Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Computer and System Sciences 70 (2005) 101–127
JOURNAL OF COMPUTER AND SYSTEM SCIENCES
www.elsevier.com/locate/jcss

First-order expressibility of languages with neutral letters or: The Crane Beach conjecture

David A. Mix Barrington[a,1], Neil Immerman[a,2], Clemens Lautemann[b], Nicole Schweikardt[c,*], Denis Thérien[d,3]

[a] *Computer Science Department, University of Massachusetts, USA*
[b] *Institut für Informatik, Johannes Gutenberg-Universität Mainz, Germany*
[c] *Institut für Informatik, Humboldt-Universität zu Berlin, Unter den Linden 6, D-10099, Berlin, Germany*
[d] *School of Computer Science, McGill University, Canada*

Received 10 October 2003; received in revised form 9 July 2004

Available online 05 November 2004

Abstract

A language L over an alphabet A is said to have a *neutral letter* if there is a letter $e \in A$ such that inserting or deleting e's from any word in A^* does not change its membership or non-membership in L.

The presence of a neutral letter affects the definability of a language in first-order logic. It was conjectured that it renders all numerical predicates apart from the order predicate useless, i.e., that if a language L with a neutral letter is not definable in first-order logic with linear order, then it is not definable in first-order logic with any set $\mathcal{N}$ of numerical predicates. Named after the location of its first, flawed, proof this conjecture is called the Crane Beach conjecture (CBC, for short). The CBC is closely related to uniformity conditions in circuit complexity theory and to collapse results in database theory.

We investigate the CBC in detail, showing that it fails for $\mathcal{N} = \{+, \times\}$, or, possibly stronger, for any set $\mathcal{N}$ that allows counting up to the m times iterated logarithm, for any constant m. On the positive side, we prove the conjecture for the case of all monadic numerical predicates, for the addition predicate $+$, for the fragment $BC(\Sigma_1)$ of first-order logic, for regular languages, and for languages over a binary alphabet. We explain the precise relation

* Corresponding author.

E-mail addresses: barring@cs.umass.edu (David A. Mix Barrington), immerman@cs.umass.edu (Neil Immerman), cl@informatik.uni-mainz.de (Clemens Lautemann), schweika@informatik.hu-berlin.de (Nicole Schweikardt), denis@cs.mcgill.ca (Denis Thérien).

[1] Supported by NSF Grant CCR-9988260.

[2] Supported by NSF Grant CCR-9877078.

[3] Supported by NSERC and FCAR.

doi:10.1016/j.jcss.2004.07.004

between the CBC and so-called natural-generic collapse results in database theory. Furthermore, we introduce a framework that gives better understanding of what exactly may cause a failure of the conjecture.

Keywords: First-order logic; Circuit uniformity; Numerical predicates

---

1. Introduction

Logicians have long been interested in the relative expressive power of different logical formalisms. In the last 20 years, these investigations have also been motivated by a close connection to computational complexity theory—most computational complexity classes have been given characterisations as finite model classes of appropriate logics, cf. [16]. In these investigations it became apparent that in order to describe computation over a finite structure, a formula has to be able to refer to some linear order of the elements of this structure. Given such an order, the universe of the structure, i.e., the set of its elements, can be identified with an initial segment of the natural numbers. In a logic with the capability to express induction we can then define predicates for arithmetical operations such as addition or multiplication on the universe, and use them in order to describe operations on time or memory locations. In weak logics, however, e.g., first-order logic, defining an order relation does not automatically make arithmetic available. In fact, even over strings, the expressive power of first-order logic varies considerably, depending on the set of numerical predicates that can be used.

As an example, if the order is the only numerical relation then the only *regular* languages that can be defined in first-order logic are the star-free languages. If, however, for every $p \in \mathbb{N}$ we have available the predicate mod_p (which holds for a number m iff $m \equiv 0 \pmod p$) then we can express regular languages that are not star-free, such as $(000+001)^*$. In fact, with these predicates we can express *all* the first-order definable regular languages, cf. [31]. Thus, even very powerful relations (arithmetical relations, or even undecidable ones) are of no further help in defining regular languages. On the other hand, with addition, we can easily express languages that are not regular, such as $\{0^n 1^n \mid n \in \mathbb{N}\}$.

First-order logic with varying numerical predicates can also be thought of as specifying circuit complexity classes with varying *uniformity conditions* [6]. The language defined by a first-order formula is naturally computed by a family of Boolean circuits with constant depth, polynomial size, and unbounded fan-in (called "AC^0 circuits"). The power of such a family depends in part on the sophistication of the connections among the nodes. A formula with only simple numerical predicates leads to a circuit family where these connections are easily computable. These are called "uniform circuits", and how uniform they are is quantified by the computational complexity of a language describing the connections. A formula with arbitrary numerical predicates leads to a circuit family with arbitrary connections—the set of languages so describable is called "non-uniform AC^0".

There are languages, such as the PARITY language, for which we can prove no AC^0 circuit exists [1,15]. A major open problem in complexity theory is to develop methods for showing languages to be outside of uniform circuit complexity classes even if they are in the corresponding non-uniform class. This is an additional motivation for the study of the expressive power of first-order logic with various numerical predicates, as this provides a parametrization of various versions of "uniform AC^0".

In an attempt to obtain a better understanding of this expressive power, Thérien considered the concept of a *neutral letter* for a language L, i.e., a letter e that can be inserted into or deleted from a string without affecting its membership in L. Since, in the presence of such a letter, membership in L cannot depend on specific (combinations of) letters being in specific (combinations of) positions, it seemed conceivable that neutral letters would render all numerical predicates, except for the order, useless. With this in mind, Thérien proposed what was later dubbed the *Crane Beach conjecture* (CBC) (named after the location of its first, flawed proof):

> If a language with a neutral letter can be defined in first-order logic using some set $\mathcal{N}$ of numerical predicates then it can be so defined using only the order relation.

One particular example of a language with a neutral letter is PARITY, consisting precisely of those 0–1-strings in which 1 occurs an even number of times. PARITY is not definable in first-order logic—no matter what numerical predicates are used (cf. [1,15]). The CBC would imply this result, since PARITY is a regular language known not to be star-free.

In this paper, we investigate the CBC in detail. We show that in general it is not true—in fact, it already fails for $\mathcal{N} = \{+, \times\}$. Furthermore, we introduce a framework that gives better understanding of what exactly may cause a failure of the conjecture. However, we also show that the conjecture is true in a number of interesting special cases, including the case of addition, i.e., when $\mathcal{N} = \{+\}$.

This work is closely related to a line of research in database theory which is concerned with so-called *collapse results* (cf. [9]). Here one considers a finite database embedded in some infinite, ordered domain, and then looks at *locally generic* queries, i.e., queries which are invariant under monotone injections of the database universe into the larger domain. In this setting, a language with a neutral letter is the special case of a locally generic (Boolean) query over monadic databases with background structure $\langle \mathbb{N}, \leqslant, \mathcal{N} \rangle$, and the conjecture then can be translated into a collapse for first-order logic.

The present paper combines results of the conference contribution [5] and the dissertation [25]. The paper is structured as follows: Section 2 fixes the basic notations concerning first-order logic and Ehrenfeucht–Fraïssé games. Section 3 gives an introduction to the Crane Beach conjecture (the CBC, for short). Section 4 presents the cases where the CBC is known to be true, explains the precise relation between the CBC and collapse results in database theory. Section 5 presents cases where the CBC is false, and Section 6 provides a framework that gives a better understanding of *what* exactly may cause the conjecture to fail. Building upon this framework we show that, in some sense, no reasonable update of the unrestricted version of the CBC is possible. Finally, Section 7 summarizes the results and points out suggestions for further work.

2. Preliminaries

2.1. First-order logic

A *signature* is a set σ containing finitely many relation, or predicate, symbols, each with a fixed arity. A σ-structure $\mathfrak{A} = \langle \mathcal{U}^{\mathfrak{A}}, \sigma^{\mathfrak{A}} \rangle$ consists of a set $\mathcal{U}^{\mathfrak{A}}$, called the *universe* of $\mathfrak{A}$ and a set $\sigma^{\mathfrak{A}}$ that contains an interpretation $R^{\mathfrak{A}} \subseteq (\mathcal{U}^{\mathfrak{A}})^k$ for each k-ary relation symbol $R \in \sigma$.

In this paper, we are concerned almost exclusively with first-order logic over *finite strings*. We write $|w|$ to denote the *length* of the string w. For an alphabet A we use the signature $\sigma_A := \{Q_a \,/\, a \in A\}$, and we identify a string $w = w_1 \cdots w_n \in A^*$ with the structure $w = \langle \{1, \ldots, n\}, \sigma_A^w, \leqslant \rangle$, where $\sigma_A^w = \{Q_a^w \,/\, a \in A\}$ and $Q_a^w = \{i \leqslant n \,/\, w_i = a\}$, i.e., $i \in Q_a^w \iff w_i = a$, for all $a \in A$.

In addition to the predicates Q_a we also have *numerical predicates*.[4] A k-ary numerical predicate P has, for every $n \in \mathbb{N}$, a fixed interpretation $P_n \subseteq \{1, \ldots, n\}^k$. Our prime example of a numerical predicate is the linear order relation $\leqslant$. Where we see no danger of confusion (i.e., almost everywhere) we will not distinguish notationally between a predicate and its interpretation.

Sometimes we will consider numerical predicates that result from relations over $\mathbb{N}$. Given a relation $R \subseteq \mathbb{N}^k$, we will write $\hat{R}$ to denote the k-ary numerical predicate with $\hat{R}_n = R \cap \{1, \ldots, n\}^k$ (for every $n \in \mathbb{N}$).

An *atomic σ-formula* is either of the form $x_1{=}x_2$, or $P(x_1, \ldots, x_k)$, where $x_1, x_2, \ldots, x_k$ are variables and $P \in \sigma$ is a k-ary predicate symbol. First-order σ-formulas are built from atomic σ-formulas in the usual way, using Boolean connectives $\wedge$, $\vee$, $\neg$, etc. and universal $(\forall x)$ and existential $(\exists x)$ quantifiers.

For every alphabet A, and every set $\mathcal{N}$ of numerical predicates, we will denote the set of first-order $\sigma_A \cup \mathcal{N}$-formulas by $FO[\mathcal{N}]$. We define the semantics of first-order formulas in the usual way. In particular, for a string $w \in A^*$ and a formula $\varphi \in FO[\mathcal{N}]$ without free variables (i.e., variables not bound by a quantifier), we will write $w \vDash \varphi$ iff φ holds on the string w. If $x_1, \ldots, x_k$ are the free variables of φ, and if $p_1, \ldots, p_k \leqslant |w|$, $w \vDash \varphi(p_1, \ldots, p_k)$ indicates that φ holds on the string w with x_i interpreted as p_i, for every $i \leqslant k$.

Every formula $\varphi \in FO[\mathcal{N}]$ without free variables defines the set L_φ of those A-strings that satisfy φ. We say that a language $L \subseteq A^*$ is *definable in $FO[\mathcal{N}]$*, and write $L \in FO[\mathcal{N}]$, if $L = L_\varphi$, for some $\varphi \in FO[\mathcal{N}]$. We will use analogous notation for subsets of $FO[\mathcal{N}]$; in particular, we will consider the set $\Sigma_1[\mathcal{N}]$ of formulas which are of the form $\exists x_1 \cdots \exists x_r \psi$, for some quantifier-free $\psi \in FO[\mathcal{N}]$, and its Boolean closure, $BC(\Sigma_1[\mathcal{N}])$. One can define a complete hierarchy of classes $\Sigma_i[\mathcal{N}]$ and $\Pi_i[\mathcal{N}]$ along with their Boolean closures, using the hierarchy of first-order formulas given by the number of quantifier alternations. But in this paper we will have need only for $BC(\Sigma_1[\mathcal{N}])$.

2.2. Ehrenfeucht–Fraïssé games

One of our main technical tools will be the *Ehrenfeucht–Fraïssé game* (EF-game, for short). In our context, the EF-game for a set of numerical predicates, $\mathcal{N}$, is played by two players, the Spoiler and the Duplicator, on two strings $u, v \in A^*$. There is a fixed number k of rounds, and in each round i

- first, the Spoiler chooses one position, a_i in u, or a position b_i in v;
- then the Duplicator chooses a position in the other string, i.e., a b_i in v, if the Spoiler's move was in u, and an a_i in u, otherwise.

After k rounds, the game finishes with positions $a_1, \ldots, a_k$ chosen in u and $b_1, \ldots, b_k$ chosen in v. The Duplicator has won if the mapping $a_i \mapsto b_i$, (for $i = 1, \ldots, k$), is a *partial $\sigma_A \cup \mathcal{N}$-isomorphism*, i.e., if

- for every $i, j \leqslant k$, $a_i = a_j \iff b_i = b_j$,
- for every $i \leqslant k$, a_i and b_i carry the same letter, i.e., $u_{a_i} = v_{b_i}$, and
- for every m-ary predicate $P \in \mathcal{N}$ and every $i_1, \ldots, i_m \leqslant k$, it holds that $P(a_{i_1}, \ldots, a_{i_m}) \iff P(b_{i_1}, \ldots, b_{i_m})$.

Since the game is finite, one of the two players must have a *winning strategy*, i.e., he or she can always win the game, no matter how the other player plays. If the Duplicator has a winning strategy in the k-round game for $\mathcal{N}$ on two strings u and v, we write $u \equiv_k^{\mathcal{N}} v$. The fundamental use of the game comes from

[4] In the literature, such predicates sometimes are also called *built-in predicates*.

the fact that it characterises first-order logic (cf., e.g., the textbooks [12,16]). In our context, this can be formulated as follows:

Theorem 2.1 (*Ehrenfeucht, Fraïssé*). *A language $L \subseteq A^*$ is definable in $FO[\mathcal{N}]$ iff there is a finite subset $\mathcal{N}'$ of $\mathcal{N}$ and a number k such that, for every $u \in L$, $v \notin L$, the Spoiler has a winning strategy in the k-round game for $\mathcal{N}'$ on u and v.*

We will also use the following variant of the game:
In the *single-round k-move game for $\mathcal{N}$* on two strings *u*, *v*

- first, the Spoiler chooses either *k* positions $a_1, \ldots, a_k$ in *u*, or *k* positions $b_1, \ldots, b_k$ in *v*;
- then the Duplicator chooses *k* positions in the other string, i.e., she chooses *k* positions $b_1, \ldots, b_k$ in *v*, if the Spoiler's move was in *u*, or she chooses *k* positions $a_1, \ldots, a_k$ in *u*, if the Spoiler's move was in *v*.

Again, the Duplicator wins iff the mapping $a_i \mapsto b_i$ (for $i = 1, \ldots, k$) is a partial isomorphism. Clearly, if the Duplicator has a winning strategy for the single-round *k*-move game on *u* and *v*, then she also has one for the single-round *h*-move game, for all $h \leqslant k$. By the standard argumentation (cf., e.g., the textbooks [12,16]), one obtains that this game characterises the expressive power of $BC(\Sigma_1[\mathcal{N}])$:

Theorem 2.2. *A language $L \subseteq A^*$ is definable in $BC(\Sigma_1[\mathcal{N}])$ iff there is a finite subset $\mathcal{N}'$ of $\mathcal{N}$ and a number k such that, for every $u \in L$, $v \notin L$, the Spoiler has a winning strategy in the single-round k-move game for $\mathcal{N}'$ on u and v.*

3. The Crane Beach Conjecture

3.1. Formulation of the CBC

As already mentioned in Section 1, first-order logic with varying *numerical predicates* can be thought of as specifying the circuit complexity class AC^0 with varying *uniformity conditions*. Roughly speaking, *simple* built-in predicates lead to circuit families that are easily computable, whereas *involved* built-in predicates may lead to circuit families that are difficult to compute, if computable at all. In [6] it was shown that $FO[\leqslant, Bit]$ and, equivalently, $FO[\leqslant, +, \times]$ precisely characterises *logtime-uniform* AC^0. Furthermore, *non-uniform* AC^0 is characterised by $FO[\leqslant, \mathcal{ARB}]$, where $\mathcal{ARB}$ is the class of *arbitrary*, i.e. all, numerical predicates.

Intuitively, since numerical predicates can only talk about *positions* in strings, it seems that they can only help to express properties that depend on certain (combinations of) letters appearing in certain (combinations of) positions. The CBC (named after the location of its first, flawed, proof) is an attempt to make that intuition precise.

Definition 3.1 (*Neutral letter*). Let $L \subseteq A^*$. A letter $e \in A$ is called *neutral* for *L* iff inserting or deleting *e*'s in any string in A^* does not change its membership or non-membership in *L*. Precisely, this means that for any $u, v \in A^*$ it holds that $uv \in L \iff uev \in L$.

For example, the letter 0 is a neutral letter of the language PARITY, consisting of exactly those {0, 1}-strings in which the letter 1 occurs an even number of times. A deep result of [1,15] states that PARITY is not definable in $FO[\leqslant, \mathcal{ARB}]$.

Membership in a language with a neutral letter cannot depend on the individual positions on which letters are: any letter can be moved away from any position by insertion or deletion of neutral letters. It seems therefore conceivable that for every such language, if it can be defined at all in first-order logic then it can be defined using the linear order as the only numerical predicate. With this intuition, the following conjecture seems plausible:

Definition 3.2 (*Crane Beach conjecture*). Let $\mathcal{N}$ be a set of numerical predicates. We say that the CBC is true for $FO[\leqslant, \mathcal{N}]$, iff for every alphabet A and every neutral letter language $L \subseteq A^*$, the following is true: If L is definable in $FO[\leqslant, \mathcal{N}]$, then L is already definable in $FO[\leqslant]$.

The CBC for any logic F other than FO is defined in the analogous way, replacing FO with F in the above definition. In other words: The CBC is true for $F[\leqslant, \mathcal{N}]$ iff $F[\leqslant]$ can define all neutral letter languages that are definable in $F[\leqslant, \mathcal{N}]$.

As we will see in the subsequent sections, the CBC is true for some cases and false for others. A summary of what is known about the CBC is given in Fig. 2 at the end of this paper.

3.2. An EF-game approach to the CBC

We now present a general methodology for EF-game lower bound proofs for languages with neutral letters. Given a set $\mathcal{N}$ of numerical predicates, our goal will be to prove that the CBC is true for $FO[\leqslant, \mathcal{N}]$. I.e., given an arbitrary neutral letter language L that is *not* definable in $FO[\leqslant]$, show that it is not definable in $FO[\leqslant, \mathcal{N}]$, either.

From Theorem 2.1 we know that for each number r of rounds, we can find strings $u \in L$ and $v \notin L$ such that the Duplicator wins the r-round game for $\{\leqslant\}$ on u and v. This game will henceforth be called the "*small game*".

Since L has a neutral letter we can assume, without loss of generality, that u and v have the same length. (If not, append u with $2^r + |v|$ neutral letters e and append v with $2^r + |u|$ neutral letters e. It is straightforward to see that the Duplicator wins the r-round game for $\{\leqslant\}$ on these padded versions of u and v.)

To show that L is not definable in $FO[\leqslant, \mathcal{N}]$ it suffices to construct (cf. Theorem 2.1), for each *finite* subset $\mathcal{N}'$ of $\mathcal{N}$, and for each number k of rounds, two strings $U \in L$ and $V \notin L$ such that the Duplicator wins the k-round game for $\{\leqslant\} \cup \mathcal{N}'$ on U and V. This game will henceforth be called the "*big game*".

The basic plan for the construction of U and V is as follows: Given $\mathcal{N}'$ and k, choose an appropriate number $r(k)$ of rounds for the "small game". Furthermore, choose strings $u \in L$ and $v \notin L$ of the same length, such that the Duplicator wins the $r(k)$-round game for $\{\leqslant\}$ on u and v. Afterwards, choose a suitable sequence $p_1 < p_2 < p_3 < \cdots$ of natural numbers, and move the letters of u and v onto the positions $p_1, p_2, \ldots$. Precisely, if $u = u_1 \cdots u_m$, then U is the string with input positions $1, \ldots, n$ (for some $n \geqslant p_m$), where the positions $p_1 < p_2 < \cdots < p_m$ carry the letters $u_1, u_2, \ldots, u_m$, and where all other positions carry the neutral letter e. In the analogous way, the string V is obtained from v.

Since e is a neutral letter for L, we know that $U \in L$ and $V \notin L$.

We have available a winning strategy for the Duplicator in the $r(k)$-round "small game" on *u* and *v*, and we want to find a winning strategy for the Duplicator in the *k*-round "big game" on *U* and *V*. To this end, we translate each move of the Spoiler in the "big game" into a number of moves for a "virtual Spoiler" in the "small game". Then we can find the answers of a "virtual Duplicator" playing according to her winning strategy in the "small game". Afterwards, we translate these answers into a move for the Duplicator in the "big game". Finally, this will give us a winning strategy for the Duplicator in the "big game" on *U* and *V*; and altogether, this will show that *L* is not definable in $FO[\leqslant, \mathcal{N}]$.

This methodology of translating a winning strategy for the "small game" into a winning strategy for the "big game" will be used for proving some of the positive instances of the CBC presented in the following section.

4. Cases where the CBC is true

In this section we present all cases where the CBC is known to be true.

4.1. The CBC for the class $\mathcal{MON}$ of monadic numerical predicates

Theorem 4.1. *Let $\mathcal{MON}$ be the class of all monadic* (*i.e.*, *unary*) *numerical predicates*. *The CBC is true for $FO[\leqslant, \mathcal{MON}]$.*

Proof. Let *L* be a language with a neutral letter that is not definable in $FO[\leqslant]$. As explained in Section 3.2, this means that for any number *k* of rounds there must be two strings $u \in L$ and $v \notin L$ of the same length *m*, such that the Duplicator wins the *k*-round game for $\{\leqslant\}$ on *u* and *v* (the "*small game*").

Now let $\mathcal{N}$ be any finite set of monadic predicates. We will show that *L* is not definable in $FO[\leqslant, \mathcal{N}]$ as follows. We will use $\mathcal{N}$ to construct two strings $U \in L$ and $V \notin L$ from *u* and *v* by a suitable padding with neutral letters. The length of *U* and *V* will be a suitably large number *n* to be defined below. Then we will show how the Duplicator can win the *k*-round game for $\{\leqslant\} \cup \mathcal{N}$ on *U* and *V*.

The construction of U and V. For every $n \in \mathbb{N}$, the predicates in $\mathcal{N}$ may be regarded as a *coloring* of the input positions from 1 to *n*, with finitely many colors. If $r < s$ are input positions, consider the colored string given by the interval from *r* to *s*, with each input position holding a neutral letter. For any two such colored strings, consider the *k*-round game for $\{\leqslant\}$. Let two strings be considered equivalent iff the Duplicator wins this game on them. Of course (cf., e.g., [16, Exercise 6.11]) there are only a *finite number* of equivalence classes.

We now define a colored undirected graph whose vertices are these *n* input positions and where the color of the edge from position *r* to position *s* represents the equivalence class of the colored string for that interval. By the Erdös–Szekeres Theorem[5] [13], as long as *n* is greater than m^d where *d* is the number of edge colors, there must be a *monochromatic path* of length at least *m*. We create *U* from *u*, and *V* from *v*, by placing the letters of the shorter strings in the locations given by the vertices of this monochromatic path (the "*special locations*"), and making all other letters neutral. We must now explain how the Duplicator can win the *k*-round game for $\{\leqslant\} \cup \mathcal{N}$ on *U* and *V* (the "*big game*").

[5] Or, almost equivalently, the finite version of the Ramsey Theorem (cf., e.g., the textbook [11]).

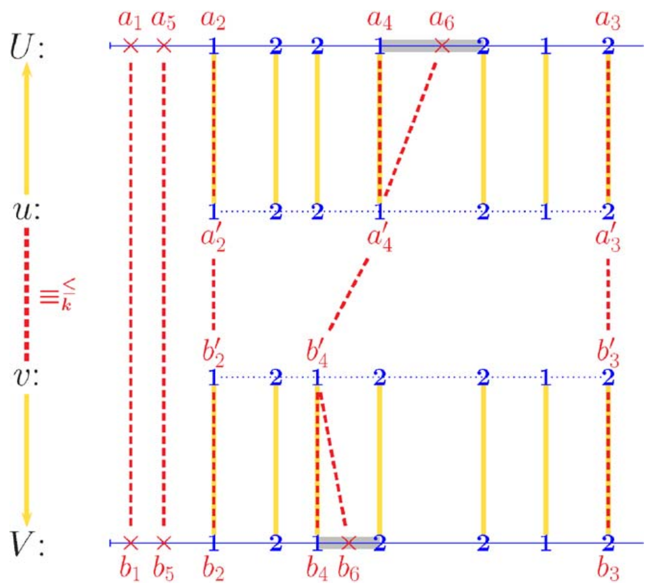

Fig. 1. Visualization of the Duplicator's strategy in the proof of Theorem 4.1 with $u = 1221212$ and $v = 1212212$. U and V are padded versions of u and v; (multiple) occurrences of the letter e are visualized by a solid line between the letters 1 and 2.

The Duplicator's strategy for the "big game": The Duplicator will model the "big game" by a series of "small games", where she already has a winning strategy for each. One small game is played on the strings u and v using only $\leqslant$, and there is another small game (using $\leqslant$ and color only) for each interval between special locations. Whenever the Spoiler moves in the "big game", the Duplicator translates this move into the "u–v small game" by moving to the position matching the next special position to the right. She also translates it into the "small game for that interval". The Duplicator's reply in the "big game" is determined by her correct move in the "u–v small game", and her correct move in the "small game for that particular interval". An illustration of this strategy is given in Fig. 1.

After k rounds the Duplicator must win the original "u–v small game" and all the "interval small games", as she has made at most k moves in each. It is easy but tedious to look at the input letters, order, equality, and monadic predicates of $\mathcal{N}$ in the "big game" and verify that the Duplicator has won that as well. $\square$

4.2. The CBC for the class $\mathcal{ARB}$ of arbitrary numerical predicates

We can use Theorem 4.1, together with a result of [4], to derive the following interesting generalization of the nonexpressibility of PARITY. Note, however, that we do not get an *independent* proof of the nonexpressibility of PARITY, because the existing proofs are used crucially to obtain the results in [4].

Corollary 4.2. *The CBC for* $FO[\leqslant, \mathcal{ARB}]$ *holds for all regular languages. That is, for the set* $\mathcal{ARB}$ *of arbitrary* (*i.e., all*) *numerical predicates and for every regular language* L *that has a neutral letter, it is true that* L *is definable in* $FO[\leqslant, \mathcal{ARB}]$ *if and only if it is definable* $FO[\leqslant]$.

Proof. This follows from Theorem 4.1 and the fact, proven in [4], that every regular language definable in $FO[\leqslant, \mathcal{ARB}]$ is also definable in $FO[\leqslant, \{mod_p \ / \ p \in \mathbb{N}\}]$, where $mod_p(i)$ is true iff $i \equiv 0 \bmod p$. □

Instead of restricting attention to *regular* languages, we can also restrict attention to languages over a *two-letter alphabet*:

Theorem 4.3. *The CBC for* $FO[\leqslant, \mathcal{ARB}]$ *holds for all languages over a two-letter alphabet. That is, if* $|A| = 2$, *then every language* $L \subseteq A^*$ *with a neutral letter that is definable in* $FO[\leqslant, \mathcal{ARB}]$ *is also definable in* $FO[\leqslant]$.

Proof. Let L be a language on $\{1, e\}$ with e as a neutral letter. Consider the set S_L of numbers n such that 1^n is in L and 1^{n+1} is not. If S_L is finite, then it is easy to see that L is regular and thus definable in $FO[\leqslant]$. We will show that if S_L is infinite, then L is not definable in $FO[\leqslant, \mathcal{A}RB]$.

Assume, then, that L is defined by a formula in $FO[\leqslant, \mathcal{A}RB]$. By Barrington et al. [6], it follows that L is recognized by a family of unbounded fan-in circuits with constant depth and polynomial size. Let n be a number in S_L and consider the circuit that decides L on inputs of length $2n$. This circuit computes a symmetric function of its inputs.

Fagin et al. [14] defined a "measure function" $\mu_{\mathcal{F}}$ for any family $\mathcal{F}$ of symmetric Boolean functions, where $\mu_{\mathcal{F}}(n)$ is the minimum number of inputs that must be fixed to make the n-bit function in $\mathcal{F}$ a constant function. We see that if we let $\mathcal{F}$ be the characteristic function of L, then $\mu_{\mathcal{F}}(2n) \geqslant n - 1$ for every n in S_L. But they proved that $\mathcal{F}$ is recognized by a family of constant-depth polynomial-size circuits only if $\mu_{\mathcal{F}} = o(n^{\varepsilon})$ for every positive ε. (In fact, given later results, their proof shows that such circuits exist iff $\mu_{\mathcal{F}}$ is polylogarithmic.) So if S_L is infinite, the circuit family cannot exist. □

Since PARITY is a non-star-free regular language over $\{0, 1\}^*$ and has a neutral letter, Theorem 4.3 implies the nonexpressibility of PARITY in first-order logic with arbitrary numerical predicates (i.e., non-uniform AC^0). Note, however, that our proof of Theorem 4.3 directly uses the existing proofs of the nonexpressibility of PARITY.

Rather than restricting the *languages* under consideration, we can also consider special cases of the CBC based on restrictions on the type of logical *formulas* allowed. For example, with arbitrary numerical predicates the conjecture does hold for Boolean combinations of Σ_1-formulas:

Theorem 4.4. *Let* $\mathcal{ARB}$ *be the set of arbitrary* (*i.e., all*) *numerical predicates. The CBC is true for* $BC(\Sigma_1[\leqslant, \mathcal{ARB}])$.

Proof. We must show that for any finite set $\mathcal{N}$ of numerical predicates and any language L with a neutral letter, L is definable in $BC(\Sigma_1[\leqslant, \mathcal{N}])$ if and only if it is definable in $BC(\Sigma_1[\leqslant])$.

Step 1: Using Theorem 2.2, we first show that the CBC for $BC(\Sigma_1[\mathcal{N}])$ is true for the special case $\mathcal{N} = \{\text{suc}, \text{min}, \text{max}\}$, where suc is the successor relation with $\text{suc}(p, q)$ iff $q = p+1$, $\langle w, p\rangle \models \min(p)$ iff $p=1$, and $\langle w, p\rangle \models \max(p)$ iff $p = |w|$.

Let *e* be the neutral letter, and assume that $L \notin BC(\Sigma_1[\leqslant])$. Then, for every *k*, there are strings $u \in L$, $v \notin L$ such that the Duplicator wins the single-round *k*-move game for $\{\leqslant\}$ on *u* and *v* (the "*small game*"). We can assume *u* and *v* to be of the same length *m* (if not, append $|v|+k$ *e*'s to *u* and $|u|+k$ *e*'s to *v*). We construct strings *U* from *u* and *V* from *v* such that $U \in L$, $V \notin L$, and the Duplicator wins the single-round *k*-move game for $\{\leqslant, \text{suc}, \text{min}, \text{max}\}$ on *U* and *V* (the "*big game*"). Then $L \notin BC(\Sigma_1[\leqslant, \text{suc}, \text{min}, \text{max}])$, which proves that the CBC is true for $BC(\Sigma_1[\leqslant, \text{suc}, \text{min}, \text{max}])$.

In order to construct *U*, insert $2k-1$ *e*'s between each pair of adjacent positions in *u*, as well as at the beginning and the end of *u*. More precisely, $U = U_1 \cdots U_{m2k+2k-1}$, with $U_{j2k} = u_j$, and $U_{j2k+i} = e$, for any $j \leqslant m$, $i < 2k$. Similarly, we construct *V* from *v*. Since *e* is neutral, we have $U \in L$, $V \notin L$.

For the "big game", assume that the Spoiler chooses positions $a_1, \ldots, a_k$ in *U* (the other case is symmetric). Some (possibly all, or none) of the U_{a_j} will be neutral letters, others will be from $A \setminus \{e\}$. For the sake of notational simplicity we will assume, without loss of generality, that $U_{a_1}, \ldots, U_{a_q} \in A \setminus \{e\}$, and $U_{a_{q+1}} = \cdots = U_{a_k} = e$. Then each a_j with $j \leqslant q$ is of the form $s_j 2k$, for some $s_j \in \{1, \ldots, m\}$. Now the Duplicator simulates a move of a "virtual Spoiler" in the "small game" on *u* and *v*, in which the "virtual Spoiler" pebbles $s_1, \ldots, s_q$ on *u*, and finds her reply, $s'_1, \ldots, s'_q$ on *v*, according to her winning strategy in the "small game". She then sets, for each *j* from 1 through *q*, b_j to be $s'_j 2k$. Then for each $j, l \leqslant q$ it holds that

- $b_j \neq b_l+1$ and $a_j \neq a_l+1$,
- $b_j \leqslant b_l \iff a_j \leqslant a_l$, and
- $V_{b_j} = v_{s'_j} = u_{s_j} = U_{a_j}$.

To complete this move, the Duplicator has to define $b_{q+1}, \ldots, b_k$ such that $V_{b_{q+1}} = \cdots = V_{b_k} = e$, and that for all $j, l \leqslant k$

- $b_j \leqslant b_l \iff a_j \leqslant a_l$,
- $b_j = b_l+1 \iff a_j = a_l+1$, and
- $b_j = 1 \iff a_j = 1$ and $b_j = |V| \iff a_j = |U|$.

Such $b_{q+1}, \ldots, b_k$ can easily be found, since between any two different b_j, b_l with $j, l \leqslant q$, there are at least $2k-1$ positions *p* where $V_p = e$.

This completes the proof of Step 1.

Step 2: Now let $\mathcal{N}$ be an arbitrary finite set of numerical predicates and assume that $L \notin BC(\Sigma_1[\leqslant])$. From what we have shown in Step 1 it follows that, for every *k*, we can find strings $u \in L$, $v \notin L$ of the same length *m* such that the Duplicator has a winning strategy in the single-round $2k+2$-move game for $\{\leqslant, \text{suc}, \text{min}, \text{max}\}$ on *u* and *v*. From now on, this game will be called the "small game".

We want to construct strings *U* and *V* by inserting neutral letters into *u* and *v*, respectively, in such a way that the original letters of *u* and *v* are moved onto positions $p_1, \ldots, p_m$ which are, in a certain sense, highly indistinguishable. To this end, we define, for every number *n*, a coloring of subsets of size $h \leqslant 2k$ of $\{1, \ldots, n\}$. This coloring was inspired by the one used by Straubing in [31], in his proof of Theorem 8. There he used the following extension of Ramsey's theorem, which will also help us here:

Theorem 4.5. *Let* $m, K, c_1, \ldots, c_k > 0$, *with* $K \leqslant m$. *Let n be sufficiently large as a function of m and the c's. If all h-element subsets of* $\{1, \ldots, n\}$, *with* $1 \leqslant h \leqslant K$, *are colored from a set of* c_h *colors, then* *there exists an m-element subset P of* $\{1, \ldots, n\}$ *such that for each* $h \in \{1, \ldots, K\}$ *there exists a color* κ_h *such that all h-element subsets of P are colored* κ_h.

For every $h \leqslant 2k$ let $\mathcal{T}_h = \{\tau_1, \ldots, \tau_q\}$ be the set of all atomic $\{\leqslant\} \cup \mathcal{N}$-formulas on the variables $x_1, \ldots, x_k, y_1, \ldots, y_h$. The $\{\leqslant\} \cup \mathcal{N}$-*type* $\alpha(r, S)$ of a tuple $r = (r_1, \ldots, r_k) \in \{1, \ldots, n\}^k$ with respect to an h-element set $S = \{s_1 < \cdots < s_h\}$ is the set of all those formulas of $\mathcal{T}_h$ that are satisfied when x_i is interpreted as r_i, and y_j as s_j, for $i \leqslant k$ and $j \leqslant h$.

We now color, for each number n and every $h \leqslant 2k$, every h-element set $S = \{s_1 < \cdots < s_h\} \subseteq \{1, \ldots, n\}$ with the set of all those $\alpha \subseteq \mathcal{T}_h$ for which there is a k-tuple r over $\{1, \ldots, n\}$ such that r has $\{\leqslant\} \cup \mathcal{N}$-type α with respect to S. Clearly, for every $h \leqslant 2k$ there is a fixed (finite) number of possible colors, independent of n. The extension of Ramsey's theorem stated above tells us that for large enough n we can find numbers $p_1 < \cdots < p_m \leqslant n$ such that, for every $h \leqslant 2k$, all h-element subsets of $\{p_1, \ldots, p_m\}$ have the same color. We now insert neutral letters into u in such a way that in the resulting string U we have $U_{p_i} = u_i$, for $i = 1, \ldots, m$, and $U_i = e$ for all $i \notin \{p_1, \ldots, p_m\}$. In the same way we construct V from v. Let us call $p_1, \ldots, p_m$ the "*special positions*".

We now show that the Duplicator has a winning strategy in the k-move game for $\{\leqslant\} \cup \mathcal{N}$ on U and V (the "*big game*"). Assume that the Spoiler chooses $a = a_1, \ldots, a_k$ in U (the other case is symmetric). Then the Duplicator finds, for every a_j the next smallest special position p_{i_j}, i.e., $p_{i_j} \leqslant a_j < p_{i_j+1}$; in the special case that $a_j < p_1$ we let $i_j := 1$; in the special case that $p_m \leqslant a_j$ we let $i_j := m$ and i_j+1 and p_{i_j+1} remain undefined. Let $S = \{p_{i_j},\ p_{i_j+1} \,/\, j = 1, \ldots, k\}$. The Duplicator now simulates a move of a "virtual Spoiler" in the single-round $2k+2$-move game for $\{\leqslant, \mathrm{suc}, \min, \max\}$ on u and v (the "*small game*"), in which the "virtual Spoiler" plays all the points i_j and i_j+1, for $j = 1, \ldots, k$ on u, as well as min and max. Using her winning strategy in this game, the "virtual Duplicator" finds a reply with which she wins the "small game". Therefore, we can safely call these points l_j, l_j+1, for $j = 1, \ldots, k$, and we know that $u_{i_j} = v_{l_j}$, for $j = 1, \ldots, k$. Let T be the set $\{p_{l_j}, p_{l_j+1} \,/\, j = 1, \ldots, k\}$. $|T| = |S| =: h \leqslant 2k$, so S and T have the same color, and this implies that there is a tuple $b = (b_1, \ldots, b_k)$ with the same $\{\leqslant\} \cup \mathcal{N}$-type as a. The Duplicator now puts her pebbles on $b_1, \ldots, b_k$ in V. We have to check the winning conditions. By construction, $\alpha(a, S) = \alpha(b, T)$. In particular, this implies that

- $(a_1, \ldots, a_k)$ and $(b_1, \ldots, b_k)$ have the same $\{\leqslant\} \cup \mathcal{N}$-type,
- $a_j \leqslant a_{j'} \iff b_j \leqslant b_{j'}$, for all j, j',
- if $a_j = p_{i_j}$ then $b_j = p_{l_j}$, and hence $U_{a_j} = u_{i_j} = v_{l_j} = V_{b_j}$. If a_j is not of this form then $p_{i_j} < a_j < p_{i_j+1}$ (or $a_j < p_1$ or $a_j > p_m$), and consequently, $p_{l_j} < b_j < p_{l_j+1}$ (or $b_j < p_1$ or $b_j > p_m$) and $U_{a_j} = V_{b_j} = e$ (note that this is the place where we need that the relations suc, min, max are present in the "small game").

This completes the proof of Step 2 and, altogether, the proof of Theorem 4.4. □

A recent result of Krol and Lautemann [17] proves another positive case of the CBC. For every string-language L that has a neutral letter, the following is true: If L is definable in $BC(\Sigma_1^{(s,p)}[\leqslant, \mathcal{ARB}])$ then L is also definable in $BC(\Sigma_1^{(s,p)}[\leqslant, \mathrm{suc}, \min, \max])$. Here, suc is the successor relation, and $\Sigma_1^{(s,p)}$ is the class of all formulas of the form $\exists^{(i,s,p)}(x_1, \ldots, x_k)\,\psi$, where $i, s, p, k \geqslant 0$ and ψ is quantifier free. Basically, such a formula expresses that the number of tuples $(x_1, \ldots, x_k)$ that satisfy ψ is congruent i

modulo (s, p), where, by definition, a number j is congruent i modulo (s, p) if and only if $(j = i \leqslant s)$ or $(j > s \text{ and } j \equiv i \bmod p)$.

4.3. The CBC with "+" as numerical predicate

As we will see in Section 5, with addition and multiplication first-order logic has enough expressive power to defeat the neutral letter. Addition alone is, in many ways much weaker than addition and multiplication together. For example, this is witnessed by the fact that the first-order theory of the natural numbers with $+$ and $\times$ is undecidable, whereas Presburger arithmetic, the first-order theory of the natural numbers with addition only, can be decided using quantifier elimination (cf., e.g., the textbook [30]). It is therefore more than conceivable that addition alone is too weak to make the conjecture fail, and we now show that this is indeed the case.

Theorem 4.6. *The CBC is true for* $FO[\leqslant, +]$, *where* $+$ *is the ternary numerical predicate which, in every universe* $\{1, \dots, n\}$, *is interpreted by the graph of the addition function*.

As indicated in the introduction, this theorem follows from collapse results for first-order queries over finite databases; in Section 4.4 we will concentrate on the correspondence between the CBC and collapse results in database theory in detail. However, the terminology in which these results (and their proofs) are formulated in the literature is rather alien to our setting here. Thus, in the following, we also will give a brief sketch of a direct, Ehrenfeucht–Fraïssé game proof of Theorem 4.6.

For simplicity, we concentrate on 0–1-strings U and V of the same (large) size and discuss what the Duplicator has to do in order to win the k-round game for $\{\leqslant, +\}$ on U and V. Let A be the set of indices a for which $U_a = 1$, similarly, $B = \{b \,/\, V_b = 1\}$. As in previous proofs, we will work with a set $P = \{p_1 < \cdots < p_m\}$ of indistinguishable positions, and choose U and V such that $A, B \subseteq P$.

Assume that, after $i-1$ rounds $a_1, \dots, a_{i-1}$ have been played in U, and $b_1, \dots, b_{i-1}$ in V. Let the Spoiler choose some element a_i in U. When choosing b_i in V, the Duplicator has to make sure that any of the Spoiler's moves for the remaining $k-i$ rounds in one structure can be matched in the other. In particular, this means that any sum over the a_j behaves in relation to A exactly as the corresponding sum over the b_j behaves in relation to B. For instance, for any sets $J, J' \subseteq \{1, \dots, i\}$, it should hold that there is some $a \in A$ that lies between $\sum_{j \in J} a_j$ and $\sum_{j' \in J'} a_{j'}$ if and only if there is some $b \in B$ that lies between $\sum_{j \in J} b_j$ and $\sum_{j' \in J'} b_{j'}$. But it is not enough to consider simple sums over previously played elements. Since with $O(r)$ additions it is possible to generate $s \cdot a_i$ from a_i, for any $s \leqslant 2^r$, we also have to consider linear combinations with coefficients as large as this. Furthermore, since the Spoiler is allowed to choose either structure to move in each time, it is necessary to deal with even more complex linear combinations. One can only handle all these complications because, as the game progresses, the number of rounds left for the Spoiler to do all these things decreases. This means, for instance, that the coefficients and the length of the linear combinations we have to consider decrease: after the last round, the only relevant linear combinations are simple additions of chosen elements.

All the technical details necessary to make this strategy work are worked out in [24] in order to prove that for each $FO[\leqslant, +]$-formula φ there is a set $Q \subseteq \mathbb{N}$ such that φ cannot distinguish between subsets of Q if they are of equal cardinality, or both large enough. Drawing on Lynch's theorem, in [21] the authors prove a theorem, which, specialised to our setting can be formulated as follows: For every $k \in \mathbb{N}$ there exists a number $r(k) \in \mathbb{N}$ and an order-preserving mapping $q : \mathbb{N} \to \mathbb{N}$ such that, for every (finite) signature σ the following holds: If σ' and σ'' are interpretations of σ over $\mathbb{N}$, and if $m', m'' \in \mathbb{N}$ such that the Duplicator has a winning strategy in the $r(k)$-round EF-game on $\langle \mathbb{N}, \sigma', m', \leqslant \rangle$ and $\langle \mathbb{N}, \sigma'', m'', \leqslant \rangle$, then the Duplicator also has a winning strategy in the k-round EF-game on $\langle \mathbb{N}, q(\sigma'), q(m'), \leqslant, + \rangle$ and $\langle \mathbb{N}, q(\sigma''), q(m''), \leqslant, + \rangle$. Here, $q(\sigma')$ is short for $\{q(R') \,/\, R \in \sigma\}$, where $q(R') = \{\big(q(i_1), \ldots, q(i_l)\big) \,/\, (i_1, \ldots, i_l) \in R'\}$. This result was further generalized in [25] to the following:

Theorem 4.7. *There is an infinite set $Q \subseteq \mathbb{N}$ such that for every finite collection $\mathcal{N}$ of subsets of Q and for every (finite) signature σ and every $k \in \mathbb{N}$ there exists a number $r(k) \in \mathbb{N}$ and an order-preserving mapping $q : \mathbb{N} \to \mathbb{N}$ such that the following is true: If σ' and σ'' are interpretations of σ over $\mathbb{N}$, and if $m', m'' \in \mathbb{N}$ such that the Duplicator has a winning strategy in the $r(k)$-round EF-game on the structures $\langle \mathbb{N}, \sigma', m', \leqslant \rangle$ and $\langle \mathbb{N}, \sigma'', m'', \leqslant \rangle$, then the Duplicator also has a winning strategy in the k-round EF-game on the structures $\langle \mathbb{N}, q(\sigma'), q(m'), \leqslant, +, \mathcal{N} \rangle$ and $\langle \mathbb{N}, q(\sigma''), q(m''), \leqslant, +, \mathcal{N} \rangle$.*

Using the above theorem, we can prove the following generalization of Theorem 4.6.

Theorem 4.8. *There is an infinite set $Q \subseteq \mathbb{N}$ such that the CBC is true for $FO[\leqslant, +, \mathcal{MON}_Q]$. Here, $\mathcal{MON}_Q = \{\hat{P} \,/\, P \subseteq Q\}$ where, in every universe $\{1, \ldots, n\}$, $\hat{P}$ is interpreted by $P \cap \{1, \ldots, n\}$.*

Proof. We follow the methodology described in Section 3.2 and use Theorem 4.7; in particular, Q is the set provided in that theorem.

Let A be an alphabet, let $L \subseteq A^*$ be a neutral letter language that is not definable in $FO[\leqslant]$, and let $\hat{\mathcal{N}}$ be a finite subset of $\mathcal{MON}_Q$. Our aim is to show that L is not definable in $FO[\leqslant, +, \hat{\mathcal{N}}]$.

In order to apply Theorem 4.7, let $\mathcal{N} := \{P \subseteq Q \,/\, \hat{P} \in \hat{\mathcal{N}}\}$, let $\sigma := \sigma_A$ be the signature associated with the alphabet A (cf. Section 2.1), and let $k \in \mathbb{N}$. Choose $r(k) \in \mathbb{N}$ and $q : \mathbb{N} \to \mathbb{N}$ according to Theorem 4.7.

Since L is not definable in $FO[\leqslant]$, there must be strings $u \in L$ and $v \notin L$ of the same length, m, such that the Duplicator has a winning strategy in the k-round game for $\{\leqslant\}$ on u and v. The strings u and v define σ-interpretations σ^u and σ^v, respectively, and the winning strategy of the Duplicator on u and v can easily be extended to the structures $\langle \mathbb{N}, \sigma^u, m, \leqslant \rangle$ and $\langle \mathbb{N}, \sigma^v, m, \leqslant \rangle$: If the Spoiler plays a position $a_i \leqslant m$ on $\langle \mathbb{N}, \sigma^u, m, \leqslant \rangle$, this corresponds to a move on u, and the Duplicator can choose her answer according to her winning strategy on v. If the Spoiler plays a position $a_i > m$ on $\langle \mathbb{N}, \sigma^u, m, \leqslant \rangle$, then the Duplicator replies with $b_i := a_i$. (The case where the Spoiler plays on $\langle \mathbb{N}, \sigma^v, m, \leqslant \rangle$ is completely symmetric.) Clearly, this defines a winning strategy for the Duplicator in the $r(k)$-round EF-game on the structures $\langle \mathbb{N}, \sigma^u, m, \leqslant \rangle$ and $\langle \mathbb{N}, \sigma^v, m, \leqslant \rangle$.

Application of Theorem 4.7 gives us a winning strategy for the Duplicator in the k-round EF-game on the structures $\langle \mathbb{N}, q(\sigma^u), q(m), \leqslant, +, \mathcal{N} \rangle$ and $\langle \mathbb{N}, q(\sigma^v), q(m), \leqslant, +, \mathcal{N} \rangle$. From this, we obtain a winning strategy for the Duplicator in the k-round game for $\{\leqslant, +\} \cup \hat{\mathcal{N}}$ on the strings U and V that are obtained from u and v by inserting neutral letters in such a way that the ith letter of u (resp. v) is placed onto the $q(i)$th position in U (resp. V): Every move of the Spoiler in U is translated into a move on $\langle \mathbb{N}, q(\sigma^u), q(m), \leqslant, +, \mathcal{N} \rangle$, and the Duplicator's reply on $\langle \mathbb{N}, q(\sigma^v), q(m), \leqslant, +, \mathcal{N} \rangle$ is translated back into a move on V. The winning condition of the Duplicator on $\langle \mathbb{N}, q(\sigma^u), q(m), \leqslant, +, \mathcal{N} \rangle$ and

$\langle \mathbb{N}, q(\sigma^v), q(m), \leqslant, +, \mathcal{N} \rangle$ directly translates into the winning condition for the Duplicator on U and V for $\{\leqslant, +\} \cup \hat{\mathcal{N}}$. Altogether, this proves Theorem 4.8. $\Box$

4.4. The CBC and collapse results in database theory

As already mentioned, so-called collapse results in database theory imply that the CBC is true for specific cases.

For a well-written concise survey on collapse results and database theory we refer to the paper [34]. A detailed and very recent overview of collapse results on finite databases is given in [22]. More information can also be found in the book [18].

A database can be viewed as a finite relational structure whose elements belong to some infinite universe $\mathbb{U}$ of "potential database elements". Sometimes, $\mathbb{U}$ also has additional, fixed relations such as a linear ordering $\leqslant$ and some further list of relations, $\mathcal{N}$. Such a structure $\langle \mathbb{U}, \leqslant, \mathcal{N} \rangle$ is called a *context structure*. A database schema can be viewed as a finite relational signature σ, whereas a finite database can be viewed as an interpretation σ' where all symbols in σ are interpreted by *finite* relations over $\mathbb{U}$ (such interpretations will be called *finite*). The *active domain* $adom(\sigma')$ is the set of all elements of $\mathbb{U}$ that occur in some tuple of some relation of σ'.

A database query (of a specific kind) can be modeled as a $FO[\sigma, \leqslant, \mathcal{N}]$-formula $\varphi(\overline{x})$. We write $\varphi(\sigma')$ to denote the evaluation of $\varphi(\overline{x})$ over σ', i.e., $\varphi(\sigma')$ is the set of all tuples $\overline{a}$ over $\mathbb{U}$ such that $\langle \mathbb{U}, \sigma', \leqslant, \mathcal{N} \rangle \models \varphi(\overline{a})$.

Basically, one speaks of a collapse result in database theory if the relations in $\mathcal{N}$ are not necessary to express queries of a specific kind. In the literature, various different kinds of collapse notions have been thoroughly investigated (cf., e.g., [22]). The specific collapse notion which perfectly fits to the CBC is fixed in the following definition.

Definition 4.9 (*locally generic; natural-generic collapse*). Let $\langle \mathbb{U}, \leqslant, \mathcal{N} \rangle$ be a context structure.

(a) Let σ be a finite relational signature. A $FO[\sigma, \leqslant, \mathcal{N}]$-formula $\varphi(\overline{x})$ is called *locally generic* iff for all finite interpretations σ' of σ over $\mathbb{U}$ the following is true: If q is a $\leqslant$-preserving mapping from $\mathrm{adom}\big(\sigma' \cup \{\varphi(\sigma')\}\big)$ to $\mathbb{U}$, then $q(\varphi(\sigma')) = \varphi(q(\sigma'))$.

(b) *FO*-logic admits the *natural-generic collapse* over $\langle \mathbb{U}, \leqslant, \mathcal{N} \rangle$ iff for every finite relational signature σ and every locally generic $FO[\sigma, \leqslant, \mathcal{N}]$-formula $\varphi(\overline{x})$ there exists a $FO[\sigma, \leqslant]$-formula $\psi(\overline{x})$ which is equivalent to $\varphi(\overline{x})$ on all structures $\langle \mathbb{U}, \sigma', \leqslant, \mathcal{N} \rangle$, for all finite interpretations σ' of σ over $\mathbb{U}$.

It is not difficult to see that the natural-generic collapse over a context structure whose universe is the set of natural numbers, implies a positive case of the CBC:

Proposition 4.10. *Let $\langle \mathbb{N}, \leqslant, \mathcal{N} \rangle$ be a structure. If FO-logic has the natural-generic collapse over $\langle \mathbb{N}, \leqslant, \mathcal{N} \rangle$, then the CBC is true for $FO[\leqslant, \hat{\mathcal{N}}]$, where $\hat{\mathcal{N}} := \{\hat{P} \,/\, P \in \mathcal{N}\}$.*

Proof. Let A be an alphabet, and let $L \subseteq A^*$ be a language with a neutral letter $e \in A$ that is definable in $FO[\leqslant, \hat{\mathcal{N}}]$. Our aim is to show that L is also definable in $FO[\leqslant]$.

Since L is definable in $FO[\leqslant, \hat{\mathcal{N}}]$, there is a first-order sentence φ over the signature $\sigma_A \cup \{\leqslant\} \cup \hat{\mathcal{N}}$ such that for all strings $w \in A^*$ we have

(I): $\quad w \in L \quad \text{iff} \quad \langle\{1, \ldots, |w|\}, \sigma_A^w, \leqslant, \hat{\mathcal{N}}\rangle \models \varphi.$

In order to apply the presumed natural-generic collapse result, we represent the string w by a finite database τ^w embedded in the context structure $\langle\mathbb{N}, \leqslant, \mathcal{N}\rangle$. The positions in w that carry non-neutral letters are exactly the active domain elements of τ^w, and τ^w contains a unary relation $\mathrm{Max}^w := \{|w|\}$ that consists of the maximum position of w. Precisely, we define the relational signature $\tau := (\sigma_A \setminus \{Q_e\}) \cup \{\mathrm{Max}\}$, and for a string w we define $\tau^w := (\sigma_A^w \setminus \{Q_e^w\}) \cup \{\mathrm{Max}^w\}$.

Let us now proceed with the proof of Proposition 4.10.

Step 1: We transform the given $FO[\sigma_A, \leqslant, \hat{\mathcal{N}}]$-sentence φ that defines L into a $FO[\tau, \leqslant, \mathcal{N}]$-sentence φ' such that for all strings $w \in A^*$ the following is true:

(II): $\quad \langle\mathbb{N}, \tau^w, \leqslant, \mathcal{N}\rangle \models \varphi' \quad \text{iff} \quad \langle\{1, \ldots, |w|\}, \sigma_A^w, \leqslant, \hat{\mathcal{N}}\rangle \models \varphi.$

The formula φ' is defined inductively via

- $\varphi' := \varphi \quad$ if φ is of the form $x<y$, $x=y$, or $Q_a(x)$, for some $a \in A \setminus \{e\}$
- $\varphi' := \bigwedge_{a\in A} \neg Q_a(x) \quad$ if $\varphi = Q_e(x)$
- $\varphi' := P(x_1, \ldots, x_k) \quad$ if $\varphi = \hat{P}(x_1, \ldots, x_k)$, for some predicate $\hat{P} \in \hat{\mathcal{N}}$
- $\varphi' := \neg\chi' \quad$ if $\varphi = \neg\chi$
- $\varphi' := \chi' \vee \zeta' \quad$ if $\varphi = \chi \vee \zeta$
- $\varphi' := \exists x\, \exists y\, (\mathrm{Max}(y) \wedge x \leqslant y \wedge \chi') \quad$ if $\varphi = \exists x\, \chi$.

It is straightforward to see that (II) is indeed true for all strings $w \in A^*$.

For technical reasons we combine φ' with a $FO[\tau]$-formula that is satisfied by a τ-structure $\langle\mathbb{N}, \tau'\rangle$ if and only if there exists a string $w \in A^*$ such that $\tau' = \tau^w$. From now on, the conjunction of this formula and the formula φ' will be called φ'.

Step 2: We show that φ' is locally generic. Let τ' be a finite interpretation of τ over $\mathbb{N}$, and let q be a $\leqslant$-preserving mapping from $adom(\tau')$ to $\mathbb{N}$. We have to show that

(III): $\quad \langle\mathbb{N}, \tau', \leqslant, \mathcal{N}\rangle \models \varphi' \quad \text{iff} \quad \langle\mathbb{N}, q(\tau'), \leqslant, \mathcal{N}\rangle \models \varphi'.$

Of course, if there is no string $w \in A^*$ such that $\tau' = \tau^w$, then both structures do *not* satisfy φ'. We therefore only have to consider the case where $\tau' = \tau^w$ for some string $w \in A^*$ of length, say, n. Let $i_1 < \cdots < i_m \leqslant n$ be exactly those positions in w that do *not* carry the letter e. Let $\tilde{n} := q(n)$. We define the string $\tilde{w}$ of length $\tilde{n}$ as follows: The positions $q(i_1) < \cdots < q(i_m)$ in $\tilde{w}$ carry the same letters as the positions $i_1 < \cdots < i_m$ in w, and all other positions in $\tilde{w}$ carry the neutral letter e. Since q is $\leqslant$-preserving, $\tilde{w}$ can be obtained from w by inserting or deleting e's, and therefore we have that $\tilde{w} \in L$ iff $w \in L$. From (I) and (II) it thus follows that $\langle\mathbb{N}, \tau^w, \leqslant, \mathcal{N}\rangle \models \varphi'$ iff $\langle\mathbb{N}, \tau^{\tilde{w}}, \leqslant, \mathcal{N}\rangle \models \varphi'$. This directly gives us (III), because by definition of $\tilde{w}$ we have $\tau^{\tilde{w}} = q\big(\tau^w\big)$.

Step 3: We use the presumed natural-generic collapse of FO-logic over $\langle\mathbb{N}, \leqslant, \mathcal{N}\rangle$. For the $FO[\tau, \leqslant, \mathcal{N}]$-sentence φ' this, in particular, gives us a $FO[\tau, \leqslant]$-sentence ψ' such that, for all $w \in A^*$,

(IV): $\quad \langle\mathbb{N}, \tau^w, \leqslant\rangle \models \psi' \quad \text{iff} \quad \langle\mathbb{N}, \tau^w, \leqslant, \mathcal{N}\rangle \models \varphi'.$

Step 4: We transform the $FO[\tau, \leqslant]$-sentence ψ' into a $FO[\sigma_A, \leqslant]$-sentence ψ such that the following is true for all strings $w \in A^*$:

(V): $\quad \langle\{1, \ldots, |w|\}, \sigma_A^w, \leqslant\rangle \models \psi \quad \text{iff} \quad \langle\mathbb{N}, \tau^w, \leqslant\rangle \models \psi'.$

For the sake of contradiction assume that ψ does not exist. Then, according to Theorem 2.1, for every $k \in \mathbb{N}$ there are strings u and v such that $\langle\mathbb{N}, \tau^u, \leqslant\rangle \models \psi'$, $\langle\mathbb{N}, \tau^v, \leqslant\rangle \not\models \psi'$, and the Duplicator has a winning strategy in the k-round game for $\{\leqslant\}$ on u and v. This strategy can easily be extended to the structures

$\langle \mathbb{N}, \tau^u, \leqslant \rangle$ and $\langle \mathbb{N}, \tau^v, \leqslant \rangle$: If the Spoiler plays a position $a_i \leqslant |u|$ on $\langle \mathbb{N}, \tau^u, \leqslant \rangle$, this corresponds to a move on u, and the Duplicator can choose her answer according to her winning strategy on v. If the Spoiler plays a position $a_i > |u|$ on $\langle \mathbb{N}, \tau^u, \leqslant \rangle$, then the Duplicator replies with $b_i := a_i - |u| + |v|$. (The case where the Spoiler plays on $\langle \mathbb{N}, \tau^v, \leqslant \rangle$ is completely symmetric.) Clearly, this defines a winning strategy for the Duplicator in the k-round EF-game on the structures $\langle \mathbb{N}, \tau^u, \leqslant \rangle$ and $\langle \mathbb{N}, \tau^v, \leqslant \rangle$. However, this contradicts the presumption that $\langle \mathbb{N}, \tau^u, \leqslant \rangle \models \psi'$ and $\langle \mathbb{N}, \tau^v, \leqslant \rangle \not\models \psi'$, and therefore the formula ψ must exist.

Step 5: From (I), (II), (IV), and (V) we obtain, for all strings $w \in A^*$, that $w \in L$ if and only if w satisfies ψ. In other words: We have shown that every language L that has a neutral letter and that is definable in $FO[\leqslant, \hat{\mathcal{N}}]$, is also definable in $FO[\leqslant]$. Hence, the proof of Proposition 4.10 is complete. □

Various different conditions on the context structure $\langle \mathbb{U}, \leqslant, \mathcal{N} \rangle$ are known which imply the natural-generic collapse (see, e.g., [8,7,3]). The most general of these conditions known by now (see [22]) is the notion of *finite Vapnik–Chervonenkis dimension* (finite VC-dimension, for short; also known as the *lack of the independence property*). For the sake of completeness, we state the precise definition of *finite VC-dimension*, basically taken from [9]:

Definition 4.11 (*Finite VC-Dimension*). Let $\langle \mathbb{U}, \leqslant, \mathcal{N} \rangle$ be a context structure.

(a) Let $\varphi(\overline{x}, \overline{y})$ be a $FO[\leqslant, \mathcal{N}]$-formula, and let $n_{\overline{x}}$ and $n_{\overline{y}}$ be the lengths of the tuples $\overline{x}$ and $\overline{y}$, respectively.
- For every $\overline{a} \in \mathbb{U}^{n_{\overline{y}}}$ the formula $\varphi(\overline{x}, \overline{y})$ defines the relation $R_{\varphi(\overline{x}, \overline{a})} := \{ \overline{b} \in \mathbb{U}^{n_{\overline{x}}} \ / \ \langle \mathbb{U}, \leqslant, \mathcal{N} \rangle \models \varphi(\overline{b}, \overline{a}) \}$.
- The formula $\varphi(\overline{x}, \overline{y})$ defines the following family of relations on $\mathbb{U}$: $F_{\varphi(\overline{x}, \overline{y})} := \{ R_{\varphi(\overline{x}, \overline{a})} \ / \ \overline{a} \in \mathbb{U}^{n_{\overline{y}}} \}$.
- A set $B \subseteq \mathbb{U}^{n_{\overline{x}}}$ is *shattered* by $F_{\varphi(\overline{x}, \overline{y})}$ iff $\{B \cap R \ / \ R \in F_{\varphi(\overline{x}, \overline{y})}\} = \{X \ / \ X \subseteq B\}$, i.e., for every $X \subseteq B$ there is a $\overline{a}_X \in \mathbb{U}^{n_{\overline{y}}}$ such that for all $\overline{b} \in B$ we have $\overline{b} \in X$ iff $\langle \mathbb{U}, \leqslant, \mathcal{N} \rangle \models \varphi(\overline{b}, \overline{a}_X)$.
- The family $F_{\varphi(\overline{x}, \overline{y})}$ has *finite* VC-*dimension* iff there exists a number $m_{\varphi(\overline{x}, \overline{y})} \in \mathbb{N}$ such that the following is true for all $B \subseteq \mathbb{U}^{n_{\overline{x}}}$:
  If B is shattered by $F_{\varphi(\overline{x}, \overline{y})}$, then $|B| \leqslant m_{\varphi(\overline{x}, \overline{y})}$.

(b) $\langle \mathbb{U}, \leqslant, \mathcal{N} \rangle$ has *finite VC-dimension* iff $F_{\varphi(\overline{x}, \overline{y})}$ has finite VC-dimension, for every $FO[\leqslant, \mathcal{N}]$-formula $\varphi(\overline{x}, \overline{y})$.

According to [22], the following deep result of [3] is the most general *natural-generic collapse* theorem that is known by now.

Theorem 4.12 (*Baldwin, Benedikt*). *If $\langle \mathbb{U}, \leqslant, \mathcal{N} \rangle$ is a context structure that has* finite VC-dimension, *then FO-logic has the natural-generic collapse over $\langle \mathbb{U}, \leqslant, \mathcal{N} \rangle$.*

Together with Proposition 4.10 this directly implies the following:

Corollary 4.13. *If a structure $\langle \mathbb{N}, \leqslant, \mathcal{N} \rangle$ has finite VC-dimension, then the CBC is true for FO$[\leqslant, \hat{\mathcal{N}}]$, where $\hat{\mathcal{N}} := \{\hat{P} \ / \ P \in \mathcal{N}\}$.*

As can be seen from Definition 4.11 it is, a priori, not at all trivial to check whether a given context structure has finite VC-dimension. Considering the CBC, context structures with universe $\mathbb{N}$ of natural numbers are of particular interest. The examples of [3,9,22] of such structures that have finite VC-dimension are $\langle \mathbb{N}, \leqslant, + \rangle$ and $\langle \mathbb{N}, \leqslant, \mathcal{MON} \rangle$, where $\mathcal{MON}$ is the class of all subsets of $\mathbb{N}$. Via Corollary 4.13 one therefore directly obtains Theorem 4.6 and a weaker version of Theorem 4.1 (recall that in Theorem 4.1 a monadic numerical predicate P may have a different interpretation for each universe size n, e.g., P may be interpreted as the set of all prime numbers for even n and as the set of all square numbers for odd n).

A recent result of Michael Taitslin [33] shows that *FO*-logic has the natural-generic collapse over $\langle \mathbb{N}, \leqslant, +, R_{2^{(\lfloor \lg x \rfloor^2)}} \rangle$, where

$$R_{2^{(\lfloor \lg x \rfloor^2)}} \;:=\; \left\{ \left(x \, , \, 2^{(\lfloor \lg x \rfloor^2)} \right) \;/\; x \in \mathbb{N} \right\}.$$

Via Proposition 4.10 one therefore obtains

Corollary 4.14. *The CBC is true for* $FO[\leqslant, +, \hat{R}_{2^{(\lfloor \lg x \rfloor^2)}}]$.

It is a further task to find more context structures with universe $\mathbb{N}$ for which *FO*-logic admits the natural-generic collapse.

Let us mention that we do not know if the converse of Proposition 4.10 is true, i.e., if a non-collapse in database theory implies that the CBC is false. The main obstacle is that a $FO[\leqslant, \mathcal{N}]$-formula that causes the non-collapse in database theory may quantify over the entire context universe $\mathbb{N}$, whereas the $FO[\leqslant, \hat{\mathcal{N}}]$-formulas that are relevant for the CBC, can only quantify over initial segments of $\mathbb{N}$. To see the enormous power gained by quantification over all of $\mathbb{N}$, recall that $FO[\leqslant, +, \times]$ on $\mathbb{N}$ can, e.g., express all semi-decidable problems (in fact, it can express the whole arithmetic hierarchy), whereas $FO[\leqslant, +, \times]$ *on initial segments* of $\mathbb{N}$ can express only properties in logtime-uniform AC^0. In particular, PARITY is definable in $FO[\leqslant, +, \times]$ on $\mathbb{N}$, but not in $FO[\leqslant, +, \times]$ on initial segments of $\mathbb{N}$.

5. Cases where the CBC is false

In this section we concentrate on cases where the CBC for is false for first-order logic. In fact, the conjecture fails for the set $\mathcal{N} = \{+, \times\}$, where $+$ and $\times$ are the ternary numerical predicates which, in every universe $\{1, \ldots, n\}$, are interpreted by the graphs of the addition function and the multiplication function, respectively. The set $\{+, \times\}$ of numerical predicates is particularly important because $FO[\leqslant, +, \times]$ corresponds to the most natural uniform version of the circuit complexity class AC^0 (cf., [6]).

Our counterexample to the CBC makes use of the well-known but somewhat counterintuitive ability of $FO[\leqslant, +, \times]$-formulas to *count* letters up to numbers polylogarithmic in the input size:

Definition 5.1 (*Definibility of Counting*). Let $f(n) \leqslant n$ be a nondecreasing function from $\mathbb{N}$ to $\mathbb{N}$. We say that a logical system can count up to $f(n)$ if there is a formula φ such that for every n and for every string $w \in \{0, 1\}^n$,

$$w \models \varphi(c) \iff c \leqslant f(n) \text{ and } c = \#_1(w),$$

where $\#_1(w)$ is the number of ones in w.

We will need to consider two functions with similar notation. We write the base-two logarithm of n as $\lg n$, the kth power of this logarithm as $(\lg n)^k$, and the kth *iterated* logarithm as $\lg^{(k)}(n)$. For example, $\lg^{(2)}(n)$ is the same as $\lg(\lg n)$. The counting capability of $FO[\leqslant, +, \times]$ can be formulated as follows:

Theorem 5.2 (*Ajtai and Ben-Or [2], Fagin et al. [14], Denenberg et al. [10], Wegener et al. [35]*). *The system $FO[\leqslant, +, \times]$ can count up to $(\lg n)^k$ for any k.*

(*However, if $f(n) = (\lg n)^{\omega(1)}$, and $\mathcal{N}$ is any set of numerical predicates, then $FO[\leqslant, \mathcal{N}]$ cannot count up to $f(n)$.*)

The polylogarithmic counting capability of $FO[\leqslant, +, \times]$ is essentially used to prove the following:

Theorem 5.3. *The CBC is false for $FO[\leqslant, +, \times]$.*

Proof. We define a language L on alphabet $\{0, 1, a\}$ as follows. For each positive integer k, L will contain a string consisting of the 2^k binary strings of length k, in order, separated by a's. The total length of the kth string in L is thus $2^k \cdot (k+1) - 1$. The first three strings in L are thus $0a1$, $00a01a10a11$, and

$$000a001a010a011a100a101a110a111.$$

We transfer L into a neutral letter language L' over the alphabet $A := \{0, 1, a, e\}$: L' is simply the set of strings w over A such that the string obtained by deleting all the e's in w is in L. Clearly, L' has a neutral letter e, as inserting or deleting e's cannot affect membership in L'. Using the *Pumping Lemma* for regular languages it is straightforward to see that L' is not regular, so it is not definable in $FO[\leqslant]$. It remains for us to prove:

Lemma 5.4. *L' is definable in $FO[\leqslant, +, \times]$.*

Proof. We need to formulate a $FO[\leqslant, +, \times]$-sentence that will hold for a string exactly if it is in L', that is, exactly if its non-neutral letters form a string in L. Recall that a string w is in L exactly if for some number k, w consists of the 2^k binary strings of length k, in order, separated by a's.

Our sentence will assert the existence of a number k such that the input string, with e's removed, is the kth string in the language L. Since the length of the kth string in L is exponential in k, and a valid input string must be at least as long, any valid k must be at most $\lg n$. Therefore by Theorem 5.2, $FO[\leqslant, +, \times]$ is able to count letters in any interval in the input string up to a limit of k.

The sentence defining L' first asserts that there are exactly k 0s and no 1s before the first a, exactly k elements from $\{0, 1\}$ between each pair of a's, and exactly k 1s (and no 0s) after the last a. It then remains to assert that each string of 0s and 1s between two a's is the successor of the previous one. I.e., when all e's are removed, between successive occurrences of a the string has to have the form

$$\cdots ab_1 \cdots b_i 011 \cdots 11ab_1 \cdots b_i 100 \cdots 00a \cdots$$

where $0 \leqslant i < k$ and $b_1, \ldots, b_i \in \{0, 1\}$.

To assert this, the sentence defining L' states that for every position y containing a 0 or 1 the following holds:

- If there is a position x left of y such that there is a 0 or 1 at x and exactly $k-1$ 0s and 1s strictly between x and y,
- then x has the same letter as y *unless*
- v has the unique a between x and y, w has the next a to the right of y or is the rightmost position if there is no such a,
- x has 1, there are no 0s between x and v, y has 0, and there are no 1s between y and w, *or*
- x has 0, there are no 0s between x and v, y has 1, and there are no 0s between y and w.

Altogether, this gives us a $FO[\leqslant, +, \times]$-sentence that defines the language L'. This proves Lemma 5.4, and thus Theorem 5.3 follows immediately. □

An obvious consequence of Theorem 5.3 is that if $\mathcal{N}$ is any class of numerical predicates such that $FO[\leqslant, \mathcal{N}]$ can express $+$ and $\times$, then the CBC is false for $FO[\leqslant, \mathcal{N}]$. Thus, we obtain the following:

Corollary 5.5. *The CBC is false for*
(a) $FO[\leqslant, \mathcal{ARB}]$,
(b) $FO[\leqslant, +, \times]$, $FO[\leqslant, Bit]$, $FO[\leqslant, \times]$, $FO[\leqslant, +, Squares]$,
where Bit is the binary numerical predicate such that $Bit(x, y)$ is true iff the yth bit in the binary representation of x is 1, *and Squares is the set of all square numbers,*
(c) $FO[\leqslant, f]$,
where f is the graph of a suitable unary function,
(d) $FO[\leqslant, \mathcal{ORD}]$,
where $\mathcal{ORD}$ is the class of all binary numerical predicates which, on each universe $\{1, \ldots, n\}$, are linear orderings.

Proof. Using Theorem 5.3, (a) is obvious. (b) is true since

$$FO[\leqslant, +, \times] = FO[\leqslant, Bit] = FO[\leqslant, \times] = FO[\leqslant, +, Squares].$$

The first equation is proved, e.g., in the textbook [16]; and, as Lee showed in [19] the construction there also suffices to prove the second equation. For the third equation it suffices to show that $\times$ is expressible in $FO[\leqslant, +, Squares]$. This can be done using a construction of [23, Lemma 1]; details can be found in [27].

(c) is true since Schwentick [29] exposed a particular unary function[6] $f : \mathbb{N} \to \mathbb{N}$ such that $FO[\leqslant, +, \times] = FO[\leqslant, f]$ (see the proof of Theorem 3 in [29]). (d) is true since Schweikardt and Schwentick [28] exposed four particular linear orderings $\leqslant_1, \leqslant_2, \leqslant_3, \leqslant_4$ on initial segments of $\mathbb{N}$ such that $FO[\leqslant, +, \times] = FO[\leqslant, \leqslant_1, \leqslant_2, \leqslant_3, \leqslant_4]$. □

Our proof of Theorem 5.3 crucially uses the fact that we can count up to $\lg n$ in $FO[\leqslant, +, \times]$. We can strengthen the construction so that it provides a counterexample using only counting up to $\lg^{(m)}(n)$, the m times iterated logarithm of n. However, we do not yet know whether this strengthening is non-trivial—it

[6] Whose graph can be interpreted as a binary numerical predicate.

may be that any set of numerical predicates that allows counting up to $\lg^{(m)}(n)$ also allows counting up to $\lg n$.

Proposition 5.6. *If $\mathcal{N}$ is a set of numerical predicates such that $FO[\leqslant, \mathcal{N}]$ can count up to $\lg^{(m)}(n)$ for some $m \in \mathbb{N}$, then the CBC is false for $FO[\leqslant, \mathcal{N}]$.*

Proof. We must show that counting up to $\lg^{(m)}(n)$ suffices to provide a counterexample to the CBC. We give the construction in some detail for $m{=}2$; afterwards we will indicate how to generalize it to arbitrary values for m. Take the alphabet $A := \{a, b, 0, 1, e\}$, and for every k consider strings of the form

$$\Big(b\,(0+1)^k\,\big(a\,(0+1)^k\,\big)^*\Big)^* b.$$

Finally, add e as a neutral letter. The letters a and b are used as markers, and we interpret the 0–1-substring between any two successive markers as the binary representation of some number between 0 and 2^k-1. If x is any position, we define $block(x)$ to be the interval between the two nearest markers to the left and to the right of x, and $num(x)$ to be the number represented by the 0–1 subsequence in $block(x)$. Using a formula that can count up to k and the construction from the proof of Theorem 5.3 we can write formulas expressing that $num(x) = num(y)$ and $num(x) + 1 = num(y)$, respectively. We can now express easily that between every successive occurrences of two b's each number from 0 to 2^k-1 is represented precisely once. In other words, this formula stipulates that the $\{a, 0, 1\}$-substring between two successive b's represent a permutation of the numbers $0, \dots, 2^k-1$. Finally, we write a formula that expresses that all permutations are represented. Altogether, our formula defines the set L of those strings which consist of a sequence of permutations of the numbers $0, \dots, 2^k-1$, for some k, containing every permutation at least once. In particular, every such string has length $\Omega(2^k!)$, whereas counting is only required up to $k = O(\lg \lg(2^k!))$.

To be more precise, the formula forces all permutations to be present as follows. It says that for every represented permutation π (starting, say, with a b at position p), and every pair of positions i, j within that permutation (i.e., $p < i < j < p'$, where p' is the smallest position $> p$ that carries a b), there is a permutation ρ (between b's at q and q', say) which is equal to π, except that $num(i)$ and $num(j)$ are swapped. In what follows we will use abbreviations $first(x)$ and $last(x)$ for formulas which express that x lies in the first, respectively last, block of some permutation; $next(x)$ will denote the first position in the block directly to the right of $block(x)$. Our formula for i and j now expresses the following for all r, s such that $p < r < p'$ and $q < s < q'$:

- if $num(r){=}num(s)$ then $num(next(r)){=}num(next(s))$ unless $\big(last(r)$ or $\{num(r), num(next(r))\} \cap \{num(i), num(j)\} \neq \emptyset\big)$
- if $\big(num(r) = num(s)$ and $num(next(r)) = num(i)\big)$ then $num(next(s)) = num(j)$
- if $\big(num(r) = num(s)$ and $num(next(r)) = num(j)\big)$ then $num(next(s)) = num(i)$
- if $\big(num(s) = num(j)$ and $\neg last(s)\big)$ then $num(next(s)) = num(next(i))$
- if $\big(num(s) = num(i)$ and $\neg last(s)\big)$ then $num(next(s)) = num(next(j))$
- if $\big(first(r)$ and $first(s)$ and $num(r) \neq num(i)\big)$ then $num(r) = num(s)$
- if $\big(first(r)$ and $first(s)$ and $num(r) = num(i)\big)$ then $num(s) = num(j)$.

Altogether, this gives us a $FO[\leqslant, \mathcal{N}]$-sentence that defines the desired language L (provided that $FO[\leqslant, \mathcal{N}]$ can count up to $O(\lg^{(2)}(n))$).

Considering $m > 2$, we can then iterate the above process, using an additional marker symbol c. After the first iteration, the resulting formula stipulates that our string represent all permutations of all the permutations of the numbers $0, \ldots, 2^k-1$. This will guarantee that string to be of length $\Omega(((2^k)!)!)$, and so forth. Finally, this proves Proposition 5.6. □

Let us note that a different proof, based on an "addressing mechanism" rather than the permutations used above, can be found in [25].

It is not difficult to code the languages above using only two non-neutral letters: just apply the homomorphism $\{a, b, 0, 1, e\}^* \to \{0, 1, e\}^*$ which maps e to e, a to 01, b to 011, 0 to 0111, and 1 to 01111. However, due to Theorem 4.3, with only one non-neutral letter there is no way of defeating the CBC.

In the following section we will introduce a framework that gives better understanding of what exactly may cause a failure of the CBC. Furthermore, we will show that, in some sense, no modified version of the CBC is true for the class $\mathcal{ARB}$.

6. The Crane Beach conjecture revisited

In Section 3 we exposed the intuition that led to the formulation of the CBC:

> Assume that L is a language that has a neutral letter e. Via inserting or deleting e's, the non-neutral letters in a given string can be moved, without changing the membership or non-membership in L, onto any combination of positions—as long as the relative ordering of the non-neutral letters remains unchanged. It therefore seems conceivable that extra numerical predicates do not help first-order logic $FO[\leqslant]$ to define neutral letter languages.

This intuition sounds convincing—but it is *wrong*. In Theorem 5.3 we saw that already the predicates $\{+, \times\}$ cause the conjecture to fail. The counterexample was a language L that is not star-free regular but definable in $FO[\leqslant, +, \times]$. Indeed, all that is needed to define this language is the ability to count up to $\lg n$. In other words: The counterexample is definable in first-order logic with unary counting quantifiers, $FOunC[\leqslant]$ (even if counting is restricted up to $\lg n$). Here, the logic $FOunC$ is obtained from FO by adding *unary counting quantifiers* of the form $\exists^{=x} y$. For an interpretation p of the variable x, a formula $\exists^{=x} y\, \varphi(y)$ expresses that there are exactly p different interpretations of the variable y such that the formula $\varphi(y)$ is satisfied.

It is quite tempting to try to find a modified version of the CBC, i.e., a new conjecture of, e.g., the following kind:

> If a language with a neutral letter can be defined in $FO[\leqslant, \mathcal{ARB}]$, then it can be defined also in $FOunC[\leqslant]$ or, as another modified version of the CBC, in $FO[\leqslant, Bit] = FO[\leqslant, +, \times]$.

However, the subsequent considerations give a new intuition that helps to refute the above versions and that gives a framework for identifying new cases for which the CBC is false.

Definition 6.1 (*Associating strings with numbers*). Let F be a class of formulas and let $\mathcal{N}$ be a set of numerical predicates. We say that $F[\leqslant, \mathcal{N}]$ *can associate strings with numbers* iff the following is true:

- there exists an alphabet B and a letter e *not* in B, and letting $A := B \cup \{e\}$,
- there exists, for every $k > 0$, a string $s(k)$ in B^* of length $\geqslant k$, where $s(k) \neq s(l)$ for all $k \neq l$, and
- there exists a $F[\sigma_A, \leqslant, \mathcal{N}]$-formula $\psi(x)$ such that

for every string $w \in A^*$ and for every position k in w, w satisfies $\psi(k)$ if and only if $s(k)$ is the string obtained by deleting all the e's in w.

In other words: The string $s(k)$ encodes the number k, and the formula $\psi(x)$ serves as a decoder that works even if neutral letters are inserted into $s(k)$.

For example, $FO[\leqslant, +, \times]$ can associate strings with numbers: In the proof of Theorem 5.3 we considered the alphabet $B := \{0, 1, a\}$ and the strings $s(k) :=$ "ordered list of all binary strings of length k, separated by the letter a", and we constructed a $FO[\leqslant, +, \times]$-formula $\psi(x)$ such that for all strings w over $A := B \cup \{e\}$ and all positions k in w it is true that w satisfies $\psi(k)$ if and only if w can be obtained by inserting e's into $s(k)$.

For convenience, given a string $s \in B^*$ we will henceforth write *neutral*(s) to denote the set of all strings over $B \cup \{e\}$ which can be obtained by inserting letters e into s.

The consequences of the ability to associate strings with numbers can be formulated as follows:

> The ability of associating strings with numbers permits access to the information stored in the numerical predicates—and this is what causes a failure of the CBC.

For example, if P is a subset of $\mathbb{N}$, then the neutral letter language

$$L_P := \bigcup_{k \in P} neutral\big(s(k)\big)$$

is definable in $FO[\leqslant, +, \times, \hat{P}]$ by the formula $\exists x\ \hat{P}(x) \wedge \psi(x)$. Consequently, since there is an uncountable number of subsets P of $\mathbb{N}$, $FO[\leqslant, \mathcal{ARB}]$ can define an uncountable number of neutral letter languages. This immediately leads to

Corollary 6.2. *Let F be a logical system such that for every finite or countable signature τ there are at most countably many $F[\tau]$-formulas.*

There is no finite or countable set $\mathcal{N}$ of numerical predicates such that $F[\leqslant, \mathcal{N}]$ can define all neutral letter languages that are definable in $FO[\leqslant, \mathcal{ARB}]$.

The tool provided by Definition 6.1 also enables us to refute the CBC for the counting logic ***FOunC***, already for the class of *monadic* predicates and two-letter alphabets:

Theorem 6.3. (a) *There is no finite or countable set $\mathcal{N}$ of numerical predicates such that **FOunC**$[\leqslant, \mathcal{N}]$ can define all neutral letter languages definable in $FO[\leqslant, \mathcal{ARB}]$.*
*In particular, the CBC is false for **FOunC**$[\leqslant, \mathcal{ARB}]$.*

(b) *Let $P \subseteq \mathbb{N}$ be a set that is not semi-linear.*[7]
*There is a neutral letter language over the alphabet $\{a, e\}$ that can be defined in **FOunC**$[\leqslant, \hat{P}]$, but not in **FOunC**$[\leqslant]$.*

[7] A set $P \subseteq \mathbb{N}$ is semi-linear iff there are $p, q \in \mathbb{N}$ such that for every $k \geqslant q$ we have $k \in P$ iff $k+p \in P$.

For example, P can be chosen to be the set Primes of all prime numbers or the set Squares of all square numbers. Consequently, the CBC is false for FOunC$[\leqslant, \mathit{Primes}]$, *FOunC*$[\leqslant, \mathit{Squares}]$, *FOunC*$[\leqslant, \mathcal{MON}]$, *and FOunC*$[\leqslant, \times]$, *even if attention is restricted to languages over a two-letter alphabet.*

Proof. (a) is a direct consequence of Corollary 6.2. The proof of (b) proceeds in three steps:

Step 1: *FOunC*$[\leqslant]$ *can associate strings with numbers.*

Let $B := \{a\}$. For every $k > 0$ let $s(k) := a^k$ be the string that consists of exactly k a's. Let $A := \{a, e\}$ and let $\psi(x) := \exists^{=x} y\, Q_a(y)$. Obviously, for all strings w in A^* and for all positions k in w, w satisfies $\psi(k)$ iff $w \in \mathit{neutral}\big(s(k)\big)$.

Step 2: Choosing an *FOunC*$[\leqslant, \hat{P}]$-definable neutral letter language L_P.

Define $L_P := \bigcup_{k \in P} \mathit{neutral}\big(s(k)\big)$. Of course, L_P is definable in *FOunC*$[\leqslant, \hat{P}]$ by the formula $\exists x\, \hat{P}(x) \wedge \psi(x)$.

Step 3: L_P is not definable in *FOunC*$[\leqslant]$.

For the sake of contradiction, assume that L_P is definable in *FOunC*$[\leqslant]$ via a sentence χ over the signature $\{Q_a, Q_e, \leqslant\}$. We first show that χ can be transformed into a *FOunC*$[\leqslant]$-formula $\varphi(x)$ that defines P in pure arithmetic on $\mathbb{N}$, i.e. $P = \{k \in \mathbb{N} \,/\, \langle \mathbb{N}, \leqslant\rangle \models \varphi(k)\}$; afterwards we will show that the existence of $\varphi(x)$ is a contradiction to the presumption that P is not semi-linear.

By definition of L_P we know that the particular string $s(k) = a^k$ belongs to L_P if and only if $k \in P$. Furthermore, the string $s(k)$ is represented by the structure $\langle \{1, \ldots, k\}, Q_a^{s(k)}, Q_e^{s(k)}, \leqslant\rangle$, where $Q_a^{s(k)} = \{1, \ldots, k\}$ and $Q_e^{s(k)} = \emptyset$. According to our assumption, χ defines the language L_P, and hence we have

$$\langle \{1, \ldots, k\}, Q_a^{s(k)}, Q_e^{s(k)}, \leqslant\rangle \models \chi \quad \text{iff} \quad s(k) \in L_P \quad \text{iff} \quad k \in P\,.$$

Let x be a new first-order variable that does not occur in χ. We replace every atom of the form $Q_a(y)$ in χ by the atom $y \leqslant x \wedge \neg y{=}0$, and we replace every atom of the form $Q_e(y)$ in χ by the atom $\neg y{=}y$. Furthermore, we relativize all quantifications to numbers that are $\leqslant x$ and $\neq 0$. It is not difficult to see that this leads to a *FOunC*$[\leqslant]$-formula $\varphi(x)$ such that the following is true for all interpretations $k > 0$ of the variable x:

$$\langle \mathbb{N}, \leqslant\rangle \models \varphi(k) \quad \text{iff} \quad \langle \{1, \ldots, k\}, Q_a^{s(k)}, Q_e^{s(k)}, \leqslant\rangle \models \chi \quad \text{iff} \quad k \in P\,.$$

Consequently, $\varphi(x)$ is an *FOunC*$[\leqslant]$-formula that defines the non-semi-linear set P in pure arithmetic on $\mathbb{N}$. However, in [27] it was shown that

FOunC$[\leqslant]$ = *FOunC*$[\leqslant, +]$ = *FO*$[\leqslant, +]$ in pure arithmetic on $\mathbb{N}$.

This gives us an *FO*$[\leqslant, +]$-formula $\varphi'(x)$ such that $P = \{k \in \mathbb{N} \,/\, \langle \mathbb{N}, \leqslant, +\rangle \models \varphi'(k)\}$. I.e., P is definable in *FO*$[\leqslant, +]$ on $\mathbb{N}$. However, this is a contradiction to the Theorem of Ginsburg and Spanier, stating that the *FO*$[\leqslant, +]$-definable sets are exactly the semi-linear sets (cf., e.g., the textbook [30, Theorem 4.10]).

In other words: The formula $\varphi'(x)$ and, consequently, the formula χ cannot exist. This completes our proof of Theorem 6.3. □

7. Discussion

A summary of what we have shown about the CBC is given in Fig. 2. Much of the above can be generalised from strings to arbitrary relational structures over the natural (or real) numbers. This programme is pursued in [21,25,26]. With regard to the questions here, the following problems remain open:

- It would be very good to have proofs of Theorem 4.3 and 4.2 that do not rely on [1,15,4]. However, since both theorems imply the nonexpressibility of PARITY, we expect this to be very difficult.
- Can we find a set of numerical predicates that allows us to count up to $\lg^{(m)}(n)$, but not to $\lg n$? What about counting up to even smaller functions? We conjecture that the CBC is true for a logical system iff it cannot count beyond a constant.
- Within $FO[\leqslant, +, \times]$, we can consider the subclasses of formulas based on the number of quantifier alternations. The lg-counting operation requires Σ_3, and the construction of the counter example adds a few more levels. This leaves a gap between the upper bound of something like Σ_5 in Theorem 5.3, and a lower bound of $BC(\Sigma_1)$ in Theorem 4.4. Since in $BC(\Sigma_2)$, counting is only possible up to a constant (cf., [14]), it is conceivable that the lower bound can be improved. However, currently the state of the CBC for arbitrary numerical predicates is not even known for $\Sigma_2 \cap \Pi_2$.
- Theorem 4.8 places limits on the power of a particular uniform circuit complexity class, an "addition and some unary predicates"-uniform version of AC^0. Can we use these techniques (or new techniques) to place limits on the power of more powerful uniform versions of AC^0 (without using the non-uniform lower bounds) or on addition-uniform versions of more powerful complexity classes? This has been done for one such class, an addition-uniform version of LOGCFL, by Lautemann et al. [20].
- From Corollary 4.13 we know that the CBC is true for $FO[\leqslant, \hat{\mathcal{N}}]$ if the structure $\langle \mathbb{N}, \leqslant, \mathcal{N} \rangle$ has finite VC-dimension. It therefore is a further task to find more structures with universe $\mathbb{N}$ that have finite VC-dimension.
- In Section 6 we have seen that the Crane Beach conjecture is false if a logic can associate strings with numbers. It is a further task to investigate what other consequences follow from the ability to associate strings with numbers.
- It would also be of interest to study the conjecture for certain extensions of *FO*, such as *FO* with modulo counting quantifiers. These each have various versions depending on the numerical predicates available.

Acknowledgements

We are indebted to Thomas Schwentick for bringing the database theory connection to our attention. He also took an active part in many discussions on the subject of this paper. In particular, the first proof of Theorem 4.1 was partly due to him. The first author in particular would like to thank Eric Allender, Pierre McKenzie, and Howard Straubing for valuable discussions on this topic, many of which occurred at a Dagstuhl workshop in March 1997. Much important work on this topic also occurred at various McGill Invitational Workshops on Complexity Theory, particularly on excursions to Crane Beach, St. Philip, Barbados. Furthermore, we would like to thank Anuj Dawar for an observation that simplified the proof of Corollary 6.2.

Cases Where the CBC is True:

in general for *all* string–languages with neutral letter:

• $FO[\leqslant, \mathcal{MON}]$	Thm. 4.1
• $FO[\leqslant, +, \mathcal{MON}_Q]$	Thm. 4.8
• $FO[\leqslant, +, \hat{R}_{2(\lfloor \lg x \rfloor^2)}]$	Cor. 4.14
• $BC(\Sigma_1[\leqslant, \mathcal{ARB}])$	Thm. 4.4
• $BC(\Sigma_1^{(s,p)}[\leqslant, \mathcal{ARB}])$	[18]
• $FO[\leqslant, \hat{\mathcal{N}}]$ if *natural–generic collapse* over $\langle \mathbb{N}, \leqslant, \mathcal{N} \rangle$	Prop. 4.10
• $FO[\leqslant, \hat{\mathcal{N}}]$ if $\langle \mathbb{N}, \leqslant, \mathcal{N} \rangle$ has *finite VC–dimension*	Cor. 4.13

for certain kinds of string–languages:

• $FO[\leqslant, \mathcal{ARB}]$ on *regular* languages	Cor. 4.2
• $FO[\leqslant, \mathcal{ARB}]$ on *two–letter* alphabets	Thm. 4.3

Cases Where the CBC is False:

• $FO[\leqslant, \mathcal{ARB}]$ (even on *three–letter* alphabets)	Thm. 5.5
• $FO[\leqslant, +, \times]$, $FO[\leqslant, Bit]$, $FO[\leqslant, \times]$, $FO[\leqslant, +, Squares]$, $FO[\leqslant, f]$, $FO[\leqslant, \mathcal{ORD}]$	Thm. 5.3, Cor. 5.5
• $FO[\leqslant, \mathcal{N}]$ if counting up to iterated logarithm	Prop. 5.6
• $FOunC[\leqslant, \mathcal{MON}]$ (even on *two–letter* alphabets)	Thm. 6.3
• $FOunC[\leqslant, \hat{P}]$ if $P \subseteq \mathbb{N}$ is not semi–linear	Thm. 6.3

No Chance for an Update of the CBC:

$FO[\leqslant, \mathcal{ARB}]$ can define an uncountable number of neutral letter languages. Consequently, for every countable $\mathcal{N}$ and every logical system F such that for every countable signature τ there are at most countably many $F[\tau]$–formulas, $FO[\leqslant, \mathcal{ARB}]$ can define neutral letter languages not definable in $F[\leqslant, \mathcal{N}]$.	Cor. 6.2

Fig. 2. Summary of what is known about the Crane Beach conjecture.

References

[1] M. Ajtai, Σ^1_1-formulae on finite structures, Ann. Pure Appl. Logic 24 (1983) 1–48.
[2] M. Ajtai, M. Ben-Or, A theorem on probabilistic constant depth computations, in: STOC'84: 16th Annual ACM Symposium on the Theory of Computing, Washington DC, 1984, pp. 471–474.
[3] J.T. Baldwin, M. Benedikt, Embedded finite models, stability theory and the impact of order, in: LICS'98: 13th Annual IEEE Symposium on Logic in Computer Science, Indianapolis, IN, USA, 1998, pp. 490–500.
[4] D.A.M. Barrington, K. Compton, H. Straubing, D. Thérien, Regular languages in NC^1, J. Comput. System Sci. 44 (1992) 478–499.
[5] D.A.M. Barrington, N. Immerman, C. Lautemann, N. Schweikardt, D. Thérien, The Crane Beach Conjecture, in: LICS'01: 16th Annual IEEE Symposium on Logic in Computer Science, Boston, MA, USA, 2001, pp. 187–196.
[6] D.A.M. Barrington, N. Immerman, H. Straubing, On uniformity within NC^1, J. Comput. System Sci. 41 (1990) 274–306.
[7] O.V. Belegradek, A.P. Stolboushkin, M.A. Taitslin, Extended order-generic queries, Ann. Pure Appl. Logic 97 (1999) 85 –125.
[8] M. Benedikt, G. Dong, L. Libkin, L. Wong, Relational expressive power of constraint query languages, J. ACM 45 (1998) 1–34.
[9] M. Benedikt, L. Libkin, Expressive power: the finite case, in: G. Kuper, L. Libkin, J. Paredaens (Eds.), Constraint Databases, Springer, Berlin, 2000, pp. 55–87.
[10] L. Denenberg, Y. Gurevich, S. Shelah, Definability by constant-depth polynomial-size circuits, Inform. Control 70 (1986) 216–240.
[11] R. Diestel, Graph Theory, Graduate Texts in Mathematics, vol. 173, 2nd ed., Springer, Berlin, 2000. Electronic version available at http://www.math.uni-hamburg.de/home/diestel/books/graph.theory/.
[12] H.-D. Ebbinghaus, J. Flum, Finite Model Theory, 2nd ed., Springer, Berlin, 1999.
[13] P. Erdös, G. Szekeres, A combinatorial problem in geometry, Compositio Math. 2 (1935) 464–470.
[14] R. Fagin, M. Klawe, N. Pippenger, L.J. Stockmeyer, Bounded-depth polynomial size circuits for symmetric functions, Theoret. Comput. Sci. 36 (1985) 239–250.
[15] M.L. Furst, J.B. Saxe, M. Sipser, Parity, circuits, and the polynomial-time hierarchy, Math. Systems Theory 17 (1984) 13 –27.
[16] N. Immerman, Descriptive and Computational Complexity, Springer, Berlin, 1999.
[17] A. Krol, C. Lautemann, Modular counting in strings, Unpublished manuscript, 2002.
[18] G.M. Kuper, L. Libkin, J. Paredaens (Eds.), Constraint Databases, Springer, Berlin, 2000.
[19] T. Lee, Arithmetical definability over finite structures, Math. Logic Quart. 49 (2003) 385–392.
[20] C. Lautemann, P. McKenzie, T. Schwentick, H. Vollmer, The descriptive complexity approach to LOGCFL, in: STACS'99: 16th Annual Symposium on Theoretical Aspects of Computer Science, Lecture Notes in Computer Science, vol. 1563, Springer, Berlin, 1999, pp. 444–454.
[21] C. Lautemann, N. Schweikardt, An Ehrenfeucht–Fraïssé approach to collapse results for first-order queries over embedded databases, in: STACS'01: 18th Annual Symposium on Theoretical Aspects of Computer Science, Lecture Notes in Computer Science, vol. 2010, Springer, Berlin, 2001, pp. 455–466
[22] L. Libkin, Embedded finite models and constraint databases. in: E. Grädel, P.G. Kolaitis, L. Libkin, M. Marx, J. Spencer, M.Y. Vardi, Y. Venema, S. Weinstein (Eds.), Finite Model Theory and Its Applications, Springer-Verlag, 2005.
[23] J.F. Lynch, Complexity classes and theories of finite models, Math. Systems Theory 15 (1982) 127–144.
[24] J.F. Lynch, On sets of relations definable by addition, J. Symbolic Logic 47 (1982) 659–668.
[25] N. Schweikardt, On the expressive power of first-order logic with built-in predicates, Ph.D. Thesis, Johannes Gutenberg-Universität Mainz, Fachbereich Mathematik und Informatik, 2001. Published at Logos-Verlag, Berlin, 2002, ISBN 3-8325-0017-0.
[26] N. Schweikardt, An Ehrenfeucht–Fraïssé game approach to collapse results in database theory, CoRR Report, arXiv:cs.LO/0212049, 20 December 2002, 70 pp, Submitted for publication.
[27] N. Schweikardt, Arithmetic, first-order logic, and counting quantifiers, ACM Trans. Comput. Logic 2004, 35 p, to appear.
[28] N. Schweikardt, T. Schwentick, Unpublished manuscript on Monadic NP with built-in grid structures.
[29] T. Schwentick, Padding and the expressive power of existential second-order logics, in: CSL'97: Annual Conference of the European Association for Computer Science Logic, Lecture Notes in Computer Science, vol. 1414, Springer, Berlin, 1997, pp. 461–477.

[30] C. Smoryński, Logical Number Theory I, Springer, Berlin, 1991.
[31] H. Straubing, Finite Automata, Formal Logic, and Circuit Complexity, Birkhäuser, Basel, 1994.
[32] H. Straubing, Languages defined with modular counting quantifiers, Inform. Comput. 166 (2001) 112–132.
[33] M.A. Taitslin, Translation results in database theory, in: Complex systems: data processing, simulation, and optimization, Tver State University, Tver, 2002, pp. 5–23. (Russian. An English abstract was presented at the FMT'03 meeting in Bedlewo, Poland).
[34] J. Van den Bussche, Applications of Alfred Tarski's ideas in database theory, in: CSL'01: Annual Conference of the European Association for Computer Science Logic, Lecture Notes in Computer Science, vol. 2142, Springer, 2001, pp. 20–37.
[35] I. Wegener, N. Wurm, S.-Z. Yi, Symmetric functions in AC^0 can be computed in constant depth and very small size, in: M. Paterson (Ed.), Boolean Function Complexity, London Mathematical Society Lecture Notes, vol. 169, 1992, pp. 129–139.