Deciding termination of query evaluation in transitive-closure logics for constraint databases

Floris Geerts* and Bart Kuijpers
University of Limburg, Belgium

Abstract

We study extensions of first-order logic over the reals with different types of transitive-closure operators as query languages for constraint databases that can be described by Boolean combinations of polynomial inequalities. We are in particular interested in deciding the termination of the evaluation of queries expressible in these transitive-closure logics. It turns out that termination is undecidable in general. However, we show that the termination of the transitive closure of a continuous function graph in the two-dimensional plane is decidable, and even expressible in first-order logic over the reals. Based on this result, we identify a particular transitive-closure logic for which termination of query evaluation is decidable and which is more expressive than first-order logic. Furthermore, we can define a guarded fragment in which exactly the terminating queries of this language are expressible.

1 Introduction

The framework of *constraint databases*, introduced in 1990 by Kanellakis, Kuper and Revesz [10] and by now well-studied [13], provides an elegant and powerful model for applications that deal with infinite sets of points in some real space $\mathbf{R}^n$, like for instance spatial databases. In the setting of the constraint model, these infinite sets are finitely represented as a Boolean combination of polynomial equalities and inequalities.

The relational calculus augmented with polynomial constraints, FO for short, is the standard first-order query language for constraint databases. Properties of this language are well-known [13], in particular, an important deficiency of FO is that its expressive power is rather limited. Therefore, more expressive extensions of FO have been introduced and studied. One such class of languages is extensions of FO with various *transitive-closure operators*. Recently, we introduced FO+TC and FO+TCS, two languages in which an operator is added to FO that allows the computation of the transitive closure of unparameterized sets in some $\mathbf{R}^{2k}$ [9]. In the latter language also FO-definable stop conditions are allowed to control the evaluation of the transitive-closure. The fragment of FO+TCS, in which multiplication is disallowed, was shown to be computationally complete on databases that can be defined by linear polynomial constraints. Later on, Kreutzer has studied a language that we refer to as FO+KTC [12]. This is an extension of FO with a transitive-closure operator that may be applied to parameterized sets. In FO+KTC, the evaluation of a transitive-closure expression may be controlled by the termination of particular paths in its computation rather than by the termination of the transitive closure of the complete set. It was shown that the fragment of FO+KTC, that does not use multiplication, is computationally complete on databases that can be defined by linear constraints [12].

In all of these transitive-closure languages, we face the well-know fact that recursion involving

*Contact author: Floris Geerts, University of Limburg, Dept. WNI, B-3590 Diepenbeek, Belgium, E-mail: floris.geerts@luc.ac.be

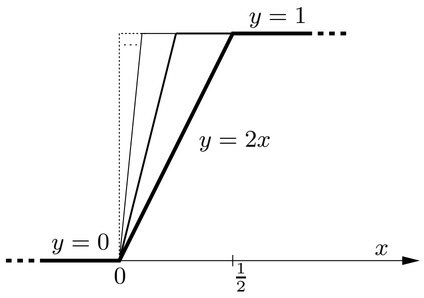

Figure 1: A function graph (thick) with non-terminating transitive closure (thin).

arithmetic over an infinite domain, such as the reals with addition and multiplication in this setting, is not guaranteed to terminate. In this paper, we are interested in termination of query evaluation in these different languages and in particular in *deciding termination.* We show that the termination of the evaluation of a given query, expressed in any of these languages, on a given input database is undecidable as soon as the transitive closure of 4-ary relations is allowed. In fact, a known undecidable problem in *dynamical systems theory*, namely deciding *nilpotency* of functions from $\mathbf{R}^2$ to $\mathbf{R}^2$ [2, 3], can be reduced to our decision problem. When the transitive-closure operator is restricted to work on binary relations, the matter is more complicated. We show the undecidability of termination for FO+TCS restricted to binary relations. However, both for FO+TC and FO+KTC restricted to binary relations, finding an algorithm for deciding termination would also solve some outstanding open problems in dynamical systems theory. Indeed, a decision procedure for FO+TC restricted to binary relations would solve the *nilpotency problem* for functions from $\mathbf{R}$ to $\mathbf{R}$ and a decision procedure for FO+KTC restricted to binary relations would solve the *point-to-fixed-point problem.* Both these problems are already open for some time [2, 11].

For FO+TC restricted to binary relations, we have obtained a positive decidability result, however. A basic problem in this context is deciding whether the transitive closure of a subset of the two-dimensional plane, viewed as a binary relation over the reals, terminates. Even if these subsets are restricted to be the graphs of possibly discontinuous functions from $\mathbf{R}$ to $\mathbf{R}$, this problem is already puzzling dynamical system theorists for a number of years (it relates to the above mentioned point-to-fixed-point problem). However, when we restrict our attention to the transitive closure of *continuous function graphs*, we can show that the termination of the transitive closure of these figures is decidable. As an illustration of possible inputs for this decision problem, two continuous function graphs are given in Figures 1 and 2. The first one has a non-terminating transitive closure, but the second terminates after four iterations. Furthermore, we show that this decision procedure is expressible in FO. In the course of our proof, we also give a stronger version of Sharkovskiĭ's theorem [1] from dynamical systems theory for terminating continuous functions. We also extend another result in this area, namely, we show that nilpotency of continuous semi-algebraic functions is decidable and that this decision procedure is even expressible in FO. Previously, this result was only stated, without proof, for continuous piecewise affine functions [2].

Based on this decision result, we define a fragment of FO+TC in which the transitive-closure operator is restricted to work on graphs of continuous functions from $\mathbf{R}$ to $\mathbf{R}$. Termination of queries in this language is shown to be decidable. Furthermore, we define a *guarded* fragment of this transitive-closure logic in which only, and all, terminating queries can be formulated. We also show that this very restricted form of transitive closure yields a language that is strictly more expressive than FO.

This paper is organized as follows. In Section 2, we define constraint databases, the query language FO and several extensions with transitive-closure operators. In Section 3, we give general undecidability results. In Section 4, we give a procedure to decide termination of the transitive closure of continuous function graphs in the plane. In Section 5, we study the extension of FO with a transitive closure operator that is restricted to work on continuous function graphs.

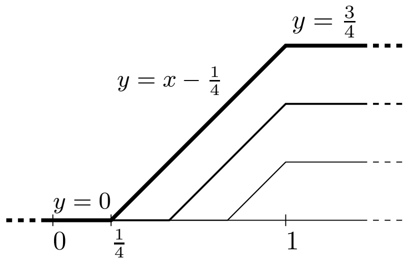

Figure 2: A function graph (thick) with terminating transitive closure (thin).

In this section, we also describe a guarded fragment of this language and give expressiveness results. The paper concludes with some remarks.

2 Definitions and preliminaries

In this section, we define constraint databases and FO, the standard first-order query language for constraint databases. We also define existing extensions of this logic with different transitive-closure operators.

2.1 Constraint databases and first-order logic over the reals

Let $\mathbf{R}$ denote the set of the real numbers, and $\mathbf{R}^n$ the n-dimensional real space (for $n \geq 1$).

Definition 1. An n-dimensional *constraint database* is a geometrical figure in $\mathbf{R}^n$ that can be defined as a Boolean combination (union, intersection and complement) of sets of the form $\{(x_1, \dots, x_n) \mid p(x_1, \dots, x_n) > 0\}$, where $p(x_1, \dots, x_n)$ is a polynomial with integer coefficients in the real variables $x_1, \dots, x_n$.[1] $\square$

We remark that in mathematical terminology, constraint databases are called *semi-algebraic sets*. If a constraint database can be described by linear polynomials only, we refer to it as a *linear constraint database*.

[1] Spatial databases in the constraint model are usually defined as finite collections of such geometrical figures. We have chosen the simpler definition of a database as a single geometrical figure, but all results carry over to the more general setting.

In this paper, we will use FO, the relational calculus augmented with polynomial inequalities as a basic query language.

Definition 2. A formula in FO, over an n-dimensional input database, is a first-order logic formula $\varphi(y_1, \dots, y_m, S)$, built, using the logical connectives and quantification over real variables, from two kinds of atomic formulas: $S(x_1, \dots, x_n)$ and $p(x_1, \dots, x_k) > 0$, where S is a n-ary relation name representing the input database and $p(x_1, \dots, x_k)$ is a polynomial with integer coefficients in the real variables $x_1, \dots, x_k$. $\square$

Variables in such formulas are assumed to range over $\mathbf{R}$. Tarski's quantifier-elimination procedure for first-order logic over the reals guarantees that FO expressions can be evaluated effectively on constraint database inputs and their result is a constraint database (in $\mathbf{R}^m$) that also can be described by means of polynomial constraints over the reals [5, 15].

If $\varphi(y_1, \dots, y_m, S)$ is a FO formula, $a_1, \dots, a_m$ are reals, and A is a n-dimensional constraint database, then we denote by $(a_1, \dots, a_m, A) \models \varphi(y_1, \dots, y_m, S)$ that $(a_1, \dots, a_m, A)$ satisfies φ.

The fragment of FO in which multiplication is disallowed is called $\mathrm{FO}_{\mathrm{Lin}}$.

2.2 Transitive-closure logics

We now define a number of extensions of FO (and of $\mathrm{FO}_{\mathrm{Lin}}$) with different types of transitive-closure operators. Recently, we introduced and studied the first two extensions, FO+TC and FO+TCS [8, 9]. The latter extension, FO+KTC, is due to Kreutzer [12].

Definition 3. A formula in FO+TC is a formula built in the same way as an FO formula, but with the following extra formation rule: if $\psi(\vec{x}, \vec{y})$ is a formula with $\vec{x}$ and $\vec{y}$ k-tuples of real variables, and $\vec{s}$, $\vec{t}$ are k-tuples of real variables, then

$$[\mathrm{TC}_{\vec{x};\vec{y}}\, \psi(\vec{x}, \vec{y})](\vec{s}, \vec{t}) \tag{1}$$

is also a formula which has as free variables those in $\vec{s}$ and $\vec{t}$. $\square$

Since the only free variables in $\psi(\vec{x}, \vec{y})$ are those in $\vec{x}$ and $\vec{y}$, parameters are not allowed in applications of the TC operator.

The semantics of a subformula of the above form (1) evaluated on a database A is defined in the following operational manner: Start computing the following iterative sequence of $2k$-ary relations: $X_0 := \psi(A)$ and $X_{i+1} := X_i \cup \{(\vec{x}, \vec{y}) \in \mathbf{R}^{2k} \mid (\exists \vec{z})\, (X_i(\vec{x}, \vec{z}) \wedge X_0(\vec{z}, \vec{y}))\}$ and stop as soon as $X_i = X_{i+1}$. The semantics of $[\mathrm{TC}_{\vec{x};\vec{y}}\, \psi](\vec{s}, \vec{t})$ is then defined as $(\vec{s}, \vec{t})$ belonging to the $2k$-ary relation X_i.

Since every step in the above algorithm, including the test for $X_i = X_{i+1}$, is expressible in FO, every step is effective and the only reason why the evaluation may not be effective is that the computation does *not terminate*. In that case the semantics of the formula (1) (and any other formula in which it occurs as subformula) is undefined.

As an example of a FO+TC formula over a 2-dimensional input database, we take

$$[\mathrm{TC}_{x;y}\, S(x, y)](s, t).$$

This expression, when applied to a 2-dimensional figure, returns the transitive closure of this figure, viewed as a binary relation over $\mathbf{R}$. For illustrations of the evaluation of this formula, we refer to Figures 1 and 2 in the Introduction.

The language $\mathrm{FO_{Lin}}$+TC consists of all FO+TC formulas that do not use multiplication.

The following language, FO+TCS, is a modification of FO+TC that incorporates a construction to specify explicit termination conditions on transitive closure computations.

Definition 4. A formula in FO+TCS is built in the same way as in FO but with the following extra formation rule: if $\psi(\vec{x}, \vec{y})$ is a formula with $\vec{x}$ and $\vec{y}$ k-tuples of real variables; σ is an FO sentence over the input database and a special $2k$-ary relation name X; and $\vec{s}$, $\vec{t}$ are k-tuples of real variables, then

$$[\mathrm{TC}_{\vec{x};\vec{y}}\, \psi(\vec{x}, \vec{y}) \mid \sigma](\vec{s}, \vec{t}) \tag{2}$$

is also a formula which has as free variables those in $\vec{s}$ and $\vec{t}$. We call σ the *stop condition* of this formula. □

The semantics of a subformula of the above form (2) evaluated on databases A is defined in the same manner as in the case without stop condition, but now we stop not only in case an i is found such that $X_i = X_{i+1}$, but also when an i is found such that $(A, X_i) \models \sigma$, whichever case occurs first. As above, we also consider the restriction $\mathrm{FO_{Lin}}$+TCS. It was shown that $\mathrm{FO_{Lin}}$+TCS is computationally complete on linear constraint databases [9].

Finally, we define FO+KTC. In finite model theory [7], transitive-closure logics, in general, allow the use of parameters. Also the language FO+KTC allows parameters in the transitive closure.

Definition 5. A formula in FO+KTC is a formula built in the same way as an FO formula, but with the following extra formation rule: if $\psi(\vec{x}, \vec{y}, \vec{u})$ is a formula with $\vec{x}$ and $\vec{y}$ k-tuples of real variables, $\vec{u}$ some further ℓ-tuple of free variables, and where $\vec{s}$, $\vec{t}$ are k-tuples of real terms, then

$$[\mathrm{TC}_{\vec{x};\vec{y}}\, \psi(\vec{x}, \vec{y}, \vec{u})](\vec{s}, \vec{t}) \tag{3}$$

is also a formula which has as free variables those in $\vec{s}$, $\vec{t}$ and $\vec{u}$. □

Since the free variables in $\psi(\vec{x}, \vec{y}, \vec{u})$ are those in $\vec{x}$, $\vec{y}$ and $\vec{u}$, here parameters are allowed in applications of the TC operator. The semantics of a subformula of the form (3), with $\vec{s} = (s_1, \ldots, s_k)$, evaluated on a database A is defined in the following operational manner: Let I be the set of indices i for which s_i is a constant. Then we start computing the following iterative sequence of $2k$-ary relations: $X_0 := \psi(A) \wedge \bigwedge_{i \in I} (s_i = x_i)$ and $X_{i+1} := X_i \cup \{(\vec{x}, \vec{y}) \in \mathbf{R}^{2k} \mid (\exists \vec{z})\, (X_i(\vec{x}, \vec{z}) \wedge \psi(\vec{z}, \vec{y}))\}$ and stop as soon as $X_i = X_{i+1}$. The semantics of $[\mathrm{TC}_{\vec{x};\vec{y}}\, \psi](\vec{s}, \vec{t})$ is then defined as $(\vec{s}, \vec{t})$ belonging to the $2k$-ary relation X_i.

We again also consider the fragment $\mathrm{FO_{Lin}}$+KTC of this language. It was shown that $\mathrm{FO_{Lin}}$+KTC is computationally complete on linear constraint databases [12].

For all of the transitive-closure logics that we have introduced in this section, we consider fragments in which the transitive-closure operator is restricted to work on relations of arity at most $2k$ and we denote this by adding $2k$ as a superscript to the name of the language. For example, in the language FO+TCS^4, the transitive closure is restricted to binary and 4-ary relations.

3 Undecidability of the termination of the evaluation of transitive-closure formulas

The decision problems that we consider in this section and the next take couples (φ, A) as input, where φ is an expression in the transitive-closure logic under consideration and A is an input database, and the answer to the decision problem is *yes* if the computation of the semantics of φ on A (as defined for that logic) terminates. We then say, for short, that φ *terminates on* A.

We give the following general undecidability result concerning termination. In the proof and further on, the notion of nilpotency of a function will be used: a function $f : \mathbf{R}^n \to \mathbf{R}^n$ is called *nilpotent* if there exists a natural number $k \geq 1$ such that for all $\vec{x} \in \mathbf{R}^n$, $f^k(\vec{x}) = (0, \ldots, 0)$.

Theorem 1. *It is undecidable whether a given formula in* FO+TC^4 *terminates on a given input database.*

Proof (sketch). We reduce deciding whether a piecewise affine function[2] $f : \mathbf{R}^2 \to \mathbf{R}^2$ is nilpotent to deciding whether the evaluation of a formula in FO+TC^4 terminates. So, assume that termination of formulas in FO+TC^4 is decidable. For a given piecewise affine function $f : \mathbf{R}^2 \to \mathbf{R}^2$, graph$(f)$, the graph of f, is a semi-algebraic subset of $\mathbf{R}^4$. We give a procedure to decide whether f is nilpotent:

Step 1. Decide whether the FO+TC^4-query $[\mathrm{TC}_{x_1,x_2;y_1,y_2}\, S(x_1, x_2, y_1, y_2)](s_1, s_2; t_1, t_2)$ terminates on the input graph(f); if the answer is *no*, then return *no*, else continue with Step 2.

Step 2. compute $f^1(\mathbf{R}^2), f^2(\mathbf{R}^2), f^3(\mathbf{R}^2), \ldots$ and return *yes* if this ends with $\{(0, 0)\}$, else return *no*.

This algorithm decides correctly whether f is nilpotent, since for a nilpotent f, the evaluation of the transitive closure of graph(f) will terminate, and the process in Step 2 is therefore also guaranteed to terminate. Since nilpotency of piecewise affine functions from $\mathbf{R}^2$ to $\mathbf{R}^2$ is known to be undecidable [3], this completes the proof. □

The following corollary follows from the previous theorem and the fact that FO+TC^4-formulas are in FO+KTC^4.

Corollary 1. *It is undecidable whether a given formula in* FO+KTC^4 *terminates on a given input database.* □

For transitive-closure logics with stop-condition, we even have undecidability for transitive closure restricted to binary relations.

Theorem 2. *It is undecidable whether a given formula in* FO+TCS^2 *terminates on a given input database.*

Proof. We prove this result by reducing the undecidability of a variant of Hilbert's 10th problem to it.[3] This problem is deciding whether a polynomial $p(x_1, \ldots, x_{13})$ in 13 real variables and with integer coefficients has a solution in the natural numbers [6].

For any such polynomial $p(x_1, \ldots, x_{13})$, let σ_p be the FO-expressible stop-condition: $(\exists x_1) \cdots (\exists x_{13})(\bigwedge_{i=1}^{13} X(-1, x_i) \wedge p(x_1, \ldots, x_{13}) = 0)$. Since, in consecutive iterations of the computation of the transitive closure of the graph of $y = x + 1$, -1 is mapped to $0, 1, 2, \ldots$, it is easy to see that $p(x_1, \ldots, x_{13})$ has an integer solution if and only if

$$[\mathrm{TC}_{x;y}\, (y = x + 1) \mid \sigma_p](s, t)$$

terminates. Since the above mentioned Diophantine decision problem is undecidable, this completes the proof. □

[2]A function $f : \mathbf{R}^n \to \mathbf{R}^n$ is called *piecewise affine* if its graph is a linear semi-algebraic set in $\mathbf{R}^n \times \mathbf{R}^n$.

[3]Because of this proof technique, this proof can be reused for Corollary 5.

The results in this section are complete for the languages FO+TC, FO+TCS and FO+KTC, apart from the cases FO+TC^2 and FO+KTC^2. The former case will be studied in the next sections. For the latter case, we remark that an open problem in dynamical systems theory, namely, the *point-to-fixed-point problem* reduces to it. This open problem is the decision problem that asks whether for a given $x_0 \in \mathbf{R}$ and a given piecewise affine function $f : \mathbf{R} \to \mathbf{R}$, the sequence $x_0, f(x_0), f^2(x_0), f^3(x_0), \dots$ reaches a fixed point. Even for piecewise linear functions with two non-constant pieces this problem is open [2, 11]. It is clear that this point-to-fixed-point problem can be expressed in FO+KTC^2. So, we are left with the following unsolved problem.

Open problem 1. *Is it decidable whether a given formula in* FO+KTC^2 *terminates on a given input database?* □

4 Deciding termination for continuous function graphs in the plane

In this section, we study the termination of the transitive closure of a constant semi-algebraic subset of the plane[4], viewed as a binary relation over $\mathbf{R}$. In the previous section, we have shown that deciding nilpotency of functions can be reduced to deciding termination of the transitive closure of their function graphs. Since deciding nilpotency of (possibly discontinuous) functions from $\mathbf{R}$ to $\mathbf{R}$ is an outstanding open problem [2, 3], this technique works only as long as these function graphs are in $\mathbf{R}^4$. We therefore have another unsolved problem.

Open problem 2. *Is it decidable whether a given formula in* FO+TC^2 *terminates on a given input database?* □

[4]Let A be a subset of $\mathbf{R}^2$. We say that A *has a terminating transitive closure*, when the evaluation of the query $[\mathrm{TC}_{x_1,x_2;y_1,y_2}\, S(x_1,x_2,y_1,y_2)](s_1,s_2;t_1,t_2)$ terminates on input A, using the semantics of FO+TC.

There are obviously classes of functions for which deciding termination of their function graphs is trivial. An example is the class of the piecewise constant functions. In this section, we concentrate on a class that is non-trivial, namely the class of the *continuous semi-algebraic*[5] *functions from* $\mathbf{R}$ *to* $\mathbf{R}$. The main purpose of this section is to prove the following theorem.

Theorem 3. *There is a decision procedure that on input a continuous semi-algebraic function f : $\mathbf{R} \to \mathbf{R}$ decides whether the transitive closure of* graph(f) *terminates. Furthermore, this decision procedure can be expressed by a formula in* FO *(over a binary input database representing the graph of he input function).* □

Before we arrive at the proof of Theorem 3, we give a series of six technical lemma's. First, we introduce some terminology.

Let $f : \mathbf{R} \to \mathbf{R}$ be a continuous function and let x be a real number. We denote by $\mathrm{Orb}(x, f)$ the set $\{f^k(x) \mid k \geq 1\}$ and call it the *orbit of* x (with respect to f). A real number x is said to be a *periodic point of* f if $f^d(x) = x$ for some natural number $d \geq 1$. And we call the smallest such d the *period of* x (with respect to f). Let $Per(f)$ be the set of periodic points in $\mathbf{R}$ of f. If a real number x is not a periodic point of f, but if $f^k(x)$ is periodic, we call x an *eventually periodic point of* f and we call the smallest such number k the *run-up of* x (with respect to f). Finally, we call f *terminating* if graph(f) has a terminating transitive closure.

The following lemma holds for arbitrary functions, not only for continuous ones. We also remark that Lemmas 1– 4 hold for arbitrary functions, not only for semi-algebraic ones.

Lemma 1. *The function $f : \mathbf{R} \to \mathbf{R}$ is terminating if and only if there exist natural numbers k and d such that for each $x \in \mathbf{R}$, $f^k(x)$ is a periodic point of f of period at most d.*

Proof. The "if" direction is straightforward, so we focus on the "only-if" direction. If f is terminating then there exists an N such that for each

[5]A function is called *semi-algebraic* if its graph is semi-algebraic.

$x \in \mathbf{R}$, the cardinality of $\mathrm{Orb}(x, f)$ is smaller than N. It is easy to see that x is eventually periodic if and only if $\mathrm{Orb}(x, f)$ is finite. It is clear that for each $x \in \mathbf{R}$ both the run-up as the period is bounded by N. This concludes the proof. □

Lemma 2. *Let $f : \mathbf{R} \to \mathbf{R}$ be a continuous function. If f is terminating, then $Per(f)$ is a non-empty, closed and connected part of $\mathbf{R}$. In particular, $Per(f) = f^k(\mathbf{R})$ for some $k \geq 1$.*

Proof. It follows from Lemma 1 that, for a terminating f, there is a bound d on the periods of f and a bound k on the run-ups of f.

Denote by C_i the set of fixed points of f^i, i.e., the set of $x \in \mathbf{R}$ for which $f^i(x) = x$. Assume that $x = \lim_{k\to\infty} x_k$ with all $x_k \in C_i$. From the continuity of f it follows that $f^i(x) = \lim_{k\to\infty} f^i(x_k) = \lim_{k\to\infty} x_k = x$. Hence C_i is closed and also $Per(f)$ since this set equals $C_1 \cup \cdots \cup C_d$.

Since all the run-ups are smaller than k, it is clear that $f^k(\mathbf{R}) \subseteq Per(f)$. On the other hand, let x be a periodic point of f with period d', with $d' \leq d$. Let $y = f^a(x)$ where a is $-k \mod d'$. Then $f^k(y) = f^{(k+a) \mod d'}(x) = f^{qd'}(x)$ for some integer $q \geq 1$. Since $f^{qd'}(x) = x$, x belongs to $f^k(\mathbf{R})$ and therefore $Per(f) \subseteq f^k(\mathbf{R})$. Since f is continuous and $\mathbf{R}$ is connected, also $f^k(\mathbf{R})$ is connected. □

Lemma 3. *Let C be a non-empty, closed and connected part of $\mathbf{R}$. If $f : C \to C$ is a continuous function and if every $x \in C$ is periodic for f, then f or f^2 is the identity mapping on C.*

Proof. We remark that C can either be $\mathbf{R}$ or of the form $[a, +\infty)$, $(-\infty, b]$ or $[a, b]$ with $a \leq b$. We will cover all these cases by taking C to be $[a, b]$, with the understanding that a can be $-\infty$ and/or b can be $+\infty$.

First of all, we observe that f must be a bijection of C. Indeed, let $y \in C$ a periodic point of period d, then $y = f^d(y) = f(f^{d-1}(y)) = f(x)$ with $x = f^{d-1}(y)$. Hence f is surjective. Next suppose that $f(x) = f(y)$. This implies that $f(x)$ and $f(y)$ are in the same orbit of f, say of period d. Therefore, $x = f^{d-1}(f(x)) = f^{d-1}(f(y)) = y$ and f is also injective.

By continuity of f we must have that either $f(a) = a$ and $f(b) = b$, or $f(a) = b$ and $f(b) = a$. To prove the lemma, it suffices to show that $f(a) = a$ and $f(b) = b$ implies that f is the identity mapping. Indeed, the second case reduces to the first when applied to f^2.

So, we assume that $f(a) = a$ and $f(b) = b$. Suppose that there exists an $x_0 \in C$ such that $f(x_0) \neq x_0$. By continuity, this means that there exists an open interval (c, d) containing x_0 such that $f(x) \neq x$ in (c, d). Let (c, d) be maximal with these properties. From the maximality of (c, d) it follows that $f(c) = c$ and $f(d) = d$ and hence $f((c, d)) = (c, d)$ (for the unbounded cases, c and/or d may be $-\infty$ and $+\infty$, or just one of them). Moreover, we have that either $f(x) > x$ for all $x \in (c, d)$, or $f(x) < x$ for all $x \in (c, d)$. Take a point $y \in (c, d)$, then $y, f(y), f^2(y), \ldots$ is a strictly increasing (if $f(x) > x$) or a strictly decreasing (if $f(x) < x$) sequence of points. Hence, (c, d) does not contain any periodic points, which contradicts the premises. Hence, f is the identity. □

Lemma 4. *For a continuous and terminating $f : \mathbf{R} \to \mathbf{R}$, $Per(f) = \{x \in \mathbf{R} \mid f^2(x) = x\}$.*

Proof. If f is terminating, then, by Lemma 2, $Per(f)$ is a closed and connected. Therefore, Lemma 3 can be applied to f restricted to $Per(f)$. This shows that $Per(f) \subseteq \{x \in \mathbf{R} \mid f^2(x) = x\}$. The other inclusion follows from the fact that any x which satisfies $f^2(x) = x$ has period 1 or 2. □

As in the proof of Lemma 2, let C_i denote the set of fixed points of f^i. From the previous lemmas it follows that for continuous and terminating f,

$$Per(f) = C_1 \cup C_2,$$

and that either $C_2 \setminus C_1$ is empty and C_1 is non-empty or $C_2 \setminus C_1$ is non-empty and C_1 is a singleton with the points of $C_2 \setminus C_1$ appearing symmetrically around C_1.

Sharkovskiĭ's theorem [1], one of the most fundamental result in dynamical system theory, tells us that for a continuous and terminating $f : \mathbf{R} \to \mathbf{R}$ only periods $1, 2, 4, \ldots, 2^d$ can appear for some integer value d. We remark that Lemmas 1–4 hold for arbitrary continuous functions, not only for functions with a semi-algebraic graph, and that therefore the previous lemma has the following corollary which strengthens the previously mentioned result of Sharkovskiĭ's.

Corollary 2. *If $f : \mathbf{R} \to \mathbf{R}$ is continuous and terminating, then f can only have periodic points with periods 1 and 2.* □

Further on, in the proof of Theorem 3, for functions f for which $Per(f)$ is $\mathbf{R}$, we only have to investigate $Per(f)$. For other f we have to do further tests. Hereto, we now describe the construction of a continuous function $\tilde{f}$ from a given continuous function f.

Let C denote the set $Per(f)$, which, by Lemma 2 and the above assumption, we can take to be $[a, b]$, $[a, +\infty)$ or $(-\infty, b]$.

First, we collapse C to $\{a\}$ if C is bounded; to $\{a\}$ if C is $[a, +\infty)$; and to $\{b\}$ if C is $(-\infty, b]$. Let us first consider the case $C = [a, b]$. Let $f_{\in C} = \{x \in \mathbf{R} \mid f(x) \in C\}$, $f_{<C} = \{x \in \mathbf{R} \mid f(x) < a\}$, and $f_{>C} = \{x \in \mathbf{R} \mid f(x) > b\}$.

We define the continuous function f_1 on $\mathbf{R}$ as $f_1(x) :=$

$$\begin{cases} f(x) & \text{if } x \in f_{<C} \text{ and } x < a \\ f(x) - (b - a) & \text{if } x \in f_{>C} \text{ and } x < a \\ f(x + (b - a)) & \text{if } x \in f_{<C} \text{ and } x > a \\ f(x + (b - a)) - (b - a) & \text{if } x \in f_{>C} \text{ and } x > a \\ a & \text{if } x \in f_{\in C}. \end{cases}$$

This construction is illustrated in Figure 3.

Let us next consider the case $C = [a, +\infty)$. Here, the function f_1 on $\mathbf{R}$ is defined as

$$\begin{cases} f(x) & \text{if } x \in f_{<C} \text{ and } x < a \\ a & \text{if } x \in f_{\in C} \text{ and } x < a \text{ or if } x \geq a. \end{cases}$$

In the case $C = (-\infty, b]$, f_1 is defined similarly to the previous case. Finally, we define

$$\tilde{f}(x) := f_1(x + \alpha) - \alpha,$$

where α is a or b, depending on the case.

The following lemma is readily verified.

Lemma 5. *Let $f : \mathbf{R} \to \mathbf{R}$ be a function. We have $f^k(\mathbf{R}) = Per(f)$ if and only if $\tilde{f}^k(\mathbf{R}) = \{0\}$.* □

As mentioned in the previous section, in the area of dynamical systems, a function $\tilde{f}$ is called *nilpotent* if $\tilde{f}^k(\mathbf{R}) = \{0\}$ for some integer k. The following lemma's show that this is a decidable property in our setting. For continuous piecewise affine functions this result was already stated (without proof) [3]. So, we extend this result to continuous semi-algebraic functions and furthermore show that the decision procedure is expressible in FO and therefore decidable for continuous semi-algebraic functions.

Lemma 6. *There is an* FO *sentence that expresses whether a continuous semi-algebraic function $f : \mathbf{R} \to \mathbf{R}$ is nilpotent.*

Proof (sketch). We describe the algorithm NILPOTENT(input f) to decide nilpotency of continuous semi-algebraic functions $f : \mathbf{R} \to \mathbf{R}$ and later on argue its correctness.

Algorithm NILPOTENT(input f):

Step 1. Compute the set $\{x \in \mathbf{R} \mid f^2(x) = x\}$. If this set differs from $\{0\}$, then answer *no*, else continue with Step 2.

Step 2. Compute the set $B = \{r \mid \gamma_{BB}(r)\}$, where $\gamma_{BB}(r)$ is the formula that defines positive real numbers r that satisfy one of the following three conditions:

1. $\lim_{x\to-\infty} f(x)$ and $\lim_{x\to+\infty} f(x)$ are constants and $f((-\infty, r]) \subset (-r, +r)$ and $f([r, +\infty)) \subset (-r, +r)$;
2. $\lim_{x\to-\infty} f(x) = +\infty$ and $\lim_{x\to+\infty} f(x)$ is a constant and $f([r, +\infty)) \subset (-r, +r)$;
3. $\lim_{x\to-\infty} f(x)$ is a constant and $\lim_{x\to+\infty} f(x) = -\infty$ and $f((-\infty, r]) \subset (-r, +r)$;

If B is empty, answer *no*, else compute the infinum r_0 of B and continue with Step 3.

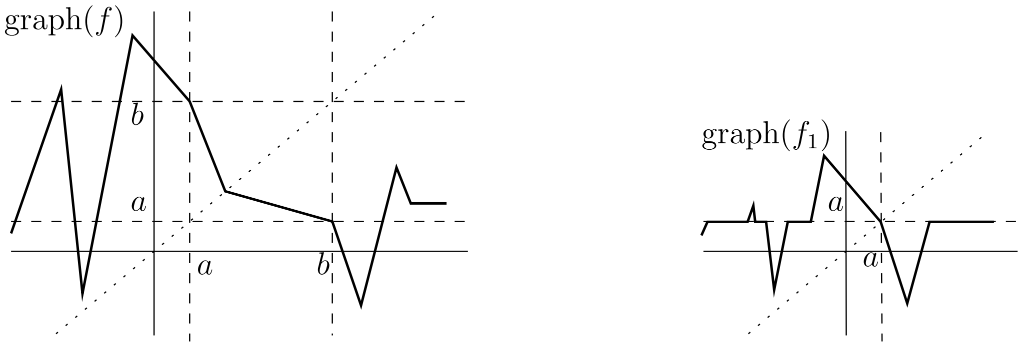

Figure 3: Illustration of the construction of f_1 (right) from f (left).

Step 3. Let g be the function defined as $g(x) := f(x)$ if $-r_0 < x < r_0$ and $g(x) := f(-r_0)$ if $x \leq -r_0$ and $g(x) := f(r_0)$ if $x \geq r_0$. Take r_1 larger than r_0 and $\max\{|g(x)| \mid x \in \mathbf{R}\}$.

If for g there exists a positive real number ε such that

1. g is constant 0 on $(-\varepsilon, +\varepsilon)$, or
2. g is constant 0 on $(-\varepsilon, 0)$ and has a right tangent with strictly negative slope in 0, or
3. g is constant 0 on $(0, +\varepsilon)$ and has a left tangent with strictly negative slope in 0,

then continue with Step 4, else answer *no*.

Step 4. If for all $x > 0$, $g(x) < x$ and $g^2(x) < x$ and for every $x < 0$, $g(x) > x$ and $g^2(x) > x$ holds, then answer *yes*, else answer *no*.

We now prove the correctness of the algorithm NILPOTENT. Clearly, if f has periodic points other than 0, then f cannot be nilpotent. This justifies the test in Step 1.

In Step 2, the consistency of the behavior of f towards $-\infty$ and $+\infty$ with nilpotency is tested. From the fact that f has a semi-algebraic graph it follows that the set B, computed in Step 2, is empty if (1) $\lim_{x\to-\infty} f(x) = -\infty$ or (2) $\lim_{x\to+\infty} f(x) = +\infty$ or (3) $\lim_{x\to-\infty} f(x) = +\infty$ and $\lim_{x\to+\infty} f(x) = -\infty$.

In Case (1), there exists a $M > 0$ such that f is strictly increasing on $(-\infty, -M]$ and for all $x < -M$, we have (1a) $f(x) < x < 0$ or (1b) $x < f(x) < 0$. Because graph(f) is semi-algebraic such an M exists (this follows from the Monotonicity Theorem [16]). Because of the test in Step 1, the case $f(x) = x$ cannot occur any more. In Case (1a), there exists an infinite orbit $\cdots < f^2(x) < f(x) < x < 0$, hence f is not nilpotent. In Case (1b), there exist arbitrary long orbits converging to x, namely from any point in the sequence $\cdots < f^{-2}(x) < f^{-1}(x) < x < 0$. Hence f is not nilpotent. For Case (2) a similar analysis can be made, again depending on the graph of f being situated below or above the diagonal. Also in Case (3), we have this phenomena, this time depending on the graph of f^2 being situated below or above the diagonal. Here, there exists an $M > 0$ such that f is strictly decreasing on $(-\infty, -M]$ and on $[M, +\infty)$ and for all $x > M$, $f(x) < 0 < x$ and (3a) $x < f^2(x)$ or (3b) $x > f^2(x)$. Because of the test in Step 1, there is no third case. In Case (3a), there exists an infinite orbit $\cdots < f^3(x) < f(x) < x < f^2(x) < f^4(x) < \cdots$, hence f is not nilpotent. In Case (3b), there exist arbitrary long orbits starting from a point in the sequence $\cdots < f^{-3}(x) < f^{-1}(x) < 0 < x < f^{-2}(x) < f^{-4}(x) < \cdots$. Hence f is not nilpotent. So, if B is empty, then f is nilpotent.

If B is non-empty, on the other hand, then $f((-\infty, r_0]) \subset (-r_0, r_0)$ and $f([r_0, +\infty)) \subset (-r_0, r_0)$, and for the function g, defined in Step 3, this also holds if you replace r_0 by r_1 and furthermore $g([-r_1, +r_1]) \subseteq [-r_1, +r_1]$ holds. By the choice of r_1, it is clear that f is nilpotent if and only if g is nilpotent.

In Steps 3 and 4, the consistency of the behavior of g in a neighborhood of 0 with nilpotency is tested. In the Cases (1), (2) and (3), $g^2(x) = 0$ holds for a small ε-environment of 0. Every different behavior of g in the neighborhood of 0, leads to infinitely long or arbitrarily long orbits of g (and hence of f). We omit the details, but this analysis can be done as in the case of Step 2.

The condition in Step 4, expresses the global convergence of g [2], which is equivalent to nilpotency of g because g^2 maps a neighborhood of 0 to 0 [3].

Finally, we remark that all computations and tests performed in the algorithm NILPOTENT, are expressible by a FO formula over the binary relation representing the graph of the input f. □

PROOF OF THEOREM 3. We describe a decision procedure TERMINATE that on input a function $f : \mathbf{R} \to \mathbf{R}$, decides whether the transitive closure of graph(f) terminates after a finite number of iterations.

Algorithm TERMINATE(input f):
Step 1. Compute the sets $C_1 = \{x \mid f(x) = x\}$ and $C_2 = \{x \mid f^2(x) = x\}$. If C_2 is an interval and if C_1 is a point with $C_2 \setminus C_1$ symmetrically around it or if $C_2 \setminus C_1$ is empty, then continue with Step 2, else answer *no*.

Step 2. If C_2 is $\mathbf{R}$, answer *yes*, else compute the function $\tilde{f}$ (as described before Lemma 5) and apply the algorithm NILPOTENT in the proof of Lemma 6 to $\tilde{f}$ and return the answer.

The correctness of this procedure follows from Lemmas 4, 5 and 6. It should be clear that all ingredients can be expressed in FO. □

We use the function f_1, given in Figure 1 in the Introduction, and the function f_2, given in Figure 2, to illustrate the decision procedure TERMINATE(input f). For f_1, $C_1 \cup C_2$ is $\{0, 1\}$, and therefore f_1 doesn't survive Step 1 and TERMINATE(input f_1) immediately returns *no*. For f_2, $C_1 \cup C_2$ is $\{0\}$, and therefore we have $\tilde{f}_2 = f_2$. Next, the algorithm NILPOTENT is called with input f_2. For f_2, the set B, computed in Step 2 of the algorithm NILPOTENT, is non-empty and r_0 is 1. So, the function g in Step 3 will be f_2 again and r_1 is 1. Since g is identical zero around the origin, finally the test in Step 4 decides. Here, we have that for $x > 0$, $g(x) < x$ and also $g^2(x) < x$ since $x - \frac{1}{4} < x$ and $x - \frac{1}{2} < x$. For $x < 0$, we have that both $g(x)$ and $g^2(x)$ are identical zero and thus the test succeeds also here. The output of NILPOTENT on input f_2 and therefore also the output of TERMINATE on input f_2 is *yes*.

For a continuous and terminating function, the periods that can appear are 1 and 2 (see Lemma 3). In dynamical systems theory, finding an upper bound on the length of the run-ups in terms of some characteristics of the function, is considered to be, even for piecewise affine functions, a difficult problem [14]. Take, for instance, the terminating continuous piecewise affine function that is constant towards $-\infty$ and $+\infty$ and that has turning points $(0, \frac{1}{3})$, $(\frac{1}{3}, \frac{2}{3} - \varepsilon)$, $(\frac{4}{9}, \frac{4}{9})$, $(\frac{5}{9}, \frac{5}{9})$, $(\frac{2}{3}, \frac{1}{3})$, and $(1, \frac{2}{3})$, with $\varepsilon > 0$ small. Here, it seems extremely difficult to find an upper bound on the length of the run-ups in terms of the number of line segments or of their endpoints. The best we can say in this context, is that from Theorem 3, it follows that also the maximal run-up can be computed.

Corollary 3. *Let $f : \mathbf{R} \to \mathbf{R}$ be a continuous, terminating semi-algebraic function. The maximal run-up of a point in $\mathbf{R}$ with respect to f is computable.* □

We end this section with a remark concerning termination of continuous functions that are defined on a connected part I of $\mathbf{R}$. Let $f : I \to I$ be such a function. We define the function $\bar{f}$ to be the continuous extension of f to $\mathbf{R}$ that is constant on $\mathbf{R} \setminus I$. It is readily verified that the transitive closure of graph(f) terminates if and only if $\bar{f}$ is terminating. We therefore have the following corollary of Theorem 3.

Corollary 4. *Let I be a connected part of $\mathbf{R}$. There is an* FO *expressible decision procedure that decides whether the transitive closure of the graph of a continuous semi-algebraic function $f : I \to I$ terminates.* □

5 Logics with transitive closure restricted to function graphs

In this section, we study fragments of FO+TC and FO+TCS where the transitive-closure operator is restricted to work only on the graphs of continuous semi-algebraic functions from $\mathbf{R}^k$ to $\mathbf{R}^k$. We will focus in particular on $k = 1$. These languages bear some similarity with *deterministic* transitive-closure logics in finite model theory [7].

If $\vec{x}$ and $\vec{y}$ are k-dimensional real vectors and if $\psi(\vec{x}, \vec{y})$ is an FO+TC-formula (resp. FO+TCS-formula), let γ_ψ be the FO+TC-sentence (resp. FO+TCS-sentence) $\gamma^1_\psi \wedge \gamma^2_\psi$, where γ^1_ψ expresses that $\psi(\vec{x}, \vec{y})$ defines the graph of a function from $\mathbf{R}^k$ to $\mathbf{R}^k$ and where γ^2_ψ expresses that $\psi(\vec{x}, \vec{y})$ defines a continuous function graph. We can express γ^2_ψ using the classical ε-δ definition of continuity. Therefore, it should be clear that γ^1_ψ and γ^2_ψ can be expressed by formulas that make direct calls to $\psi(\vec{x}, \vec{y})$. Thus, the following property is readily verified.

Property 1. *Let $\psi(\vec{x}, \vec{y})$ be an* FO+TC*-formula (resp.* FO+TCS*-formula). The evaluation of $\psi(\vec{x}, \vec{y})$ on an input database A terminates if and only if the evaluation of γ_ψ on A terminates.* □

Definition 6. We define FO+cTC (resp. FO+cTCS) to be the fragment of FO+TC (resp. FO+TCS) in which only TC-expressions of the form $[\mathrm{TC}_{\vec{x};\vec{y}}\, \psi(\vec{x}, \vec{y}) \wedge \gamma_\psi](\vec{s}, \vec{t})$ (resp. $[\mathrm{TC}_{\vec{x};\vec{y}}\, \psi(\vec{x}, \vec{y}) \wedge \gamma_\psi \mid \sigma](\vec{s}, \vec{t})$) can occur. □

We again use superscript numbers to denote restrictions on the arities of the relations of which transitive closure can be taken.

5.1 Deciding termination of the evaluation of FO+cTC^2 queries

Since, when $\psi(x, y)$ is $y = x + 1$, γ_ψ is *true*, from the proof of Theorem 2 the following negative result follows.

Corollary 5. *It is undecidable whether a given formula in* FO+cTCS^2 *terminates on a given input database.* □

We remark that for this undecidability it is not needed that the transitive closure of continuous functions on an *unbounded* domain is allowed ($f(x) = x + 1$ in the proof of Theorem 2). Even when, for example, only functions from $[0, 1]$ to $[0, 1]$ are allowed, we have undecidability. We can see this by modifying the proof of Theorem 2 as follows. For any polynomial $p(x_1, \ldots, x_{13})$, let σ_p be the FO-expressible stop-condition:

$$(\exists x_1) \cdots (\exists x_{13})(\bigwedge_{i=1}^{13}((\exists y_i)(x_i y_i = 1 \wedge X(1, y_i)) \vee x_i = 0 \vee x_i = 1) \wedge p(x_1, \ldots, x_{13}) = 0).$$

Since, in consecutive iterations, the function $\bar{f}$, for $f : [0, 1] \to [0, 1]$, with $f(x) = \frac{x}{x+1}$, maps 1 to $\frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \ldots$, it is then easy to see that $p(x_1, \ldots, x_{13})$ having an integer solution is equivalent to

$$[\mathrm{TC}_{x;y}\, \mathrm{graph}(\bar{f}) \mid \sigma_p](s, t)$$

terminating (remark again that the $\gamma_{\mathrm{graph}(\bar{f})}$ is *true*).

The main result of this section is the following.

Theorem 4. *It is decidable whether a given formula in* FO+cTC^2 *terminates on a given input database. Moreover, this decision procedure is expressible in* FO+cTC^2.

Proof (sketch). Given a formula φ in FO+cTC^2 and an input database A, we can decide whether the evaluation of φ on A terminates by first evaluating the deepest FO-formulas on which a TC-operator works on A and then using Theorem 3 to decide whether the computation of transitive closure halts on this set. If it does not terminate, we answer *no*, else we compute the result and continue recursively to less deep occurrences of TC-operators in φ. We continue this until the complete formula φ is processed. Only if we reach the end and all intermediate termination tests returned *yes*, we output *yes*.

The expressibility of the decision procedure in FO can also be proven by induction on the structure of the formula. □

5.2 A guarded fragment of FO+cTC^2

The fact that termination of FO+cTC^2-formulas is expressible in FO+cTC^2, allows us to define a *guarded* fragment, FO+cTC$^2_{\mathcal{G}}$, of this language. Indeed, if ψ is a formula in FO+cTC^2 of the form $[\mathrm{TC}_{\vec{x};\vec{y}}\,\psi(\vec{x},\vec{y})](\vec{s},\vec{t})$, let τ_ψ be the FO+cTC^2-sentence that expresses that this TC-expression terminates (obviously, τ_ψ also depends on the input database). We can now define the guarded fragment of FO+cTC^2, in which every TC-expression is accompanied by a *termination guard*.

Definition 7. We define FO+cTC$^2_{\mathcal{G}}$ to be the fragment of FO+cTC^2 in which only TC-expressions of the form $[\mathrm{TC}_{\vec{x};\vec{y}}\,\psi(\vec{x},\vec{y}) \wedge \tau_\psi](\vec{s},\vec{t})$ are allowed. □

The following property follows from the above remarks.

Property 2. *In the language* FO+cTC$^2_{\mathcal{G}}$*, every query terminates on all possible input databases. Furthermore, all terminating queries of* FO+cTC^2 *are expressible in* FO+cTC$^2_{\mathcal{G}}$*.* □

5.3 Expressivity results

Even the least expressive of the transitive-closure logics is more expressive than first-order logic.

Theorem 5. *The language* FO+cTC$^2_{\mathcal{G}}$ *is more expressive than* FO *on finite constraint databases.*

Proof. Consider the following query Q_{int} on 1-dimensional databases S: "Is S a singleton that contains a natural number?". The query Q_{int} is not expressible in FO (if it would be expressible, then also the predicate $\mathrm{nat}(x)$, expressing that x is a natural number, would be in FO). The query Q_{int} is expressible in FO+cTC$^2_{\mathcal{G}}$ by the sentence that says that S is a singleton that contains 0, 1 or an element $r > 1$ such that $(\exists s)(\exists t)([\mathrm{TC}_{x;y}\,\psi(x,y) \wedge \gamma_\psi \wedge \tau_{\psi(x,y)\wedge\gamma_\psi}](s,t) \wedge s = 1 \wedge t = 0)$, where $\varphi(r,x,y)$ defines the graph of the continuous piecewise affine function that maps x to

$$y = \begin{cases} 0 & \text{if } x \leq \frac{1}{r}, \\ x - \frac{1}{r} & \text{if } \frac{1}{r} < x < 1, \\ 1 - \frac{1}{r} & \text{if } x \geq 1, \end{cases}$$

and where $\psi(x,y)$ is the formula $(\exists r)(S(r) \wedge \varphi(r,x,y))$. Remark that γ_ψ is always *true*. The sentence $\tau_{\psi(x,y)\wedge\gamma_\psi}$ is *true* when the database is a singleton containing a number that is 0, 1, or larger that 1. The function given by $\varphi(r,x,y)$ is illustrated in Figure 2 for $r = 4$. The evaluation of this transitive closure is guaranteed to end after at most $\lceil r \rceil$ iterations and this sentence indeed expresses Q_{int} since $(1,0)$ belongs to the result of the transitive closure if and only if $r > 1$ is a natural number. □

6 Concluding remarks

We conclude with a number of remarks. One of our initial motivations to look at termination of query evaluation in transitive closure logics was to study the expressive power of FO+TC compared to that of FO+TCS. As mentioned in the Introduction, the latter language is computationally complete on linear databases. It is not clear whether FO+TC is also complete. In general, we have no way to separate these languages. But if we restrict ourselves to their fragments FO+cTC^2 and FO+cTCS^2, the fact that for the former termination is decidable, whereas it is not for the latter, might give the impression that at least these fragments can be separated. But this is not the case, since equivalence of formulas in these languages is undecidable. In fact, the expressions used in the proof of Theorem 2, are expressible in FO+TC (they don't even use an input database).

A last remark concerns the validity of the results in Section 4 for more general settings. Lemmas 1– 5 are also valid for arbitrary real closed fields R. One could ask whether the same is true for Lemma 6. However, the proof of the correctness of the FO-sentence which decides global convergence in Step 4 [2], relies on the Bolzano-Weierstrass theorem, which is known not to be valid for arbitrary real closed fields [4]. Furthermore, we can even prove that no FO-sentence exists that decides termination of semi-algebraic functions $f : R \to R$ for arbitrary real closed fields R.

References

[1] Ll. Alsedà, J. Llibre, and M. Misiurewicz. *Combinatorial Dynamics and Entropy in Dimension One*, volume 5 of *Advances Series in Nonlinear Dynamics*. World Scientific, 1993.

[2] V.D. Blondel, O. Bournez, P. Koiran, C.H. Papadimitriou, and J.N. Tsitsiklis. Deciding stability and mortality of piecewise affine dynamical systems. *Theoretical Computer Science*, 255(1-2):687–696, 2001.

[3] V.D. Blondel, O. Bournez, P. Koiran, and J.N. Tsitsiklis. The stability of saturated linear dynamical systems is undecidable. *Journal of Computer and System Sciences*, 62(3):442–462, 2001.

[4] J. Bochnak, M. Coste, and M.-F. Roy. *Real Algebraic Geometry*, volume 36 of *Ergebenisse der Mathematik und ihrer Grenzgebiete. Folge 3*. Springer, 1998.

[5] G.E. Collins. Quantifier elimination for real closed fields by cylindrical algebraic decomposition. *Lecture Notes in Computer Science*, 33:134–183, 1975.

[6] Martin Davis, Yuri Matijasevič, and Julia Robinson. Hilbert's Tenth Problem. Diophantine equations: positive aspects of a negative solution. In *Mathematical Developments Arising from Hilbert Problems*, volume 28, pages 323–378, Providence, Rhode Island, 1976. American Mathematical Society.

[7] H.-D. Ebbinghaus and J. Flum. *Finite Model Theory*. Springer-Verlag, 1995. second edition.

[8] F. Geerts. Linear approximation of semi-algebraic spatial databases using transitive closure logic, in arbitrary dimension. In G. Ghelli and G. Grahne, editors, *Proceedings of Database Programming Languages—DBPL'01*, Lecture Notes in Computer Science.

[9] F. Geerts and B. Kuijpers. Linear approximation of planar spatial databases using transitive-closure logic. In *Proceedings of the 19th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS'00*, pages 126–135. ACM Press, 2000.

[10] P.C. Kanellakis, G.M. Kuper, and P.Z. Revesz. Constraint query languages. *Journal of Computer and System Science*, 51(1):26–52, 1995. A preliminary report appeared in the *Proceedings 9th ACM Symposium on Principles of Database Systems (PODS'90)*.

[11] P. Koiran, M. Cosnard, and M. Garzon. Computability with low-dimensional dynamical systems. *Theoretical Computer Science*, 132:113–128, 1994.

[12] S. Kreutzer. Operational semantics for fixed-point logics on constraint databases. In R. Nieuwenhuis and A. Voronkov, editors, *LPAR*, volume 2250 of *Lecture Notes in Computer Science*, pages 470–484. Springer, 2001.

[13] G.M. Kuper, J. Paredaens, and L. Libkin. *Constraint databases.* Springer-Verlag, 1999.

[14] J. Llibre and C. Preston. Personal communication.

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

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