensional cell and a vertex. The same statements hold for \( \text{cl}(C_{\bar{b}}) \) in view of the above equivalences.

We next show that for any cell \( B \in C, B_{\bar{a}} \) is bounded iff \( B_{\bar{b}} \) is bounded. Indeed, \( B_{\bar{a}} \) is bounded iff \( \text{cl}(B_{\bar{a}}) \) is bounded. By a simple induction on dimension, \( \text{cl}(B_{\bar{a}}) \) 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 \( A_{\bar{a}} \) contained in \( \text{cl}(B_{\bar{a}}) \) is adjacent to two vertices (0-dimensional cells). Since the adjacency structures of \( C_{\bar{a}} \) and \( C_{\bar{b}} \) are the same, we obtain from here that in \( \text{cl}(B_{\bar{b}}) \) every 1-dimensional face is bounded, and thus \( B_{\bar{b}} \) is bounded.

Next, we need the following observation. Let \( P_1 \) and \( P_2 \) be two 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 [34], this in particular implies that \( \text{bd}(P_1) \) is homeomorphic to \( \text{bd}(P_2) \). Fix any homeomorphism \( g : \text{bd}(P_1) \to \text{bd}(P_2) \). 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, \( \text{bd}(X) = h_1(\text{bd}(P_1)) = h_2(\text{bd}(P_2)) \)). Consider a map \( v \) from \( \text{bd}(X) \) 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 $B_k = \{ \bar{x} \mid \| \bar{x} \| \leq 1 \}$ (if it is bounded) or to $D_k = [0, 1]^k$ (if it is unbounded), see [34]. In the first case, a homeomorphism $g : \text{bd}(B_k) \to \text{bd}(B_k)$ is extended as follows. The origin is mapped to itself. Given $\bar{x} \in B_k$, let the ray from the origin in the direction of $\bar{x}$ intersect $\text{bd}(B_k)$ at $\bar{y}$. Then $G(\bar{x})$ is the point $\bar{x}_0$ on the segment between the origin and $g(\bar{y})$ such that $\| \bar{x} \| = \| \bar{x}_0 \|$. In the second case, consider any $\bar{x}$ in the interior of $D_k$. Let 1 stand for $(1, \ldots, 1)$. Let the ray originating in 1 and passing through $\bar{x}$ intersect $\text{bd}(D_k)$ at $\bar{y}$. Consider a point $\bar{x}_0$ on the segment between 1 and $g(\bar{y})$ such that

$$\frac{d(\bar{x}_0, 1)}{d(\bar{y}, 1)} = \frac{d(\bar{x}, 1)}{d(\bar{y}, 1)},$$

(where $d(\cdot, \cdot)$ is the usual Euclidean distance), and let $G(\bar{x}) = \bar{x}_0$. It is routine to verify that in both cases $G$ is a homeomorphism extending $g$.

Let now $C_a^k$ be the union of cells $C_a$ whose dimension is at most $k$, and likewise for $C_b^k$. Since for each cell $C_a$, $\text{cl}(C_a)$ is the union of lower-dimensional cells in $C_a$, we obtain that $C_a^k$ and $C_b^k$ are closed. We now conclude the proof of a) by induction, by showing that for every $k$, there is a homeomorphism $h_k : C_a^k \to C_b^k$ such that $h_k(B_a) = B_b$ for any cell $B$ with $\dim(B_a) \leq k$. For $k = 0$, the statement follows from the fact that $\dim(B_a) = \dim(B_b)$ for every $B \in C$; thus, $h_0$ maps every 0-dimensional cell (point) of the form $B_{a_0}$ to the point $B_{b_0}$.

For the induction step, assume we have already constructed $h_k$. Consider any cell $B \in C$ such that $\dim(B_a) = \dim(B_b) = k + 1$. Consider $\text{cl}(B_a) = B_a \cup \text{bd}(B_a)$. We know that $\text{bd}(B_a)$ is a subset of $C_a^k$ and moreover a union of cells. We thus have a mapping $g_B$ which is a restriction of $h_k$ on $\text{bd}(B_a)$ and therefore a homeomorphism $\text{bd}(B_a) \to \text{bd}(B_b)$ (because $\text{bd}(B_b)$ is a union of cells too, and the adjacency structures of $C_a$ and $C_b$ are the same). Note that $\text{cl}(B_a)$ and $\text{cl}(B_b)$ are both $k + 1$-dimensional polyhedra, none containing a line, and $\text{cl}(B_a)$ is bounded iff $\text{cl}(B_b)$ is. Applying the claim above, we extend $g_B$ to a homeomorphism $G_B : \text{cl}(B_a) \to \text{cl}(B_b)$; note that $G_B(B_a) = B_b$ as $G_B(\text{bd}(B_a)) = \text{bd}(B_b)$.

Let $B_1, \ldots, B_s$ be all the cells with $\dim(B_a^i) = k + 1$. Then $C_a^{k+1} = \bigcup_i \text{cl}(B_a^i)$ and $C_b^{k+1} = \bigcup_i \text{cl}(B_b^i)$. We have homeomorphisms $G_{B_i} : \text{cl}(B_a^i) \to \text{cl}(B_b^i)$ for each $i$. Note that $B_i$'s are pairwise disjoint, and for any $x \in \text{cl}(B_a^i) \cap \text{cl}(B_b^i)$ it is the case that $x \in C_a^k$ and $G_{B_i}(x) = G_{B_i}(x) = h_k(x)$. We thus can define $h_{k+1}$ as the union of all $G_{B_i}$. Clearly, it extends $h_k$ and $h_{k+1}(B_a^i) = B_b^i$ for all $i$. Elementary topology shows 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 $X_i$s and $Y_i$s 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 $Y$ (continuity of $f$ follows since if a sequence $(a_j)$ in $X$ converges, then a subsequence $a_{j_k}$ lying within one $X_i$ converges, and hence $f(a_{j_k})$ converges by continuity of the restriction. Applying the same argument to $f^{-1}$ gives that $f$ is a homeomorphism). This implies that $h_{k+1}$ is a homeomorphism $C_a^{k+1} \to C_b^{k+1}$, 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 $h(C_a) = C_b$ for every cell $C \in C$. This completes the proof of a).

We finally prove c). Start with a cell decomposition $C$ of $\mathbb{R}^{n+m}$ such that each $S_i$ is a union of cells,
and \(\pi(C)\) is a cell decomposition of \(\mathbb{R}^n\) (recall that \(\pi\) here is the natural projection onto the first \(n\) coordinates). It is known that each cell is definable in the structure \(\langle \mathbb{R}, <, S_1, \ldots, S_k \rangle\) [35]. To see what is needed in order to obtain formulæ defining each cell, one can check all the steps in the proof of cell decomposition for \(o\)-minimal structures (see, e.g., [30, 35]) and observe that the only step that is needed to ensure effectiveness is the calculation of uniform bounds. That is, for a formula \(\alpha(x, y)\), one should be able to calculate a number \(K\) such that for each \(\bar{a}\), the set \(\{c \mid M \models \alpha(c, \bar{a})\}\) is composed of fewer than \(K\) intervals. This can be done using the decidability of \(M\). For each number \(i\), we can write a sentence \(\Phi_i^\alpha\) stating that the set \(\{c \mid M \models \alpha(c, \bar{a})\}\) is composed of fewer than \(i\) intervals for all \(i\), and then check if \(M = \Phi_i^\alpha\). The uniform bounds theorem [30] says that there is a number \(K\) such that \(M = \Phi^\alpha_0\), and thus it can be found since \(M\) is decidable.

Now we have a decomposition of \(\mathbb{R}^{n+m}\) into, say, \(s\) cells. We consider a cell \(C' \in \pi(C)\) (note that the decomposition \(\pi(C)\) 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 \(\bar{a} \in C'\) such that for every \(1 \leq i, j, k \leq t\),
\[
(A_i^j \cap \text{cl}(A_i^j)) / \sigma(i, j) \emptyset
\]
and
\[
(A_i^j \cap \text{cl}(A_i^j) \cap \text{cl}(A_i^j)) / \theta(i, j, k) \emptyset.
\]
Since the closure of a definable set is definable in any \(o\)-minimal structure on \(\mathbb{R}\) [35], we obtain that \(C'_{\sigma, \theta}\) is definable by a formula \(\beta_{\sigma, \theta}(\bar{x})\). We now consider the collection \(F\) of all the \(2^{2}+t^3\) formulæ \(\beta_{\sigma, \theta}\), as \(\sigma\) and \(\theta\) 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 \(\hat{C}\) of \(\mathbb{R}^n\) so that every cell in \(\pi(C)\) and every set definable by \(\beta_{\sigma, \theta}\) is a union of cells. By the same argument as in the previous paragraph, if \(M\) is decidable we can effectively construct formulæ defining the cells of \(\hat{C}\).

Let \(\hat{C}\) 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 \(\hat{C}\) and \(A\) ranges over \(\hat{C}\). Clearly, \(\hat{C}\) is a decomposition of \(\mathbb{R}^{n+m}\) which is cylinder over \(\mathbb{R}^n\). Furthermore, every cell in the projection \(\pi(\hat{C})\) is a cell of \(C\). Next, fix a cell \(A\) in \(\pi(C)\). Let \(\bar{a}, \bar{b} \in \hat{A}\). Assume that for two cells \(C^1, C^2\) of \(\hat{C}\), we have \(C^1_a \cap \text{cl}(C^2_a) \neq \emptyset\). Since \(C^1 = C^2 \cap (A \times \mathbb{R}^m)\) for some cell \(C^2_0\) of \(C\) in the same cylinder over \(\mathbb{R}^n\), we have \(C^1_b \cap \text{cl}(C^2_b) \neq \emptyset\) if \(C^1_b \cap \text{cl}(C^2_b) \neq \emptyset\). The proof that \(C^1_b \cap \text{cl}(C^2_b) \cap \text{cl}(C^2_b) \neq \emptyset\) if \(C^1_b \cap \text{cl}(C^2_b) \cap \text{cl}(C^2_b) \neq \emptyset\) for any \(C^1, C^2, C^3 \in \hat{C}\) is identical. This shows that \(\hat{C}\) 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 as soon as \(M\) is decidable. This completes the proof of d), and the lemma.

\[\square\]

### 3.2 Closure theorem

We now prove the closure result for topological properties. Formally, a topological property \(\text{Top}\) is a collection \(\{T_1, \ldots, T_n, \ldots\}\) where \(T_n\) is a family of sets in \(\mathbb{R}^n\) such that if \(X \in T_n\), then for
each homeomorphism $h$ of $\mathbb{R}^n$, $h(X) \in \mathcal{T}_n$. For example, $\text{Top}$ 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 $X \in \text{Top}$ instead of $X \in \mathcal{T}_n$.

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

For instance, the query “the intersection of regions $R$ and $S$ connected” could be written as $C. R(\vec{x}) \land S(\vec{x})$ (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 $\varphi(x) \equiv C(y, z)$. $S(x, y, z)$ 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 $\text{FO}(\Omega) + \mathcal{T}$ is $\text{PTIME}^T$ if $\text{FO}(\Omega) + \mathcal{T}$ queries can be evaluated in polynomial time in the size of the database, assuming an oracle that can test each $\text{Top} \in \mathcal{T}$ in constant time.

**Theorem 3** Let $\mathcal{T}$ be any set of topological properties. Then $\text{FO} + \text{LIN} + \mathcal{T}, \text{FO} + \text{POLY} + \mathcal{T}$ and $\text{FO}(\Omega) + \mathcal{T}$ are closed, for $(\mathbb{R}, \Omega)$ an $\omega$-minimal expansion of the real field. Furthermore, the data complexity of $\text{FO} + \text{LIN} + \mathcal{T}$ and $\text{FO} + \text{POLY} + \mathcal{T}$ is $\text{PTIME}^T$.

**Proof.** The result is by a simple induction on the formulae. The only case to prove is $\psi(\vec{x}) \equiv \text{Top} \vec{y}$, $\varphi(\vec{x}, \vec{y})$ for $\text{Top} \in \mathcal{T}$. 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 2, there exists a decomposition $C$ 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 $C$. Let $C'$ be the projection of $C$ onto $\mathbb{R}^n$, and $C$ a cell in $C'$. 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 $\text{Top}$. Therefore, the output of $\psi$ on $D$ is a union of (finitely) many cells in $C'$; since each cell is definable, the output is definable, too.

To get the complexity bound for $\text{FO} + \text{LIN} + \mathcal{T}$ and $\text{FO} + \text{POLY} + \mathcal{T}$, we show by induction that for each query $\varphi$, there exists a number $k$ such that the complexity of evaluating $\varphi$ on $D$ is $O(N^k)$, where $N$ is the size of a given representation of $D$ (assuming $\text{Top}$ can tested in constant time). Again, the only case to consider is that of $\psi(\vec{x}) \equiv \text{Top} \vec{y}$, $\varphi(\vec{x}, \vec{y})$, as others follow from the standard bounds on quantifier-elimination. Given $S = \varphi(D)$ computed in $O(N^k)$, we can find, by Lemma 2, a trivial decomposition $C$ 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 $\varphi$, we can construct both $C$ and $C'$ in time $O(N^{k_1})$. We next select a point $\vec{a}$ in each cell of $C'$ 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 [10, 8], 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 $\text{Top}$. Thus, the total complexity is polynomial in $N$, with the exponent depending on $\psi$ only. This completes the proof. $\square$

Now consider the case when $\mathcal{T}$ 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) [17], the proof of the complexity bounds in Theorem 3 implies the following.
Corollary 4 FO + LIN + C and FO + POLY + C are closed, and the queries they define have PTIME data complexity.

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-T0 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 $\mathbb{R}^{nk}$. We call $T$ topological if for any homeomorphism $h : \mathbb{R}^m \rightarrow \mathbb{R}^m$ and any $(\bar{x}_1, \ldots, \bar{x}_k) \in T(S)$, for $S \subseteq \mathbb{R}^m$, we have $(h(\bar{x}_1), \ldots, h(\bar{x}_k)) \in T(h(S))$. For example, the mapping $\text{Conn} : 2^{\mathbb{R}^m} \rightarrow 2^{\mathbb{R}^m \times \mathbb{R}^m}$ such that $(\bar{x}_1, \bar{x}_2) \in \text{Conn}(S)$ iff $\bar{x}_1, \bar{x}_2$ 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 [35], the topological query $\text{Conn}$ is definable over such structures.

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

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

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

In the semi-algebraic case or in any o-minimal expansion, the Local Triviality Theorem used in the proof of Lemma 2 can also be used to prove the following:

Theorem 5 For every definable topological map $T$, $\text{FO} + \text{POLY} + T$ is closed. Moreover, $\text{FO}(\Omega) + T$ is closed, whenever $(\mathbb{R}, \Omega)$ is an o-minimal expansion of the real field.

Proof. As usual this is proved by induction on the structure of the formulae. We only need to prove the case of $\psi(\bar{x}, \bar{y}_1, \ldots, \bar{y}_k) \equiv T\bar{y} : \varphi(\bar{x}, \bar{y})$. Assume $\varphi$ defines $S \subseteq \mathbb{R}^n \times \mathbb{R}^m$: $S = \{(\bar{a}, \bar{b}) \mid D \models \varphi(\bar{a}, \bar{b})\}$. By Lemma 2, there exists a decomposition $C$ trivial over $\mathbb{R}^n$. Let $C_1, \ldots, C_p$ be all the cells in the projection of $C$ onto $\mathbb{R}^n$. The proof of b) in Lemma 2 (which is just an application of Local Triviality) shows that for every $i$, and every $\bar{a}, \bar{b} \in C_i$, there is a definable homeomorphism $h^i_{\bar{a}, \bar{b}} : \mathbb{R}^m \rightarrow \mathbb{R}^m$ such that $h^i_{\bar{a}, \bar{b}}(S_\bar{a}) = S_\bar{b}$. Since $T$ is topological, it implies that $(\bar{e}_1, \ldots, \bar{e}_k) \in T(S_\bar{a})$ if and only if $(h^i_{\bar{a}, \bar{b}}(\bar{e}_1), \ldots, h^i_{\bar{a}, \bar{b}}(\bar{e}_k)) \in T(S_\bar{b})$.

Since each cell $C_i$ is a definable set, it has a definable representative $\bar{e}_i \in C_i$ [35]. Thus, a tuple $(\bar{a}, \bar{b}_1, \ldots, \bar{b}_k)$ is in $\psi(D)$ iff for $i \in 1, \ldots, p$ such that $\bar{a}$ is in $C_i$, the following holds:

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

Since $S_\bar{a}$ is definable, and $h^i_{\bar{a}, \bar{e}_j}$ is a definable homeomorphism, $T(S_\bar{a})$ is definable, which implies that $\psi(D)$ is a definable set, and proves closure.
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 $\text{FO+LIN}$, gives it the full power of $\text{FO+POLY}$ [2]. One can find definable operators that, when added to $\text{FO+POLY}$, 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 $\text{Conn}$ is definable and topological (over the real field and its o-minimal expansions), we conclude from Theorem 5 that $\text{FO + POLY + Conn}$ is closed. However, the proof above cannot possibly be extended to $\text{FO + LIN}$. Indeed, we used not only the triviality of the partition which is guaranteed by Lemma 2, but also the fact that homeomorphisms between fibers are definable. This latter condition fails over $(\mathbb{R}, +, -, 0, 1, <)$. Nevertheless, we can show that $\text{FO + LIN + Conn}$ is closed.

**Proposition 6** $\text{FO + LIN + Conn}$ is closed; that is, it defines a semi-linear output on a semi-linear input. Furthermore, $\text{FO + LIN + Conn}$ queries have PTIME data complexity.

**Proof.** The proof is by induction on the formulae, with only the case of applying the $\text{Conn}$ operator being nontrivial. Suppose we are given $\varphi(x, y)$ 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 $(\bar{a}, \bar{b}_1, \bar{b}_2)$ such that $\bar{b}_1$ and $\bar{b}_2$ are in the same connected component of $S_{\bar{a}}$, 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 2. We compute in PTIME a decomposition $C$ 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 $\pi(C)$, 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 2 that for any $\bar{a}, \bar{b}, C^i_{\bar{a}}$ adj $C^j_{\bar{a}}$ iff $C^i_{\bar{b}}$ adj $C^j_{\bar{b}}$ iff $C^i$ adj $C^j$ where $X$ adj $Y$ means $X \cap \text{cl}(Y) \neq \emptyset$ or $Y \cap \text{cl}(X) \neq \emptyset$. 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 $P^C_{\bar{a}}$ as

$$
\bigcup_{l=1}^{p} \left( \bigcup_{i \in K_l} C^i_{\bar{a}} \right) \times \left( \bigcup_{i \in K_l} C^j_{\bar{a}} \right) \subseteq \mathbb{R}^n \times \mathbb{R}^m
$$

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

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

It is known [35] 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 $(\bigcup_{i \in K_l} C^i_{\bar{a}})$ are exactly the connected components of $S_{\bar{a}}$, and hence $S'$ is the result of $\text{Conn}. \varphi$. By converting the above into a FO definition, we obtain PTIME data complexity as the number of quantifiers only depends on $n$ and $m$.

We note that the results on closure under topological operators and $\text{Conn}$ leave something to be desired. First, the proof of Theorem 5 does not produce a complexity bound, as it is not immediately clear how hard it is to compute definable homeomorphisms between fibers. We will see in the next
section that the data complexity is PTIME (Corollary 15). 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 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 and allows to express properties of paths with respect to given sets in the Euclidean space $\mathbb{R}^n$. For now, let us think of a path as a continuous curve in $\mathbb{R}^n$. For example, to express that a set $S \subseteq \mathbb{R}^n$ is connected we would say “for all $\bar{x}, \bar{y} \in S$ there exists a path such that for all points $p$ on this path which appear between $\bar{x}$ and $\bar{y}$ we have $p \in S$”. Formally, we would write this in the form

$$\forall \bar{x} \forall \bar{y} \left( (S(\bar{x}) \land S(\bar{y})) \rightarrow \mathbf{EP} \exists p \exists q \left( p = \bar{x} \land q = \bar{y} \land \forall r ((p < r \land r < q) \rightarrow S(r)) \right) \right).$$

Let us try to decode this formula: The first line is just first-order, its obvious meaning is “for all tuples $\bar{x}, \bar{y} \in \mathbb{R}^n$ which are both contained in $S$ we have.” Then $\mathbf{EP}$ in the second line says “there exists a path,” i.e. a continuous curve in $\mathbb{R}^n$. We shall assume that all paths have a starting point; that is, they are continuous maps $f : \mathbb{R}_+ \rightarrow \mathbb{R}^n$ where $\mathbb{R}_+ = \{ r \in \mathbb{R} \mid r \geq 0 \}$. Next we quantify over a new type of variables, path variables $p, q, r$, which range over the points of the path. We say that “there exist points $p, q$ on the path such that $p$ equals $\bar{x}$ and $q$ equals $\bar{y}$ (if we consider them as points of $\mathbb{R}^n$), and all points $r$ between $p$ and $q$ are contained in $S$.” Here the order in “between” is just the natural order on $\mathbb{R}$. So formally the meaning of the second line of our formula is

$$\exists r, u \in \mathbb{R}_+ \left( f(t) = \bar{x} \land f(u) = \bar{y} \land \forall v \in (t, u) f(v) \in S \right).$$

Similarly, we can formulate statements “for all paths” by using a universal quantifier $\mathbf{AP}$ instead of $\mathbf{EP}$.

Before we give the formal definition of the syntax and semantics of our logic, let us give one more intuitive example. The following query defines the set of all $\bar{y} \in \mathbb{R}^2$ such that if one wants to go from a point in Portugal ($P \subseteq \mathbb{R}^2$) to $\bar{y}$ on land ($L$) then one has to go through Spain ($S$) and then France ($F$):

$$\forall \bar{x} \left( P(\bar{x}) \rightarrow \mathbf{AP} \left( \left( \forall p \ L(p) \land \exists q \exists s \left( p < q \land q = \bar{x} \land q = \bar{y} \right) \right) \rightarrow \exists r \exists s \left( p < r < s < q \land \forall v \in (r, s) F(v) \right) \right) \right).$$

Thus the query returns, for example, all points $\bar{y}$ in France, Germany, and Italy, but no points in Spain or England.

In Subsection 4.1, we formally define the path logic $L_{\text{PATH}}$. In Subsection 4.2, we give more examples and analyze the expressive power. In Subsection 4.3, we show that $L_{\text{PATH}}$ is closed and
tractable over polynomial and linear constraints (more generally, the closure is shown for $\omega$-minimal structures). In the subsequent section, 5, we give an application of path logic to hybrid systems.

4.1 Definition of the path logic

Formulas in the logic may have two sorts of variables, element variables $x, y, \ldots$ and path variables $p, q, \ldots$. There are also two kinds of formulae: state formulae and path formulae. Associated with each path formula is an arity $n \geq 1$. (An $n$-ary path formula speaks about a path in $\mathbb{R}^n$.)

**Syntax** Given a database schema $SC$ and a structure $\mathcal{M} = (\mathbb{R}, \Omega)$, formulae of $L_{\text{PATH}}(\Omega)$ are defined inductively as follows:

1. Every $\text{FO} + \Omega$ formula $\varphi$ is a state formula.
2. State formulae are closed under the Boolean connectives $\lor, \land, \lnot$, and quantification $\exists, \forall$.
3. If $\varphi$ is a state formula, $\vec{x} = (x_1, \ldots, x_n)$ is an $n$-tuple of element variables, and $p$ is a path variable, then $\varphi[p \leftarrow \vec{x}]$ is a path formula of arity $n$.
4. If $n \geq 1$ and $p, q$ are path variables, then $p = q$, $p < q$, and $p > q$ are path formulae of arity $n$. (To be formally correct, we should write $p =_n q$, $p <_n q$, or $p >_n q$ for the $n$-ary versions of these formulae, but we can safely omit this.)
5. Path formulae of the same arity are closed under the Boolean connectives $\lor, \land, \lnot$.
6. If $\varphi$ is a path formula and $p$ a path variable, then $\exists p \varphi$ and $\forall p \varphi$ are path formulae of the same arity as $\varphi$.
7. If $\varphi$ is a path formula without free path variables, then $\text{EP} \varphi$ and $\text{AP} \varphi$ are state formulae.

To make the last point of the definition precise, we have to define the set $\text{FV}_p(\psi)$ of free path variables of a formula $\psi$: If $\psi \equiv \varphi[p \leftarrow \vec{x}]$ then $\text{FV}_p(\psi) = \{p\}$. For the other types of path formulae, $\text{FV}_p$ is defined in the usual way, for example $\text{FV}_p(p < q) = \{p, q\}$, $\text{FV}_p(\varphi_1 \lor \varphi_2) = \text{FV}_p(\varphi_1) \cup \text{FV}_p(\varphi_2)$, and $\text{FV}_p(\exists p \varphi) = \text{FV}_p(\varphi) \setminus \{p\}$. If $\psi$ is a state formula, then $\text{FV}_p(\psi) = \emptyset$.

This completes the definition of the syntax of $L_{\text{PATH}}(\Omega)$. When $\Omega = (+, -, 0, 1, <)$, we use the notation $L_{\text{PATH}}(\text{Lin})$ for $L_{\text{PATH}}(\Omega)$; for $\Omega = (+, -, 0, 1, <)$, we use the notation $L_{\text{PATH}}(\text{Poly})$.

We shall usually write $p = \vec{y}$ instead of $(\vec{x} = \vec{y})[p \leftarrow \vec{x}]$. Similarly, if $R$ is a relation name, then we shall write $R(p)$ instead of $R(\vec{x})[p \leftarrow \vec{x}]$. (We have already used these conventions in the examples at the beginning of this section.)

To be able to define the semantics, we also have to define the set $\text{FV}_e(\psi)$ of free element variables of a formula $\psi$: If $\psi$ is an $\text{FO} + \Omega$-formula, then $\text{FV}_e(\psi)$ is the set of free variables of $\psi$ defined in the usual way. Similarly, if $\psi$ is a Boolean combination of two state or path formulae, then we apply the usual rules to define $\text{FV}_e(\psi)$. If $\psi \in \{\exists x \varphi, \forall x \varphi\}$ (for an element variable $x$) then $\text{FV}_e(\psi) = \text{FV}_e(\varphi) \setminus \{x\}$.

If $\psi \in \{\exists p \varphi, \forall p \varphi\}$ (for a state variable $p$) then $\text{FV}_e(\psi) = \text{FV}_e(\varphi)$. If $\psi \in \{\text{EP} \varphi, \text{AP} \varphi\}$ then $\text{FV}_e(\psi) = \text{FV}_e(\varphi)$. Finally, if $\psi \equiv \varphi[p \leftarrow (x_1, \ldots, x_n)]$ then $\text{FV}_e(\psi) = \text{FV}_e(\varphi) \setminus \{x_1, \ldots, x_n\}$.

Semantics In our informal discussion starting this section we defined a path in \( \mathbb{R}^n \) to be an arbitrary continuous mapping \( P : \mathbb{R}_+ \rightarrow \mathbb{R}^n \). However, continuous functions can oscillate very wildly, and including pathological curves may lead to counterintuitive truth values for sentences. Therefore, we define our semantics with respect to the set of non-zeno paths, which are reasonably smooth and can only oscillate mildly. Formally, given an \( \alpha \)-minimal structure \( \mathcal{M} \), we say that a path \( P : \mathbb{R}_+ \rightarrow \mathbb{R}^n \) is non-zeno with respect to \( \mathcal{M} \) if for any set \( X \subseteq \mathbb{R}^n \) definable in any \( \alpha \)-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, every semi-algebraic path is non-zeno with respect to \( \mathbb{R}_{\text{lin}} \) and \( \mathbb{R} \). Another typical class of non-zeno paths can be obtained as follows: 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 \( \epsilon \). Let \( P : \mathbb{R}_+ \rightarrow \mathbb{R}^n \) 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 \( \mathbb{R}_{\text{lin}} \) and \( \mathbb{R} \). An example is given by \( P : \mathbb{R}_+ \rightarrow \mathbb{R}^2 \) defined as \( P(x) = (x, x - \lfloor x \rfloor) \) if \( \lfloor x \rfloor \) is even, and \( P(x) = (x, \lfloor x \rfloor - x) \) if \( \lfloor x \rfloor \) is odd. An example of a path that is non-zeno with respect to \( \mathbb{R}_{\text{lin}} \) and \( \mathbb{R} \), but not piecewise semi-algebraic, is the path \( s : \mathbb{R} \rightarrow \mathbb{R}^2 \) defined by \( s(x) := (x, \sin(x)) \). An example of a path which is not non-zeno with respect \( \mathbb{R}_{\text{lin}} \) and \( \mathbb{R} \) is the path defined by \( s(x) := (x, x \sin(1/x)) \).

We now give the formal definition of the semantics of \( L_{\text{PATH}}(\Omega) \). Whereas for state formulas the satisfaction relation is defined with respect to databases \( D \) in the usual way, the satisfaction relation for path formulas is defined with respect to pairs \( (D, P) \) consisting of a database \( D \) and a non-zeno path \( P \).

The cases of FO + \( \Omega \) formulae, as well as first-order quantification and Boolean connectives are standard.

For an \( n \)-ary path formula \( \psi(p, q) \equiv \varphi(x, y)[p \leftarrow \bar{x}], \) a database \( D \), a non-zeno path \( P : \mathbb{R}_+ \rightarrow \mathbb{R}^n \), a \( t \in \mathbb{R}_+ \), and a tuple \( \bar{a} \in \mathbb{R}^m \) with \( m = \| \bar{y} \| \), we have \( (D, P) \models \psi(t, \bar{a}) \) if \( D \models \varphi(P(t), \bar{a}) \). For \( \varphi(p, q) \equiv (p \theta q) \), with \( \theta \in \{ <, =, > \} \), we define \( (D, P) \models \varphi(t, t') \) iff \( t \theta t' \). Finally, \( D \models \mathbf{EP} \varphi(\bar{a}) \) iff there exists a non-zeno path \( P \) such that \( (D, P) \models \varphi(\bar{a}) \).

This completes the definition of the semantics of \( L_{\text{PATH}} \).

We close this section with an example that shows why we have to be very careful in defining a path logic that is closed and decidable. Note that in formulae \( \varphi(q) \models (x_2 = 0) \), only one path variable can get instantiated. Intuitively, this corresponds to coloring a path with previously defined regions. The following example shows why one cannot bind two path variables at the same time, i.e. admit formulas of the form \( \varphi[p \leftarrow \bar{x}, q \leftarrow \bar{y}] \) (with the obvious meaning).

Example 7 Let \( \alpha(x_1, x_2) \equiv (x_2 = 0) \). Then the path formula
\[
\beta(p, q) \equiv \left( \alpha[p \leftarrow \bar{x}] \land \alpha[q \leftarrow \bar{x}] \land p < q \land \neg \exists r (\alpha[r \leftarrow \bar{x}] \land p < r \land r < q) \right)
\]
says that \( p \) and \( q \) are two consecutive intersections of a path with the \( x_1 \)-axis. Now let
\[
\gamma(x_1, x_2, y_1, y_2) \equiv (x_2 = 0) \land (y_2 = 0) \land (y_1 = x_1 + 1)
\]
and consider the formula
\[
\varphi(z) \equiv \mathbf{EP} \left( \exists p \exists q (p = (0,0) \land q = (0,z) \land p < q) \land \forall p' \forall q' (\delta(p,q) \rightarrow \gamma[p \leftarrow \overline{x}, q \leftarrow \overline{y}]) \right).
\]

It says that there is a path from \((0,0)\) to \((z,0)\) 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, \(\varphi(z)\) holds iff \(z\) is a positive integer, while \(\mathbb{N}\) is not a semi-algebraic set.

### 4.2 Expressive power

In this section we collect a few facts we know about the expressive power of \(L_{\text{PATH}}\). We have indicated that \textsc{connected} and \textsc{connects-to} can be expressed: for example, to test that \(\overline{x}, \overline{y}\) are in the same connected component of \(R\), one writes \((R(\overline{x}) \land R(\overline{y})) \land \mathbf{EP} (\forall p \ R(p) \land \exists p_1 (p_1 = \overline{x}) \land \exists p_2 (p_2 = \overline{y}))\).

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 \(\overline{x}, \overline{y}\) such that every path from \(\overline{x}\) to \(\overline{y}\) intersects \(R\). Clearly, this can be expressed in \(L_{\text{PATH}}\).

The example above is an instance of a general result, saying that the language \(L_{\text{PATH}}\) is quite expressive when it comes to topological queries in 2-dimensional space. Proposition 8 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 \textbf{topological invariant} [28]. \(I(D)\) 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 \(L_{\text{PATH}}\).

**Proposition 8** The topological invariant of semi-linear or semi-algebraic 2-dimensional spatial database instances is definable in \(L_{\text{PATH}}\). 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 \(L_{\text{PATH}}(\text{Poly})\).

**Proof:** We briefly recall the definition of topological invariant. The reader is referred to [28, 33] for a more precise description. Given a spatial instance \(D\) over a database schema \textit{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 \textit{exterior-cell}.
2. A unary function \(\text{Dim}\), which associates a dimension to each cell.
3. A binary relation \textit{Adjacent} providing the topological adjacency relationship between the cells.

4. For each region name \( p \in \text{Reg} \), a unary relation \( p \) providing the set of cells contained in region \( p \).

5. \textit{Orientation} is a 5-ary relation providing the clockwise and counterclockwise orientation of edges incident to each cell of dimension 0. More precisely, \((+, v, e_1, e_2, e_3) \in \text{Orientation} \) 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 \((-+, v, e_1, e_2, e_3) \in \text{Orientation} \) 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 \( \text{inv} (\text{Reg}) \) denote the above schema. We want to prove that, given a semi-algebraic or semi-linear instance \( D \) over \( \text{Reg} \), there exists a formula in \( L_{\text{PATH}} \) 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 [35].)

The cells of dimension 0 are non-regular points and can be defined in \( \text{FO}(\prec) \).

A cell of dimension 1 is a connected set of points with the same boundary cone type (cone type as defined in [22, 33], a boundary cone type is the cone type of a point that lies on the common boundary of several region of \( \text{Reg} \)). The fact that two points \( p \) and \( p' \) have the same boundary cone type is expressible in \( \text{FO}(\prec) \); this follows from [22, 24, 33]. From the definition of topological invariant we know that, 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 \( L_{\text{PATH}} \).

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 \( \text{FO}(\prec) \) and therefore its connected components are definable in \( L_{\text{PATH}} \).

The function \( \text{Dim} \) 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 \( \text{FO}(\prec) \).

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 \( L_{\text{PATH}} \) (the cell of \( p \) is the set of points \( q \) such that \( E(p, q) \)), and testing for adjacency is \( \text{FO}(\prec) \). Therefore \( A \) is \( L_{\text{PATH}} \) definable.

We now show how to define the relation \textit{Orientation} in \( L_{\text{PATH}} \). From the above we know how to check in \( L_{\text{PATH}} \) 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 \( L_{\text{PATH}} \) whether \( e_2 \) lies between \( e_1 \) and

19
$e_3$ 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 $S$ $v_3$ lies on.

\[ \square \]

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

**Proposition 9** Every query on finite constraint databases that is computable in NL is expressible in $L_{\text{PATH}}$.

**Proof**: 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 [19] 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 $L_{\text{PATH}}$ is closed under first-order reductions, it suffices to prove that directed reachability is expressible in $L_{\text{PATH}}$.

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 $L_{\text{PATH}}$-formula:

$$\text{EP} \left( \exists p_1 \exists p_2 (p_1 = x \land p_2 = y) \land \forall q_1 \forall q_2 ((V(q_1) \land V(q_2) \land \neg \exists q_3 (V(q_3) \land q_1 < q_3 \land q_3 < q_2)) \rightarrow E(z_1, z_2)[z_1 \leftrightarrow q_1, z_2 \leftrightarrow q_2]) \right).$$

(The reason that this is not an $L_{\text{PATH}}$-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 $L_{\text{PATH}}$-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 $\gamma(x, y, \bar{z}) \in FO(<)$ such that for all $a, b \in \mathbb{R}$, $\bar{c} \in \mathbb{R}^3$ we have $\langle \mathbb{R}, < \rangle \models \gamma(a, b, \bar{c})$ if, and only if, $\bar{c}$ 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 $\varphi_V(\bar{z}), \varphi_E(\bar{z})$ in FO over $< \land E$, $V$ 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 $\bar{E}$.  

20
To encode the direction of the edges we define two new sets $\text{Tail} := \{(a, b, 0) \in \mathbb{R}^3 : (a, b) \in E\}$ and $\text{Head} := \{(b, 0, a) \in \mathbb{R}^3 : (a, b) \in E\}$. Let $\varphi_{\text{Tail}}(\vec{x}), \varphi_{\text{Head}}(\vec{x})$ in FO over $\langle \mathbb{R}, E, V \rangle$ be formulas defining those 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 $\vec{V}$ on this path that is not the final point is followed by a point in $\text{Tail}$, every point in $\text{Tail}$ is followed by a point in $\text{Head}$, and every point in $\text{Head}$ is followed by a point in $\vec{V}$. To formalize this in $L_{\text{PATH}}$, we let

$$\varphi_{\text{Empty}}(q_1, q_2) = \forall q_3 \left( (q_1 < q_3 \land q_3 < q_2) \rightarrow \neg (\varphi_V(q_3) \lor \varphi_{\text{Tail}}(q_3) \lor \varphi_{\text{Head}}(q_3)) \right).$$

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

$$V(x) \land V(y) \land \exists \exists (x, 0, 0) \land (y, 0, 0) \land p_1 \leq p_2 \rightarrow
\exists q_2 \left( \varphi_{\text{Tail}}(q_2) \land \varphi_{\text{Empty}}(q_1, q_2) \right)$$

Of course this proposition is only a small step towards understanding the expressive power of $L_{\text{PATH}}$. We would like to prove the converse statement that every $L_{\text{PATH}}$-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 $L_{\text{PATH}}$-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 $L_{\text{PATH}}$, however.

Another difficult problem is to compare the expressive power of $L_{\text{PATH}}$ with that of the various extensions of FO by the connectivity quantifiers. We believe that directed reachability in finite graphs is not expressible in FO + POLY + CORN, 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 $L_{\text{PATH}}$ 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 in Example 7 that if one extends $L_{\text{PATH}}$ by allowing new binary predicates on path variables, the resulting logic is neither closed nor decidable. We now show that $L_{\text{PATH}}$ has very desirable closure and tractability properties for domains relevant in spatial applications. More precisely, we say that $L_{\text{PATH}}(\Omega)$

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

Our goal is to prove the following result.

**Theorem 10** Let \( \mathcal{M} = \langle \mathbb{R}, \Omega \rangle \) be \( \alpha \)-minimal and decidable. Then \( L_{\text{PATH}}(\Omega) \) is closed and decidable. Furthermore, if \( \mathcal{M} \) admits effective quantifier-elimination, then \( L_{\text{PATH}}(\Omega) \) admits effective query evaluation, and for \( L_{\text{PATH}}(\text{Lin}) \) and \( L_{\text{PATH}}(\text{Poly}) \) the data complexity is \( \text{PTIME} \).

**Proof.** We prove the theorem in two steps: we first show that an \( L_{\text{PATH}} \) query can be transformed into a “discrete path query” in which instead of quantifying over any point 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 \( \mathcal{A} = \{ A_1, \ldots, A_k \} \) be some finite collection of \( \Omega \)-definable sets in \( \mathbb{R}^n \). Fix a non-zero path \( P : \mathbb{R}_+ \to \mathbb{R}^n \). For \( r, s \in \mathbb{R}_+ \), we say that \( r \) and \( s \) agree on \( \mathcal{A} \) if \( P(r) \in A_i \Leftrightarrow P(s) \in A_i \) for all \( i = 1, \ldots, k \). We say that \( r, s \in \mathbb{R}_+ \) are \( \mathcal{A} \)-equivalent (and write \( r =_A s \)) if there is an open interval \( I \) such that \( r, s \in I \), and all \( r', s' \in I \) agree on \( \mathcal{A} \). We write \( r <_A s \) if \( r < s \) and \( r \neq_A s \). Note that the equivalence classes of \( =_A \) are either open intervals or single points.

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

We now introduce a new path formula \( p <_{A(\bar{y})} q \) with free path variables \( p, q \) and free element variables \( \bar{y} = (y_1, \ldots, y_m) \), which holds in a database interpreting the predicates in \( \mathcal{A} \) by \( \Omega \)-definable sets \( X_1^1, \ldots, X_k^1 \), for a non-zero path \( P \) in \( \mathbb{R}^n \), path elements \( p_0, q_0 \) and \( \bar{c} \in \mathbb{R}^m \) exactly when \( p_0 < x q_0 \) where \( \mathcal{X} = \{ X_j^i \} \). Recall that \( X_j^i = \{ \bar{b} \mid (\bar{b}, \bar{c}) \in X_j^i \} \subseteq \mathbb{R}^n \). Similarly we introduce the formula \( =_{A(\bar{y})} \), saying two path variables are in the same equivalence class modulo \( \mathcal{A}(\bar{y}) \). 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 \( \mathcal{A} = \{ A_i(\bar{x}, \bar{y}) \} \) of first-order formulae (over \( \Omega \) and the schema predicates) we have new atomic formulae:

- For every path variable \( p \) of arity the same as that of \( \bar{x} \), \( A_i(\bar{y})([p]_{A_i}) \) is an atomic formula with \( p \) and \( \bar{y} \) free, with meaning the same as \( A_i(p \leftarrow \bar{x}, \bar{y}) \).

- For every path variable \( p \) with arity the same as that of \( \bar{x} \), we have a formula \( O(p, A(\bar{x}, \bar{y}), \bar{y}) \), with free variables \( p \) and \( \bar{y} \) meaning that the equivalence class of \( p \) under \( =_{A(\bar{y})} \) is an open interval. If free element variables are clear from the context, we use the abbreviation \( O([p]_{A_i}) \). Similarly, we have formulae \( C([p]_{A_i}), CO([p]_{A_i}), OC([p]_{A_i}) \), meaning that the equivalence class of \( p \) under \( =_{A(\bar{y})} \) is closed, open on right and closed on left, closed on right open on left, respectively.

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

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 $\overline{L}^{-}_\text{PATH}(\Omega)$, we have only boolean operators (where the set $\mathcal{A}$ must be the same in each operand) and path-variable quantification. Neither path quantification $\mathbf{EP}$ nor element quantification is present.

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

With $\mathcal{A}(\vec{x}, \vec{y})$ a collection of first-order formulae as above, and $p$ a path variable, let $\text{First}_\mathcal{A}(p)$ be a new formula that says $p$ is the first element in the path in its $\mathcal{A}$ equivalence class. Similarly let $\text{Last}_\mathcal{A}(p)$ 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 $L\text{PATH}$ can say about a set of path variables.

**Lemma 11** Suppose $\varphi(\vec{y}, \vec{w})$ is an $L\text{PATH}(\Omega)$ path formula (where $\vec{y}$ are path variables and $\vec{w}$ are element variables) that has no occurrences of $\mathbf{EP}$. Then from $\varphi$ we can effectively find a collection $\mathcal{A}$ of first-order formulae and a formula $\varphi'$ that is equivalent to $\varphi$ (on every database and every non-zero path), and such that $\varphi'$ is a boolean combination of formulae of $L\overline{\text{PATH}}$ and atomic formulae $p_1 \triangleleft \mathcal{A} p_2$, $\text{First}_\mathcal{A}(p_1)$, and $\text{Last}_\mathcal{A}(p_1)$. In particular, if $\varphi$ has no free path-variables, then it is equivalent to a formula of $L\overline{\text{PATH}}(\Omega)$.

**Proof of Lemma 11.** 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 $\mathcal{A} \mathcal{B} = \mathcal{A} \cup \mathcal{B}$. 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 $\varphi_i([p_1], \ldots, [p_k], \mathcal{A})$, $\text{First}_\mathcal{A}(p_i)$, and $\text{Last}_\mathcal{A}(p_i)$, and a formula $\gamma$ that is a boolean combination of $\gamma([p_1], \ldots, [p_k], \mathcal{B})$, $\text{First}_\mathcal{B}(p_i)$, and $\text{Last}_\mathcal{B}(p_i)$, and writing it in terms of formulae mentioning only equivalence modulo $\mathcal{A} \mathcal{B}$. Here $\mathcal{A}$ and $\mathcal{B}$ are collections of formulae with free variables $\vec{x}, \vec{y}$, and the formula $\varphi_i$ 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

$$\forall r \ ( [p] \leq_{\mathcal{X}} [r] \lor [q] \leq_{\mathcal{X}} [r] \leq_{\mathcal{X}} [p])$$

$$\rightarrow \bigwedge_i A_i([r], \mathcal{X}) \leftrightarrow A_i([q], \mathcal{X}) \land A_i([r], \mathcal{X}) \leftrightarrow A_i([p], \mathcal{X}).$$

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

The formula $\mathbf{O}([p]_\mathcal{A})$ is equivalent to
\[ \exists q, r \left( (O([q], \mathcal{A}) \lor OC([q], \mathcal{A})) \land (O([r], \mathcal{A}) \lor CO([r], \mathcal{A})) \land (q = A p = A r) \land (q' < \mathcal{A} q \rightarrow (q' = A p)) \land (r' > \mathcal{A} r \rightarrow (r' = A p)) \right) , \]

which can then be converted to the proper form, since we have seen above how to convert formulae using the relation \( =_A \) on path variables. The other interval types are similar, as are \( First_A \) and \( Last_A \).

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

The last step is existential path variable quantification. Suppose we have a formula \( \forall p_1 \varphi(p_1, p_2, \ldots, p_k, \bar{y}) \). By induction, \( \varphi \) is a Boolean combination of \( L_{PATH}^\mathcal{A} \) formulae, inequalities among the \( p_i \), and \( First_A(p_i) \) and \( Last_A(p_i) \) statements. By combining sets, we can assume that these last statements all refer to the same set of formulas \( \mathcal{A}_i(\bar{x}, \bar{y}) \). Without loss of generality, \( \varphi \) is

\[ \bigvee_i (\varphi_i([p_1]_A, \ldots, [p_k]_A) \land t_i(p_1, \ldots, p_k) , \]

where \( \mathcal{A} = \{ \mathcal{A}_i(\bar{x}, \bar{y}) \} \) is a collection of first-order formulae, \( \varphi_i \) is a statement about the ordering and interval types of equivalence classes, and the \( \mathcal{A}_i \) labels of equivalence classes, while \( t_i \) is a simple ordering statement about \( \bar{p} \), giving inequalities between them and which ones are first in their equivalence classes. We may assume w.l.o.g. that \( \varphi_i \) completely specifies the ordering relations \( <_A \) 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 \( <_A \) given in \( \varphi_i \) (otherwise, we can eliminate this disjunct).

Let \( S \) be the collection of \( i \) such that \( \varphi_i \) 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 \( \exists p_1 \varphi_i([p_1]_A, \ldots, [p_k]_A) \land t_i(p_1, \ldots, p_k) \) is equivalent to \( \exists p_1 \varphi_i([p_1]_A, \ldots, [p_k]_A) \land t'_i(p_2, \ldots, p_k) \), where \( t'_i \) is obtained from \( t_i \) by eliminating all inequalities involving \( p_1 \). This is true because the \( <_A \) inequalities involving \( p_1 \) in \( \varphi_i \) already imply the inequalities involving \( p_1 \) in \( t_i \).

For \( i \) in \( S \), the formula \( \exists p_1 \varphi_i([p_1]_A, \ldots, [p_k]_A) \land t_i(p_1, \ldots, p_k) \) is equivalent to

\[ \varphi_i([p_{\delta(i)}]_A, \ldots, [p_k]_A) \land \exists p_1 (p_1 =_A p_{\delta(i)} \land t_i(p_1, \ldots, p_k)) \].

Furthermore, \( \exists p_1 (p_1 =_A p_{\delta(i)} \land t_i(p_1, \ldots, p_k)) \) can be converted into an atomic formula \( t'_i(p_2, \ldots, p_n) \), due to the fact that \( \varphi_i \) specifies the interval type (closed, open, etc.) of \([p_{\delta(i)}]_A \), 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 [9].

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 $L_{\text{PATH}}^-$ formulae and inequalities of the form $p_i < p_j$. This complete the proof of Lemma 11. □

Our next lemma shows that $L_{\text{PATH}}^-$ formulae can be evaluated effectively.

**Lemma 12** For every $L_{\text{PATH}}^-(\Omega)$ query $\varphi(\vec{x})$, where the $\vec{x}$ are free element variables, and there are no free path variables, and for every $\Omega$-definable database $D$, there is a first-order $\Omega$-formula $\varphi_D(\vec{x})$ such that for every $\vec{a}$, $\mathcal{D}c \models EP \varphi(\vec{a})$ iff $\mathcal{M} = \varphi_D(\vec{a})$.

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

**Proof of Lemma 12.** Let $\mathcal{A} = \{A_i(\vec{x}, \vec{y}) \mid i \leq K\}$ be the set of formulae over $\Omega$ (unioned with the schema) which are used in $\varphi$. For any $\Omega$-definable database $D$, we construct a family of finite discrete structures parameterized by $\vec{y}$.

We do it as follows. Let $A_i^D = \{(\vec{a}, \vec{b}) \mid D \models A_i(\vec{a}, \vec{b})\}$. Applying Lemma 2, 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 $\mathbb{R}^k$ by letting $\vec{b} \equiv \vec{c}$ iff $\vec{b}$ and $\vec{c}$ are in the same cell of the projection of $\mathcal{B}$ 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 $i, j \leq N$, $B^i_\vec{b} \text{ Adj } B^j_\vec{b}$ iff $B^i_\vec{c} \text{ Adj } B^j_\vec{c}$, where $E \text{ Adj } F$ means $\text{cl}(E) \cap F \neq \emptyset$.
- 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 $E \cap \text{cl}(F) \cap \text{cl}(G) \neq \emptyset$.

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 $\text{Adj}$ 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 $\langle (N(i), E(i)) \mid i \in \mathbb{N} \rangle$, 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 13** If $\vec{c} \equiv \vec{d}$, then they agree on $EP \varphi$; that is, $EP \varphi(\vec{c})$ holds iff $EP \varphi(\vec{d})$ holds.
Proof of Claim 13. Suppose we have \( \tilde{c} \equiv \tilde{d} \). Then the adjacency structures of \( \tilde{c} \) and \( \tilde{d} \) are isomorphic (with the natural isomorphism that sends \( B^n_{\tilde{c}} \) to \( B^n_{\tilde{d}} \)). Suppose we are given a non-zero path \( P \) that witnesses that \( EP \varphi(\tilde{c}) \). Then (by non-zero-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 \( P' \) running through the adjacency graph of \( \tilde{c} \), 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. Since the adjacency graph of \( \tilde{d} \) is isomorphic to that of \( \tilde{c} \), there is a corresponding path \( Q' = \{ (N(i), X(i)) \mid i \in \mathbb{N} \} \) through the adjacency graph of \( \tilde{d} \) 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-zero 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 \( P'_{n-1} = \langle M(n-1), O \rangle \), then \( Q_{n-1} \) takes all its values in \( N(n-1) \).
- If \( P'_{n-1} = \langle M(n-1), C \rangle \), then \( Q_{n-1} \) takes all its values in \( N(n-1) \).
- If \( P'_{n-1} = \langle M(n), O \rangle \), then \( Q_n \) is closed on the right and the value of \( Q_n \) at the right endpoint is in the closure of \( N(n+1) \).
- If \( P'_{n-1} = \langle M(n), C \rangle \), then \( Q_n \) is open on the right and the limit of the path \( Q_n \) as we approach the right endpoint is in \( N(n+1) \).
- \( J_n \), 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.
- Suppose \( J_n \) 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 \( N(n+2) \).

**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 \( Q_{n-1} \) 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 [35], it can be chosen to be definable in \( M \). 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 to \( y \) 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 definable connected [35]), the resulting path \( Q \) can be made to be non-zero. We want now to verify that \( Q \) also witnesses that \( \text{EP} \varphi\langle d \rangle \), using that \( P \) witnesses \( \text{EP} \varphi\langle e \rangle \). This follows easily from the fact that \( L_{\text{PATH}} \) can only describe properties of equivalence classes of elements (Lemma 11). This completes the proof of the claim. \( \square \)

Given Claim 13, and the fact that each adjacency type is described by a first-order formula, we
conclude that that on $D$, the formula $\text{EP}\, \varphi$ is equivalent to a first-order formula. Hence we have completed the proof of the first part of Lemma 12.

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 $\text{EP}\, \varphi$ on $D$. Since in the linear or polynomial cases the adjacency structures are of polynomial size in the complexity of $D$ (see Lemma 2), and can be produced from $D$ in polynomial time, the problem reduces to the following. Given $\varphi$ and an adjacency structure $AS$, determine efficiently in the size of $AS$ whether $\text{EP}\, \varphi$ 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 $\tau'$ of each tuple $\tau$ of the form $(B, C, E, O, F)$ where $E \cap \text{cl}(B) \cap \text{cl}(F) \neq \emptyset$. This tuple $\tau'$ has the same transitions in and out of it as $\tau$ does. We also add nodes of the form $(\text{START}, H, X, I)$, where $H$ and $I$ are cells and $X$ is a transition in $AS$. For nodes $\tau$ of the form $(\text{START}, H, O, I)$, we add a copy $\tau'$. Both $\tau$ and $\tau'$ 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 $\Sigma$ that consists of symbols $L_{ij}$, with $i \in \{1, 2, 3\}$ and $1 \leq j \leq K = |\mathcal{A}|$, $L_{CO}, L_{OC}, L_{CC}, L_{OO}, L_{nsn}, L_{nsn}, L_{START}, L_{START,O}$ and $L_{START,C}$.

We now show how to label nodes in $K(AS)$ by symbols in $\Sigma$. 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 $i$th cell in the tuple is in $A_j$. Label elements of the form $(\text{START}, H, O, I)$ 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 $\tau$ of the form $(B, C, E, O, F)$ where $E \cap \text{cl}(B) \cap \text{cl}(F) \neq \emptyset$, we label $\tau'$ with $L_{nsn}$ and $\tau$ with $L_{nsn}$. For each tuple $\tau$ of the form $(\text{START}, H, O, I)$, we label $\tau'$ with $L_{nsn}$ and $\tau$ with $L_{nsn}$. Finally, we label each tuple of the form $(\text{START}, H, X, I)$ with $L_{START}$. We label $(\text{START}, H, O, I)$ with $L_{START,O}$ and $(\text{START}, H, C, I)$ with $L_{START,C}$.

We convert $\varphi$ into a first-order formula $\varphi'$ over $\omega$-words in $\Sigma'$ with the language $\prec_{\mathcal{A}}$ unioned with the label alphabet $\Sigma$. We can assume that there are no occurrences of $=_{\mathcal{A}}$, since they can be reduced to $\prec_{\mathcal{A}}$. We translate atomic formulae $[p] \prec_{\mathcal{A}} [p']$ as

$$([p] < [p']) \land \bigwedge_i \neg L_{2i}(p) \leftrightarrow L_{2i}(p').$$

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

$$\exists p' \exists p'' \left( p' < p < p'' \land (L_{OC}(p') \lor L_{CO}(p')) \land (L_{OC}(p'') \lor L_{CC}(p'')) \land \forall q (p' \leq q \leq p'' \rightarrow \bigwedge_i (L_{2i}(q) \leftrightarrow L_{2i}(p))) \right)$$

and similarly for other interval types (analogously to the proof of the conjunction step in Lemma
11).

We translate the formula $S([p], A)$ into the disjunction of the formula

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

with

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

Finally, we translate statements $A_j([p], A)$ by the corresponding labels $L_{2j}(p)$. Now a straightforward modification of the proof of Claim 13 shows the following.

**Claim 14** 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 $(\text{START}, B, X, E)$, such that the $\omega$-word corresponding to the path satisfies $\varphi'$ iff $EP \varphi(\vec{c})$ is satisfied.

**Proof of Claim 14.** Given a path $P$ that witnesses $EP \varphi(\vec{c})$, 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 $A$-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 $(\text{START}, B, X, E)$: we take $\tau'$ if the initial equivalence class in the quotient path through $AS$ is a singleton (in which case, that singleton must have been contained in cell $B$), and take $\tau$ 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 $(\text{START}, E, X, F)$ 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-zero path $P$ just as in the proof of Claim 13. 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 14.

The formula $\varphi'$ 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. [36]) 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 2 implies decidability for decidable $\omega$-minimal structures, and polynomial time data complexity for $R_{lin}$ and $R$. This completes the proof of Lemma 12.
Proof of Theorem 10. We finally prove closure and effective query evaluation by induction on the complexity of the formula $\varphi$. Clearly, atomic state formulae can be evaluated effectively if $\Omega$ 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 [25]. The interesting case is where we have a query of the form $\text{EP } \psi(y)$. By the induction hypothesis, we can assume that every proper state subformula $\psi_i$ of $\psi$ 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 $\psi$ as a query over the outputs of these subformulae and thus can assume that $\psi$ has no existential path quantifiers within it. By Lemma 11, $\psi$ can be transformed into an $L_\text{PATH}$ query $\psi'$ over these predicates, and by Lemma 12, $\text{EP } \psi'$ 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 2, a), implies that for fibers of three cells, $C_0, C_1, C_2$, one has $C_0 \cap \text{cl}(C_1) \cap \text{cl}(C_2) \neq \emptyset$ iff $C_0 \cap \text{cl}(C_1) \neq \emptyset$ and $C_0 \cap \text{cl}(C_2) \neq \emptyset$. Thus, $T$ can be reconstructed from $C$ and $O$ edges of the adjacency structure, which simplifies the construction of the Kripke structure $K(AS)$. □

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

Corollary 15 $\text{FO + POLY + Conn}$ queries have PTIME data complexity. □

5 Model-checking for hybrid systems

We mentioned that our logic $L_\text{PATH}$ 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 [18]. 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 $\text{LTL}$ for one class of hybrid systems, the $\omega$-minimal hybrid systems of [26, 27] 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 $\text{CTL}$ and $\text{CTL}^\star$.

A hybrid system (cf. [18, 26]) 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 \( \rightarrow \subseteq S \times (E \cup \{c\}) \times S \), where \( c \) is a new symbol not contained in \( E \). We write \( s \xrightarrow{e} s' \) instead of \((s, e, s') \in \rightarrow\). 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**: \((q, \vec{x}) \xrightarrow{q'} (q', \vec{x}')\) 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 \( s_1 e_1 s_2 e_2 \ldots \) 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 \( \Pi \) that assigns to each state \( s \in S \) a subset of \( \Sigma \). Then \( \Pi \) associates with each trajectory \( \tau = s_1 e_1 s_2 e_2 \ldots \) of \( H \) an \( \omega \)-word \( \Pi(\tau) := \Pi(s_1)\Pi(s_2)\ldots \) over the alphabet \( 2^\Sigma \). We assume that the reader is familiar with the linear time temporal logic LTL (interpreted over \( \omega \)-words), see [11]. The LTL-model checking problem for hybrid systems is defined as follows:

**Input:** An interpreted hybrid system \((H, \Pi)\)
and an LTL-formula \(\varphi\).

**Problem:** Decide if for every trajectory \( \tau \) of \( H \)
the word \( \Pi(\tau) \) satisfies \( \varphi \).

Let \( \mathcal{M} \) be an \( \alpha \)-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 \( T := \{(q, \vec{x}, \vec{y}) \mid (q, \vec{x}) \xrightarrow{e} (q, \vec{y})\} \) are definable in \( \mathcal{M} \). A hybrid system is \( \alpha \)-minimal if it is definable in some \( \alpha \)-minimal structure over \( \mathbb{R} \). An interpreted hybrid system \((H, \Pi)\) is \( \mathcal{M} \)-definable if \( H \) is \( \mathcal{M} \)-definable, and for every \( \tau \) in the signature, the set \( \Pi^{-1}(\tau) \) is definable.

\(^1\)We assume that there exists a finite set of \( \text{card}(Q) \) definable constants. This is certainly true for all structures of practical interest like \( \mathbb{R}_{\text{ss}} \) and \( \mathbb{R} \); otherwise one can restate the definition by talking about definability of the fibers of \( S, S_F, T \), etc. over \( q \) for each \( q \in Q \).
Theorem 16 Let $\mathcal{M} = (\mathbb{R}, \Omega)$ be such that its expansion with $+,-,0,1$ is a decidable $\alpha$-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 $\mathcal{M} = \mathbb{R}_{\text{dim}}$ or $\mathbb{R}$, for every fixed LTL-formula $\varphi$ 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. 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 $\sigma := \{\prec, P_1, \ldots, P_m\}$, 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 $(H, \Pi)$, we also assume it to be $\mathcal{M}$-definable and of signature $\Sigma$.

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

It is well-known that every LTL-formula $\varphi$ of signature $\Sigma$ can effectively be transformed into an equivalent $FO[\sigma]$-sentence $\varphi'$. What we actually show in the following is how to translate an $FO[\sigma]$-sentence $\varphi$ into an $L_{\text{PATH}}$-formula $\varphi^*$ such that for every interpreted hybrid system $(H, \Pi)$ we have $(H, \Pi) \models \varphi$ if, and only if, there exists a trajectory $\tau$ 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 $\varphi^*$ be a formula of the form $\text{EP} \varphi'$, where $\varphi'$ is more or less the same as our original $\varphi$. The path whose existence we state by $\text{EP}$ 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 $\bar{x} \in \mathbb{R}^n$. However, in a path-formula we cannot say that our path is on such a curve (unless we treat $\bar{x}$ 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 $r_i = (q, \bar{x})$, $r'_i = (q, \bar{x}')$ and $x_i : [0, \delta] \to \mathbb{R}^n$ is a curve such that $x_i(0) = \bar{x}$, $x_i(\delta) = \bar{x}'$ and for every $t \in [0, \delta]$ it satisfies $x_i(t) = F(q, x_i(t))$ and $x_i(t) \in I(q)$. Furthermore, for all $i \geq 1$ there is a discrete transition $r'_i \xrightarrow{\epsilon} r_{i+1}$ (for an $\epsilon \in E$).
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 \(x_i = 0\) and \(r_i = r'_i\), or \(r_i = (q, \bar{x})\), \(r'_i = (q, \bar{x}')\) and \(x_i : [0, \delta] \to \mathbb{R}^n\) for some \(\delta > 0\) and \(x_i(0) = \bar{x}\), \(x_i(\delta) = \bar{x}'\). 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 \(r_{f(i)} = (q, \bar{x})\) for some \(q \in \mathbb{R}\), \(\bar{x} \in \mathbb{R}^n\) 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 \(\nu\) consisting of the binary relation symbol \(\langle\rangle\) that is always interpreted as a linear order of the universe and finitely many unary relation symbols. In particular, let an \(\omega\)-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 \(\nu\) is the set of all \(\text{FO}[\nu]\)-sentences of quantifier-rank at most \(r\) it satisfies. Note that there are only finitely many \(r\)-types of a fixed vocabulary.

**Lemma 17** There is a computable function \(f : \mathbb{N} \to \mathbb{N}\) such that the following holds: Let \(r \geq 1\) and \(W, W'\) be words of vocabulary \(\sigma\) 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 \(\text{FO}[\tau]\)-sentence \(\psi\) 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 \(\psi\).

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 \(r = (q, \bar{x})\), \(r' = (q, \bar{x}')\) with \(q \in \mathbb{R}\), \(\bar{x}, \bar{x}' \in \mathbb{R}^n\) and \(x : [0, \delta] \to \mathbb{R}^n\) we let \(W(r, x, r')\) be the word of vocabulary \(\sigma\) with universe \([0, \delta]\) and \(\Pi_i := \{t \in [0, \delta] \mid \pi_i \in \Pi(q(x(t)))\}\). 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 \(x_i = 0\) then \(\Theta_i = \emptyset\).
- If \(x_i : [0, \delta] \to \mathbb{R}^n\), then \(\Theta_i\) is the \(f(r)\)-type of the word \(W(r_i, x_i, r_i')\). Here \(f\) is taken from Lemma 17.
Let $\sigma^+$ be the extension of the vocabulary $\sigma$ that contains a new unary relation symbol $P_0$ for every $f(r)$-type $\Theta$ of vocabulary $\sigma$. We let $\Pi(\rho_i)$ be the $\omega$-word $W$ with $P^W_i := \{ j \mid \pi_i \in \Pi(r_j) \}$ and $\Theta_j = \emptyset$ (for $1 \leq i \leq m$) and $P_0 := \{ j \mid \Theta_j = \Theta \}$ (for every $f(r)$-type $\Theta$). For a trajectory $\tau = (s_i, e_i)_{i \geq 1}$ we let $\Pi(\tau)$ be the $\omega$-word $(\Pi(s_i))_{i \geq 1}$.

The following lemma is a generalization of Lemma 17 that is proved using “Referman-Vaught”-type arguments (See chapter 6 of [9]) and the fact that for every $f(r)$-type $\Theta$ there is a FO-sentence $\theta$ of quantifier-rank $f(r)$ such that a word satisfies $\theta$ if, and only if, its $r$-type is $\Theta$.

**Lemma 18** Let $f$ be as in Lemma 17. Let $\rho, \rho'$ be runs such that $\Pi(\rho_i)$ and $\Pi(\rho_i')$ have the same $r$-type. Then for every trajectory $\tau$ consistent with $\rho$ there is a trajectory $\tau'$ consistent with $\rho'$ such that $\Pi(\tau)$ and $\Pi(\tau')$ have the same $r$-type.

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

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

Applying Lemma 18, we first translate $\varphi$ to a sentence $\varphi^+$. In the remaining proof we show how to translate $\varphi^+$ to an LPATH-sentence $\varphi'$ such that for every interpreted hybrid system $(H, \Pi)$ we have $(H, \Pi) \models \varphi'$ if, and only if, there is a run $\rho$ of $H$ such that $\Pi(\rho_i)$ satisfies $\varphi^+$. 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 $\Theta_1, \ldots, \Theta_t$ be an enumeration of all $f(r)$-types of vocabulary $\sigma$. 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 $\Theta$ 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 $(H, \Pi)$) if $s, s' \in S$ and either $t = 0$ and $s = s'$ or $s \not\sim s'$ via a curve $x : [0, \delta) \to \mathbb{R}^n$ such that the $f(r)$-type of the word $W(s, x, s')$ is $\Theta_i$. The first thing we do is define an LPATH-formula $\chi$ such that for every interpreted hybrid system $(H, \Pi)$ and $\vec{x} \in \mathbb{R}^{2n+3}$ we have $(H, \Pi) \models \chi(\vec{x})$ if, and only if, $\vec{x}$ is good. Let $\Theta_i$ be an FO[$\sigma$]-formula defining the type $\Theta_i$. Recall that $T := \{(q, \vec{x}, \vec{y}) \mid (q, \vec{x}) \overset{\sim}{\rightarrow} (q, \vec{y})\}$ is definable. Then the following formula says that $(q, \vec{x}) \overset{\sim}{\rightarrow} (q', \vec{x}')$ 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 $\Theta_i$:

$$\xi(q, \vec{x}, q', \vec{x}'): = (q = q') \land \text{EP} \left( \exists p (p = \vec{x} \land \forall p' (p \leq p')) \\
\land \exists p' (p' = \vec{x}' \land \forall p'' (p'' \leq p')) \\
\land \forall p'' (T(q, \vec{x}, \vec{y}) \land I(q, \vec{y})) [p'' \leftarrow \vec{y}] \\
\land \theta_i' \right),$$

where $\theta_i'$ is the formula obtained from $\theta_i$ 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 $\xi$, it is easy to define the desired $\chi$ defining the good tuples.
It will be convenient to make the following assumption for every hybrid system $H$:

\((\ast)\) 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 $(s_1, t, s_1') \xrightarrow{e} (s_2, t, s_2')$ if $(s_1, s_1'), (s_2, s_2')$ are good and $s_1' \xrightarrow{e} s_2$ in $H$.

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 $\gamma^o(e, \vec{z}, \vec{z}')$ 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

\[
\Gamma^o(e) := \bigcup_{\vec{z} \xrightarrow{e} \vec{z}'} \gamma^o(e, \vec{z}, \vec{z}'),
\]

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

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

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 $\beta$ from $(0, 0, 0, \vec{z})$ to $(0, 0, 0, \vec{z}')$ such that the interior $\beta^o$ of $\beta$ is contained in $\Gamma^o(e)$. (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 $(q, x_1) \xrightarrow{e} (q', x_1')$ and $(q, x_2) \xrightarrow{e} (q', x_2')$ implies $(q, x_1) \xrightarrow{e} (q', x_2')$.) It follows that there exists an $e \in E$ such that $\vec{z} \xrightarrow{e} \vec{z}'$ if there is a path $\beta$ from $(0, 0, \vec{z})$ to $(0, 0, \vec{z}')$ such that $\beta^o \subseteq \Gamma^o$.

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

\[
\text{GOOD} := \{0\} \times \{0\} \times \{0\} \times \{\vec{z} \in \mathbb{R}^{2n+3} \mid \vec{z} \text{ good}\},
\]

\[
\text{TAIL} := (\mathbb{R}^2 \times \{1\} \times \mathbb{R}^{2n+3}) \cap \Gamma^o,
\]

\[
\text{HEAD} := (\mathbb{R}^2 \times \{3\} \times \mathbb{R}^{2n+3}) \cap \Gamma^o,
\]

\[
\text{REST} := \Gamma^o \setminus (\text{TAIL} \cup \text{HEAD}).
\]
Since the good tuples and the set $\Gamma^*$ are definable in $L_{\text{PATH}}$, these sets are also definable.

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

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

  $$P^1_\alpha := \{(0, 0, s, t, s') \in \text{GOOD} \mid t = 0 \text{ and } \pi_i \in \Pi(s)\} \quad \text{(for } 1 \leq i \leq m\),$$

  $$P^2_\alpha := \{(0, 0, s, t, s') \in \text{GOOD} \mid t = i\} \quad \text{(for } 1 \leq i \leq l\)$$

satisfies $\varphi^+$.

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

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 17 for $\omega$-words. We omit the details.

\[\square\]

6 Conclusion

Reachability between points in a region is a fundamental notion in spatial reasoning. 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 $L_{\text{PATH}}$ 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 $L_{\text{PATH}}$ is interesting as a synthesis of the temporal languages for verification of discrete systems with first-order constraint query languages.

Although we approached $L_{\text{PATH}}$ 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 $L_{\text{PATH}}$. 

36
We do not claim that $L_{\text{PATH}}$ 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 $L_{\text{PATH}}$ (beyond the PTIME complexity bound), and we know even less about languages FO$+T$. We conjecture that FO + C $\subseteq$ FO + Con $\subseteq$ $L_{\text{PATH}}$, but we know of no techniques for proving separation results of this kind.

Remark C. Giannella and D. Van Gucht independently discovered the closure of FO + LIN and FO + POLY under connectivity operators C and Con [14] (our Corollary 4, Proposition 6, and a remark preceding that Proposition). Their proof uses cylindrical algebraic decomposition instead of Local Triviality, and consequently cannot show the closure under all topological properties as we do here.

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

References


