Hierarchies in Transitive Closure Logic, Stratified Datalog and Infinitary Logic*

Erich Grädel
Mathematische Grundlagen der Informatik
RWTH Aachen
D-52056 Aachen
Germany
graedel@informatik.rwth-aachen.de

Gregory L. McColm
Department of Mathematics
University of South Florida
Tampa, FL 33620
U.S.A
mccolm@math.usf.edu

January 1993
Revised August 1994

Abstract

We establish a general hierarchy theorem for quantifier classes in the infinitary logic $L^{\omega}_{\infty\omega}$ on finite structures. In particular, it is shown that no infinitary formula with bounded number of universal quantifiers can express the negation of a transitive closure.

This implies the solution of several open problems in finite model theory: On finite structures, positive transitive closure logic is not closed under negation. More generally the hierarchy defined by interleaving negation and transitive closure operators is strict. This proves a conjecture of Immerman.

We also separate the expressive power of several extensions of Datalog, giving new insight in the fine structure of stratified Datalog.

1 Introduction

It is well-known that the expressive power of first order logic (FO) is limited by the lack of a recursion mechanism. One of the simplest and most fundamental queries that is not first-order expressible is *transitive closure*, denoted TC. It assigns to a given binary relation E on universe U its reflexive transitive closure, i.e., the set of all pairs $(x, y) \in U \times U$ such that there exist $z_0, \ldots, z_r \in U$ with $z_0 = x$, $z_r = y$ and Ez_iz_{i+1} for all $i < r$. It was shown by Fagin [12], that TC is not expressible in FO (in fact not even in monadic-Σ^1_1).

On the other hand, almost every logic or database query language with recursive constructs is strong enough to define transitive closure. In fact, when logics like fixpoint logic or database query languages like datalog are introduced, the example usually given is a formula (or a program) that defines TC.

*To appear in Annals of Pure and Applied Logic. In this version the pictures are missing. For a paper version with pictures, write to Greg McColm. A preliminary version of this paper has appeared in the Proceedings of the 33rd Annual IEEE Symposium on Foundations of Computer Science (1992)

It is therefore natural to define a logic that extends FO precisely by the ability to define transitive closures. This logic, called *transitive closure logic* and denoted (FO + TC), was introduced by Immerman [21]. It increases the expressive power of first-order logic by a TC-operator which, applied to a formula $\varphi(\bar{x}, \bar{y})$ with two k-tuples of free variables, defines the transitive closure of the binary relation on k-tuples, which is defined by φ.

Another important motivation for transitive closure logic comes from the fundamental discovery in complexity theory that many of the basic complexity classes can be characterized in terms of logic — or more precisely, finite model theory — as natural extensions of first order logic, provided that an ordering of the underlying structures is available. The first discovery of this sort was made by Fagin [12], who proved that the queries in NP are precisely the Σ_1^1-definable queries. This readily generalizes to the result that the polynomial-time hierarchy consists of the queries in second order logic [33]. Later, Immerman [20] and Vardi [35] independently identified PTIME with least fixed point logic (FO + LFP). Immerman [21] then proved that the queries (always on ordered structures) definable by transitive closure logic coincide with the queries that are computable in NLOGSPACE and that LOGSPACE is captured by a deterministic analogue of (FO + TC), denoted (FO + DTC).

An important fragment of transitive closure logic is (FO + pos TC), which only allows *positive applications* of TC operators. In fact, Immerman's original result in [21] only identified (FO + pos TC) with NLOGSPACE. At that time most people believed that NLOGSPACE would not be closed under complementation, and therefore (FO + pos TC) not be equivalent to (FO + TC). However, the closure of of NSPACE complexity classes under complementation, proved by Immerman and Szelepcsényi in 1988 [22, 34], implied that *on ordered structures*, transitive closure logic collapses to its positive fragment.

It has been one of the more notable open problems in finite model theory (posed by Immerman in 1983 [21]) whether this is also true in the absence of order, i.e., whether (FO + pos TC) coincides with (FO + TC) on arbitrary finite structures. Immerman also formulated the more general problem of whether the hierarchy formed by interleaving TC-operators and negations collapses. Generally it has been suspected that this is not the case, i.e., that (FO + pos TC) is strictly weaker than (FO + TC) on finite structures and that the TC/negation-hierarchy is infinite.

These conjectures have motivated our investigation presented here, and they will be proved in this paper. But we will treat these questions in a more general context: Transitive closure logic can be embedded into the infinitary logic with bounded variables, denoted $L^\omega_{\infty\omega}$. We will develop a theory of quantifier classes in $L^\omega_{\infty\omega}$ — a variation of the usual prefix classes in first order logic. In particular, we will consider quantifier classes that are defined by words over the alphabet $\{\exists, \forall, \exists^*, \forall^*\}$ (where, say, $\exists^*$ represents unbounded blocks of existential quantifiers), corresponding to quantifier classes in (FO + TC), defined by words over the alphabet $\{\exists, \forall, T, N\}$ (where, say, T represents an application of the TC operator). It turns out that (for $\exists^*$ corresponding to T and $\forall^*$ to N) for all words $p \in \{\exists, \forall, T, N\}^*$, the queries definable in the quantifier class $\mathrm{TC}(p)$ in (FO + TC) are also definable in the quantifier class $L^\omega_{\infty\omega}(p)$ in infinitary logic. Our main result is a strong hierarchy theorem for these quantifier classes. It essentially says that if a word q cannot be embedded into a word p, then the quantifier class $\mathrm{TC}(q)$ is not contained in $L^\omega_{\infty\omega}(p)$.

The embedding of logics and database query languages into infinitary logic is not new. Inexpressibility results for fixpoint logic and for Datalog using game theoretic arguments that actually apply to infinitary logic have been used by several authors since the original work in [3] and [19]. Kolaitis and Vardi showed that certain queries cannot be defined in the existential negation-free fragment of $L^\omega_{\infty\omega}$ and are therefore not computable by Datalog programs [27]. Our more general treatment of quantifier classes in $L^\omega_{\infty\omega}$ yields structural results on certain extensions of Datalog, extending those in [31] and thus gives insight on the relationship between the fine structure of (FO + TC) and the fine structure of Stratified Datalog.

Here are the contents of this paper: In section 2 we recall the basic definitions on infinitary logic and introduce *quantifier classes* in $L^\omega_{\infty\omega}$. In section 3 we present the necessary background on transitive closure logic, discuss quantifier classes in this logic and relate them to $L^\omega_{\infty\omega}$. In section 4 we show how the Ehrenfeucht-Fraïssé games technique can be adapted to the quantifier classes in $L^\omega_{\infty\omega}$. In section 5 we use this game to prove the general hierarchy theorem on quantifier classes in infinitary logic; in particular we show that non-reachability is not in (FO + pos TC). There are several implications of our main result: First, Immerman's question about the negation/TC-hierarchy is answered. Second, it gives a new proof of our earlier result that (FO + DTC) is weaker than (FO + TC) [15]. Third, it implies a hierarchy theorem for narrow Henkin quantifiers, via results of Blass and Gurevich [4]. Finally, it extends our earlier results in [14] (on a non-collapsing hierarchy inside (FO + pos TC), formed by interleaving TC-operators and universal quantifiers). In section 6 we consider the implications of our main result on the fine structure of stratified Datalog. In the last section, we discuss the question whether our main theorem also holds for prefix classes (as opposed to quantifier classes) in first order logic.

2 Quantifier classes in infinitary logic

The infinitary logic with bounded number of variables, $L^\omega_{\infty\omega}$, extends first-order logic by allowing infinite disjunctions and conjunctions, but requires that every formula contains only a finite number of variables.

Definition 1 Let σ be a finite vocabulary of relational and constant symbols[1]. The class $L^k_{\infty\omega}$ over σ is the smallest set of formulae that

- includes the first-order formulae over σ that contain only variables among $x_1, \ldots, x_k$;
- contains with every formula φ also its negation $\neg\varphi$;
- contains with every formula φ also the formulae $(\exists x_i)\varphi$ and $(\forall x_i)\varphi$, where $1 \leq i \leq k$;
- contains with every set Φ of formulae also the disjunction $\bigvee \Phi$ and the conjunction $\bigwedge \Phi$ over all formulae of Φ.

Finally, let $L^\omega_{\infty\omega} := \bigcup_{k\in\mathbb{N}} L^k_{\infty\omega}$.

Example. It is well-known that the transitive closure can be expressed in $L^3_{\infty\omega}$: Let

$$\begin{aligned}\varphi_0(x,y) &\equiv (x=y)\\ \varphi_{n+1}(x,y) &\equiv (\exists z)(Exz \wedge (\exists x)(x=z \wedge \varphi_n(x,y))\end{aligned}$$

Then the formula

$$\psi(x,y) \equiv \bigvee_{n\in\mathbb{N}} \varphi_n(x,y)$$

[1] All vocabularies in this paper are finite and contain only relational and constant symbols (no function symbols)

expresses that there is path from x to y.

With every formula $\psi \in L^{\omega}_{\infty\omega}$, we can associate its *quantifier structure* which is a *set* of words over the alphabet $\{\exists, \forall\}$. Intuitively the quantifier structure $P(\psi)$ of a formula $\psi \in L^{\omega}_{\infty\omega}$ is constructed as follows: Consider the formula as a tree; push all negation symbols through to the leaves (the atomic formulae) using de Morgan's laws. Then a word $p \in \{\exists, \forall\}^*$ belongs to $P(\psi)$ if and only if it is a sequence of quantifiers that occurs on a path through ψ from the root to a leaf.

For a more formal definition, let's first introduce the following notation: If $P, Q \subseteq \{\exists, \forall\}^*$, then let $PQ = \{pq \mid p \in P, q \in Q\}$; in case Q is a singleton $\{q\}$ we will write Pq rather that $P\{q\}$. We write $p \leq q$ if p is a subword of q, i.e. if p can be obtained by deleting some symbols from q; let $P \leq Q$ that for all $p \in P$ there exists a $q \in Q$ with $p \leq q$. Finally for $p \in \{\exists, \forall\}^*$, let $\overline{p}$ be the dual word obtained from p by replacing all $\exists$ by $\forall$ and vice versa.

Definition 2 With every formula $\psi \in L^{\omega}_{\infty\omega}$ we associate its *quantifier structure* $P(\psi) \subseteq \{\exists, \forall\}^*$ as follows:

- if ψ is quantifier-free than $P(\psi) = \{\epsilon\}$ where ϵ is the empty word;
- if $\psi \equiv \neg(\varphi)$, then $P(\psi) = \{\overline{p} \mid p \in P(\varphi)\}$;
- if $\psi \equiv (\exists x_i)\varphi$, then $P(\psi) = \exists P(\varphi)$; similarly, if $\psi \equiv (\forall x_i)\varphi$, then $P(\psi) = \forall P(\varphi)$;
- if $\psi \equiv \bigvee \Phi$ or $\psi \equiv \bigwedge \Phi$, then

$$P(\psi) = \bigcup_{\varphi \in \Phi} P(\varphi).$$

Remark. If ψ is a first-order formula in prenex normal form whose prefix (in the usual, first-order sense) is $p \in \{\exists, \forall\}^*$, then obviously $P(\psi) = \{p\}$; however, if ψ is not in prenex normal form, then $P(\psi)$ can be a set with more than one element. For instance, if $\psi \equiv (\forall w)[(\exists x)\alpha \vee (\forall x)(\forall y)\beta]$ with α and β quantifier-free, then $P(\psi) = \{\forall\exists, \forall^3\}$. *Warning:* while the prefixes are motivated by the tree of subformulas of an $L^{\omega}_{\infty\omega}$ formula, we are not trying to embed trees of nodes labelled $\exists$, $\forall$ into other such trees; in particular, if we say that $P(\psi) \leq P$, we are not saying that the tree whose branches are listed in $P(\psi)$ is embeddable in a tree whose branches are listed in P: note that $\{\forall\exists, \forall^3\} \leq \{\exists\forall\exists, \forall\exists\forall^5\}$.

Definition 3 A set $P \subseteq \{\exists, \forall\}^*$ defines in $L^{k}_{\infty\omega}$ the *quantifier class*

$$L^{k}_{\infty\omega}(P) := \{\psi \in L^{k}_{\infty\omega} \mid P(\psi) \leq P\}.$$

Similarly, $L^{\omega}_{\infty\omega}(P) = \bigcup_{k \in \mathbb{N}} L^{k}_{\infty\omega}(P)$.

We are especially interested in quantifier classes that are defined by strings over the alphabet $\{\exists, \forall, \exists^*, \forall^*\}$.

Definition 4 Let S be the monoid generated by $\{\exists, \forall, \exists^*, \forall^*\}$, with the relations

$$\exists\exists^* = \exists^*\exists = \exists^*\exists^* = \exists^*$$
$$\forall\forall^* = \forall^*\forall = \forall^*\forall^* = \forall^*$$

A word $p \in S$ defines a *set* of words over $\{\exists, \forall\}$, when $\exists^*$ and $\forall^*$ are identified with th e sets $\{\exists^k \mid k \in \mathbb{N}\}$ and $\{\forall^k \mid k \in \mathbb{N}\}$, respectively. E.g., the word $\exists^*\forall\exists\forall^*$ denotes the set $\{\exists^k\forall\exists\forall^m \mid k, m \in \mathbb{N}\}$.

Well-known examples of such classes are Σ_k and Π_k which are defined as follows: Let $\Sigma_0 = \Pi_0 = \{\epsilon\}$, and, for $k > 0$, $\Sigma_k = \exists^*\Pi_{k-1}$ and $\Pi_k = \forall^*\Sigma_{k-1}$. Note that $\exists^*$ and $\forall^*$ *sometimes* allow us to collapse infinite P into a finite set of finite words.

Example. Connectivity is expressible in $L^3_{\infty\omega}(\forall\forall\exists^*)$.

The words in S are partially ordered: We write $p \leq q$ if the set defined by p is a subset of the set defined by q. Our main result will show that every quantifier class $L^\omega_{\infty\omega}(q)$ contains a formula ψ_q which is equivalent to a formula in $L^\omega_{\infty\omega}(p)$ if and only if $q \leq p$.

Often, we are interested in coarser formula classes. These are defined as unions $\bigcup_{p\in Q} L^\omega_{\infty\omega}(p)$ where $Q \subseteq S$.

The most interesting examples are $\bigcup_{k\in\mathbb{N}} L^\omega_{\infty\omega}(\Sigma_k)$, containing those formulae with *bounded quantifier alternations*, and the the class $L^\omega_{\infty\omega}(\mathrm{BU})$ of infinitary formulae with bounded universal depth, which is defined as follows:

Definition 5 Let $U_0 := \exists^*$, and $U_{n+1} := U_n\forall\exists^*$. The set of *infinitary formulae with bounded number of universal quantifiers* is

$$L^\omega_{\infty\omega}(\mathrm{BU}) = \bigcup_{n\in\mathbb{N}} L^\omega_{\infty\omega}(U_n).$$

Equivalently, we can say that $L^\omega_{\infty\omega}(\mathrm{BU})$ consists of those formulae in $L^\omega_{\infty\omega}$ whose quantifier structure is dominated by a word p over the alphabet $\{\exists^*, \forall\}$.

3 Transitive closure logic

The logic that originally motivated this research is transitive closure logic, introduced by Immerman [21]. An earlier variant is the database query language QBE, introduced by Zloof [38], in which one can ask transitive closure queries (unlike SQL which is based on first-order logic).

Definition 6 *Transitive closure logic*, denoted (FO + TC), is obtained by augmenting the syntax of first order logic by the following rule for building formulae:

Let $\varphi(\bar{x}, \bar{y})$ be a formula with (at least) the $2k$ free variables $\bar{x} = x_1, \ldots, x_k$ and $\bar{y} = y_1, \ldots, y_k$, and let $\bar{u}$ and $\bar{v}$ be two k-tuples of terms. Then

$$\psi \equiv [\mathrm{TC}_{\bar{x},\bar{y}}\ \varphi(\bar{x}, \bar{y})](\bar{u}, \bar{v})$$

is a formula, which says that the pair $(\bar{u}, \bar{v})$ is contained in the reflexive, transitive closure of the binary relation on k-tuples that is defined by φ.

Of course, it is understood that φ can contain other free variables than $\bar{x}$ and $\bar{y}$; these will be free also in ψ. Moreover, transitive closure logic is closed under the usual first order operations. We thus can build Boolean combinations of TC-formulae, we can nest TC-operators etc.

Examples. Let E be a binary relation symbol. Then the formula

$$(\forall u)(\forall v)[\mathrm{TC}_{x,y}\ Exy](u, v)$$

expresses that the digraph defined by E is strongly connected. The formula

$$\neg[\mathrm{TC}_{x,y}\ (\exists z)(Exz \wedge Ezy)](x,y)$$

says that there is no walk of even length from x to y.

Immerman was interested in this logic because of its significance for his long-term project to provide logical descriptions of complexity classes (see [16, 23] for surveys on this topic). He proved that on ordered structures, transitive closure logic can express precisely the queries that are computable with nondeterministic logarithmic space [21]. More precisely: Let $\mathcal{O}$ be the class of *finite successor structures*, i.e. structures whose vocabulary contains a binary predicate S, interpreted by a successor relation, and two constants 0 and e, interpreted by the first and the last element of S.

Theorem 1 (Immerman) *Let $L \subseteq \mathcal{O}$ be a class of σ-structures for some vocabulary σ. The following are equivalent*

(i) $L \in$ NLOGSPACE;

(ii) there is a formula $\psi \in$ (FO + TC) such that L is the set of finite models of ψ;

(iii) there is a quantifier-free formula $\varphi(\bar{x},\bar{y})$ such that L is the set of finite models of

$$[\mathrm{TC}_{\bar{x},\bar{y}}\ \varphi(\bar{x},\bar{y})](\bar{0},\bar{e}).$$

Theorem 1 strongly depends on the presence of the successor relation. Without successor, only the (trivial) implications $(i) \Longleftarrow (ii) \Longleftarrow (iii)$ survive whereas the reverse directions fail. It has been known for a long time that on unordered structures, the usual logics and query languages do not have the ability to count. For instance, the query EVEN (which asks whether the cardinality of a given structure is even) is clearly computable in LOGSPACE but cannot be expressed in first order logic, transitive closure logic or even $L^{\omega}_{\infty\omega}$.

To see that the implication $(ii) \Longrightarrow (iii)$ fails, observe that for any quantifier-free formula $\varphi(\bar{x},\bar{y})$, the query defined by $\psi(\bar{u},\bar{v}) \equiv [\mathrm{TC}_{\bar{x},\bar{y}}\ \varphi](\bar{u},\bar{v})$ is *monotone*. This means that for all k-tuples $\bar{u}$ and $\bar{v}$ of elements of a structure $\mathfrak{A}$ and for all extensions $\mathfrak{B}$ of $\mathfrak{A}$,

$$\mathfrak{A} \models \psi(\bar{u},\bar{v}) \quad \Longrightarrow \quad \mathfrak{B} \models \psi(\bar{u},\bar{v}).$$

Clearly this does not even hold for all first-order formulae.

It should be noted that Immerman's original result was weaker than the present statement of Theorem 1; it said that on finite ordered structures, NLOGSPACE is captured by the logic (FO + pos TC), the restriction of (FO + TC) where the operator TC can occur only positively. However, the closure of NLOGSPACE under complementation [22, 34] implies the equivalence of (FO + pos TC) with (FO + TC). See [23] for a direct proof of this fact.

In this paper we show that this is not the case on arbitrary finite structures.

Quantifier classes in transitive closure logic. Note that the TC-operator in a formula $\psi \equiv [\mathrm{TC}_{\bar{x},\bar{y}}\ \varphi(\bar{x},\bar{y})](\bar{u},\bar{v})$ where $\bar{x}$ and $\bar{y}$ are k-tuples, can be considered as a sort of generalized quantifier that binds $2k$ variables. So we define, that $\mathrm{depth}(\psi) := 2k + \mathrm{depth}(\varphi)$.

We define quantifier classes in (FO + TC) in a very similar way as in $L^{\omega}_{\infty\omega}$. The symbol T stands for a TC-operator, and N stands for a negated TC-operator.

Definition 7 For every word p over the alphabet $\{\exists,\forall,T,N\}$, we define a *quantifier class* $\mathrm{TC}(p)$ in (FO + TC) inductively as follows:

- $\mathrm{TC}(\epsilon)$ contains the quantifier-free formulae.
- For a quantifier $Q \in \{\exists, \forall\}$, the class $\mathrm{TC}(Qp)$ is the closure under conjunctions and disjunctions of the class $\mathrm{TC}(p) \cup \{(Qx_i)\varphi \mid \varphi \in \mathrm{TC}(p)\}$.
- $\mathrm{TC}(Tp)$ is the closure under conjunctions and disjunctions of the class of all formulae of the form $[\mathrm{TC}_{\bar{x},\bar{y}}\ \varphi(\bar{x},\bar{y})](\bar{u},\bar{v})$ where $\varphi \in \mathrm{TC}(p)$;
- $\mathrm{TC}(Np)$ is the closure under conjunction and disjunction of the class of all formulae of the form $\neg[\mathrm{TC}_{\bar{x},\bar{y}}\ \neg\varphi(\bar{x},\bar{y})](\bar{u},\bar{v})$ where $\varphi \in \mathrm{TC}(p)$.

Obviously, $(\mathrm{FO} + \mathrm{TC}) = \bigcup_p \mathrm{TC}(p)$, and $(\mathrm{FO} + \mathrm{pos\ TC})$ is the union over those classes $\mathrm{TC}(p)$ where p does not contain the symbol N.

If two distinct constants are available, then quantifier classes can often be simplified: Let R be the monoid that is generated by $\{\exists, \forall, T, N\}$ with the relations

$$\exists T = T\exists = TT = T$$
$$\forall N = N\forall = NN = N.$$

A word $p \in \{\exists, \forall, T, N\}^*$ that cannot be simplified anymore via these relations is called *reduced.*

Proposition 2 *Let σ be a signature that contains at least two constants a, b. Then, on the class of all σ-structures that satisfy $a \neq b$, every formula in* $(\mathrm{FO} + \mathrm{TC})$ *of signature σ is equivalent to a formula in some* $\mathrm{TC}(p)$ *where $p \in R \cap \{T, N\}^*$; moreover, every σ-formula in* $(\mathrm{FO} + \mathrm{pos\ TC})$ *is equivalent to a formula in* $\mathrm{TC}(p)$ *for some $p \in R \cap \{\forall, T\}^*$.*

Proof. Existential quantifiers can always be replaced by TC-operators: A formula $(\exists z)\psi(z)$ is equivalent to

$$[\mathrm{TC}_{x,y}\ \psi(x) \vee \psi(y)](a, b).$$

By dual arguments, universal quantifiers are replaced by N.

The proposition is now implied by the following facts, that were proved by Immerman: Formulae of the form

- $[\mathrm{TC}_{\bar{x},\bar{y}}\ \psi(\bar{x},\bar{y})](\bar{a},\bar{b}) \vee [\mathrm{TC}_{\bar{u},\bar{v}}\ \varphi(\bar{u},\bar{v})](\bar{a},\bar{b})$
- $[\mathrm{TC}_{\bar{x},\bar{y}}\ \psi(\bar{x},\bar{y})](\bar{a},\bar{b}) \wedge [\mathrm{TC}_{\bar{u},\bar{v}}\ \varphi(\bar{u},\bar{v})](\bar{a},\bar{b})$
- $[\mathrm{TC}_{\bar{x}\bar{y}}\ [\mathrm{TC}_{\bar{u},\bar{v}}\ \psi(\bar{x},\bar{y},\bar{u},\bar{v})](\bar{a},\bar{b})](\bar{a},\bar{b})$

are equivalent to formulae of the form

$$[\mathrm{TC}_{\bar{x}\bar{u},\bar{y}\bar{v}}\ \alpha(\bar{x},\bar{u},\bar{y},\bar{v})](\bar{a}\bar{a},\bar{b}\bar{b})$$

where α is formed by conjunctions and disjunctions of instances of ψ and φ and of equalities and inequalities. ∎

Remark. It should be noted that Immerman proved his results on successor structures. However, the proofs of the facts mentioned above go through for arbitrary structures. In addition Immerman showed, that on successor structures, the formulae $(\forall\bar{u})[\mathrm{TC}_{\bar{x},\bar{y}}\ \psi](\bar{a},\bar{b})$ and $[\mathrm{TC}_{\bar{x},\bar{y}}\ (\forall\bar{u})\psi](\bar{a},\bar{b})$ can be brought to the form

$$[\mathrm{TC}_{\bar{x}\bar{u},\bar{y}\bar{v}}\ \alpha(\bar{x},\bar{u},\bar{y},\bar{v})](\bar{a}\bar{a},\bar{b}\bar{b})$$

where α may contain the successor predicate. Our results will show that this is not true on arbitrary finite structures.

Transitive closure logic and infinitary logic. The quantifier classes $\mathrm{TC}(p)$ in transitive closure logic can be embedded in the quantifier classes in $L^{\omega}_{\infty\omega}$:

Proposition 3 *Let* $\mathrm{TC}(p)$ *be defined by a word* $p \in \{\exists, \forall, T, N\}^*$*, and let* $\tilde{p}$ *be the word obtained from* p *by replacing* T *by* $\exists^*$ *and* N *by* $\forall^*$*. Then*

$$\mathrm{TC}(p) \subseteq L^{\omega}_{\infty\omega}(\tilde{p}).$$

In particular, $(\mathrm{FO} + \text{pos TC}) \subseteq L^{\omega}_{\infty\omega}(\mathrm{BU})$.

Proof. By induction: Let $\alpha(x,y)$ be a formula in $\mathrm{TC}(p) \subseteq L^{\omega}_{\infty\omega}(\tilde{p})$. The formula

$$[\mathrm{TC}_{x,y}\ \alpha(x,y)](u,v)$$

in $\mathrm{TC}(Tp)$ is expressed in infinitary logic by the formula $\psi \equiv \bigvee_{n\in\mathbb{N}} \varphi_n(x,y)$ with

$$\begin{aligned} \varphi_0(x,y) &\equiv (x = y) \\ \varphi_{n+1}(x,y) &\equiv (\exists z)(\alpha(x,z) \wedge \varphi_n(z,y)) \end{aligned}$$

(where $\varphi_n(z,y)$ is understood as the formula obtained from $\varphi_n(x,y)$ by replacing all occurrences of x by z and vice versa). It follows that $\varphi_n \in L^{\omega}_{\infty\omega}(\exists^n \tilde{p})$ and thus $\psi \in L^{\omega}_{\infty\omega}(\exists^* \tilde{p})$.

Similarly, a formula

$$\neg[\mathrm{TC}_{x,y}\ \neg\alpha(x,y)](u,v)$$

from $\mathrm{TC}(Np)$ is equivalent to $\psi' \equiv \bigwedge_{n\in\mathbb{N}} \varphi'_n(x,y)$ with

$$\begin{aligned} \varphi'_0(x,y) &\equiv (x \neq y) \\ \varphi'_{n+1}(x,y) &\equiv (\forall z)(\alpha(x,z) \vee \varphi'_n(z,y)). \end{aligned}$$

Thus $\varphi'_n \in L^{\omega}_{\infty\omega}(\forall^n \tilde{p})$ and $\psi' \in L^{\omega}_{\infty\omega}(\forall^* \tilde{p})$. ■

The words $p \in R$ can be partially ordered as follows. We say that $p \leq q$ if p can be embedded into q, possibly by replacing some occurrences of subwords $\exists^k$ or $\forall^k$ by T or N respectively. Note that this produces an infinite, nonlinear partial order with no infinite antichains.

It is obvious that $q \leq p$ implies $\mathrm{TC}(q) \subseteq \mathrm{TC}(p)$. Our aim is to show that the converse is also true: There exist formulae in $\mathrm{TC}(q)$ which are not equivalent (on finite structures) to any formula in $\bigcup_{q \not\leq p} \mathrm{TC}(p)$: our main result will show that

$$\mathrm{TC}(q) \not\subseteq \bigcup_{q \not\leq p} L^{\omega}_{\infty\omega}(\tilde{p}).$$

4 The Ehrenfeucht-Fraïssé game for $L^k_{\infty\omega}(P)$

Ehrenfeucht-Fraïssé-games [13, 11] provide a powerful tool for proving inexpressibility results for various logics. In their classical form they give a criterion for the indistinguishability of two (classes of) structures by means of first order formulae. They were modified by Barwise [3] and Immerman [19] to deal with other logics, especially infinitary logic and fixpoint logic. Using these games, Kolaitis and Vardi [27] have established, e.g., the 0-1 law for infinitary logic. Also, variants of Ehrenfeucht-Fraïssé games have been designed and used for monadic second order logic [12], first-order logic with counting [6], Datalog [29, 28], fragments of fixpoint logic [31] and transitive closure logic [14]. We will be using a variant of the game in [31].

Definition 8 Suppose we have two structures $\mathfrak{A}$ and $\mathfrak{B}$ of the same vocabulary σ. Let $c_1, \ldots, c_s$ and $d_1, \ldots, d_s$ be the interpretations of the constants of σ in $\mathfrak{A}$ and $\mathfrak{B}$, respectively. The *(infinitary) k-pebble game* on the pair $(\mathfrak{A}, \mathfrak{B})$ is played by two players, called the *Spoiler* and the *Duplicator*[2], according to the following rules: There are k pairs $(u_1, v_1), \ldots, (u_k, v_k)$ of pebbles. Each round of the game consists of either an $\exists$-move or an $\forall$-move:

The $\exists$-move. The Spoiler picks a pebble u_i and places it on an element of $\mathfrak{A}$. The Duplicator answers by putting the corresponding pebble v_i on $\mathfrak{B}$.

The $\forall$-move. Similarly but with 'reversed board': the Spoiler places v_i on $\mathfrak{B}$; the Duplicator responds with u_i on $\mathfrak{A}$.

After each move, the Spoiler wins if the pebbles do not determine a *partial isomorphism* from $\mathfrak{A}$ to $\mathfrak{B}$. More precisely: Let $a_1, \ldots, a_k$ and $b_1, \ldots, b_k$ be the elements carrying the pebbles $u_1, \ldots, u_k$ and $v_1, \ldots, v_k$, respectively. If the mapping f with

$$f(a_i) = b_i \ \text{ for } i = 1, \ldots, k \ \text{ and } f(c_i) = d_i \ \text{ for } i = 1, \ldots, s$$

does not determine an isomorphism between the substructures of $\mathfrak{A}$ and $\mathfrak{B}$ that are generated by the pebbled elements and the constants, then the Spoiler wins; otherwise the game goes on. The Duplicator wins if the Spoiler never does.

In the classical game, the Spoiler is free to choose between $\exists$-moves and $\forall$-moves.The crucial result, relating expressibility and pebble games is:

Theorem 4 *Let $\mathfrak{A}$ and $\mathfrak{B}$ be two structures over the same vocabulary and let $k \in \mathbb{N}$. Then the following two statements are equivalent:*

(i) For all $\varphi \in L^k_{\infty\omega}$

$$\mathfrak{A} \models \varphi \implies \mathfrak{B} \models \varphi.$$

(ii) The Duplicator has a winning strategy for the k-pebble game on $(\mathfrak{A}, \mathfrak{B})$.

This result is implicit in a paper by Barwise [3], although pebble games are not explicitly mentioned there. In more explicit form it is proved by Immerman in [19], but only the first-order fragment of $L^k_{\infty\omega}$ is mentioned. The most accessible proof of Theorem 4 is probably the one given by Kolaitis and Vardi in [28].

[2] As usual we denote the Spoiler by male and the Duplicator by female pronouns.

By imposing restrictions on the admissible sequences of $\exists$- and $\forall$-moves one defines games that characterize the expressive power of quantifier classes in $L^k_{\infty\omega}$: Let $P \subseteq \{\exists, \forall\}^*$ be closed under subwords. The $L^k_{\infty\omega}(P)$-game on $(\mathfrak{A}, \mathfrak{B})$ is the k-pebble game on $(\mathfrak{A}, \mathfrak{B})$ where the Spoiler must play in such a way that the sequence of his moves defines a word in P.

Remark. For sets that are defined by a word $p \in \{\exists, \forall, \exists^*, \forall^*\}$ it is convenient to consider also $\exists^*$ and $\forall^*$-moves:

The $\exists^*$-move. The Spoiler has the option to make an arbitrary number of $\exists$-moves, each answered at once by the Duplicator. This continues until until the Spoiler wants to stop.

The $\forall^*$-move. This is similar, except that the Spoiler makes $\forall$-moves until he gets bored.

Thus, to play the p-game, where $p = Qq$, the players make the Q-move, and then, from there, play a q-game, until they reach ϵ, when they stop (notice that if, during an $\forall^*$-move or $\exists^*$-move, partial isomorphism is broken, then the Spoiler can stop and win at once).

We will show that our modified games characterize the classes $L^k_{\infty\omega}(P)$. For this purpose, we will give a more formal definition for the concept of a winning strategy for the Duplicator:

Definition 9 Let $\mathfrak{A}$ and $\mathfrak{B}$ be two structures over the vocabulary σ and let $c_1, \ldots, c_s$ and $d_1, \ldots, d_s$ be the interpretations of the constant symbols of σ in $\mathfrak{A}$ and $\mathfrak{B}$ respectively. Let $P \subseteq \{\exists, \forall\}^*$ be closed under subwords. We say that *the Duplicator has a winning strategy for $L^k_{\infty\omega}(P)$ on $(\mathfrak{A}, \mathfrak{B})$* if there exists a family $(I_p)_{p\in P}$ of nonempty sets of partial isomorphisms from $\mathfrak{A}$ to $\mathfrak{B}$ such that:

- *Each I_p is closed under subfunctions:* if $f \in I_p$ and $\{(c_1, d_1), \ldots, (c_s, d_s)\} \subseteq g \subseteq f$, then $g \in I_p$.
- *"Forth" property up to k in P:* If $q = \exists p \in P$, $f \in I_p$ with $|f| < s + k$, then for every element $a \in \mathfrak{A}$ there exists an element $b \in \mathfrak{B}$ such that $f \cup \{(a, b)\} \in I_q$.
- *"Back" property up to k in P:* If $q = \forall p \in P$, $f \in I_p$ with $|f| < s + k$, then for every element $b \in \mathfrak{B}$ there exists an element $a \in \mathfrak{A}$ such that $f \cup \{(a, b)\} \in I_q$.

Theorem 5 *Let $\mathfrak{A}$ and $\mathfrak{B}$ be two structures over the same vocabulary, let $P \subseteq \{\exists, \forall\}^*$ and let $k \in \mathbb{N}$. Then the following two statements are equivalent:*

(i) For all $\varphi \in L^k_{\infty\omega}(P)$

$$\mathfrak{A} \models \varphi \implies \mathfrak{B} \models \varphi.$$

(ii) The Duplicator has a winning strategy for the $L^k_{\infty\omega}(P)$-game on $(\mathfrak{A}, \mathfrak{B})$.

Proof. The proof is a variation of the proof of Theorem 4 in [28]. Given a word $p \in P$, let $Q_p = \{q \in \{\exists, \forall\}^* \mid qp \in P\}$. In particular $Q_\epsilon = P$. Note that even for $p \neq \epsilon$, Q_p need not be a proper subset of P: If $P = \{\exists^k \mid k \in \mathbb{N}\}$ then $Q_p = P$ for every $p \in P$.

First suppose that for all $\psi \in L^k_{\infty\omega}(P)$

$$\mathfrak{A} \models \psi \implies \mathfrak{B} \models \psi.$$

The desired family $(I_p)_{p\in P}$ of sets of partial isomorphisms is defined as follows: I_p consists of the functions

$$\{(c_1,d_1),\dots,(c_s,d_s)\}\cup\{(a_1,b_1),\dots,(a_m,b_m)\}$$

such that $m\le k$ and such that for all formulae $\varphi(x_1,\dots,x_m)\in L^k_{\infty\omega}(Q_p)$

$$\mathfrak{A}\models\varphi(a_1,\dots,a_m)\Longrightarrow\mathfrak{B}\models\varphi(b_1,\dots,b_m).$$

Clearly, each I_p is closed under subfunctions and non-empty. It remains to show that the back and forth properties are satisfied. Assume that one of these, say the forth property is violated: There is a $q=\exists p\in P$, a partial isomorphism $f=\{(c_1,d_1),\dots,(c_s,d_s)\}\cup\{(a_1,b_1),\dots,(a_m,b_m)\}$ in I_p with $m<k$ and an element $a\in\mathfrak{A}$ such that $f\cup\{(a,b)\}$ does not belong to I_q for any $b\in\mathfrak{B}$. It follows that for every $b\in\mathfrak{B}$, there exists a formula $\varphi_b(x_1,\dots,x_m,x_{m+1})\in L^k_{\infty\omega}(Q_q)$ such that

$$\mathfrak{A}\models\varphi_b(a_1,\dots,a_m,a)\ \text{ but }\ \mathfrak{B}\models\neg\varphi_b(b_1,\dots,b_m,b).$$

But this implies a contradiction, because the formula

$$\psi(x_1,\dots,x_m)\equiv(\exists x_{m+1})\Big(\bigwedge_{i=1}^{m}(x_i\neq x_{m+1})\wedge\bigwedge_{b\in B}\varphi_b(x_1,\dots,x_m,x_{m+1})\Big)$$

belongs to $L^k_{\infty\omega}(\exists Q_q)=L^k_{\infty\omega}(Q_p)$ and because

$$\mathfrak{A}\models\psi(a_1,\dots,a_m)\ \text{ but }\ \mathfrak{B}\models\neg\psi(b_1,\dots,b_m).$$

The argument for the back property is similar, with an infinitary disjunction over elements of $\mathfrak{A}$.

For the converse, assume that the Duplicator has a winning strategy for the $L^k_{\infty\omega}(P)$-game on $(\mathfrak{A},\mathfrak{B})$. We claim that for every $p\in P$, for every partial isomorphism $f\in I_p$ and for any choice of $m\le k$ (not necessarily distinct) elements $a_1,\dots,a_m$ in the domain of f the following holds: If $\psi(x_1,\dots,x_m)$ is a formula in $L^k_{\infty\omega}(Q_p)$ then

$$\mathfrak{A}\models\psi(a_1,\dots,a_m)\ \Longrightarrow\ \mathfrak{B}\models\psi(f(a_1),\dots,f(a_m)).$$

For $p=\epsilon$ and $m=0$, this implies the desired result.

We prove the claim by induction on the formulae $\psi(x_1,\dots,x_m)$. It is obvious for quantifier-free formulae and for infinitary conjunctions and disjunctions over formulae for which it the claim is already established. So suppose that $\psi(x_1,\dots,x_m)\equiv(\exists x)\varphi(x_1,\dots,x_m,x)$ is a formula in $L^k_{\infty\omega}(Q_p)$ and that the claim is true for φ. Note that $\varphi\in L^k_{\infty\omega}(Q_q)$ for $q=\exists p$.

First, let us assume that x is different from $x_1,\dots,x_m$; it follows that $m<k$, thus the forth property applies: Given any element $a\in\mathfrak{A}$, such that $\mathfrak{A}\models\varphi(a_1,\dots,a_m,a)$ there exists a $b\in\mathfrak{B}$ such that $f\cup\{(a,b)\}\in I_q$. By induction hypothesis $\mathfrak{B}\models\varphi(f(a_1),\dots,f(a_m),b)$. It follows that

$$\mathfrak{A}\models\psi(a_1,\dots,a_m)\ \Longrightarrow\ \mathfrak{B}\models\psi(f(a_1),\dots,f(a_m)).$$

If x is one of the variables x_i, then x_i no longer occurs free in ψ. The subfunction g of f with domain $\{c_1,\dots,c_s,a_1,\dots,a_{i-1},a_{i+1},\dots,a_m\}$ still belongs to I_p because I_p is closed under subfunctions. Now $|g|<s+k$, so we can again apply the forth property: for every $a\in\mathfrak{A}$ there exists a $b\in\mathfrak{B}$ such that $g\cup\{(a,b)\}\in I_q$. By induction hypothesis

$$\mathfrak{A} \models \varphi(a_1, \ldots, a_{i-1}, a, a_{i+1}, \ldots, a_m) \implies \mathfrak{B} \models \varphi(g(a_1), \ldots, g(a_{i-1}), b, g(a_{i+1}), \ldots, g(a_m)).$$

It follows that also in this case

$$\mathfrak{A} \models \psi(a_1, \ldots, a_m) \implies \mathfrak{B} \models \psi(f(a_1), \ldots, f(a_m)).$$

The arguments for formulae whose leading quantifier is universal are similar, using the back property instead of the forth property. ∎

This theorem will be our major tool for proving the main theorem.

5 The Hierarchy Theorem

In this section, we will present our main results.

Definition 10 For each $i \geq 0$, let Σ_i^{TC} denote the set of formulae in (FO + TC) where $i-1$ alternations of TC operators and negations are allowed, starting with TC. More formally, let $\Sigma_0^{\mathrm{TC}} = \Pi_0^{\mathrm{TC}} = \mathrm{FO}$, let Π_i^{TC} be the set of negations of Σ_i^{TC}-formulae and let $\Sigma_{i+1}^{\mathrm{TC}}$ be the closure of Π_i^{TC} under conjunction, disjunction, existential and universal quantification, and under (positive applications of) the TC-operator.

Immerman conjectured that for each i, $\Sigma_i^{\mathrm{TC}} \subsetneq \Sigma_{i+1}^{\mathrm{TC}}$. We prove a much more general hierarchy theorem for infinitary logic that implies this conjecture:

Take any quantifier class $\mathrm{TC}(q) \subseteq (\mathrm{FO} + \mathrm{TC})$ where $q \in R$, i.e., q is a reduced word over $\{\exists, \forall, T, N\}$. Further, recall that S is the set of reduced words over $\{\exists, \forall, \exists^*, \forall^*\}$. As observed in section 2, $\mathrm{TC}(q)$ can be embedded into the quantifier class $L^\omega_{\infty\omega}(\tilde{q})$ where $\tilde{q}$ is obtained by replacing T by $\exists^*$ and N by $\forall^*$.

Theorem 6 (Hierarchy Theorem for (FO + TC) and $L^\omega_{\infty\omega}$)

1. *Non-reachability is not in* $L^\omega_{\infty\omega}(\mathrm{BU})$*, hence not in* (FO + pos TC).
2. *Let* $q \in R$*. Then the quantifier class* $\mathrm{TC}(q)$ *in* (FO + TC) *contains a formula* ψ_q *which is equivalent to a formula in* $L^\omega_{\infty\omega}(p)$ *if and only if* $\tilde{q} \leq p$*. Thus*

$$\mathrm{TC}(q) \not\subseteq \bigcup_{\tilde{q} \not\leq p} L^\omega_{\infty\omega}(p).$$

Proof. Let $q \in \{\exists, \forall, T, N\}^*$ and $p \in \{\exists, \forall, \exists^*, \forall^*\}^*$ be reduced. To simplify notation we will write $q \leq p$ rather than $\tilde{q} \leq p$ from now on.

We will construct:

- for every $q \in R$, a formula $\psi_q(x, y) \in \mathrm{TC}(q)$;
- for every $k \in \mathbb{N}$ and $q \in R$, a pair of structures $\mathfrak{C}^k(q)$ and $\mathfrak{D}^k(q)$

such that the following conditions are satisfied:

(i) $\mathfrak{C}^k(q) \models \psi_q(a, b)$ and $\mathfrak{D}^k(q) \models \neg\psi_q(a, b)$;

(ii) If $q \not\leq p$ and $k > \mathrm{length}(p)$, then the Duplicator wins the $L^k_{\infty\omega}(p)$-game on $\mathfrak{C}^k(q)$ and $\mathfrak{D}^k(q)$.

The Theorem then follows immediately from Theorem 5.

The formulae ψ_q and the structures $\mathfrak{C}^k(q)$ and $\mathfrak{D}^k(q)$ will have the vocabulary $\sigma_m = \{E, F_0, \ldots, F_{m-1}, a, b\}$ where $m = |q|$, where E and $F_0, \ldots, F_{m-1}$ are binary predicates and a and b are constants[3]. We will proceed by induction on the length of q. A word q is called *existential* if it begins with $\exists$ or T, and *universal* if it begins with $\forall$ or N. For $q = \epsilon$, nothing has to be proved.

The structures $\mathfrak{C}^k(q)$ and $\mathfrak{D}^k(q)$ are defined inductively. To start the process we define the structures $\mathfrak{C}^k(\epsilon)$ consisting of the nodes a and b with an E-arc from a to b, and $\mathfrak{D}^k(\epsilon)$, consisting of two isolated nodes a and b. To build the structures $\mathfrak{C}^k(q)$ and $\mathfrak{D}^k(q)$, we will use coloured digraphs I^k and J^k: Let I^k consist of source a, sink b and intermediate vertices $u_1, \ldots, u_k, v_1, \ldots, v_k$ with arcs coloured amber (A) from a to each u_i and from each v_i to b, and with arcs coloured blue (B) from a to each v_i and from each u_i to b. Let J^k be I^k with an additional vertex w and with amber arcs from a to w and from w to b.

Given $\mathfrak{A}$, a digraph whose arcs are either amber (A) or blue (B), and given $\mathfrak{C}$, $\mathfrak{D}$, each with binary relations E, F_0, ..., $F_{\ell-1}$, and each with source a and sink b, let

$$\mathfrak{A} \otimes_{A,B} (\mathfrak{C}, \mathfrak{D})$$

be the result of going through $\mathfrak{A}$ and replacing each amber arc with a copy of $\mathfrak{C}$ and each blue arc with a copy of $\mathfrak{D}$ as follows. Start with $F_\ell := \emptyset$. Given an amber arc (x, y), replace it with a copy of $\mathfrak{C}$, identifying x with $a^{\mathfrak{C}}$ and y with $b^{\mathfrak{C}}$, and setting $F_\ell := F_\ell \cup \{(x, y)\}$. Replace blue arcs with copies of $\mathfrak{D}$ in the same way.

Let $\delta(\mathfrak{A})$ be the result of taking $\mathfrak{A}$ above and eliminating the relations $F_0, \ldots, F_\ell$. We now construct our structures, taking q to be reduced: If $\bar{q}$ is the dual of q, then $\mathfrak{C}^k(\bar{q}) = \mathfrak{D}^k(q)$, so we define the structures for existential q and the rest follows. Given q, let

$$\begin{aligned} \mathfrak{C}^k(\exists q) &= J^k \otimes_{A,B} (\mathfrak{C}^k(q), \mathfrak{D}^k(q)) \\ \mathfrak{D}^k(\exists q) &= I^k \otimes_{A,B} (\mathfrak{C}^k(q), \mathfrak{D}^k(q)). \end{aligned}$$

Now, let $I_1^k = I^k$ and $J_1^k = J^k$ and

$$\begin{aligned} I_{j+1}^k &= I^k \otimes_{A,B} (J_j^k, I_j^k) \\ J_{j+1}^k &= J^k \otimes_{A,B} (J_j^k, I_j^k), \end{aligned}$$

so that J_j^k admits an amber path from a to b (while I_j^k does not), and let

$$\begin{aligned} \mathfrak{C}^k(Tq) &= \delta(J_k^k) \otimes_{A,B} (\mathfrak{C}^k(q), \mathfrak{D}^k(q)) \\ \mathfrak{D}^k(Tq) &= \delta(I_k^k) \otimes_{A,B} (\mathfrak{C}^k(q), \mathfrak{D}^k(q)). \end{aligned}$$

See figures 1 and 2 for $\mathfrak{C}^2(\forall\exists) = \mathfrak{D}^2(\exists\forall)$ and $\mathfrak{D}^2(\forall\exists) = \mathfrak{C}^2(\exists\forall)$ respectively, figures 3 and 4 for I_3 and J_3 respectively, and figures 5 and 6 for I_2^2 and J_2^2 respectively.

`Note for printer:  Include Figures 1 to 6 here.`

The predicates F_i will be needed in the following construction to identify source-sink pairs of substructures of level i. We start with $\psi_\epsilon(x, y) \equiv Exy$; once again, if $\bar{q}$ is the dual of $q \neq \epsilon$, then

[3]We will show below that the Hierarchy Theorem holds also with a uniform signature that does not depend on m

$\psi_q \equiv \neg\psi_{\bar{q}}$. Suppose that $|q| = m$ and that we have already constructed the formula ψ_q. Then, the desired formulas for existential Qq are

$$\begin{aligned}
\psi_{\exists q}(x,y) &\equiv (\exists u)(F_m xu \wedge F_m uy \wedge \psi_q(x,u) \wedge \psi_q(u,y)),\\
\psi_{Tq}(x,y) &\equiv [\mathrm{TC}_{u,v}(F_m uv \wedge \psi_q(u,v))](x,y),
\end{aligned}$$

which are in $\mathrm{TC}(\exists q)$ and $\mathrm{TC}(Tq)$ respectively.

It follows immediately from the construction that $\mathfrak{C}^k(q) \models \psi_q(a,b)$ and $\mathfrak{D}^k(q) \models \neg\psi_q(a,b)$. So we have to show that if $q \not\leq p$, and k is large enough (i.e., $k > \mathrm{length}(p)$), then the Duplicator wins the $L^k_{\infty\omega}(p)$-game on $\mathfrak{C}^k(q)$ and $\mathfrak{D}^k(q)$. This is proven by induction on the length of q.

Note that this is vacuously true for $q = \epsilon$. Suppose that $q = Qq'$, and the theorem is true for q', i.e., if $q' \not\leq p'$ and $k > \mathrm{length}(p')$, then the Duplicator wins the $L^k_{\infty\omega}(p')$-game on $\mathfrak{C}^k(q')$ and $\mathfrak{D}^k(q')$. We want to prove that if $q \not\leq p$ and $k > \mathrm{length}(p)$, then the Duplicator also wins the $L^k_{\infty\omega}(p)$-game on $\mathfrak{C}^k(q)$ and $\mathfrak{D}^k(q)$. We need a lemma:

Lemma 7 *Choose $q' \neq \epsilon$, and suppose $q \in \{\exists, \forall, T, N\}^*$ and $p \in \{\exists, \forall, \exists^*, \forall^*\}^*$ are reduced.*

1. *Suppose that $q = \exists q' \not\leq p$ and that p' is maximal with the property that $p = p''p'$ and $q' \not\leq p'$. Then $p \leq \forall^*\exists p'$.*
2. *Suppose that $q = \forall q' \not\leq p$ and that p' is maximal with the property that $p = p''p'$ and $q' \not\leq p'$. Then $p \leq \exists^*\forall p'$.*
3. *Suppose that $q = Tq' \not\leq p$ and that p' is maximal with the property that $p = p''p'$ and $q' \not\leq p'$. Then for some $t <$ length(p), $p \leq (\forall^*\exists)^t p'$.*
4. *Suppose that $q = Nq' \not\leq p$ and that p' is maximal with the property that $p = p''p'$ and $q' \not\leq p'$. Then for some $t <$ length(p), $p \leq (\exists^*\forall)^t p'$.*

PROOF. If $p' = p$, then the claim is trivially true: suppose that $p' \neq p$.

Proof of (1): As $p \neq p'$, p'' is not the empty word; so let Q be the last symbol of p'' and write $p = p'''Qp'$. By the maximality of p', it follows that $q' \leq Qp'$ and thus $p''' \leq \forall^*$, because otherwise we would have $q \leq p$. So we have shown that $p \leq \forall^*Qp'$. It remains to prove that $Q \neq \exists^*$. But this is clear: if $Q = \exists^*$, then $q' \leq \exists^*p'$ and therefore $q = \exists q' \leq \exists\exists^*p' = \exists^*p' \leq p$ which contradicts the assumption that $q \not\leq p$.

Proof of (2): Similar to the proof of (1).

Proof of (3): Again, $p'' = p'''Q$ for some Q and p'''. As $q' \leq Qp'$, q' begins with a Q — and $Q \leq \forall^*$ as q is reduced. Thus as $T \not\leq p'''$, $p''' \leq (\forall^*\exists)^t$ for some finite t.

Proof of (4): Similar to the proof of (3). ■

And now for the induction. Assume that the Duplicator wins the k-pebble q'-game on $\mathfrak{C}^k(q')$ and $\mathfrak{D}^k(q')$ for any p' such that $q' \not\leq p'$ and $k > \mathrm{length}(p')$.

Case 1: $q = \exists q'$. If $q \not\leq p$, we can assume without loss of generality that $p = \forall^*\exists p'$ for some p' such that $q' \not\leq p'$. This means that the Spoiler begins the game with an arbitrary number of $\forall$-moves, i.e., he places and moves around as long as he likes at most k pebbles on $\mathfrak{D}^k(q)$. Note that J^k is an extension of I^k, so there exists an isomorphic embedding $g\colon \mathfrak{D}^k(q) \hookrightarrow \mathfrak{C}^k(q)$. Thus, the inital $\forall$-moves can be answered by 'isomorphic' moves: if the Spoiler pebbles $d \in \mathfrak{D}^k(q)$, the Duplicator simply responds by pebbling $g(d)$. Then the Spoiler makes a single $\exists$-move, i.e. he moves one pebble on $\mathfrak{C}^k(q)$. If he chooses c in the range of g then the Duplicator pebbles $g^{-1}(c)$.

The only interesting case is when the Spoiler plays outside the range of g, that is in the "line" that connects a over w to b through two copies of $\mathfrak{C}^k(q')$. By symmetry we can assume that he pebbles an element c inside the $\mathfrak{C}^k(q')$ copy that leads from a to w. Since there are only k pebbles, at least one "line" in $\mathfrak{D}^k(q)$ from a over some u_i to b carries no pebble. Recall that a is connected to u_i by a copy of $\mathfrak{C}^k(q')$, while the way from u_i to b goes through a copy of $\mathfrak{D}^k(q')$. The Duplicator moves her free pebble on an element c' (corresponding to c) in the $\mathfrak{C}^k(q')$-copy from a to u_i. Now the Duplicator wins the remaining $L^k_{\infty\omega}(p')$-game. She just has to maintain the partial isomorphism of the "line" through w in $\mathfrak{C}^k(q)$ with the line through u_i in $\mathfrak{D}^k(q)$, using her winning strategy for the $L^k_{\infty\omega}(p')$-game on $\mathfrak{C}^k(q')$ and $\mathfrak{D}^k(q')$.

Case 2: $q = \forall q'$. The argument is similar to that of Case 1.

Case 3: $q = Tq'$. If $q \not\leq p$, then for some finite t, $p \leq (\forall^*\exists)^t p'$, where $q' \not\leq p'$. We claim that the Duplicator wins the k-pebble $(\forall^*\exists)^t p'$-game on $\mathfrak{C}^k(q)$ and $\mathfrak{D}^k(q)$. The argument is similar to that of Case 1: we start with an embedding $g\colon \mathfrak{D}^k(q) \hookrightarrow \mathfrak{C}^k(q)$, and we repeat the argument t times.

Recall that $\mathfrak{C}^k(q) = J^k_k \otimes_{A,B} (\mathfrak{C}^k(q'), \mathfrak{D}^k(q'))$ and $\mathfrak{D}^k(q) = I^k_k \otimes_{A,B} (\mathfrak{C}^k(q'), \mathfrak{D}^k(q'))$ and $t < \text{length}(p) < k$. First, by the same argument as that of Case 1, the Duplicator wins the k-pebble $\forall^*\exists p'$-game on $J^k_{k-t+1} \otimes_{A,B} (\mathfrak{C}^k(q'), \mathfrak{D}^k(q'))$ and $I^k_{k-t+1} \otimes_{A,B} (\mathfrak{C}^k(q'), \mathfrak{D}^k(q'))$. So it suffices to prove that if $t' < t$ and the Duplicator wins the k-pebble $(\forall^*\exists)^{t'} p'$-game on $J^k_{k-t+t'} \otimes_{A,B} (\mathfrak{C}^k(q'), \mathfrak{D}^k(q'))$ and $I^k_{k-t+t'} \otimes_{A,B} (\mathfrak{C}^k(q'), \mathfrak{D}^k(q'))$, then she also wins the k-pebble $(\forall^*\exists)^{t'+1} p'$-game on $J^k_{k-t+t'+1} \otimes_{A,B} (\mathfrak{C}^k(q'), \mathfrak{D}^k(q'))$ and $I^k_{k-t+t'+1} \otimes_{A,B} (\mathfrak{C}^k(q'), \mathfrak{D}^k(q'))$. But that also follows from the same argument as that of Case 1, and we are done.

Case 4: $q = Nq'$. The argument is similar to that of Case 3.

Note that for any k,

$$\mathfrak{C}^k(N) \models \neg[\text{TC}_{x,y}\ Exy](a,b) \quad \text{while} \quad \mathfrak{D}^k(N) \models [\text{TC}_{x,y}\ Exy](a,b).$$

If non-reachability could be expressed in $L^\omega_{\infty\omega}(\text{BU})$, then it could be expressed in $L^\omega_{\infty\omega}((\exists^*\forall)^k)$ for some $k \in \mathbb{N}$. But since the Duplicator wins the $(k+1)$-pebble $(\exists^*\forall)^k$-game on $\mathfrak{C}^{k+1}(N)$ and $\mathfrak{D}^{k+1}(N)$, this cannot be the case, and it follows in particular that non-reachability is not in (FO + pos TC). ■

We can now prove Immerman's conjecture.

Corollary 8 *For all $i \geq 0$, $\Sigma^{\text{TC}}_i \subsetneq \Sigma^{\text{TC}}_{i+1}$.*

Proof. We define, by induction on i, subsets E_i and A_i of $\{\exists, \forall, \exists^*, \forall^*\}^*$: Let

$$E_0 = A_0 = \{\exists, \forall\}^*$$
$$E_{i+1} = (\exists^* E_0)^* A_i$$
$$A_{i+1} = (\forall^* E_0)^* E_i.$$

It is easy to see that, for all i,

$$\Sigma^{\text{TC}}_i \subseteq \bigcup_{p \in E_i} L^\omega_{\infty\omega}(p).$$

Let $q_i = TNT\cdots$ be an alternating sequence of length i of the symbols T and N, starting with T. Obviously $\mathrm{TC}(q_i) \in \Sigma_i^{\mathrm{TC}}$. By Theorem 6, for all $i \geq 1$

$$\mathrm{TC}(q_i) - \bigcup_{p \in E_{i-1}} L^{\omega}_{\infty\omega}(p) \quad \text{is not empty.}$$

■

Avoiding increasing signatures in the Hierarchy Theorem. At the expense of a slightly more complicated construction, it is possible to get rid of the indefinite number of predicates $F_0, F_1, \ldots$ that appear in the proof of the Hierarchy Theorem and to prove the theorem for a uniform signature. Uwe Bosse [5] came up with a construction with a 4-ary predicate, three binary predicates, and two constant symbols; his construction can be modified into one with three binary predicates and two constant symbols.

Recall that the predicates F_m were needed to identify source-sink pairs of the substructures $\mathfrak{C}^k(q)$ and $\mathfrak{D}^k(q)$ of level m. If $|q| = m$ then the predicate F_m appears in the formulae $\psi_{Qq}(x, y)$. For instance

$$\psi_{\exists q}(x, y) \equiv (\exists u)(F_m xu \wedge F_m uy \wedge \psi_q(x, u) \wedge \psi_q(u, y)).$$

We will now replace F_m by a subformula $\beta_q \in \mathrm{TC}(q)$; then $\psi_{Qq}(x, y)$ remains inside $\mathrm{TC}(Qq)$. The formulae $\beta_q(x, y)$ will be defined inductively and have to satisfy the property that for every (reduced) $q'' = q'q$ and for $\mathfrak{B} = \mathfrak{C}^k(q'')$ or $\mathfrak{B} = \mathfrak{D}^k(q'')$,

$$\mathfrak{B} \models (\forall x)(\forall y)(\beta_q(x, y) \leftrightarrow F_m xy).$$

In addition, on $\mathfrak{B} = \mathfrak{C}^k(q'')$ or $\mathfrak{B} = \mathfrak{D}^k(q'')$, where $q'' = q'q$, the formula $\beta_q(x, y)$ and $\beta_{\bar{q}}(x, y)$ are equivalent: they both can replace the predicate F_m expressing that "between" x and y lies a copy of $\mathfrak{C}^k(q)$ or $\mathfrak{D}^k(q)$.

To achieve this we work with the uniform signature $\sigma = \{E, F_0, G, a, b\}$ where E, F_0, a, b are as before and where G is a binary predicate to be interpreted in $\mathfrak{C}^k(q'')$ and $\mathfrak{D}^k(q'')$ as $\bigcup_{0<i\leq|q''|} F_i$.

First, we set $\beta_\epsilon(x, y) \equiv F_0 xy$. Then set

$$\begin{aligned}
\beta_\exists(x, y) &\equiv Gxy \wedge (\exists u)[F_0 xu \wedge F_0 uy] \\
\beta_\forall(x, y) &\equiv Gxy \wedge (\forall u)[\neg Gxu \vee \neg Guy].
\end{aligned}$$

We then define

$$\begin{aligned}
\beta_{\exists q}(x, y) &\equiv Gxy \wedge (\exists u)[\beta_q(x, u) \wedge \beta_q(u, y)] \\
\beta_{\forall q}(x, y) &\equiv Gxy \wedge (\forall u)[(Gxu \wedge Guy) \rightarrow (\beta_q(x, u) \wedge \beta_q(u, y))].
\end{aligned}$$

And then we write

$$\begin{aligned}
\beta_{Tq}(x, y) &\equiv Gxy \wedge \neg\beta_{\bar{q}}(x, y) \wedge [\mathrm{TC}_{u,v}\, \beta_q(u, v) \wedge (Gxu \rightarrow \beta_q(x, u))](x, y) \\
\beta_{Nq}(x, y) &\equiv Gxy \wedge \neg \bigvee_{p \leq q} \beta_{\bar{p}}(x, y) \wedge \neg[\mathrm{TC}_{u,v}\, Guv \wedge (u, v) \neq (x, y) \wedge \neg \bigvee_{p \leq q} \beta_{\bar{p}}(u, v)](x, y).
\end{aligned}$$

It is not difficult to see that β_q is a q-formula. To see that $\beta_q \equiv F_{|q|}$ for non-null q, proceed by induction on $|q|$. Easily, if $\beta_q \equiv F_{|q|}$, then $\beta_{\exists q} \equiv F_{|q|+1}$ or $\beta_{\forall q} \equiv F_{|q|+1}$ as appropriate. And recalling that $\beta_{\bar{q}} \equiv F_{|q|} \equiv \beta_q$, β_{Tq} asserts that $F_m xy$ for some $m \neq |q|$ and that going along $x = u_0$, u_1, u_2, ..., where $F_{|q|} u_k u_{k+1}$ and $\neg F_h x u_k$ for any $h \neq |q|$, we reach $u_j = y$; clearly we must have $m = |q| + 1$, and $\beta_{Tq} \equiv F_m$. Similarly, β_{Nq} says that $F_m xy$ for some $m > |q|$, while any attempt to go from x to y along steps $\bigvee_{h > |q|+1} F_h$ will fail: again, this forces $m = |q| + 1$ and $\beta_{Nq} \equiv F_m$.

Thus, the formula $\psi_q(x,y) \wedge \beta_q(x,y)$ is in $\mathrm{TC}(q)$ and expresses that x and y are source and sink of a $\mathfrak{C}^k(q)$-copy while $\neg\psi_q(x,y) \wedge \beta_{\bar{q}}$ is in $\mathrm{TC}(\bar{q})$ and says that x and y are the source-sink pair of a $\mathfrak{D}^k(q)$-copy.

We conclude this section with a list of implications that follow from our main results:

Deterministic transitive closure. Immerman has introduced also a deterministic version of transitive closure logic, denoted (FO + DTC). It is the closure of first-order logic under the DTC-operator which is defined by

$$[\mathrm{DTC}_{\bar{x},\bar{y}}\ \varphi(\bar{x},\bar{y})](\bar{u},\bar{v}) \equiv [\mathrm{TC}_{\bar{x},\bar{y}}\ \Big(\varphi(\bar{x},\bar{y}) \wedge \forall \bar{z}(\varphi(\bar{x},\bar{z}) \rightarrow \bar{y} = \bar{z})\Big)](\bar{u},\bar{v}).$$

On successor structures, (FO + DTC) captures LOGSPACE. In a previous paper [15], we showed by an entirely different method, that deterministic transitive closure logic (FO + DTC) is weaker than (FO + TC). It is easy to see that (FO + pos DTC) = (FO + DTC), so Theorem 6 also implies that (FO + DTC) $\subsetneq$ (FO + pos TC). However, it should be noted that the results in [15] prove more than the mere separation of (FO + DTC) from (FO + TC): We show that on every class that satisfies a certain criterion, called the *short path criterion*, (FO + DTC) collapses to first-order logic. One of the most important classes satisfying the short path criterion is the class of all hypercubes.

Henkin quantifiers. Blass and Gurevich [4] have shown that transitive closure logic is equivalent to (FO + NH), the extension of first-order logic by *narrow Henkin quantifiers*. More precisely, positive occurrences of TC operators correspond to negative occurrences of narrow Henkin quantifiers, and vice versa. Thus, our results imply that (FO + pos NH) is not closed under complementation, and the hierarchy, formed by interleaving narrow Henkin quantifiers and negations, is infinite.

The structure of positive transitive closure logic. In an earlier paper [14], one of us showed that there is an infinite hierarchy inside (FO + pos TC): Consider a signature with at least two constants a and b. TC^0 is the set of formulae $[\mathrm{TC}_{\bar{x},\bar{y}}\ \varphi(\bar{x},\bar{y})](\bar{x},\bar{y})$ where φ is a first-order formula. Then $\forall$-TC^m is the universal closure of TC^m and TC^{m+1} is the set of formulae $[\mathrm{TC}_{\bar{x},\bar{y}}\ \varphi(\bar{x},\bar{y})](\bar{x},\bar{y})$ where $\varphi \in \forall$-TC^m. Consequently, for all m, $\mathrm{TC}^m \subsetneq \forall$-$\mathrm{TC}^m \subsetneq \mathrm{TC}^{m+1}$. Also, the Hierarchy Theorem subsumes the Main Theorem of [31].

6 Extensions of Datalog

We now discuss a class of languages intermediate between (FO + TC) and $L^{\omega}_{\infty\omega}$. They are fragments of fixpoint logic (see [17, 20, 32]) and were described in game-theoretic terms in [31]. Here we describe them as extensions of Datalog.

Datalog is a now very popular database query language which, in its pure form, consists of function-free and negation-free Horn clauses. Although Datalog has very nice properties and deals with recursion in a very nice way, it also has severe limitations. For instance, even very simple first-order properties are not computable by Datalog programs. This comes from the notorious difficulties that appear with negations in logic programming. An important extension of Datalog, called *Stratified Datalog*, solves part of this problem by partitioning the program into strata, so that negation over predicates that are computed on lower strata can be used in higher ones. We consider here the expressive power of different extensions of Datalog and provide insights into the fine structure of stratified Datalog.

Let's first recall the basic definitions of Datalog:

Definition 11 A *Datalog program* Π consists of a finite set of rules of the form

$$H \leftarrow B_1 \wedge \cdots \wedge B_m$$

where H is an atomic formula $S(x_1, \ldots, x_r)$, called the *head* of the rule, and $B_1 \wedge \cdots \wedge B_m$ is a conjunction of atomic formulae $R(x_1, \ldots, x_m)$ and equalities $x_i = x_j$. Every x_i is either a variable or a constant. A predicate that occurs in the head of some rule is called an *intensional database predicate*, abbreviated IDB predicate; a predicate occuring only in the bodies of the rules is called an *extensional database predicate*, or EDB predicate. One of the IDB predicates is the *goal predicate* of the program. The extensional vocabulary of Π is formed by all EDB predicates and by all constants occurring in Π; a finite structure of this vocabulary is called an *extensional database* EDB for Π.

Given any extensional database $\mathfrak{B}$, the program computes intensional relations, by the usual fixpoint semantics (or, equivalently, minimum model semantics). The result of Π on $\mathfrak{B}$ is the value of the goal predicate after the computation is terminated.

Example. The TC query is computed by the following Datalog program:

$$Txx; \qquad Txy \leftarrow Exy; \qquad Txz \leftarrow Txy \wedge Tyz$$

It is easy to see that every query computed by a Datalog program is preserved by homomorphisms and extensions of the EDB. Since one may also want to compute queries that do not have this property, the following extensions of Datalog have been considered:

Definition 12 Datalog($\neq$) extends Datalog by allowing inequalities $x_i \neq x_j$ in the body of the rules. Datalog($\neg$) allows also negated atomic formula $\neg R(x_1, \ldots, x_r)$ provided R is an extensional predicate. Finally, *Stratified Datalog* [8, 36], denoted S-Datalog, allows a limited form of negation over the IDB predicates: The IDB predicates are partitioned into strata, say $I_1, \ldots, I_m$; for simplicity, let I_0 denote the set of EDB predicates. In a rule

$$S(\bar{x}) \leftarrow B_1 \wedge \cdots \wedge B_m$$

where S is an IDB predicate from stratum I_j (with $j > 0$), the body may contain atomic formulae $R(\bar{y})$ provided that R does not belong to a higher stratum than S; and it may contain negated atomic formulae $\neg R(\bar{y})$ provided R belongs to a strictly lower stratum than S, i.e. $R \in \bigcup_{k<j} I_k$.

The semantics of Datalog readily extends to Datalog($\neq$) and Datalog($\neg$). The intensional predicates of a stratified Datalog program are computed stratum by stratum; in fact the set of rules whose head is in I_j may be considered as a Datalog($\neg$) program whose EDB predicates are $\bigcup_{k<j} I_k$.

Datalog($\neq$) and Datalog($\neg$) also have monotonicity properties: Queries computed by Datalog($\neg$) are preserved by extensions; queries computable by Datalog($\neq$) are preserved by extensions and one-to-one homomorphisms. This excludes, e.g., that these programs can compute all first order queries. The question has been raised whether Datalog, Datalog($\neq$) and Datalog($\neg$) are strong enough to compute all polynomial time queries that satisfy the respective monotonicity properties. This has recently been answered negatively [2].

Stratified Datalog is a proper extension of first order logic; it has the same expressive power as (stratified) existential fixpoint logic. It had been claimed that S-Datalog has the same expressive power as fixpoint logic, but Dahlhaus [10] and Kolaitis [26] exhibited fixpoint queries that are not computable by any stratified Datalog program.

Definition 13 A program in (an extension of) Datalog is *linear* if the body of each rule contains at most one IDB predicate.

The Datalog program for TC given above is not linear. But there is of course a linear Datalog program for TC, namely

$$Txx; \qquad Txz \leftarrow Txy \wedge Eyz.$$

However, we pay a price for the linearization: the original Datalog program reaches the fixpoint after $O(\log n)$ iterations while the linear program requires n iterations (where n is the length of the longest path).

We will now define two more extensions of Datalog, situated between Datalog($\neg$) and S-Datalog. We first want to define Datalog(FO), introduced in [14] which is arguably the simplest extension of Datalog that includes all FO-queries.

Definition 14 Datalog(FO) extends Datalog by allowing rules where the expressions in the body may be atomic formulae or arbitrary first order formulae $\varphi(\bar{x})$ over the EDB predicates. The semantics of Datalog(FO) is defined in the same way as for Datalog.

Intuitively, Datalog(FO) programs apply the recursion mechanism of Datalog not just to the EDB predicates but to any predicate computed by a FO-query on the extensional database.

Next, we define a stronger extension, which allows universal projection of already computed relations.

Definition 15 Datalog(FO,$\forall$) extends Datalog(FO) in the following way: Let I_0 be the EDB predicates and let the IDB predicates be partitioned into strata $I_1, \ldots, I_m$. A rule with head $S(\bar{x})$ may contain the following expressions in the body:

- first order formulae over the extensional predicates;
- atoms $R(\bar{y})$ provided R does not belong to a higher stratum than S;
- formulae $(\forall \bar{z})R(\bar{y}, \bar{z})$ provided R belongs to a strictly lower stratum than S.

Again, we have fixpoint sematics for Datalog(FO,$\forall$) by computing the the least fixpoints for lower strata before going to a higher one. Datalog (FO,$\forall$) was introduced independently in [14] and in [31] where it was called (FO + pos EFP).

Transitive closure logic and linear Datalog programs. There is a very close relationship between transitive closure logic and linear programs in extensions of Datalog. First of all, letting S-LinDatalog be stratified linear Datalog,

Theorem 9 [9] S-LinDatalog = (FO + TC).

This can be refined in terms of the subclasses of TC^m of (FO + pos TC), defined at the end of section 5. Recall that TC^0 is the class of all formulae

$$[\mathrm{TC}_{\bar{x},\bar{y}}\ \varphi(\bar{x},\bar{y})](\bar{u},\bar{v})$$

where φ is first-order. Thus in terms of the quantifier classes $\mathrm{TC}(p)$:

$$\mathrm{TC}^0 = \bigcup_{p\in\{\exists,\forall\}^*} \mathrm{TC}(Tp).$$

For $m > 0$, TC^m is the class of all formulae

$$[\mathrm{TC}_{\bar{x},\bar{y}}\ (\forall\bar{z})\varphi(\bar{x},\bar{y},\bar{z})](\bar{u},\bar{v})$$

with $\varphi \in \mathrm{TC}^{m-1}$.

Theorem 10 [14] *Let $m \in \mathbb{N}$. A query is expressible in TC^m if and only if it is computable by a linear* Datalog(FO,∀)*-program with $m+1$ strata. In particular:*

$$\begin{aligned} \text{LinDatalog(FO)} &= \mathrm{TC}^0 \\ \text{LinDatalog(FO,}\forall) &= \text{(FO + pos TC)} \end{aligned}$$

Furthermore we would like to mention that not all Datalog programs are linearizable. In fact there even exist Datalog programs that are not expressible in transitive closure logic, i.e . they are not even equivalent to any *stratified* linear Datalog program:

Theorem 11 Datalog − (FO + TC) $\neq \emptyset$.

This result is implicit already in an early paper of Immerman [18]. In more explicit form it appears in [1]. In fact it also follows from Immerman's proof that the *alternating graph accessibility problem* AGAP, even restricted to graphs with outdegree at most two, is not expressible in (FO + TC). On the other hand both this problem and its complement are in Datalog(¬). Thus, Datalog(¬) ∩ Co-Datalog(¬) $\not\subseteq$ (FO + TC).

Datalog and infinitary logic. It is easy to see (and well-known) that Datalog is included in the existential fragment of $L^\omega_{\infty\omega}$ (over the class of finite structures) — see [27] for inexpressibility results on Datalog that are based on this fact — and that stratified Datalog programs can be expressed by infinitary formulae with bounded alternations:

Proposition 12 *(i)* Datalog $\subsetneq$ Datalog(¬) $\subsetneq L^\omega_{\infty\omega}(\exists^*)$.

(ii) (FO + TC)$\subsetneq$ S-Datalog $\subsetneq \bigcup_{k\in\mathbb{N}} L^\omega_{\infty\omega}(\Sigma_k)$.

The inclusions are strict, due to the results mentioned above and the fact that $L^\omega_{\infty\omega}(\exists^*)$ can express uncountably many properties. Also the extensions Datalog(FO) and Datalog(FO,∀) can be embedded into $L^\omega_{\infty\omega}$. A program in Datalog(FO) can be expressed by a formula $\psi \in L^\omega_{\infty\omega}$ with quantifier structure $P(\psi) = \exists^* p$ for some fixed $p \in \{\exists,\forall\}^*$. The fragment of infinitary logic that extends Datalog(FO,∀) is $L^\omega_{\infty\omega}(\mathrm{BU})$, the set of infinitary formulae with bounded number of universal quantifiers, defined at the end of section 2.

Proposition 13 *(i)* $\mathrm{TC}^0 \subsetneq \mathrm{Datalog(FO)} \subsetneq \bigcup\limits_{p \in \{\exists,\forall\}^*} L^{\omega}_{\infty\omega}(\exists^* p)$.

(ii) $(\mathrm{FO} + \mathrm{pos\ TC}) \subsetneq \mathrm{Datalog(FO,}\forall) \subsetneq L^{\omega}_{\infty\omega}(\mathrm{BU})$.

Fine structure of stratified Datalog. Our hierarchy theorem allows to separate different extensions of Datalog and extends the Main Theorem of [31] on the fine structure of fixpoint logic, which, in the notation of this paper, asserts the following: Let p and q be two words of the form

$$f_0 P_1 f_1 P_2 \cdots P_n f_n$$

where $f_i \in \{\forall, \exists\}^*$ and $P_i \in \{\exists^*, \forall^*\}$ such that p cannot be embedded into q. Then there then there exists a p-definable query that is not q-definable.

In particular we can conclude

Corollary 14

1. Datalog(FO) *and* Datalog(FO,$\forall$) *are not closed under complementation. Thus* Datalog(FO,$\forall$) $\subsetneq$ S-Datalog.
2. *For all k, there exists a (linear)* Datalog(FO,$\forall$)*-program with $k+1$ strata which is not expressible in $\bigcup_{m \in \mathbb{N}} L^{\omega}_{\infty\omega}((\exists^*\forall^m)^k)$, which therefore is not equivalent to any* Datalog(FO,$\forall$)*-program with k strata.*

We can summarize these results in the following diagram:

$$\begin{array}{ccccccccc}
 & & & & \mathrm{LinDatalog} & \subset & \mathrm{Datalog} & \subset & L^{\omega}_{\infty\omega}(\exists^*) \\
 & & & & \cap & & \cap & & \cap \\
\mathrm{FO} & \subset & \mathrm{TC}(0) & = & \mathrm{LinDatalog(FO)} & \subset & \mathrm{Datalog(FO)} & \subset & \bigcup\limits_{p \in \{\exists,\forall\}^*} L^{\omega}_{\infty\omega}(\exists^* p) \\
 & & \cap & & \cap & & \cap & & \cap \\
 & & \vdots & & \vdots & & \vdots & & \vdots \\
 & & \cap & & \cap & & \cap & & \cap \\
 & & (\mathrm{FO} + \mathrm{pos\ TC}) & = & \mathrm{LinDatalog}(\mathrm{FO},\forall) & \subset & \mathrm{Datalog}(\mathrm{FO},\forall) & \subset & L^{\omega}_{\infty\omega}(\mathrm{BU}) \\
 & & \cap & & \cap & & \cap & & \cap \\
 & & (\mathrm{FO} + \mathrm{TC}) & = & \text{S-LinDatalog} & \subset & \text{S-Datalog} & \subset & \bigcup_k L^{\omega}_{\infty\omega}(\Sigma_k) \\
 & & & & & & \cap & & \cap \\
 & & & & & & (\mathrm{FO} + \mathrm{LFP}) & \subset & L^{\omega}_{\infty\omega}
\end{array}$$

The vertical dots stand for the infinite hierarchies defined by the number of strata or equivalently, the number of quantifier alternations (as in Corollary 14); $\subset$ means proper inclusion.

7 Some Conjectures about Prefixes

In this paper, we have been investigating "quantifier classes". Traditionally, the FO queries were characterized by "prefix classes", i.e., a prefix is a string of $\exists$s and $\forall$s, and given a prefix Q, a FO query R is Q-definable if there exists a quantifier-free formula $\theta(\bar{x}, \bar{y})$ such that for all $\mathfrak{A}$,

$\mathfrak{A} \models \forall \bar{x}[R(x) \leftrightarrow Q\bar{y}\theta(\bar{x}, \bar{y})]$.

A large amount of classical model theory is devoted to expressibility classes based on prefixes. The standard is the Levy hierarchy, based on the number of alternations: Σ_1 is the set of $\exists \cdots \exists$-expressible queries, Π_1 is the set of $\forall \cdots \forall$-expressible queries, Σ_{n+1} is the set of all $\exists \cdots \exists Q$-expressible queries (where Q is Π_n), Π_{n+1} is the set of all $\forall \cdots \forall Q$-expressible queries (where Q is Σ_n), and so on. Chandra and Harel [7] proved that on the class of all finite digraphs, these expressibility classes are distinct. Meanwhile, Walkoe [37] constructed a class of infinite structures such that if Q and Q' are distinct prefixes of the same length, then there exists a Q-expressible query that is not Q'-expressible in that class. Keisler and Walkoe [25] then proved that for each N, over the class of finite structures with one N-ary relation, if Q and Q' are distinct prefixes of length N, then there exists a Q-expressible query that is not Q'-expressible. And that is where the matter stands at the moment.

One question that probably comes to mind is: Does the Main Theorem hold for prefix classes? Let $\mathrm{FO}(q)$ be the class of FO-definable prefixes with prefix q.

Conjecture 1 *Over the class of all structures with a binary relation, if $q \not\leq q'$, then* $\mathrm{FO}(q) \not\subseteq \mathrm{FO}(q')$.

It should be noted that the techniques of this paper do not seem sufficient to settle this conjecture, because Ehrenfeucht-Fraïssé games deal with quantifier classes rather than prefix classes. Obviously, Conjecture 1 generalizes the results of Chandra-Harel and Keisler-Walkoe. In fact if Conjecture 1 is true, then the latter holds uniformly, i.e., it does not require structures admitting relations of large arity.

Furthermore, it should be noted that Conjecture 1 (and the Main Theorem itself) fails for structures admitting only unary relations:

Proposition 15 *Over the class of structures admitting only unary relations, all FO-definable queries are Δ_2-definable.*

Proof. Let $\mathbf{R} = R_1, \ldots, R_m$ be unary, and let $\mathrm{FO}(\mathbf{R})$ be the class of queries FO-definable from $\mathbf{R}$. For each $\iota = \iota_1\iota_2 \cdots \iota_m \in \{0,1\}^m$, let

$$R_\iota(x) \equiv \bigwedge_i [\iota_i = 1 \leftrightarrow R_i(x)].$$

Let $M = 2^m$, and for any fixed r, and any tuple $\mathbf{r} = (r_1, \ldots, r_M)$, where $r_\iota \leq r$ for each ι, let $\mathcal{C}(\mathbf{r})$ be the class of structures $\mathfrak{A}$ such that for each ι,

$$r_\iota < r \Rightarrow |\{a \in |\mathfrak{A}| \colon R_\iota(a)\}| = r_\iota$$

and

$$r_\iota = r \Rightarrow |\{a \in |\mathfrak{A}| \colon R_\iota(a)\}| \geq r.$$

By a simple Ehrenfeucht game argument, if $\mathfrak{A}, \mathfrak{B} \in \mathcal{C}(\mathbf{r})$, then $\mathfrak{A} \equiv_r \mathfrak{B}$. Hence it suffices to prove that the queries "$\mathfrak{A} \in \mathcal{C}(\mathbf{r})$"are are Δ_2-definable.

Fix $\mathbf{r} = r_1, \ldots r_M$. For each ι, let

$$\theta_\iota \equiv \exists x_1 \cdots x_{r_\iota} \forall y [\bigwedge_{i \neq j} x_i \neq x_j \wedge \bigwedge_i R_\iota(x_i) \wedge (R_\iota(y) \rightarrow \bigvee_i y = x_i)],$$

and as θ_ι is also Π_2, it is Δ_2, and hence

$$\mathfrak{A} \in \mathcal{C}(\mathbf{r}) \equiv \mathfrak{A} \models \bigwedge_\iota \theta_\iota$$

is Δ_2-expressible. ∎

A more practical problem is this: given a query and a prefix or quantifier class, can we determine if that query can be expressed with that class? For each $n \in \{0, 1, 2, \ldots\}$ and each $a \leq n$, let $\mathfrak{N}_n(a) = \langle\{0, \ldots, n\}, <_n, G_{+,n}, G_{\times,n}, G_{\uparrow,n}, 0, a\rangle$, where $G_{+,n}$, $G_{\times,n}$, and $G_{\uparrow,n}$ are the graphs of the functions $x+y$, $x \times y = xy$, and $x \uparrow y = x^y$ restricted to $\{0, 1, \ldots, n\}$. Let $\mathrm{NF} = \{\mathfrak{N}_n(a) \colon a \leq n \in \omega\}$. Recall that the recursive functions can be generated from Zero, the Successor, and the Projection functions via Composition, Primitive recursion, and Minimalization. All this can be done in finitary ways on structures in NF. Let Δ_1' be the class of FO formulas in the language of arithmetic whose quantifications are bounded (i.e., of the form $(Qx < y)[\cdots]$), and let $\Sigma_1' = \{\exists x\theta \colon \theta \in \Delta_1'\}$ —- say that a Σ_1' formula is in *normal form* if it is of the form $\exists\bar{x}\theta$, where θ is Δ_1'. We need a bit of lore resembling Fact 2.1 of [30]:

Lemma 16 *If $f \colon \omega^k \to \omega$ is recursive, then there exists Σ_1' formulas θ_f and ψ_f in the above language such that the following hold:*

- *If $\mathfrak{N} \models f(\bar{x})\downarrow$, then for cofinitely many n, $\mathfrak{N}_n \models \psi_f(\bar{x})$. If $\mathfrak{N} \models f(\bar{x})\uparrow$, then for all n, $\mathfrak{N}_n \not\models \psi_f(\bar{x})$.*
- *If $f(\bar{x}) \simeq y$, then for all $n \geq \max\{\bar{x}, y\}$, $\mathfrak{N}_n \models \psi_f(\bar{x}) \to \theta_f(\bar{x}, y)$, while $f(\bar{x}) \not\simeq y$ implies that for all such n, $\mathfrak{N}_n \not\models \theta_f(\bar{x}, y)$.*

It is not hard to prove that, in general, there is no effective procedure to determine if, given a query and a prefix or quantifier class, the query is an element of the class. The proof for the class of all structures is immediate. Given a finite quantifier structure P, let $|P|$ be maximum length of any element of P, and for any n, let $\mathrm{FO}(n) = \bigcup\{\mathrm{FO}(P) \colon |P| = n\}$; by Ehrenfeucht's [11] theorem that $\equiv_r$ has finitely many equivalence classes, $\mathrm{FO}(n)$ is finite. For each P, let $\mathrm{Rec}(P)$ be the class of recursive functions defined on $\mathfrak{N}_\infty(a) = \langle\omega, <, G_+, G_\times, G_\uparrow, 0, a\rangle$ by Σ_1' formulas with prefixes in P. Clearly, $\mathrm{Rec}(P)$ is finite and nonempty for each P. Hence, by Rice's theorem, it follows that $\{\theta \colon$ "θ defines a function in $\mathrm{Rec}(P)$"$\}$ is undecidable. But what if our interest is restricted to finite structures?

Proposition 17 *There is no effective procedure to determine if, given a query over all finite structures, and a prefix or quantifier class, the query is an element of that class.*

Proof. As any prefix P can express only finitely many Σ_1' queries, there will exist a Σ_1' query f that is not P-expressible. Note that if ψ is a Σ_1' unary formula in normal form, then if n is sufficiently large with respect to a, $\mathfrak{N}_n(a) \models \psi \Leftrightarrow \mathfrak{N}(a) \models \psi$. By Rice's Theorem, there is no effective procedure for separating the Σ_1' formulas for the empty function from the Σ_1' formulas for f. Hence, there is no effective procedure for determining if a given query is P-expressible. ∎

We would like to ask a more general question. We might want to know if a given query is, say, Σ_5-expressible. Can this be done effectively? We suspect that the answer is no: If $\mathcal{P}$ is a collection of finite prefix structures such that

$$[P \in \mathcal{P} \ \& \ P' \ \leq P] \Rightarrow P' \in \mathcal{P},$$

call $\mathcal{P}$ *regular.* If $\mathcal{P}$ is regular, call a query $\mathcal{P}$-*expressible* if it is P-expressible for some $P \in \mathcal{P}$.

Conjecture 2 If $\mathcal{P}$ is a regular, recursive, nontrivial class of prefix structures, then the class of queries that are $\mathcal{P}$-expressible is not decidable.

We suspect that Conjecture 2 holds even if we allow $\mathcal{P}$ to contain infinite prefix structures.

Acknowledgement. The first author would like to thank the participants of the Basel-Freiburg seminar on finite model theory, in particular Uwe Bosse and Martin Otto, for valuable suggestions and discussions.

References

[1] F. Afrati and S. Cosmadakis, *Expressiveness of Restricted Recursive Queries*, Proceedings of 21st ACM Symposium on Theory of Computing (1989), 113–126.

[2] F. Afrati, S. Cosmadakis and M. Yannakakis, *On Datalog vs. Polynomial Time*, Proceedings of 10th ACM Symposium on Principles of Database Systems (1991).

[3] J. Barwise, *On Moschovakis closure ordinals*, J. Symbolic Logic **42** (1977), 292–296.

[4] A. Blass and Y. Gurevich, *Henkin quantifiers and complete problems*, Annals of Pure and Applied Logic **32** (1986), 1–16.

[5] U. Bosse, personal commmunication (1992).

[6] J. Cai, M. Fürer and N. Immerman, *An Optimal Lower Bound on the Number of Variables for Graph Identification*, Proceedings of 30th IEEE Symposium on Foundations of Computer Science (1989), 612–617.

[7] A. Chandra and D. Harel, *Structure and Complexity of Relational Queries*, J. Comp. Syst. Sciences **25** (1982), 99–128.

[8] A. Chandra and D. Harel, *Horn Clause Queries and Generalizations*, J. Logic Programming **1** (1985), 1–15.

[9] M. Consens and A. Mendelzon, *GraphLog: a Visual Formalism for Real Life Recursion*, Proceedings of 9th ACM Symposium on Principles of Database Systems (1990), 404–416.

[10] E. Dahlhaus, *Skolem Normal Forms Concerning the Least Fixed Point*, in: "Computation Theory and Logic" (E. Börger, Ed.), Lecture Notes in Computer Science Nr. 270, Springer 1987, 101–106.

[11] A. Ehrenfeucht, *An application of games to the completeness problem for formalized theories*, Fund. Math. **49** (1961) 129–141.

[12] R. Fagin, *Monadic generalized spectra*, Zeitschrift für Math. Logik Grundlagen d. Math. **21** (1975), 89–96.

[13] R. Fraïssé, *Sur quelques classifications des systèmes de relations*, Publications Scientifique de l' Université d' Alger, Série A **1** (1954), 35–182.

[14] E. Grädel, *On transitive closure logic*, Proceedings of 5th workshop on Computer Science Logic CSL '91, Lecture Notes in Computer Science Nr. 626, Springer 1992, 149–163.

[15] E. Grädel and G. McColm, *Deterministic versus Nondeterministic Transitive Closure Logic*, Proceedings of 7th IEEE Conference on Logic in Computer Science (1992), 58–63.

[16] Y. Gurevich, *Logic and the Challenge of Computer Science*, in: E. Börger (Ed), Trends in Theoretical Computer Science, Computer Science Press (1988), 1–57.

[17] Y. Gurevich and S. Shelah, *Fixed Point Extensions of First Order Logic*, Annals of Pure and Applied Logic **32** (1986), 265–280.

[18] N. Immerman, *Number of Quantifiers is Better than Number of Tape Cells*, J. Comp. Syst. Sciences **22** (1981), 65–72.

[19] N. Immerman, *Upper and lower bounds for first-order expressibility*, J. Comp. Syst. Sciences **25** (1982), 76–98.

[20] N. Immerman, *Relational queries computable in polynomial time*, Information & Control 68 (1986), 86-104.

[21] N. Immerman, *Languages that Capture Complexity Classes*, SIAM J. Comput. **16** (1987), 760–778. A preliminary version appeared in the Proceedings of 15th ACM Symposium on Theory of Computing (1983), 347–354.

[22] N. Immerman, *Nondeterministic space is closed under complementation*, SIAM J. Comput. **17** (1988), 935–939.

[23] N. Immerman, *Descriptive and Computational Complexity*, in: J. Hartmanis (Ed.), Computational Complexity Theory, Proc. of AMS Symposia in Appl. Math. **38** (1989), 75–91.

[24] P. Kannellakis, *Elements of Relational Database Theory*, in: J. van Leeuwen (Ed.), Handbook of Theoretical Computer Science, vol. B, North Holland, Amsterdam 1990, pp. 1073–1156.

[25] H. J. Keisler & W. Walkoe, Jr. *The diversity of quantifier prefixes*, J. Sym. Logic **38** (1973), 79–85

[26] Ph. Kolaitis, *The Expressive Power of Stratified Logic Programs*, Information and Computation **90** (1991), 50–66.

[27] Ph. Kolaitis and M. Vardi, *On the Expressive Power of Datalog: Tools and a Case Study*, Proceeding of 9th ACM Symposium on Principles of Database Systems (1990), 61–71.

[28] Ph. Kolaitis and M. Vardi, *Infinitary logics and 0-1 laws*, Information and Computation **98** (1992), 258–294.

[29] V. Lakshmanan and A. Mendelzon, *Inductive pebble games and the expressive power of datalog*, Proceedings of 8th ACM Symposium on Principles of Database Systems (1989), 301–310.

[30] G. McColm, *Parametrization over inductive relations of a bounded number of variables*, Ann. Pure & Applied Logic **48** (1990), 103-134.

[31] G. McColm, *Pebble Games and the Fine Structure of Fixedpoint Logic*, submitted for publication.

[32] Y. Moschovakis, *Elementary Induction on Abstract Structures*, North-Holland, Amsterdam 1974.

[33] L. Stockmeyer, *The polynomial time hierarchy*, Theoretical Computer Science **3:1** (1976), 1–22.

[34] R. Szelepcsényi, *The Method of Forced Enumeration for Nondeterministic Automata*, Acta Informatica **26** (1988), 279–284.

[35] M. Vardi, *Complexity of relational query languages*, Proceedings of 14th STOC (1982), 137-146.

[36] A. Van Gelder, *Negation as Failure using tight derivations for general logic programs*, Proceedings of 3rd IEEE Symposium on Logic Programming (1986), 137–146.

[37] W. Walkoe, *Finite partially-ordered quantification*, J. Sym. Logic **35** (1970), 535–555.

[38] M. Zloof, *Query-by-Example: Operations on the Transitive Closure*, IBM Research Report RC5526 (1976).

Figure 1

Figure 2

Figure 3

AMBER
BLUE

Figure 4

AMBER
BLUE

Figure 5

AMBER
BLUE

Figure 6

AMBER
BLUE