ACADEMIC PRESS

Available at
www.ComputerScienceWeb.com
POWERED BY SCIENCE @ DIRECT®

Information and Computation 185 (2003) 159–181

Information and Computation

www.elsevier.com/locate/ic

Refining the hierarchy of blind multicounter languages and twist-closed trios

Matthias Jantzen* and Alexy Kurganskyy

Universität Hamburg FB Informatik, Universität Hamburg, Vogt-Kölln-Straße 30, 22527 Hamburg, Germany

Received 6 April 2001; revised 9 April 2002

Abstract

We introduce the new families (k, r)-RBC of languages accepted in quasi-realtime by one-way counter automata having k blind counters, of which at least r are reversal-bounded. It is proved, that these families form a strict and linear hierarchy of semi-AFLs within the the family BLIND $= \mathcal{M}_{\cap}(C_1)$ of blind multicounter languages with generator $C_1 = \{w \in \{a_1, b_1\}^* \mid |w|_{a_1} = |w|_{b_1}\}$. This thereby combines the families BLIND and RBC from [13] to one strict hierarchy and generalizes and sharpens Greibachs results. The strict inclusions between the k-counter families (k, r)-RBC are proved using linear algebra techniques. We also study the language theoretic monadic operation *twist* [18,20], in connection with the semi-AFLs of languages accepted by multicounter and multipushdown acceptors, all restricted to reversal-bounded behavior. It is verified, that the family (k, r)-RBC is *twist*-closed if and only if $r = 0$, in which case $(k, 0)$-RBC $= \mathcal{M}(C_k)$, C_k being the k-fold shuffle of disjoint copies of C_1. We characterize the family $\mathcal{M}_{\cap}(PAL)$ of languages accepted in quasi-realtime by nondeterministic one-way reversal-bounded multipushdown acceptors as the least *twist*-closed trio $\mathcal{M}_{twist}(PAL)$ generated by the set of palindromes $PAL = \{w \in \{a, b\}^* \mid w = w^{rev}\}$.
© 2003 Elsevier Science (USA). All rights reserved.

Keywords: Reversal-bounded multicounter automata; Shuffle; Intersection-closed semi-AFL; *Twist*-closed semi-AFLs; Hierarchy of semi-AFLs; Blind multicounter languages; Linear algebra; Multipushdown acceptors

1. Introduction

The families RBC, of languages accepted by one-way reversal-bounded multicounter automata in quasi-realtime, and BLIND, where the counters are unrestricted, are identical and form the well known

* Corresponding author. Fax: +49-40-42883-2246.
E-mail addresses: jantzen@informatik.uni-hamburg.de (M. Jantzen), Kurgansk@gmx.de (A. Kurganskyy).

0890-5401/$ - see front matter © 2003 Elsevier Science (USA). All rights reserved.
doi:10.1016/S0890-5401(03)00087-7

semi-AFL $\mathcal{M}_\cap(B_1)$. This semi-AFL is principal as an *intersection*-closed semi-AFL with generator $B_1 := \{a_1^n b_1^n \mid n \in \mathbb{N}\}$, but is not a principal semi-AFL, see [9,13]. (For the definition of the languages B_i, C_1, and C_i see Definition 2.2 below.) Greibach [13] has shown $\bigcup_{i\geqslant 1}\mathcal{M}(C_i) = \mathcal{M}_\cap(C_1) = \text{BLIND} = \bigcup_{i\geqslant 1}\mathcal{M}(B_i) = \mathcal{M}_\cap(B_1) =$ RBC. Moreover, we have $\mathcal{M}(B_i) \subsetneq \mathcal{M}(B_{i+1})$, see [10], $\mathcal{M}(C_i) \subsetneq \mathcal{M}(C_{i+1})$, and $\mathcal{M}(B_i) \subseteq \mathcal{M}(C_i)$ for all $i \geqslant 1$, as shown in [12,13,27–29]. It was verified by Latteux in [27] that BLIND is a *slip* family, i.e., has a semilinear Parikh-image, thus each language $L \in$ BLIND is letter equivalent to some regular set. Counter automata that may use some number of blind and some reversal-bounded counters have not been considered before.

We define and study the families (k, r)-RBC of languages accepted in quasi-realtime by one-way (or on-line) counter automata having k blind counters of which (at least) $r \leqslant k$ are reversal-bounded and prove (k_1, r_1)-RBC $\subsetneq$ (k_2, r_2)-RBC if and only if $k_1 < k_2$ or $k_1 = k_2$ and $r_1 > r_2$. The strict inclusions between the families having the same number of blind counters, but which differ only by the number of reversal-bounded counters are for the first time proved using linear algebra techniques, as reported in [21,22]. In this situation the usual technique by comparing the dimension of the storage space is not applicable, since the total number of counters is not changed.

In connection with a representation of Petri net languages by Dyck-reductions of (linear) context-free sets the operation *twist* was defined and used for the first time, see [19,20]. The definition of this new language theoretic operation is based upon a mapping from strings to strings which rearranges letters depending solely on their positions. For a string $w := x_1x_2\cdots x_{n-1}x_n$ the unique new string is $\mathit{twist}(w) := x_1x_nx_2x_{n-1}\cdots x_{\lfloor n/2\rfloor+1}$. For example $\mathit{twist}(abcxyz) = azbycx$ and $\mathit{twist}(abcxy) = aybxc$. The mapping *twist* can be regarded as a permutation of the n distinct positions for the symbols of a string of length n. Hence, $\mathit{twist} : \Sigma^* \to \Sigma^*$ is a bijection and its inverse yields the unique string $v = \mathit{twist}^{-1}(w)$.

As monadic language theoretic operation *twist* is generalized to languages and families of languages in the obvious way, see Definition 4.1.

It is a nice exercise to prove that the family $\mathcal{R}eg$ of regular sets is closed with respect to *twist* (see [20] for a proof). The inclusion $\mathit{twist}(\mathcal{R}eg) \subsetneq \mathcal{R}eg$ must be proper since $\mathit{twist}(\mathit{MIR}) = \{a^2, b^2\}^*$, where $\mathit{MIR} := \{ww^{rev} \mid w \in \{a, b\}^*\}$ is the nonregular context-free set of palindromes of even length. This means, that the regular set $\{a^2, b^2\}^*$ will never appear as $\mathit{twist}(R)$ for any regular set $R \in \mathcal{R}eg$. Notice, $\mathit{twist}^{-1}(\mathit{MIR}) = \mathit{COPY} := \{ww \mid w \in \{a, b\}^*\}$. In [20] it was proved (Theorem 2.11) that the family $\mathcal{L}_0 := \mathcal{M}_\cap(D_1)$ is closed with respect to the operation *twist*. Again, $\mathit{twist}(\mathcal{L}_0) \subsetneq \mathcal{L}_0$, since $\mathit{twist}(MIR) \in \mathcal{L}_0$ but $MIR \notin \mathcal{L}_0$ follows from [13,14] using [24,30].

In this work, we verify that the family (k, r)-RBC is *twist*-closed, only if $r = 0$, in which case $(k, 0)$-RBC $= \mathcal{M}(C_k)$. It follows, that $\bigcup_{k\geqslant 1}(k, 0)$-RBC $= \bigcup_{i\geqslant 1}\mathcal{M}(C_i) = \mathcal{M}_\cap(C_1)$ forms a hierarchy of *twist*-closed semi-AFLs and therefore is not principal as *twist*-closed semi-AFL.

In [15] a new morphic characterization of the recursively enumerable sets was given by $\mathcal{R}e = \hat{\mathcal{H}}(\mathcal{H}^{-1}(\mathit{twist}(\mathit{lin}\,\mathcal{C}f))) = \hat{\mathcal{M}}_{\mathit{twist}}(\mathit{PAL})$. Similar results are known for principal *intersection*-closed full trios (see [1]) and for full principal trios, the generator of which is as rich in structure as the twinshuffle language L_{TS} (see [31], Chapter 6, for a condensed presentation. L_{TS} was there abbreviated by L_Σ). A slight improvement of Theorem 4.5 of [6] has been shown by Engelfriet in [7] for the reverse twinshuffle language L_{RTS}.

In all the above characterizations of the recursively enumerable sets, the use of arbitrary (erasing) homomorphisms was essential. The nonfull semi-AFL's with these generators have not been studied there. Some results in this direction have been obtained in [5] when studying the family of languages accepted by reversal-bounded Turing-machines in linear time. Each such language is also accepted in quasi-real-time by a reversal-bounded multipushdown automaton. It was proved in [5] that the *intersection*-closed trio $\mathcal{M}_\cap(PAL)$ equals the class of languages accepted by one-way reversal-bounded multipushdown automata in real-time and that three pushdown stores suffice. Hence, this family is a principal semi-AFL that is intersection-closed. In addition, we show in this work $\mathcal{M}_\cap(PAL) = \mathcal{M}_{twist}(PAL)$ directly and thereby improve the techniques used in [15–17].

2. Basic definitions

Definition 2.1. Let $\mathcal{R}eg$ (resp. $lin\,\mathcal{C}f$, $\mathcal{C}f$, $\mathcal{C}s$, $\mathcal{R}ec$, $\mathcal{R}e$) denote the families of regular sets (linear context-free, context-free, context sensitive, recursive, and recursively enumerable languages, respectively).

Definition 2.2. The symbol Σ will be used for any finite alphabet needed, while specific languages we consider are constructed using the alphabets $\Gamma := \{a, b\}$ and Γ_n specified for each $n \in \mathbb{N}, n \geqslant 1$ by: $\Gamma_n := \{a_i, b_i \mid 1 \leqslant i \leqslant n\}$. Especially in the language D_n defined below, the symbols a_i and b_i can be regarded as an pair of matching brackets. For an easier definition of certain languages we use the codings $ind_i : \Gamma^* \to \Gamma_n^*$ for each index $1 \leqslant i \leqslant n$, that are defined by: $ind_i(a) := a_i$ and $ind_i(b) := b_i$. Also the projection homomorphisms π_i onto the ith pair of brackets will be helpful and is defined by:

$$\pi_i(x) := \begin{cases} \lambda & \text{if } x \notin \{a_i, b_i\}, \\ x & \text{else.} \end{cases}$$

By $|w|_x$ we denote the number of occurrences of the symbol $x \in \Sigma$ within the string $w \in \Sigma^*$ and $|w| := \sum_{x\in\Sigma} |w|_x$ denotes the length of w.

$$\begin{aligned}
B_n &:= \{w \in \Gamma_n^* \mid \forall 1 \leqslant i \leqslant n \exists m \in \mathbb{N} : \pi_i(w) = a_i^m b_i^m\}, \\
C_n &:= \{w \in \Gamma_n^* \mid \forall 1 \leqslant i \leqslant n : |w|_{a_i} = |w|_{b_i}\}, \\
D_n &:= \{w \in C_n^* \mid \forall 1 \leqslant i \leqslant n : \forall w = uv : |u|_{a_i} \geqslant |u|_{b_i}\},
\end{aligned}$$

$$\begin{aligned}
MIR &:= \{ww^{rev} \mid w \in \Gamma^*\}, \\
dMIR &:= \{ind_1(w) ind_2(w^{rev}) | w \in \Gamma^*\}, \\
PAL &:= \{w \mid w = w^{rev}, w \in \Gamma^*\}, \\
COPY &:= \{ww \mid w \in \Gamma^*\}, \\
dCOPY &:= \{ind_1(w) ind_2(w) \mid w \in \Gamma^*\}.
\end{aligned}$$

With this definition, B_n (and D_n) is the shuffle of n disjoint copies, "colors", of the language $B_1 = \{a_1^n b_1^n \mid n \in \mathbb{N}\}$ (of D_1, respectively). And in the deterministic versions of the languages *COPY* and *MIR* the two halves of each string are build over disjoint alphabets.

For an easier reading, let us repeat the basic notions and results from AFL-theory, details of which are to be found in the textbooks of Ginsburg [10], and Berstel [2].

A family of languages $\mathcal{L}$ is called trio if it is closed under inverse homomorphisms, intersection with regular sets, and nonerasing homomorphisms. The least trio containing the family $\mathcal{L}$ is written $\mathcal{M}(\mathcal{L})$. If $\mathcal{L} := \{L\}$, then L is a generator of the trio $\mathcal{M}(\mathcal{L})$, shortly written as $\mathcal{M}(L)$, and the trio is then called principal. A union-closed trio is called semi-AFL and any principal trio is in fact a semi-AFL. If a trio is closed under arbitrary homomorphisms, then it is called a full trio, written $\hat{\mathcal{M}}(\mathcal{L})$.

A family of languages $\mathcal{L}$ is called an AFL (or full AFL) if it is a trio (full trio, resp.) which is closed under the operations union, product and Kleene plus. The smallest AFL (or full AFL) containing the family $\mathcal{L}$ is written $\mathcal{F}(\mathcal{L})$ ($\hat{\mathcal{F}}(\mathcal{L})$, resp.). Each full AFL is closed with respect to Kleene star.

If a trio $\mathcal{M}(\mathcal{L})$ (or an AFL $\mathcal{F}(\mathcal{L})$) is in addition closed with respect to one further operation $\circledast$ then this family will be called $\circledast$-closed and we use $\mathcal{M}_{\circledast}(\mathcal{L})$ (resp. $\mathcal{F}_{\circledast}(\mathcal{L})$) to denote the smallest trio (AFL, resp.) containing $\mathcal{L}$ and closed with respect to $\circledast$.

PAL, *MIR* and *dMIR* are well-known context-free generators of the family $lin\,\mathcal{C}f$ of linear context-free languages: $lin\,\mathcal{C}f = \mathcal{M}(dMIR) = \mathcal{M}(MIR) = \hat{\mathcal{M}}(PAL)$. These languages are precisely the languages accepted by nondeterministic one-way single pushdown acceptors which operate in such a way that in every computation the pushdown store makes at most one reversal. And this family is not closed with respect to product or Kleene plus.

The *intersection*-closed semi-AFL $\mathcal{M}_{\cap}(PAL)$ can be identified with the family of languages accepted by nondeterministic one-way multipushdown acceptors which operate in such a way that in every computation each pushdown makes at most one reversal and that work in quasi-realtime, see [3]. This family becomes the set of recursively enumerable languages if erasing is allowed and was characterized in [1] by $\mathcal{R}e = \hat{\mathcal{M}}_{\cap}(PAL) = \hat{\mathcal{M}}(twinPAL)$.

The language D_1 defined above is the so-called semi-Dyck language on one pair of brackets (also abbreviated by $D_1'^*$, see, e.g. [27,29] or [2]). In this work D_n denotes the n-fold shuffle of disjoint copies of the semi-Dyck language D_1 and it is known [13] that $\bigcup_{i\geqslant 1}\mathcal{M}(D_i) = \mathcal{M}_{\cap}(D_1) = \text{PBLIND}(n)$. The latter family is the family of languages accepted in quasi-realtime by nondeterministic one-way multi-counter acceptors which operate in such a way that in every computation no counter can store a negative value, and whether or not the value stored in a counter is *zero* is not used for deciding the next move.

The languages C_n are the (symmetric) Dyck languages on n pairs of brackets a_i, b_i (also denoted by D_n^*, see again [27,29] or [2]). Greibach [13] has shown that $\bigcup_{i\geqslant 1}\mathcal{M}(C_i) = \mathcal{M}_{\cap}(C_1) = \text{BLIND} = \text{BLIND}(lin) = \text{BLIND}(n) = \bigcup_{i\geqslant 1}\mathcal{M}(B_i) = \mathcal{M}_{\cap}(B_1) = \text{RBC}(n) = \text{RBC} \subsetneq \text{PBLIND}$.

Here BLIND (resp. BLIND(lin), BLIND(n)) denotes the family of languages accepted by nondeterministic one-way multicounter acceptors (in linear or quasi-real time, respectively) which operate in such a way that in every computation all counters may store arbitrary integers, and the information on the contents of the counters is not used for deciding the next move. The family RBC(n) is the family of languages accepted by nondeterministic one-way multicounter acceptors performing at most one reversal in each computation. The formal definition is to be found in Section 3.

The least *intersection*-closed full semi-AFL $\hat{\mathcal{M}}_{\cap}(B_1)$ has been characterized in [1] as the family of languages accepted by nondeterministic one-way multicounter acceptors which operate in such a way that in every computation each counter makes at most one reversal. It was there shown that this class contains only recursive sets, i.e., $\hat{\mathcal{M}}_{\cap}(B_1) \subseteq \mathcal{R}ec$.

3. Blind *k*-counter automata with $r \leqslant k$ reversal-bounded counters

In [18–20] it has been shown that $\mathcal{M}_{\cap}(D_1) = \text{PBLIND}(n)$ is closed with respect to *twist* without answering the question of *twist*-closure for its subfamilies $\mathcal{M}(D_k)$. We will now show that for each $k \geqslant 1$ the family $\mathcal{M}(C_k)$ of languages accepted by blind k-counter automata in quasi-realtime is *twist*-

closed, too, and answer the question for some of its subfamilies. To do this, let us more formally define the notation for those counter automata that have k blind counters of which at least r counters are reversal-bounded. We shall deal only with counter-automata that have a one-way read-only input tape (also known as on-line automata) and are restricted to work in realtime, quasi-realtime or in linear time.

Definition 3.1. A blind k-counter automaton $M := (Q, \Sigma, \delta, q_0, Q_{fin})$ consists of a finite set of states Q, a designated initial state $q_0 \in Q$, a designated set of final states $Q_{fin} \subseteq Q$, a finite input alphabet Σ, and a transition function $\delta : Q \times (\Sigma \cup \{\lambda\}) \to 2^{Q\times\{+1,0,-1\}^k}$.

An instantaneous description (*ID*) of M is an element of $Q \times \Sigma^* \times \mathbb{Z}^k$. We write $(q_1, aw, z_1, \ldots, z_k) \vdash_M (q_2, w, z_1 + \vec{\Delta}(1), \ldots, z_k + \vec{\Delta}(k))$ if $(q_2, \vec{\Delta}) \in \delta(q_1, a)$ where $(\vec{\Delta}(1), \ldots, \vec{\Delta}(k)) = \vec{\Delta}'$ is the transpose of the column vector $\vec{\Delta}$ and we omit the subscript M if no confusion will arise. $\vdash_M^*$ denotes the reflexive transitive closure of the computation relation $\vdash_M$ and is defined as usual from the n-step computation relations $\vdash_M^n := \vdash_M^{n-1} \circ \vdash_M$ by $\vdash_M^* := \bigcup_{i\geqslant 0} \vdash_M^i$, where $\vdash_M^0$ is the identity relation on the *ID*'s of the nondeterministic automaton *M*.

$ID_i \vdash_M^* ID_j$ is an accepting computation for w iff $ID_i := (q_0, w, 0, \ldots, 0))$ and $\exists q_e \in Q_{fin}$ such that $ID_j := (q_e, \lambda, 0, \ldots, 0))$.

$L(M) := \{w \in \Sigma^* \mid M$ has an accepting computation for $w\}$ is the language accepted by M.

A specific k-counter automaton M can be described by a finite state transition diagram in which a directed arc from state q_1 to q_2 is inscribed by the input symbol x to be processed and a vector $\vec{\Delta} \in \{+1, 0, -1\}^k$ used for updating the counters by adding the component $\vec{\Delta}(i)$ of $\vec{\Delta}$ to the current contents z_i of the ith counter. This will be written as $q_1 \xrightarrow[\Delta]{x} q_2$.

Definition 3.2. A blind k-counter automaton $M := (Q, \Sigma, \delta_M, q_0, Q_{fin})$ accepts $L(M)$ in linear time with factor $d \in \mathbb{N}$, if for any $w \in L(M)$ there exists an accepting n-step computation $ID_0 \vdash_M^n ID_1$ for w such that $n \leqslant d \cdot max(|w|, 1)$.

If there exists $d \in \mathbb{N}$ such that $(q_1, \lambda, z_1, \ldots, z_k) \vdash_M^n (q_2, \lambda, z_1', \ldots, z_k')$ implies $n \leqslant d$, then the automaton M is said to work in quasi-realtime of delay d. If in this case $d = 0$ then M works in realtime.

The ith counter $(1 \leqslant i \leqslant k)$ of some blind k-counter automaton M is reversal-bounded iff for any subcomputation $(q_0, w, 0, \ldots, 0) \vdash_M^* (q_1, w_1, x_1, \ldots, x_k) \vdash_M^* (q_2, w_2, y_1, \ldots, y_k) \vdash_M^* (q_3, w_3, z_1, \ldots, z_k)$ $x_i > y_i$ implies $y_i \geqslant z_i$.

By this definition, a reversal-bounded counter has to be increased first and decreased after its reversal. Counters that are first decreased and solely increased after one reversal can be replaced by those required by Definition 3.2. In addition, reversal bounded counters are forced by the finite control to perform at most one reversal on each computation, even in the nonaccepting ones!

Definition 3.3. For all $k, r \in \mathbb{N}$ let (k, r)-RBC denote the family of languages accepted by (k, r)-counter automata in quasi-realtime, i.e., are accepted in quasi-realtime by on-line counter automata having k blind counters of which at least r are reversal-bounded.

Obviously we have $\mathcal{M}(C_k) = (k, 0)$-RBC and $\mathcal{M}(B_k) = (k, k)$-RBC.

We will now identify the trio generator for the family (k, r)-RBC, which is the shuffle of disjoint copies of the respective generators of the one-counter families:

Definition 3.4. For $k, r \in \mathbb{N}$ with $k > 1$ and $r \leqslant k$ let $L_{k,r} := \{w \in \Gamma_k^* \mid \forall_{1 \leqslant i \leqslant r} \exists_{m_i \in \mathbb{N}} : \pi_i(w) = a_i^{m_i} b_i^{m_i} \wedge \forall_{r+1 \leqslant i \leqslant k} : |\pi_i(w)|_{a_i} = |\pi_i(w)|_{b_i}\}$.

Thus, $L_{k,r}$ is the shuffle of r disjoint copies of B_1, the generator of the trio of the reversal-bounded one-counter languages, and $k - r$ disjoint copies of C_1, the generator of the trio of the blind one-counter languages. By results from Ginsburg [10, Prop. 3.6.1, Prop. 5.1.1 and Theorem 5.5.1] and Ginsburg and Greibach [11, Corr. 3] it follows that $L_{k,r}$ is a generator of the family (k, r)-RBC. We do not give a detailed explanation using these standard techniques and state Lemma 3.5 without proof:

Lemma 3.5. *(k, r)-RBC $= \mathcal{M}(L_{k,r})$, for each $k, r \in \mathbb{N}$ with $k > 1$ and $r \leqslant k$.*

Ginsburg [10, Example 4.5.2] has shown $\mathcal{M}(B_k) \subsetneq \mathcal{M}(B_{k+1})$ and in [12,14] it was shown that $B_{k+1} \notin \mathcal{M}(D_k)$. Both results relied on the different dimension of the storage space, which is given by the number of counters.

We will sharpen the above results by proving $(k, r+1)$-RBC $\neq (k, r)$-RBC for all $k, r \in \mathbb{N}$ with $k \geqslant 1$ and $r < k$.

Since the families $(k, r+1)$-RBC and (k, r)-RBC are both defined by counter automata having the same number of counters, their distinct capability cannot be proved by arguments that base on the dimension of the storage space. We somehow have to look at the computations itself and the different loops that are possible in them. Since a computation which uses the reversal bounded counters may at the same time also use the other counters, we have to separate the effect of the computations on the different counters. We will give an informal outline of the proof, before starting with the formal details.

We first define a typical language, called $B_{k,r}$, which is an infinite subset of the generator of the trio (k, r)-RBC.

For a proof by contradiction, we next assume that the set $B_{k,r}$ is accepted by some $(k, r+1)$-counter automaton B with one more reversal-bounded counters as used for the class (k, r)-RBC.

As defined for any (k, r)-counter automaton, also for B the two matrices $B_{\bar{\Delta}}$ and B_Γ are defined. These matrices describe the behavior of the (k, r)-counter automaton, see Lemmas 3.8 and 3.10, that together give the equality $\{v \in R_B \mid B_{\bar{\Delta}} \cdot \psi(v) = \vec{0}\} = \{v \in R_B \mid \binom{B_{\bar{\Delta}}}{B_\Gamma} \cdot \psi(v) = \vec{0}\}$ which holds for all (k, r)-counter automata accepting subsets of the generator language C_k.

In several steps of constructions, we will define four infinite, nonregular subsets of the set R_B of all valid computation paths in B, accepting and nonaccepting ones as well. This yields the following sequence of inclusions: $R_B \supsetneq K_0 \supsetneq K_1 \supsetneq K_2 \supsetneq K_3$.

The sets K_i can most easily be described by the strings that are the input for the automaton B on the paths, that are coded by their elements:

K_0 consists of strings corresponding to accepting computations for words of the form $v_i = a_1^i b_1^i a_2^i b_2^i \cdots a_r^i b_r^i a_{r+1}^i b_{r+1}^{2i} a_{r+1}^i \cdots a_k^i b_k^{2i} a_k^i$, $i \in \mathbb{N}$.

In K_1, there exists only one computation path for each $v_i \in K_0$.

Then, K_2 consists of those path descriptions $w \in K_1$, which (for a fixed $p \in \mathbb{N}$) allow a decomposition $w = u_1 u_2 \cdots u_p$, such that (a) the string read by B in a subcomputation u_i is an element of $\{x\}^*$ for some symbol $x \in \Gamma_k$, and (b) no reversal takes place on any of the $r + 1$ reversal-bounded counters in such a subcomputation.

K_3 is, basically, only a convenient infinite subset of K_2, and still nonregular.

We then identify a certain regular subset L with $K_3 \subsetneq L \subsetneq R_B$. Since L is regular, its Parikh image $\psi(L)$ is a semilinear vector set.

From this set we extract a linear set of vectors $\psi(K)$ for $K \subseteq L$, of which we can show the existence of $r+1$ linearly independent elements, by proving $rank(\binom{B_{\vec{\Delta}}}{B_\Gamma} \cdot P) > rank\left(B_{\vec{\Delta}} \cdot P\right)^1$ for any $(k, r+1)$-counter automaton accepting $B_{k,r}$. But the above general equation for (k,r)-counter automata induces an equation over sets defined by the matrices of B and of K (see Eq. $(*)$ in the Proof of Lemma 3.13), which can be fulfilled only by sets of rank r, i.e., having at most r linearly independent elements. This then yields the desired contradiction.

We now give the notations and definitions, needed to allow the use of techniques from linear algebra in the proof of the main result (Lemma 3.13).

Definition 3.6. For any (k,r)-counter automaton $A := (Q, \Sigma, \delta_A, q_0, Q_{fin})$ let $G_A \subseteq Q \times \Sigma \times \{+1, 0, -1\}^k \times Q$ be the finite set defined by $G_A := \{(p, x, \vec{\Delta}, q) \mid (q, \vec{\Delta}) \in \delta_A(p, x)\}$, which is in bijection with the arcs of A's transition diagram. For later use let $n_A := |G_A|$ be the number of elements in the arbitrarily but fixed ordered set $G_A = \{g_1, g_2, \ldots, g_{n_A}\}$. (The ordering that is actually used will be described and determined later.)

The four mappings $f_i,\ 1 \leqslant i \leqslant 4$, are defined by: $f_1, f_4 : G_A \to Q$ with $f_1((p, x, \vec{\Delta}, q)) := p$ and $f_4((p, x, \vec{\Delta}, q)) := q$ as projections are mere codings, whereas $f_2 : G_A \to \Gamma$ and $f_3 : G_A \to \{+1, 0, -1\}^k$ are extended to homomorphisms in the obvious way. f_2 gives the input string that is composed from the projections onto the second coordinates, and f_3 yields the counter update after adding all the vectors in the projections onto the third coordinates. More precisely: $\forall : u, v \in G_A^*$ let $f_2 : G_A^* \to \Gamma^*$ with $f_2(uv) = f_2(u) f_2(v)$ and $f_3 : G_A^* \to \mathbb{Z}^k$ with $f_3(uv) = f_3(u) + f_3(v)$ be homomorphisms, where $+$ is the componentwise addition of the vectors $f_3(u)$ and $f_3(v)$. For an easier readability, let $\vec{\Delta}_{g_i} := f_3(g_i)$ denote the counter update induced by $g_i \in G_A$.

Let $R_A := \{g_0 g_1 \cdots g_t \mid \exists \in \mathbb{N} : \forall i \in \{0, \ldots t\} : (g_i \in G_A) \wedge (f_1(g_0) = q_0) \wedge (f_4(g_t) \in Q_{fin}) \wedge (f_4(g_i) = f_1(g_{i+1}) \text{ for } i \neq t)\} \subseteq G_A^*$ be the regular set describing all the accepting paths in A's transition diagram.

Of course, $w \in R_A$ does not imply that $f_2(w)$ will be accepted by A, since the final counter value may not be equal to zero.

On the basis of a given (k,r)-counter automaton $A := (Q, \Sigma, \delta_A, q_0, Q_{fin})$ two matrices $A_{\vec{\Delta}}$ and A_Γ are defined. $A_{\vec{\Delta}}$ collects all the possible counter changes of A's transitions and A_Γ accounts for the symbols from Γ_k that are to be read within each transition $g_i \in G_A$. These matrices help us to describe the behavior of the (k,r)-counter automaton A by matrix multiplication, see Lemmas 3.8 and 3.10, that together give the equality

$$\{v \in R_A \mid A_{\vec{\Delta}} \cdot \psi(v) = \vec{0}\} = \left\{ v \in R_A \;\middle|\; \begin{pmatrix} A_{\vec{\Delta}} \\ A_\Gamma \end{pmatrix} \cdot \psi(v) = \vec{0} \right\}$$

which holds for those (k,r)-counter automata that accept subsets of C_k. This is formulated in Lemma 3.11.

1 The rank of a matrix A (or a set L of vectors) is the maximal number of linearly independent rows of A (of elements within L, respectively).

Definition 3.7. $A_{\vec{\Delta}} \in \mathbb{Z}^{k \times n_A}$ is defined componentwise by $A_{\vec{\Delta}}(i, j) := \vec{\Delta}_{g_j}(i)$, for $1 \leqslant i \leqslant k$, $1 \leqslant j \leqslant n_A$.

Hence $A_{\vec{\Delta}}$ can be written as composite matrix: $A_{\vec{\Delta}} = (\vec{\Delta}_{g_1} \vec{\Delta}_{g_2} \cdots \vec{\Delta}_{g_{n_A}})$.

With the notation from Definition 3.6 we see that $A_{\vec{\Delta}} \cdot \psi(v) = f_3(v)$ for each $v \in G_A^*$ and the following is a immediate consequence of the definition of acceptance for (k, r)-counter automata:

Lemma 3.8. *Let $A := (Q, \Sigma, \delta_A, q_0, Q_{fin})$ be an arbitrary (k, r)-counter automaton then*

$$\forall v \in R_A : A_{\vec{\Delta}} \cdot \psi(v) = \vec{0} \text{ iff } f_2(v) \in L(A).$$

Proof. $v \in R_A$ ensures that there exists a path in the transition diagram of A beginning in q_0 and ending in some final state of Q_{fin}. If in addition $f_3(v) = A_{\vec{\Delta}} \cdot \psi(v) = \vec{0}$, then $f_2(v) \in L(A)$. Conversely, for any $w \in L(A)$ there exists an accepting path in A having a corresponding string $v' \in R_A$ with $w = f_2(v')$. Since a (k, r)-counter automaton accepts if the k-counters are empty at the beginning and at the end, it follows that $f_3(v') = A_{\vec{\Delta}} \cdot \psi(v') = \vec{0}$. $\square$

Definition 3.9. For each (k, r)-counter automaton $A := (Q, \Gamma_k, \delta_A, q_0, Q_{fin})$ the following matrix $A_\Gamma \in \{+1, 0, -1\}^{k \times n_A}$ is defined for each component $A_\Gamma(i, j)$, $1 \leqslant i \leqslant k$, $1 \leqslant j \leqslant n_A$, by:

$$A_\Gamma(i, j) := \begin{cases} 1 & \text{if } f_2(g_j) = a_i \\ -1 & \text{if } f_2(g_j) = b_i \\ 0 & \text{if } f_2(g_j) \notin \{a_i, b_i\}. \end{cases}$$

Without loss of generality the ordering of the elements in G_A is such, that

$$A_\Gamma = \begin{pmatrix} 1 & \dots & 1 & -1 & \dots & -1 & \dots & 0 & \dots & 0 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & \dots & 0 & 0 & \dots & 0 & \dots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \dots & 0 & \dots & 0 & 0 & \dots & 0 & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & 0 & \dots & 0 & \dots & 1 & \dots & 1 & -1 & \dots & -1 & 0 & \dots & 0 \end{pmatrix}$$
$$= \left(\vec{\gamma}_1 \vec{\gamma}_2 \cdots \vec{\gamma}_{n_A}\right),$$

where γ_j denotes the jth column of A_Γ, hence $\gamma_j(i) := A_\Gamma(i, j)$, for each $1 \leqslant j \leqslant n_A$ and $1 \leqslant i \leqslant k$. The columns $\gamma_j = \vec{0}$ on the right account for those transitions of A which occur when λ is used as input, i.e., $f_2(g_j) = \lambda$.

The next fact is obvious from the definitions and formulated without detailed proof:

Lemma 3.10. *Let $A := (Q, \Gamma_k, \delta_A, q_0, Q_{fin})$ be some (k, r)-counter automaton then*

$$\forall v \in G_A^* : \quad A_\Gamma \cdot \psi(v) = \vec{0} \text{ iff } f_2(v) \in C_k.$$

Combining the preceding Lemmas 3.8 and 3.10 we obtain an equality which is independent from the number of reversal bounded counters. However, through R_A, this equality is not independent of the language accepted:

Lemma 3.11. *Let $A := (Q, \Gamma_k, \delta_A, q_0, Q_{fin})$ be some (k, r)-counter automaton accepting $L(A) \subseteq C_k$, and let $\binom{A_{\vec{\Delta}}}{A_\Gamma}$ denote the compound matrix of dimension $2k \times n_A$ then*

$$\{v \in R_A \mid A_{\vec{\Delta}} \cdot \psi(v) = \vec{0}\} = \left\{v \in R_A \,\middle|\, \begin{pmatrix} A_{\vec{\Delta}} \\ A_\Gamma \end{pmatrix} \cdot \psi(v) = \vec{0}\right\}.$$

The following language $B_{k,r}$ is a bounded subset of $L_{k,r} \subsetneq C_k$ that captures all relevant information. The first r parts equal the product $B_1 B_2 \cdots B_r$, while the remaining parts are subsets of $\{a_s\}^*\{b_s\}^*\{a_s\}^*$, $r < s \leqslant k$, which, if separated as single languages, cannot be accepted by a 1-counter automaton allowing only one reversal. This fact can be shown for the 1-counter case by standard methods, since the subsets $\{a_s^{i_s} b_s^{i_s+j_s} a_s^{j_s} \mid i_s, j_s \in \mathbb{N}\}$ are not linear context-free. In case of more than one counter, this argument is no longer applicable.

Definition 3.12.

$$B_{k,r} := \{a_1^{i_1} b_1^{i_1} \cdots a_r^{i_r} b_r^{i_r} a_{r+1}^{i_{r+1}} b_{r+1}^{i_{r+1}+j_{r+1}} a_{r+1}^{j_{r+1}} \cdots a_k^{i_k} b_k^{i_k+j_k} a_k^{j_k} \mid \forall \mu : i_\mu, j_\mu \in \mathbb{N}\}.$$

Lemma 3.13. *$(k, r+1)$-RBC $\neq (k, r)$-RBC for all $k \in \mathbb{N}$ and $0 \leqslant r < k$.*

We will in fact prove $B_{k,r} \notin (k, r+1)$-RBC by contradiction, assuming that $B_{k,r} = L(B)$ for some $(k, r+1)$-counter automaton B. We will see, that the equation of Lemma 3.11 implies the following equation about certain semilinear sets of vectors (see Proof of Lemma 3.13):

$$\left\{\vec{Y} \in \mathbb{N}^h \mid B_{\vec{\Delta}} \cdot P \cdot \vec{Y} = -(B_{\vec{\Delta}}) \cdot \vec{C}\right\} = \left\{\vec{Y} \in \mathbb{N}^h \,\middle|\, \begin{pmatrix} B_{\vec{\Delta}} \\ B_\Gamma \end{pmatrix} \cdot P \cdot \vec{Y} = -\begin{pmatrix} B_{\vec{\Delta}} \\ B_\Gamma \end{pmatrix} \cdot \vec{C}\right\},$$

which cannot be satisfied, if $B_{k,r}$ is accepted by using $r+1$ reversal bounded counters. This will follow from $rank(\binom{A_{\vec{\Delta}}}{A_\Gamma} \cdot P) > rank(A_{\vec{\Delta}} \cdot P)$, as will be proved in Lemma 3.20 and this finally leads to the desired contradiction.

The linear set $\psi(K) = \{\vec{C} + P\vec{Y} \mid \vec{Y} \in \mathbb{N}^h\}$, of which we take the matrix P and the constant vector $\vec{C}$, used in the above equation, stems from the automaton B that is supposed to accept $B_{k,r}$. The set K is an infinite subset of the regular set R_B of paths in B's finite control. These notions will be defined later.

That this suffices is obvious, since $B_{k,r}$ is obtained from $L_{k,r}$ by intersection with an appropriate bounded regular set, hence $B_{k,r} \in (k, r)$-RBC is easily seen.

The proof of Lemma 3.13 is quite involved and needs a lot of definitions first. For the sake of contradiction, let us assume $B_{k,r} \in (k, r+1)$-RBC and let $B := (S_B, \Gamma_k, \delta_B, q_0, Q_{fin})$ be a blind k-counter automaton having $r+1$ reversal bounded counters that accepts $B_{k,r} = L(B)$. Without loss of generality, we assume that the first $r+1$ counters are reversal-bounded. As before, we let $n_B := |G_B|$ be the number of elements in the new set $G_B = \{g_1, g_2, \ldots, g_{n_B}\}$ as described earlier for arbitrary (k, r)-counter automata. $R_B \subseteq G_B^*$ is the regular set of accepting paths in B's finite control.

Within the set R_B we, step by step, identify infinite and nonregular subsets $K_i \subsetneq R_B$, $0 \leqslant i \leqslant 3$, to be used for the proof of Lemma 3.19. Each string in a set K_i describes a computation in B for a certain input from the accepted language $B_{k,r}$.

Definition 3.14. We define the set

$$K_0 := \{w \in R_B \mid \exists i \in \mathbb{N} : f_2(w) = a_1^i b_1^i a_2^i b_2^i \cdots a_r^i b_r^i a_{r+1}^i b_{r+1}^{2i} a_{r+1}^i \cdots a_k^i b_k^{2i} a_k^i\}$$

as a nonregular subset of R_B of which we then select the set $K_1 \subseteq K_0 \subsetneq R_B$ where no two different strings have an identical f_2-projection:

$K_1 := \{w \in K_0 \mid \forall w' \in K_0 : f_2(w) = f_2(w')$ implies $w = w'\}$.

By w_i we denote the unique string in K_1, for which

$$f_2(w_i) = a_1^i b_1^i a_2^i b_2^i \cdots a_r^i b_r^i a_{r+1}^i b_{r+1}^{2i} a_{r+1}^i \cdots a_k^i b_k^{2i} a_k^i.$$

In order to put our hands on the actions that automaton B performs while reading an input from the set $B_{k,r}$, we define a symmetric and reflexive relation ξ_k^l on G_B. For a step-by-step definition of the infinite subset $K_2 \subseteq K_1$ we use the property $p_{\xi_k^{r+1}}$ to specify certain strings within K_1.

Definition 3.15. For each $k \in \mathbb{N}, k \neq 0$ and each l, $1 \leqslant l < k$ let $\xi_k^l \subseteq G_B \times G_B$ be defined by

$$(g, g') \in \xi_k^l \text{ iff } \begin{cases} \exists x \in \Gamma_k : & f_2(g), f_2(g') \in \{x, \lambda\}, \quad \text{and} \\ \forall 1 \leqslant j \leqslant l : & \vec{\Delta}_g(j) > 0 \Longrightarrow \vec{\Delta}_{g'}(j) \geqslant 0, \\ & \vec{\Delta}_{g'}(j) > 0 \Longrightarrow \vec{\Delta}_g(j) \geqslant 0. \end{cases}$$

$(g, g') \in \xi_k^l$ means that the counter automaton B does not read two different symbols from the input by using g and g', if any at all, and these arcs do not force a reversal on any of the counters with index less or equal to l. The remaining counters with index strictly larger than l do not have any restriction on their updating. The relation ξ_k^l is obviously symmetric and reflexive but not necessarily transitive. So we can only find subsets of $C \subseteq G_B \times G_B$ which are transitively closed. Any such set will be called a ξ_k^l-clique. Such a clique reflects those parts of B's computation on substrings of a_s^* or of b_s^*, $s \leqslant k$, on which the counters with index less or equal l do not perform any reversal. Transitions $g, g' \in G_B$ may be used within loops while B accepts strings of the language $B_{k,r}$.

Definition 3.16. $p_{\xi_k^{r+1}} : G_B^* \times \mathbb{N} \to \{\mathit{true}, \mathit{false}\}$ is defined by: $p_{\xi_k^{r+1}}(w, p) = \mathit{true}$ iff $\exists u_1, \ldots, u_p \in G_B^*$:

(1) $w = u_1 u_2 \cdots u_p$ and

(2) $\forall j,\ 1 \leqslant j \leqslant p : \forall g, g' \in G_B : g, g' \sqsubseteq u_j \Longrightarrow (g, g') \in \xi_k^{r+1}$

(3) $\forall j,\ 1 \leqslant j < p : \exists g, g' \in G_B : g \sqsubseteq u_j \wedge g' \sqsubseteq u_{j+1} \wedge (g, g') \notin \xi_k^{r+1}$.

For each $u \in G_B^*$ let $G(u)$ denote the set of symbols from G_B that occur in the string u, i.e., $G(u) := \{g \in G_B \mid g \sqsubseteq u\}$, where $\sqsubseteq$ denotes the substring relation.

$G(u_j)$ forms a ξ_k^{r+1}-clique for each u_j of the decomposition $w = u_1 u_2 \cdots u_p$. If two arcs $g, g' \in G_B$ are in the same ξ_k^{r+1}-clique, then there exists $x \in \Gamma_k$ such that $f_2(g), f_2(g') \in \{\lambda, x\}$ and their f_3-projections do not lead to a reversal on one of the first $r+1$ counters. The change between two ξ_k^{r+1}-cliques can thus be forced either by changing the symbols ($\neq \lambda$) of the f_2-projections or by performing a reversal on one of the first $r+1$ counters.

For each $w_i \in K_1$ we have $p_{\xi_k^{r+1}}(w_i, p) = \mathit{true}$ implies $p \leqslant 3k+1$. This is seen as follows: There exist at most $2r + 3(k-r)$ different ξ_k^{r+1}-cliques with a component from Γ_k, since there are at most that many different blocks of consecutive identical symbols. Because $(g, g') \in \xi_k^{r+1}$ also allows $f_2(g) = f_2(g') = \lambda$, some of these λ-arcs may fall into the neighboring ξ_k^{r+1}-clique, as long as the according transitions do not force a reversal on one of the first $r+1$ counters. At most $r+1$ reversals may fall into the $2r+3(k-r)$ different blocks, which allows for $r+1$ additional substrings in the decomposition of $w_i = u_1u_2\cdots u_p$ and $p \leqslant 2r+3(k-r)+r+1 = 3k+1$.

Since k is a constant and K_1 is infinite, there exists some $p \leqslant 3k+1$ such that infinitely many strings $w \in K_1$ satisfy $p_{\xi_k^{r+1}}(w,p) = \mathit{true}$. This gives rise to the subset $K_2 \subseteq K_1$ defined next.

Definition 3.17. Let $p \leqslant 3k+1$ be fixed and such that $K_2 := \{w \in K_1 \mid p_{\xi_k^{r+1}}(w,p) = \mathit{true}\}$ is infinite.

Since G_B is finite there exists a fixed string $w_g := g_{l,1}g_{l,2}\cdots g_{l,p} \in G_B^*$ where $g_{l,j}$ is the leftmost symbol of u_j for each $1 \leqslant j \leqslant p$ in the decomposition of $w = u_1u_2\cdots u_p$ for infinitely many strings $w \in K_2$. These strings are collected in the set $K_3 \subseteq K_2$:

Definition 3.18. Let $w_g = g_{l,1}g_{l,2}\cdots g_{l,p} \in G_B^*$ be fixed and such, that $K_3 := K_2 \cap \{g_{l,1}\}G(u_1)^*\{g_{l,2}\}G(u_2)^*\cdots\{g_{l,p}\}G(u_p)^*$ is infinite.

The set K_3, $K_3 \subseteq K_2 \subseteq K_1$, is not regular, but we shall find an infinite regular set $L \subseteq \{g_{l,1}\}G(u_1)^*\{g_{l,2}\}G(u_2)^*\cdots\{g_{l,p}\}G(u_p)^*$ such that $K_3 \subsetneq L \subsetneq R_B$.

Such a set is is needed, because we then have a semilinear Parikh image available, from which we can extract an infinite linear subset that will be plugged into the matrix equation of Lemma 3.11.

For each j, $1 \leqslant j \leqslant p$, let L_j be the regular set accepted by the finite automaton $B_j := (Q_j, G(u_j), \delta_j, f_1(g_{l,j}), Q_{j,\mathit{fin}})$, where
(1) $Q_j := \{f_1(g), f_4(g) \mid g \in G(u_j)\}$,
(2) $\delta_j : Q_j \times G(u_j) \to Q_j$ is given by $\delta_j(f_1(g), g) := f_4(g)$,
(3) $Q_{j,\mathit{fin}} := f_4(g')$, where g' is the rightmost symbol of u_j.
Each automaton B_j accepts all substrings from $R_B \subseteq G_B^*$ that begin in that state of B's finite control which is the source state $f_1(g_{l,j})$ of the leftmost transition symbol $g_{l,j}$ of u_j and end in the target state $f_4(g')$ of the last transition symbol of u_j. Hence, among many other strings, also $u_j \in L_j$ for the subcomputation u_j in an element of K_3. Since each accepting path in the automaton B_j is a part of an accepting path in B's transition diagram, we see that $K_3 \subseteq L \subseteq R_B$ for $L := L_1L_2\cdots L_p$. Moreover, at least $3k-r$ languages among the $L_1, L_2, \ldots .L_p$ must be infinite, since the projection of the elements of K_3 onto the elements of Γ_k are infinite for each of the $2r+3(k-r)$ blocks of identical symbols. Since L is regular, the Parikh-image $\psi(L) = \sum_{1\leqslant j\leqslant p}\psi(L_j)$ is a semilinear set and infinite, too. The sum is understood elementwise for the p semilinear sets $\psi(L_j)$. Each linear subset of $\psi(L_j)$ has a representation of the form:

$$\{\vec{C}_j + P_j\vec{Y} \mid \vec{Y} \in \mathbb{N}^{h_j}\} \text{ for some } h_j \geqslant 1,\ \ \vec{C}_j \in \mathbb{N}^{n_B},\ \text{ and }\ P_j \in \mathbb{N}^{n_a \times h_j}.$$

With these preliminaries, we can formulate and prove the following important result:

Lemma 3.19. *There exists an infinite set $K \subseteq R_B$ such that (a)–(c) hold:*
(a) $\psi(K) = \{\vec{C} + P\vec{Y} \mid \vec{Y} \in \mathbb{N}^h\}$ *for some* $h \in \mathbb{N}$, $\vec{C} \in \mathbb{N}^{n_B}$, *and* $P \in \mathbb{N}^{n_B \times h}$,
(b) *If* $P(s,j)\cdot P(t,j) \neq 0$ *for* $1 \leqslant j \leqslant h$, $1 \leqslant s,t \leqslant n_B$ *then* $(g_s, g_t) \in \xi_k^{r+1}$,
(c) $\forall n_0 \in \mathbb{N} : \exists \vec{Y}_0 \in \mathbb{N}^h : (\forall j : 1 \leqslant j \leqslant h \wedge \vec{Y}_0(j) > n_0) \wedge \vec{C} + P\vec{Y}_0 \in \psi(K)$.

Proof. By definition of the finite automaton B_j the matrix P_j satisfies (b) of Lemma 3.19. Given $L := L_1 L_2 \cdots L_p$ we choose for each L_j a linear subset $S_j := \{\vec{C}_j + P_j \vec{Y} \mid \vec{Y} \in \mathbb{N}^{h_j}\} \subseteq \psi(L_j)$ which should be infinite whenever L_j is infinite. The set $S := \{\vec{C}_S + P_S \vec{Y} \mid \vec{Y} \in \mathbb{N}^{h_S}\}$ combines the linear sets S_j and is linear and infinite, too. The linear set S is specified by $\vec{C}_S := \sum_{j=1}^{p} \vec{C}_j$, where the sum of the column vectors are taken componentwise, $h_S := \sum_{j=1}^{p} h_j$, and the compound matrix is given by $P_S := \left(P_1 P_2 \cdots P_p\right) \in \mathbb{N}^{n_B \times h_S}$. The matrix P_S satisfies property (b) of Lemma 3.19, since each submatrix P_j fulfilled this property. Now, $K := L \cap \psi^{-1}(S)$ is an infinite subset of L containing infinitely many elements from K_3, thus satisfying properties (a) and (b) of Lemma 3.19. Of course we had to choose the appropriate linear subsets of each L_j to see that $\psi^{-1}(S)$ contains infinitely many elements of $K_3 \subsetneq L = L_1 L_2 \cdots L_p$, but this is obviously possible.

Now we modify the matrix P_S by omitting certain columns to obtain a matrix P that in addition satisfies (c) of the Lemma. First, $K \cap K_3 \subseteq R_B$ is infinite, so that there exists an infinite set $M \subseteq \mathbb{N}^{h_S}$ such that $\psi(K \cap K_3) = \{\vec{C}_S + P_S \vec{Y} \mid \vec{Y} \in M\}$. Let $\vec{m}_0, \vec{m}_1, \ldots, \vec{m}_i, \ldots,$ be any enumeration of the elements of $M = \{\vec{m}_i \mid i \in \mathbb{N}\}$. Then there exists a subset $M' = \{\vec{m}_{i_j} \mid \forall j \in \mathbb{N} : i_j \in \mathbb{N} \wedge \vec{m}_{i_j} \in M \wedge i_j < i_{j+1}\} \subseteq M$ such that for each t, $1 \leqslant t \leqslant h_S$:

either	$\vec{m}_{i_1}(t) = \vec{m}_{i_2}(t)$	for all $i_1, i_2 \in \mathbb{N}$,
or	$\vec{m}_{i_1}(t) < \vec{m}_{i_2}(t)$	for all $i_1 < i_2$.

This means, that for each coordinate t, $1 \leqslant t \leqslant h_S$ either $\vec{m}(t)$ is constant for all $\vec{m} \in M'$ or is strictly increasing. This result is a variant of Dickson's Lemma and shown in Appendix A as Lemma A.3.

From M' we deduce the following index sets and constants:

$$\begin{aligned}
&I_{le} := \{j \mid 1 \leqslant j \leqslant h_S,\ \forall l \geqslant 1 : \vec{m}_{i_l}(j) < \vec{m}_{i_{l+1}}(j)\},\\
&I_{eq} := \{j \mid 1 \leqslant j \leqslant h_S,\ \forall l \geqslant 1 : \vec{m}_{i_l}(j) = \vec{m}_{i_{l+1}}(j)\}, \text{ and}\\
&c_j := \vec{m}_{i_1}(j) \text{ for each } j \in I_{eq}.
\end{aligned}$$

Now,

$$\psi(K \cap K_3) = \{\vec{C}_S + P_S \vec{Y} \mid \vec{Y} \in M\} \supseteq \left\{ \vec{C}_S + \sum_{j \in I_{eq}} P_S(:, j) c_j + P\vec{Y} \mid \vec{Y} \in M'' \right\},$$

where $P \in \mathbb{N}^{n_B \times h}$ is obtained from P_S by omitting the columns $P_S(:, j)$ having index $j \in I_{eq}$, $h := h_S - \left|I_{eq}\right|$, and M'' is obtained from M' by omitting all components j, where $j \in I_{eq}$. Thereby, $M'' \subsetneq \mathbb{N}^h$ is an infinite set which can be linearly ordered by $<$ and this relation applies to all components of its elements. Thus, also property (c) of Lemma 3.19 is satisfied, and the proof is finished. □

The existence of an arbitrarily large vector in the set of factors defining the set $\psi(K)$ was shown with the help of Lemma A.3, a variant of the well known Lemma of Dickson. By further applying facts from linear algebra (Lemma A.1, see any text book on this subject, e.g. [26]), it is possible to verify the necessity of more independent elements in the solution set of an matrix equation, than actual possible.

Lemma 3.20.

$$rank\left(\begin{pmatrix} B_{\vec{\Delta}} \\ B_\Gamma \end{pmatrix} \cdot P\right) > rank\left(B_{\vec{\Delta}} \cdot P\right).$$

Proof. Let

$$\begin{pmatrix} \vec{Y}_1 & \vec{Y}_2 & \cdots & \vec{Y}_h \\ \vec{Z}_1 & \vec{Z}_2 & \cdots & \vec{Z}_h \end{pmatrix} = \begin{pmatrix} B_{\vec{\Delta}} \\ B_\Gamma \end{pmatrix} \cdot P,$$

where for each $1 \leqslant j \leqslant h$, the columns $\vec{Y}_j := (B_{\vec{\Delta}} \cdot P)(:, j)$ of $B_{\vec{\Delta}} \cdot P$ are given by $\vec{Y}_j(l) := \sum_{i=1}^{n_B} B_{\vec{\Delta}}(l, i) P(i, j) := \sum_{i=1}^{n_B} \vec{\Delta}_{g_i}(l) \cdot P(i, j)$ for $1 \leqslant l \leqslant k$ and likewise $\vec{Z}_j := (B_\Gamma \cdot P)(:, j)$ denotes the jth column of $B_\Gamma \cdot P$ with $\vec{Z}_j(l) := \sum_{i=1}^{n_B} B_\Gamma(l, i) P(i, j)$. From (b) in Lemma 3.19 one concludes that each column $\vec{Z}_j$, $1 \leqslant j \leqslant h$, has at most one nonzero component: if $\vec{Z}_j(l) \neq 0$ then $\vec{Z}_j(i) = 0$ for each $i \neq l$.

We still have to verify:

$$rank\begin{pmatrix} \vec{Y}_1 & \vec{Y}_2 & \cdots & \vec{Y}_h \\ \vec{Z}_1 & \vec{Z}_2 & \cdots & \vec{Z}_h \end{pmatrix} > rank\left(\vec{Y}_1 \quad \vec{Y}_2 \quad \cdots \quad \vec{Y}_h\right).$$

By the definition of matrix B_Γ (Definition 3.9) and the construction of P (Lemma 3.19) one readily verifies that the rows of the compound matrix $(\vec{Z}_1 \quad \vec{Z}_2 \quad \cdots \quad \vec{Z}_h)$ are linearly independent. Let $\alpha_1, \alpha_2, \ldots \alpha_k$ and $\beta_1, \beta_2, \ldots \beta_k$ denote the rows of $(\vec{Y}_1 \quad \vec{Y}_2 \quad \cdots \quad \vec{Y}_h)$, respectively, those of $(\vec{Z}_1 \quad \vec{Z}_2 \quad \cdots \quad \vec{Z}_h)$.

Each word $w_i \in K \cap K_3$, can be written as

$$w_i = u_{1,1}^{(i)} u_{1,2}^{(i)} u_{2,1}^{(i)} u_{2,2}^{(i)} \cdots u_{r,1}^{(i)} u_{r,2}^{(i)} w_{r+1,1}^{(i)} w_{r+1,2}^{(i)} w_{r+1,3}^{(i)} \cdots w_{k,1}^{(i)} w_{k,2}^{(i)} w_{k,3}^{(i)},$$

where

(1) $f_2(u_{s,1}^{(i)}) = a_s^i$ and $f_2(u_{s,2}^{(i)}) = b_s^i$ for each s, $1 \leqslant s \leqslant r$,

(2) $f_2(w_{s,1}^{(i)}) = f_2(w_{s,3}^{(i)}) = a_s^i$ and $f_2(w_{s,2}^{(i)}) = b_s^{2i}$ for each s, $r+1 \leqslant s \leqslant k$.

If $P(i, j) \neq 0$ then for each $n_0 \in \mathbb{N}$ there exists $w \in K$ such that $|w|_{g_i} > n_0$. This follows from (c) in Lemma 3.19. Let $G_1 := \{g \in G_B \mid \forall n_0 \in \mathbb{N} : \exists w \in K : |w|_{g_i} > n_0\}$ be the set of all these arcs. Now we want to show that the row-space $\{\alpha_1, \alpha_2, \ldots, \alpha_k, \beta_1, \beta_2, \ldots, \beta_k\}$ contains strictly more linearly independent elements than the row-space $\{\beta_1, \beta_2, \ldots \beta_k\}$ if $B_{k,r} = L(B)$ for the counter automaton B having r+1 reversal bounded counters. We distinguish two cases:

Case 1. Assume that one of the reversal bounded counters will be changed by an arc $g \in G(w_{l,1}^{(i)} w_{l,2}^{(i)} w_{l,3}^{(i)}) \cap G_1$ for some $l, r+1 \leqslant l \leqslant k$. W.l.o.g. we assume that this is the first counter, hence $\vec{\Delta}_g(1) = f_3(g)(1) \neq 0$. By choosing two more arcs from $G(w_{l,1}^{(i)} w_{l,2}^{(i)} w_{l,3}^{(i)}) \cap G_1$ we can always find three elements $g_{\mu_1}, g_{\mu_2}, g_{\mu_3} \in G_A$, $1 \leqslant \mu_1, \mu_2, \mu_3 \leqslant n_B$, such that:

(1) $g \in \{g_{\mu_1}, g_{\mu_2}, g_{\mu_3}\}$,

(2) $g_{\mu_1} \sqsubseteq w_{l,1}^{(i)}$, $g_{\mu_2} \sqsubseteq w_{l,2}^{(i)}$, and $g_{\mu_3} \sqsubseteq w_{l,3}^{(i)}$,

(3) if $g \neq g_{\mu_j}$ for some $1 \leqslant j \leqslant 3$ then $f_2(g_{\mu_j}) \neq 0$.

Now consider two triples $\vec{y} := (y_1, y_2, y_3)$ and $\vec{z} := (z_1, z_2, z_3)$, where y_1, y_2, and y_3 are entries of the matrix $(\vec{Y}_1 \quad \vec{Y}_2 \quad \cdots \quad \vec{Y}_h)$ and z_1, z_2, and z_3 are entries of the matrix $(\vec{Z}_1 \quad \vec{Z}_2 \quad \cdots \quad \vec{Z}_h)$. We specify $\vec{y}$ and $\vec{z}$ as follows: for $1 \leqslant j \leqslant 3$ the elements y_j are located in the first row α_1 of $(\vec{Y}_1 \quad \vec{Y}_2 \quad \cdots \quad \vec{Y}_h)$

and the elements z_j are located in the lth row β_l of $(\vec{Z}_1 \quad \vec{Z}_2 \quad \cdots \quad \vec{Z}_h)$ and their crossing with some column $\begin{pmatrix}\vec{Y}_{m_j}\\ \vec{Z}_{m_j}\end{pmatrix}$, where $1 \leqslant m_j \leqslant h$ for $j \in \{1, 2, 3\}$, and m_j is such, that $P(\mu_j, m_j) \neq 0$. As mentioned before, each column of $(\vec{Z}_1 \quad \vec{Z}_2 \quad \cdots \quad \vec{Z}_h)$ has at most one single non*zero* entry. Since $f_2(g_{\mu_1}), f_2(g_{\mu_2}), f_2(g_{\mu_3}) \in \{a_l, b_l, \lambda\}$ these entries must occur in the lth row β_l of $(\vec{Z}_{m_1} \quad \vec{Z}_{m_2} \quad \vec{Z}_{m_3})$, hence applies to the elements z_1, z_2, and z_3. Consequently, if $\vec{y}$ is linearly independent of $\vec{z}$ then also the first row α_1 would be linearly independent of the rows $\beta_1, \beta_2, \ldots, \beta_k$. This (together with Lemma A.2 from Appendix A) would imply the statement of Lemma 3.20. Thus it suffices to prove that indeed $\vec{y}$ and $\vec{z}$ are linearly independent.

Among the cases $g = g_{\mu_1}$, $g = g_{\mu_2}$, or $g = g_{\mu_3}$ we select $g := g_{\mu_1}$ as subcase 1.1 (the remaining cases are similar):

By the choice of g we have $\vec{\Delta}_g(1) \neq 0$ which implies $y_1 \neq 0$. Now, either $\vec{z} = (0, -1, 1)$ or $\vec{z} = (1, -1, 1)$ by definition of $\{g_{\mu_1}, g_{\mu_2}, g_{\mu_3}\}$. Since $\vec{z} = (0, -1, 1)$ means independence of $\{\vec{y}, \vec{z}\}$ we proceed by assuming $\vec{z} = (1, -1, 1)$. Since the first counter is reversal bounded, only the following choices are possible for $\vec{y}$: $y_1 > 0,\ y_2 > 0,\ y_3 \neq 0$, *or* $y_1 > 0,\ y_2 < 0,\ y_3 < 0$, *or* $y_1 < 0,\ y_2 < 0,\ y_3 < 0$. It is immediately verified that in all these cases $\vec{y}$ is linearly independent of $\vec{z}$.

Case 2. We next have to consider the case, that for each $l,\ r + 1 \leqslant l \leqslant k$ no arc $g \in G(w^{(i)}_{l,1} w^{(i)}_{l,2} w^{(i)}_{l,3}) \cap G_1$ updates one of the $r + 1$ reversal bounded counters. Let $G_2 := G(w^{(i)}_{r+1,1} w^{(i)}_{r+1,2} w^{(i)}_{r+1,3} \cdots w^{(i)}_{k,1} w^{(i)}_{k,2} w^{(i)}_{k,3}) \cap G_1$ be the relevant set of these arcs. Again we consider matrices defined from columns of

$$\begin{pmatrix}\vec{Y}_1 & \vec{Y}_2 & \cdots & \vec{Y}_h\\ \vec{Z}_1 & \vec{Z}_2 & \cdots & \vec{Z}_h\end{pmatrix}.$$

Let $(\vec{Y}_{m_1} \quad \vec{Y}_{m_2} \quad \cdots \quad \vec{Y}_{m_q})$ and $(\vec{Z}_{m_1} \quad \vec{Z}_{m_2} \quad \cdots \quad \vec{Z}_{m_q})$ be the matrices consisting of those columns of $(\vec{Y}_1 \quad \cdots \quad \vec{Y}_h)$, respectively, of $(\vec{Z}_1 \quad \cdots \quad \vec{Z}_h$, for which $P(j, m_i) \neq 0$ and $g_j \in G_2$ where $1 \leqslant j \leqslant n_A$, and $1 \leqslant m_i \leqslant h$ for all $1 \leqslant i \leqslant q$. Now, $g \in G_2$ implies $\vec{\Delta}_g(j) = 0$ for $1 \leqslant j \leqslant r + 1$, since none of the reversal bounded counters is modified by an arc from the set G_2. Consequently, $\vec{Y}_{m_i}(j) = 0$ for $1 \leqslant j \leqslant r + 1$ and $1 \leqslant i \leqslant q$, so that $rank(\vec{Y}_{m_1} \quad \vec{Y}_{m_2} \quad \cdots \quad \vec{Y}_{m_q}) \leqslant k - (r + 1)$. On the other hand, $rank(\vec{Z}_{m_1} \quad \vec{Z}_{m_2} \quad \cdots \quad \vec{Z}_{m_q}) = k - r$, since $rank(\vec{Z}_1 \quad \vec{Z}_2 \quad \cdots \quad \vec{Z}_h) = k$ and r rows of $(\vec{Z}_{m_1} \quad \vec{Z}_{m_2} \quad \cdots \quad \vec{Z}_{m_q})$ have an entry equal to *zero* (recall $f_2(g) \notin \Gamma_r$ for $g \in G_2$). Also in case 2 the statement of the lemma has been proved. □

Proof of Lemma 3.13. From Lemma 3.11 we know that $L(B) = B_{k,r} \subseteq C_k$ implies $\{v \in R_B \mid B_{\vec{\Delta}} \cdot \psi(v) = \vec{0}\} = \{v \in R_B \mid \binom{B_{\vec{\Delta}}}{B_\Gamma} \cdot \psi(v) = \vec{0}\}$. We choose $K \subseteq R_B$ from Lemma 3.19(a) with $\psi(K) = \{\vec{C} + P \cdot \vec{Y} \mid \vec{Y} \in \mathbb{N}^h\}$. By Lemma 3.19(c) we find, that for each $n_0 \in \mathbb{N}$ there exists $\vec{Y}_0 \in \mathbb{N}^h$ such that $\forall j, 1 \leqslant j \leqslant h : \vec{Y}_0(j) > n_0$, and that there exists a string $w \in K \subsetneq R_B$ such that, $B_\Gamma \cdot \psi(w) = \vec{0}$, $B_{\vec{\Delta}} \cdot \psi(w) = \vec{0}$, $\vec{C} + P \cdot \vec{Y}_0 = \psi(w)$, and $f_2(w) \in B_{k,r}$. This yields the equation

$$(*) : \left\{\vec{Y} \in \mathbb{N}^h | B_{\vec{\Delta}} \cdot P \cdot \vec{Y} = -(B_{\vec{\Delta}}) \cdot \vec{C}\right\} = \left\{\vec{Y} \in \mathbb{N}^h \,\middle|\, \begin{pmatrix}B_{\vec{\Delta}}\\ B_\Gamma\end{pmatrix} \cdot P \cdot \vec{Y} = -\begin{pmatrix}B_{\vec{\Delta}}\\ B_\Gamma\end{pmatrix} \cdot \vec{C}\right\},$$

and we find an arbitrarily large vector $\vec{Y}_0 \in \{\vec{Y} \in \mathbb{N}^h | B_{\vec{\Delta}} \cdot P \cdot \vec{Y} = -(B_{\vec{\Delta}}) \cdot \vec{C}\}$.

By Lemmas A.1 and 3.20 we then see

$$rank\left\{\vec{Y} \in \mathbb{N}^h | B_{\vec{\Delta}} \cdot P \cdot \vec{Y} = -(B_{\vec{\Delta}}) \cdot \vec{C}\right\} > rank\left\{\vec{Y} \in \mathbb{N}^h \left| \begin{pmatrix} B_{\vec{\Delta}} \\ B_\Gamma \end{pmatrix} \cdot P \cdot \vec{Y} = -\begin{pmatrix} B_{\vec{\Delta}} \\ B_\Gamma \end{pmatrix} \cdot \vec{C}\right.\right\},$$

so that equation **(**$*$**)** cannot be fulfilled and $B_{k,r} \notin (k, r+1)$-RBC. □

The above results yield our main Theorem:

Theorem 3.21. *(k_1, r_1)-RBC $\subsetneq$ (k_2, r_2)-RBC iff $(k_1 < k_2)$ or $(k_1 = k_2$ and $r_1 > r_2)$.*

Proof. The mere inclusion (k_1, r_1)-RBC $\subseteq (k_2, r_2)$-RBC if $(k_1 < k_2)$ or $(k_1 = k_2$ and $r_1 > r_2)$, follows from the definition of the family (k, r)-RBC (Definitions 3.1 to 3.3). The strictness of (k_1, r_1)-RBC $\subsetneq$ (k_2, r_2)-RBC if $k_1 < k_2$ is verified as follows:

By definition we have (k, r_1)-RBC $\subseteq (k, 0)$-RBC for any $r_1 \leqslant k$, the strict inclusion $(k, 0)$-RBC $= \mathcal{M}(C_k) \subsetneq \mathcal{M}(B_{k+1}) = (k+1, k+1)$-RBC is Theorem 4.5, and again by definition $(k+1, k+1)$-RBC $\subseteq (k+1, r_2)$-RBC for any $r_2 \leqslant k+1$. Finally, the inequality $(k, r+1)$-RBC $\neq (k, r)$-RBC for all $k \in \mathbb{N}$ and $0 \leqslant r < k$ has been shown in Lemma 3.13. □

Corollary 3.22. *(k, r)-RBC is a twist-closed semi-AFL if and only if $r = 0$.*

Proof. We know $\mathcal{M}(B_k) = (k, k)$-RBC $\subseteq (k, r)$-RBC $\subseteq (k, 0)$-RBC $= \mathcal{M}(C_k)$ for each r with $k \geqslant r \geqslant 0$ from the definition. Now, if (k, r)-RBC $(k \geqslant r \geqslant 0)$ would be *twist*-closed, then it would be identical to the family $\mathcal{M}_{twist}(B_k) = \mathcal{M}(C_k)$ for all $k \geqslant 1$ by Lemma 4.3 and Theorem 4.4 and this would contradict Theorem 3.21. □

4. The language theoretic operation *twist*

Definition 4.1. Let Σ be an alphabet, then *twist* : $\Sigma^* \to \Sigma^*$ is defined for any $w \in \Sigma^*$ and $a \in \Sigma$ by: $twist(aw) := a \cdot twist(w^{rev})$, and $twist(\lambda) := \lambda$.

For languages L and families of languages $\mathcal{L}$ the operation *twist* is generalized as usual:

$$twist(L) := \{twist(w) \mid w \in L\} \text{ and } twist(\mathcal{L}) := \{twist(L) \mid L \in \mathcal{L}\}.$$

We can informally describe the *twist* of a string $w = uv$ as the exact (or literal) shuffle of its first half u with the reversal of its second half v. More precisely, one observes that $twist(x_1 x_2 \cdots x_{n-1} x_n) = x_1 x_n x_2 x_{n-1} \cdots x_{\lfloor n/2 \rfloor + 1}$ for any $x_i \in \Sigma$.

Twisting a context-free language obviously yields a context-sensitive language, and this inclusion must be proper since $twist(L)$ has a semilinear Parikh image whenever L has this property, i.e., $twist(\mathcal{C}f) \subsetneq \mathcal{C}s$.

Note that $twist(L)$ may not be context-free, even for L being linear context-free or a one-counter language:

It is easily verified that $twist(L_{lin}) \notin \mathcal{C}f$ and $twist(L_{count}) \notin \mathcal{C}f$ for $L_{lin} := \{a^{3m} b^n c^n d^m \mid n, m \in \mathbb{N}\}$ and $L_{count} := \{a^{3m} b^m c^n d^n \mid n, m \in \mathbb{N}\}$. One verifies $twist(C) \cap \{ad\}^* \{ac\}^* \{ab\}^* = \{(ad)^m (ac)^m (ab)^m \mid m \in \mathbb{N}\} \notin \mathcal{C}f$ for $C \in \{L_{lin}, L_{count}\}$.

In order to avoid cumbersome constructions and notation, it is helpful to informally understand the techniques that, formally written down, yield the proofs verifying the power of the *twist* in connection with trio operations, also known as rational transductions, which are generalized gsm-mappings.

In order to obtain an arbitrary shuffle of the reversal of the substring v of a string $w = uv \in \{a, b, c, x, y, z\}^*$ with the substring u one has to first certify that u and v use disjoint alphabets, say, $u \in \{a, b, c\}^*$ and $v \in \{x, y, z\}^*$. Then one has to insert anchor symbols $\$^k$ at those positions within the left part u where substrings of v^{rev} of length k should appear. Those parts of the left substring u which should not be changed have to be paired during the *twist* with marker symbols, say ¢, that had to be placed before applying the *twist* operation at the appropriate positions within the right substring v. Inserting the anchor and marker symbols by an inverse homomorphism yields an arbitrary number of them and at any position. In order to select only those strings that have the desired number and position of each symbol, one applies an inverse homomorphism h^{-1} that finally takes care for using only the correct number of $|v|$ many \$ and $|u|$ many ¢ symbols. The inverse homomorphism h^{-1} should find a pre-image only for those substrings where each anchor symbol \$ is to the left of a symbol from v's alphabet $\{x, y, z\}$ and any marker symbol ¢ is to the right of a symbol from u's alphabet $\{a, b, c\}$. Thus, h has to be defined on the alphabet $\{a, b, c, x, y, z\}$ by $h(a) := a¢$, $h(b) := b¢$, $h(c) := c¢$, $h(x) := \$x$, $h(y) := \$y$, and $h(z) := \$z$.

As a small example take $abcyxyz$. Inserting anchor and marker symbols at the desired positions gives $w := \$a\$b\$\$b¢\ yx¢\ y¢\ z$. Now we obtain $twist(w) = \$za¢\ \$yb¢\ \$x\$yb¢$ and $h^{-1}(twist(w)) = zayb\,xyb$. If one wants to obtain the reversal of a string v, one sets $u := \lambda$ and uses only the anchor symbols \$, hence takes $twist(\$^{|v|} \cdot v)$.

Using this technique one immediately obtains the following:

Lemma 4.2. *Any trio which is closed with respect to twist is also closed under reversal.*

There exist families of languages that are closed with respect to the operations *twist* and *product* but not w.r.t. intersection! The family $\mathcal{L}_{slip}$ of languages having a semi-linear Parikh image, i.e., being letter equivalent to regular sets, is such a family. This is because this family is not a trio since it is not even closed with respect to intersection by regular sets! To see this, consider the language $L := \{ab^{2^n} \mid n \in \mathbb{N}\} \cup \{b\}^*\{a\}^* \in \mathcal{L}_{slip}$, where one has $L \cap \{a\}\{b\}^* \notin \mathcal{L}_{slip}$.

This observation indicates that it might not be easy to express the operation *twist* by means of known operations in abstract formal language theory.

It is easy to construct a blind k-counter automaton M_2 that accepts L^{rev} from the automaton M_1 that accepts $L \in \mathcal{M}(C_k)$. In fact, there exist two slightly different methods: In the first, one just has to revert the arcs in the state transition diagram of M_1, replace $\vec{\Delta}$ by $-\vec{\Delta}$ in all their inscriptions and exchange the sets of final and initial states. In an accepting computation of length r in the new automaton M_2 the counter contents in the ith step will be the same as the one of M_1 in step $r - i$ of its mirrored computation.

The version of k-counter automata, used here, is one that starts and finishes an accepting computation with empty counters and allows arbitrary integers as counter contents. Hence, by the second method of proof, the reversal L^{rev} of $L = L(M_1)$ is accepted by an automaton M_3 obtained from M_1 by exchanging the sets of final and initial states and only reverting the arcs in M_1's state transition diagram without changing the counter updates $\vec{\Delta}$. If the counter contents of all of M_3's counters in the ith step of an accepting computation of length r equals $z \in \mathbb{Z}$, then $-z$ is the counter contents in step $r - i$ of its mirrored computation in automaton M_1.

Having this in mind, it is not difficult to show that each family of blind k-counter languages is *twist*-closed.

Lemma 4.3. *Each family $\mathcal{M}(C_k)$ is twist-closed, that is,* $\forall k \in \mathbb{N} : \mathcal{M}_{twist}(C_k) = \mathcal{M}(C_k)$.

Proof. Let $L \in \mathcal{M}(C_k)$, $L \subseteq \Sigma^*$, then there exists a blind k-counter automaton M which accepts $L = L(M)$ in quasi-realtime. For $k = 0$ the accepted set is regular and the proof is a nice exercise. In order to accept the set *twist*(L) in all the remaining cases ($k \neq 0$), we modify the k-counter automaton M to a new automaton M_{twist} that uses two copies of the original finite control of automaton M in alternating steps. One copy is identical to M and the other accepts $L(M)^{rev}$ and uses the unmodified counter updates, as described by the second method in the remark before. Formally, M_{twist} is defined as follows: Let Q (Q_0, and Q_{f}) be the set of states (initial and final states, resp.) of the automaton M, then $Q_{twist} := Q^2 \times \{o, e\}$ is the set of states of M_{twist}. The sets of initial (and final) states $Q_{0,twist}$ ($Q_{f,twist}$, resp.) of M_{twist} are given by $Q_{0,twist} := \{(p_0, p_f, o) \mid p_0 \in Q_0, p_f \in Q_{\mathrm{f}}\}$ and $Q_{f,twist} := \{(p, p, o), (p, p, e) \mid p \in Q\}$. Let $w \in L,\ w = x_1x_2x_3 \cdots x_{n-1}x_n,\ x_i \in \Sigma$ be a string of length $|w| = n$ which is accepted by M in a sequence of transitions $q_0 \xrightarrow[\vec{\Delta}^{(1)}]{x_1} q_1 \xrightarrow[\vec{\Delta}^{(2)}]{x_2} q_2 \xrightarrow[\vec{\Delta}^{(3)}]{x_3} q_3 \cdots q_{n-1} \xrightarrow[\vec{\Delta}^{(n)}]{x_n} q_n$.

The new automaton M_{twist} now uses the finite control of M in the first components of the elements in Q_{twist} in each odd step beginning with the first move, while it is used every second (even) step in the second components backwards. The sequence of transitions of M_{twist} accepting *twist*(w) now would be

$$\begin{pmatrix} q_0 \\ q_n \\ o \end{pmatrix} \xrightarrow[\vec{\Delta}^{(1)}]{x_1} \begin{pmatrix} q_1 \\ q_n \\ e \end{pmatrix} \xrightarrow[\vec{\Delta}^{(n)}]{x_n} \begin{pmatrix} q_1 \\ q_{n-1} \\ o \end{pmatrix} \xrightarrow[\vec{\Delta}^{(2)}]{x_2} \begin{pmatrix} q_2 \\ q_{n-1} \\ e \end{pmatrix} \xrightarrow[\vec{\Delta}^{(n-1)}]{x_{n-1}} \cdots \xrightarrow[\vec{\Delta}^{(\lfloor \frac{n}{2} \rfloor + 1)}]{x_{\lfloor \frac{n}{2} \rfloor + 1}} \begin{pmatrix} q_{\lceil \frac{n}{2} \rceil} \\ q_{\lceil \frac{n}{2} \rceil} \\ \xi \end{pmatrix},$$

where $\xi \in \{o, e\}$ depends on the length of the input string. The last step of this computation is

$$\begin{pmatrix} q_{\frac{n}{2}} \\ q_{\frac{n}{2}+1} \\ e \end{pmatrix} \xrightarrow[\vec{\Delta}^{(\frac{n}{2}+1)}]{x_{\frac{n}{2}+1}} \begin{pmatrix} q_{\frac{n}{2}} \\ q_{\frac{n}{2}} \\ o \end{pmatrix} \quad \text{if } n \text{ is even,}$$

and is

$$\begin{pmatrix} q_{\lfloor \frac{n}{2} \rfloor} \\ q_{\lceil \frac{n}{2} \rceil} \\ o \end{pmatrix} \xrightarrow[\vec{\Delta}^{(\lceil \frac{n}{2} \rceil)}]{x_{\lceil \frac{n}{2} \rceil}} \begin{pmatrix} q_{\lceil \frac{n}{2} \rceil} \\ q_{\lceil \frac{n}{2} \rceil} \\ e \end{pmatrix} \quad \text{otherwise.}$$

Conversely, every accepting computation in M_{twist} can be unfolded to yield a valid computation in M showing that only strings of the form *twist*(w), $w \in L(M)$ are accepted by M_{twist}. □

From what we have seen before, it is no surprise, that the generator C_k is obtainable from B_k using *twist* and rational transductions. This gives the following characterization, showing that the family $\mathcal{M}(C_k)$ of blind k–counter languages not only is *twist*-closed by itself, but equals the least *twist*-closed trio containing the family of reversal-bounded multicounter languages:

Theorem 4.4. $\mathcal{M}_{twist}(B_k) = \mathcal{M}(C_k)$, *for each* $k \in \mathbb{N}$ *with* $k > 1$.

Proof. From $B_k \in \mathcal{M}(C_k)$ and Lemma 4.3 we see $\mathcal{M}_{twist}(B_k) \subseteq \mathcal{M}(C_k)$. To show the converse we have to verify $C_k \in \mathcal{M}_{twist}(B_k)$.

Let $\not{c} \notin \Gamma_k$ and again

$$h_{\not{c}}(x) := \begin{cases} \lambda & \text{if } x = \not{c}, \\ x & \text{else,} \end{cases}$$

be the $\not{c}$-erasing homomorphism. Then $L_k := \mathit{twist}(h_{\not{c}}^{-1}(B_k)) \cap \{\not{c}b_i, a_j\not{c} \mid b_i, a_j \in \Gamma_k\}^*$ is an element of $\mathcal{M}_{\mathit{twist}}(B_k)$ and the paired symbols $\not{c}b_i$ and $a_j\not{c}$ may appear in any order for all $1 \leqslant i, j \leqslant k$. Hence, applying the inverse homomorphism $g : \Gamma_k^* \to (\Gamma_k \cup \{\not{c}\})^*$ with $g(a_i) := a_i\not{c}$ and $g(b_i) := \not{c}b_i$ we see $C_k = g^{-1}(L_k) \in \mathcal{M}_{\mathit{twist}}(B_k)$. $\square$

We slightly improve the statement $C_1 \in \mathcal{M}(B_3)$ shown by Greibach (Lemma 1 in [13]) by showing that it is sufficient to use only one more counter to accept C_k using only k+1 reversal-bounded counters:

Theorem 4.5. $\mathcal{M}(C_k) \subsetneq \mathcal{M}(B_{k+1})$, *for each* $k \in \mathbb{N}$ *with* $k > 1$.

Proof. "$\subseteq$". Let C_k be accepted by some blind k-counter automaton working in quasi-realtime. The new reversal-bounded $(k + 1)$-counter automaton M_{k+1} is given as follows: One counter, call it z_0, is used to nondeterministically find the middle of each string $w \in C_k$ by adding 1 in each step when reading a prefix u of $w = uv$. We call this the first phase of the work of M_{k+1}. Then, nondeterministically this phase is stopped and in the following, second, phase the counter z_0 is decreased by 1 in each and every step. One has $|u| = |v| = |w|/2$ if and only if this counter reached zero after reading the last symbol of w. All other counters z_1 to z_k are treated differently in the first and the second phase of M_{k+1}'s computation: If $\vec{\Delta}_1{}' = (\vec{\Delta}_1(1), \ldots, \vec{\Delta}_1(k))$[2] is a counter-update used in the first phase of M_k, then $\vec{\Delta}_2{}' := (+1, \vec{\Delta}_2(1) + 1, \ldots, \vec{\Delta}_2(k) + 1) \in \{2, 1, 0\}^{k+1}$ is the nondecreasing counter-update in M_{k+1} to be used instead. Likewise, if $\vec{\Delta}_3{}' = (\vec{\Delta}_3(1), \ldots, \vec{\Delta}_3(k))$ is a counter-update used in the second phase of M_k, then $\vec{\Delta}_3{}' := (-1, \vec{\Delta}_3(1) - 1, \ldots, \vec{\Delta}_3(k) - 1) \in \{-2, -1, 0\}^{k+1}$ is the nonincreasing counter-update to be used in M_{k+1} instead. Since the first and the second phase consist of equally many steps, the overall change of the counters is zero again, and exactly the strings form C_k are accepted using k+1 reversal-bounded partially blind counters. Since the new counter-updates now are elements of $\{-2, -1, 0, 1, 2\}^{k+1}$ and not of $\{-1, 0, 1\}^{k+1}$ we have to modify the automaton M_{k+1} by splitting moves that increase (or decrease) a counter by 2 into two moves that increase (or decrease) this counter by 1 and all other counters are treated as before in the first step and stay stationary in the second step. This modification gives a partially blind $(k + 1)$-counter automaton that accepts C_k in quasi-realtime and with at most one reversal on each of its counters, hence $\mathcal{M}(C_k) \subseteq \mathcal{M}(B_{k+1})$.

In order to see the strictness of this inclusion one uses known results from Ginsburg [10], Greibach [12], or Latteux [28,29], where it was shown that $B_{k+1} \cap \{a_1\}^* \cdots \{a_{k+1}\}^*\{b_{k+1}\}^* \cdots \{b_1\}^*$ is not an element of $\mathcal{M}(C_k)$, hence $\mathcal{M}(C_k) \subsetneq \mathcal{M}(B_{k+1}) \subseteq \mathcal{M}(C_{k+1})$ for each $k \geqslant 1$. $\square$

Consequently, $\bigcup_{i \geqslant 0} \mathcal{M}(C_i) = \mathcal{M}_{\cap}(C_1)$ forms a strict hierarchy of *twist*-closed semi-AFLs and therefore is not principal as *twist*-closed semi-AFL.

Using standard techniques we show that *PAL*, *dMIR*, *MIR*, *dCOPY*, and *COPY* all are generators of the same *twist*-closed trio $\mathcal{M}_{\mathit{twist}}(\mathit{PAL})$.

[2] $\vec{\Delta}_1{}'$ denotes the transpose of the column vector $\vec{\Delta}_1$.

Theorem 4.6. $\mathcal{M}_{twist}(dCOPY) = \mathcal{M}_{twist}(COPY) = \mathcal{M}_{twist}(MIR) = \mathcal{M}_{twist}(PAL) = \mathcal{M}_{twist}(dMIR)$.

Proof.
(a) $COPY \in \mathcal{M}_{twist}(dCOPY)$ follows since *COPY* is obtained from *dCOPY* by a simple coding.
(b) $MIR \in \mathcal{M}_{twist}(COPY)$ follows by observing that $MIR = twist(COPY)$, which can be shown by induction on the length and structure of the strings involved.
(c) $dMIR \in \mathcal{M}_{twist}(MIR)$, (d) $PAL \in \mathcal{M}_{twist}(dMIR)$, and (e) $MIR \in \mathcal{M}_{twist}(PAL)$ follow from the well known: $\mathcal{M}(dMIR) = \mathcal{M}(MIR) = \mathcal{M}(PAL)$.
(f) $dCOPY \in \mathcal{M}_{twist}(dMIR)$: obviously $K_2 := \{g_1(w)\$^i g_2(w^{rev})\ ¢^j \mid w \in \{a,b\}^*,\ i, j \in \mathbb{N}\} \in \mathcal{M}(dMIR)$. Likewise, $K_3 := twist(K_2) \cap (\{a_1, b_1\}\{¢\})^* \cdot (\{\$\}\{a_2, b_2\})^* \in \mathcal{M}_{twist}(dMIR)$ and then $dCOPY = h^{-1}(K_3)$ follows with $h : \Gamma_2^* \to (\Gamma_2 \cup \{\$, ¢\})^*$ defined by $h(a_1) := a_1¢,\ h(b_1) := b_1¢,\ h(a_2) := \$a_2,\ h(b_2) := \$b_2$. Consequently, $dCOPY \in \mathcal{M}_{twist}(dMIR)$. □

5. Reversal-bounded multipushdown languages

Since the mapping *twist* only performs a permutation of the symbols that form a string it is easily seen that $\mathcal{Re}$, $\mathcal{Rec}$, and $\mathcal{Cs}$ are *twist*-closed families. The family $\mathcal{M}_\cap(PAL)$ of quasi-realtime reversal-bounded multipushdown languages [4,5] will be shown to be yet another *twist*-closed family.

Lemma 5.1. *The family $\mathcal{M}_\cap(PAL)$ is closed with respect to the operation twist.*

Proof. Let $L \in \mathcal{M}_\cap(PAL)$ be accepted by some nondeterministic one-way reversal-bounded multipushdown automaton M_L which operates in such a way that in every computation each pushdown makes at most one reversal and runs in linear time, see [5]. In order to accept $K := twist(L)$ we use the automaton M_L and add one further pushdown store to obtain automaton M_K that accepts K as follows: M_K reads the symbols at odd positions of an input string $w \in K$, beginning with the first symbol of *w* and behaves on them exactly as the automaton M_L. Beginning with the second symbol of *w* the symbols at even positions alternatively are pushed onto the new pushdown. After having read the last symbol of the input string the symbols from the pushdown are popped and now treated as input for the automaton M_L. M_K accepts if the new pushdown is emptied and M_L accepts its input $twist^{-1}(w)$. Hence, M_K accepts $twist(L)$ and operates on each pushdown with only one reversal. It must be observed that M_K works in linear but not in quasi-realtime. That this is not a loss, follows from a result in [5] stating that the class $\mathcal{M}_\cap(dMIR) = \mathcal{M}_\cap(PAL)$ is closed with respect to linear erasing homomorphisms. □

Lemma 5.1 showed $\mathcal{M}_{twist}(PAL) \subseteq \mathcal{M}_\cap(PAL)$ and by the following result we will in fact prove equality of these two classes.

In [5], Theorem 3.1, it was proved that each $L \in \mathcal{M}_\cap(PAL)$ is the length preserving homomorphic image of the intersection of three linear context-free languages.

Definition 5.2. Let L_{rbc} denote the shuffle of three disjoint copies of *MIR*, which is formally defined by: $L_{\mathrm{rbc}} := \{w \in \Gamma_3^* \mid \forall i \in \{1, 2, 3\} : \pi_i(w) \in ind_i(MIR)\}$.

As before, one deduces with the results from Ginsburg and Greibach, and from Theorem 3.1 in [5], that the language L_{rbc} is a generator for $\mathcal{M}_\cap(PAL)$ as a trio $\mathcal{M}(L_{\mathrm{rbc}}) = \mathcal{M}_\cap(PAL)$.

We will now show, how to obtain the generator L_{rbc} from *MIR* by using only gsm-mappings, or the slightly more general rational transductions, and *twist*-operations, and thereby prove the following characterization:

Theorem 5.3. $\mathcal{M}_{twist}(PAL) = \mathcal{M}_{\cap}(PAL)$.

From *dMIR* we will obtain the generator L_{rbc} by using finitely man *twist* and rational transductions. The technique described to verify Lemma 4.2 will be used extensively and the various steps will be described informally, only. Formal details would hide the main concepts more than they clarify.

Proof. Let us first define L_1 as suitable variant of the language *dMIR* by $L_1 := \{w_1 w_2 w_3 \overline{w}_3^{rev} \overline{w}_2^{rev} \overline{w}_1^{rev} \in \Gamma_6^* \mid \exists u, v, w \in \Gamma^* : w_1 = ind_1(u), \overline{w}_1 = ind_2(u), w_2 = ind_3(v), \overline{w}_2 = ind_4(v), w_3 = ind_5(w), \overline{w}_3 = ind_6(w)\}$.

Obviously, $L_1 \in \mathcal{M}(dMIR) = \mathcal{M}(PAL)$ and the six components use disjoint alphabets. Starting with a string

$$\alpha := w_1 w_2 w_3 \overline{w}_3^{rev} \overline{w}_2^{rev} \overline{w}_1^{rev} \in L_1$$

one places anchor and marker symbols by a rational transduction at the appropriate places and obtains a string of the form:

$$\alpha' := w_1 \$^{k_1} w_2 \$^{k_2} w_3 \overline{w}_3^{rev} ¢^{k_3} \overline{w}_2^{rev} ¢^{k_4} \overline{w}_1^{rev} ¢^{k_5} \in \mathcal{M}(L_1) \text{ for some } k_i \in \mathbb{N}.$$

Twisting α' and applying an inverse homomorphism as a simple rational transduction yields:

$$\beta := w_1 \overline{w}_1 w_2 \overline{w}_2 w_3 \overline{w}_3^{rev}.$$

From β one gets

$$\gamma := (\overline{w}_1 w_2 \overline{w}_2 w_3 \overline{w}_3^{rev})^{rev} w_1 = \overline{w}_3 w_3^{rev} \overline{w}_2^{rev} w_2^{rev} \overline{w}_1^{rev} w_1.$$

Now, suitable placement of anchor and marker symbols yields the string

$$\gamma' := \$^{k_1} \overline{w}_3 w_3^{rev} \$^{k_2} \overline{w}_2^{rev} ¢^{k_3} w_2^{rev} ¢^{k_4} \overline{w}_1^{rev} w_1.$$

And one more twisting, followed by an invers homomorphism finally gives:

$$\delta := w_1^{rev} \overline{w}_1 \overline{w}_3 w_3^{rev} w_2 \overline{w}_2^{rev} \in dMIR \cdot dMIR' \cdot dMIR''.$$

The language L_2 obtained this way from L_1 is the product of three disjoint copies of *dMIR*.

From strings $uvw \in dMIR \cdot dMIR' \cdot dMIR''$ one obtains strings in $(u \sqcup\!\sqcup w)\{v\}$ using one more application of the *twist* operation as described for the proof of Lemma 4.2. Here $\sqcup\!\sqcup$ denotes the shuffle operation, which to two strings defines the set $u \sqcup\!\sqcup v = \{w \mid w = u_1 v_1 u_2 v_2 \cdots u_n v_n, u = u_1 u_2 \cdots u_n, v = v_1 v_2 \cdots v_n\}$, and is generalized to sets of strings by: $M_1 \sqcup\!\sqcup M_2 := \bigcup_{\substack{u \in M_1 \\ v \in M_2}} u \sqcup\!\sqcup v$.

With the same technique we get the strings from $(u \sqcup\!\sqcup w) \sqcup\!\sqcup v$, thus finding $L_{\text{rbc}} \in \mathcal{M}_{twist}(PAL)$, which proves the theorem since $\mathcal{M}(L_{\text{rbc}}) = \mathcal{M}_{\cap}(PAL)$. □

A consequence of this new characterization of the family of languages accepted in quasi-realtime by reversal-bounded one-way multipushdown automata we obtain jet another characterization of the recursively enumerable languages that was proven by a slightly different method in [15,16].

Corollary 5.4. $\mathcal{R}e = \hat{\mathcal{M}}_{twist}(PAL)$

Similar characterizations are to be found in [1,31] and later [7,8] where the class $\mathcal{M}_\cap(PAL)$, however, was not considered separately. With the techniques described and used before, we have not been able to show that both the twinshuffle language L_{TS} and the reverse twinshuffle language L_{RTS} are trio generators for the family $\mathcal{M}_\cap(PAL)$, but can use the *twist*-closure to formulate Corollary 5.5. The proof is a simple exercise, similar to the proof of Theorem 4.6(f), and left for the reader, who can find definitions of the languages L_{RTS} and L_{TS} in the references, cited above.

Corollary 5.5. $\mathcal{M}_{twist}(L_{TS}) = \mathcal{M}_{twist}(L_{RTS}) = \mathcal{M}_\cap(PAL)$

Acknowledgments

We thank Olaf Kummer for Lemma 4.3 and Theorem 4.5 and Berndt Farwer for the fruitful and detailed discussions on the preliminary version of this work. We gratefully acknowledge the hints from the referees for STACS 2001 and the competent and helpful comments by the referee for Information and Computation.

Appendix A

Let $A \in \mathbb{Z}^{n\times h}$ be of rank r, $\vec{B} \in \mathbb{Z}^n$, and $L := \{\vec{x} \in \mathbb{N}^h \mid A\vec{x} = \vec{B}\}$ be the set of all nonnegative solutions of the linear equation $A\vec{x} = \vec{B}$. It is known from linear algebra that each subset $M \subseteq L$ of linearly independent elements has cardinality of at most $(h - r)$.

Lemma A.1. *If $L := \{\vec{x} \in \mathbb{N}^h \mid A\vec{x} = \vec{B}\}$ for some $A \in \mathbb{Z}^{n\times h}$ of rank r and $\vec{B} \in \mathbb{Z}^n$. If for each $n \in \mathbb{N}$ there exists $\vec{x} \in L$ such that $\vec{x}(i) > n$ for each i, $1 \leqslant i \leqslant h$, then L contains a subset $M = \{\vec{x}_1, \vec{x}_2, \dots, \vec{x}_{h-r}\}$ of linearly independent elements.*

Proof. Let $\vec{y}_1, \vec{y}_2, \dots, \vec{y}_{h-r} \in \mathbb{Z}^h$ be linearly independent solutions of the homogenous linear equation $A\vec{x} = \vec{0}$ and define $n_0 := max\{|\vec{y}_i(j)| \,|1 \leqslant i \leqslant h, 1 \leqslant j \leqslant r - h\}$. Now, if $\vec{x}_0 \in L$ is a solution of the inhomogeneous linear equation $A\vec{x} = \vec{B}$ that satisfies $\vec{x}_0(i) > n_0$, then $\vec{x}_0 + \vec{y}_1\, \vec{x}_0 + \vec{y}_2, \dots, \vec{x}_0 + \vec{y}_{h-r}$ are linearly independent and nonnegative solutions of the equation $A\vec{x} = \vec{B}$. □

Lemma A.2. *Let $\vec{\alpha}_1, \vec{\alpha}_2, \dots, \vec{\alpha}_k, \vec{\beta}_1, \vec{\beta}_2, \dots, \vec{\beta}_k \in \mathbb{Z}^n$, $n \geqslant k$. If for some $\vec{\alpha} \in \{\vec{\alpha}_1, \vec{\alpha}_2, \dots, \vec{\alpha}_k\}$ the set $\{\vec{\alpha}, \vec{\beta}_1, \vec{\beta}_2, \dots, \vec{\beta}_k\}$ is a set of linearly independent elements, then there exists $\vec{\beta} \in \{\vec{\beta}_1, \vec{\beta}_2, \dots, \vec{\beta}_k\}$ such that $\{\vec{\beta}, \vec{\alpha}_1, \vec{\alpha}_2, \dots, \vec{\alpha}_k\}$ is a set of linearly independent elements, too.*

Proof. Assume that $\vec{\alpha}, \vec{\beta}_1, \vec{\beta}_2, \dots, \vec{\beta}_k$ are linearly independent for some $\vec{\alpha} \in \{\vec{\alpha}_1, \vec{\alpha}_2, \dots, \vec{\alpha}_k\}$ and each $\vec{\beta} \in \{\vec{\beta}_1, \vec{\beta}_2, \dots, \vec{\beta}_k\}$ is a linear combination of $\vec{\alpha}_1, \vec{\alpha}_2, \dots, \vec{\alpha}_k$. That means, that there exists some $(k \times k)$-matrix A, such that $(\vec{\alpha}_1\, \vec{\alpha}_2 \cdots \vec{\alpha}_k) \cdot A = (\vec{\beta}_1\, \vec{\beta}_2 \cdots \vec{\beta}_k)$, where $(\vec{\alpha}_1\, \vec{\alpha}_2 \cdots \vec{\alpha}_k) \in Z^{n\times k}$ is the matrix composed by the column-vectors $\vec{\alpha}_i$. Here A is of rank k, since $\{\vec{\beta}_1, \vec{\beta}_2, \dots, \vec{\beta}_k\}$ is a set of linearly independent elements and the inverse A^{-1} exists. Now $(\vec{\alpha}_1\, \vec{\alpha}_2 \cdots \vec{\alpha}_k) = (\vec{\beta}_1\, \vec{\beta}_2 \cdots \vec{\beta}_k) \cdot A^{-1}$ shows that each $\vec{\alpha} \in \{\vec{\alpha_1}, \vec{\alpha_2}, \ldots, \vec{\alpha_k}\}$ is a linear combination of $\vec{\beta_1}, \vec{\beta_2}, \ldots, \vec{\beta_k}$, contradicting the assumption. Consequently not all elements of the set $\{\vec{\beta_1}, \vec{\beta_2}, \ldots, \vec{\beta_k}\}$ can be a linear combination of $\vec{\alpha_1}, \vec{\alpha_2}, \ldots, \vec{\alpha_k}$ and there exists at least one $\vec{\beta} \in \{\vec{\beta_1}, \vec{\beta_2}, \ldots, \vec{\beta_k}\}$ that is linearly independent of $\{\vec{\alpha_1}, \vec{\alpha_2}, \ldots, \vec{\alpha_k}\}$. □

Lemma A.3 is only slightly stronger than the one by Dickson, stating that each infinite sequence of vectors contains an infinite nondecreasing subsequence.

Lemma A.3. *Each infinite sequence $\sigma := \vec{m}_1, \vec{m}_2, \ldots, \vec{m}_i, \ldots$ of elements $\vec{m}_i \in N^n$ has an infinite subsequence $\sigma' := \vec{m}_{i_1}, \vec{m}_{i_2}, \ldots, \vec{m}_{i_r}, \ldots$, such that for each j, $1 \leqslant j \leqslant n$, either $\vec{m}_{i_r}(j) > \vec{m}_{i_{r+1}}(j)$ for all $r \geqslant 1$ or $\vec{m}_{i_r}(j) = \vec{m}_{i_{r+1}}(j)$ for all $r \geqslant 1$.*

Proof. For each infinite sequence $\sigma := \vec{m}_1, \vec{m}_2, \ldots, \vec{m}_i, \ldots$ of elements $\vec{m}_i \in N^n$ let $M_\sigma := \{\vec{m}_i \mid \vec{m}_i$ occurs within $\sigma\}$. We proceed by induction on n:

Basic step: For $n = 1$, $\sigma := \vec{m}_1, \vec{m}_2, \ldots, \vec{m}_i, \ldots$ is an infinite sequence of not necessarily different elements. If M_σ is finite, then there exists (at least one) $\vec{m} \in M_\sigma$ such that $\{i \mid \vec{m}_i = \vec{m}\}$ is infinite and the following subsequence σ' can be chosen: $\sigma' := \vec{m}_{i_1}, \vec{m}_{i_2}, \ldots, \vec{m}_{i_j}, \ldots$ where $\vec{m}_{i_r} = \vec{m}$ for each $r \geqslant 1$. Here $\{j \mid \vec{m} = \vec{m}_j\} = \{i_j \mid \vec{m} = \vec{m}_{i_j}\}$ and σ' satisfies the requirement.

On the other hand, if M_σ is not finite, then there exists an infinite subsequence with property **(☆)**: M_σ is infinite and each $m \in M_\sigma$ occurs exactly once within the sequence σ', that is, $\vec{m}_{i_s} \neq \vec{m}_{i_t}$ iff $s \neq t$.

This sequence will be defined by repeatedly applying the following transformation **R** on the current sequence σ. **R** has two parameters: σ and some $i \in \mathbb{N}$ and is defined as follows:
$\mathbf{R}(\sigma, i)$ is: remove all elements $\vec{m}_t$ from σ for which $t > i$ and $\vec{m}_t = \vec{m}_i$.

If $\sigma' := \mathbf{R}(\sigma, i)$ is the infinite sequence obtained by performing **R** once for some fixed $i \in \mathbb{N}$, then obviously $M_\sigma = M_{\sigma'}$ and σ' still is an infinite sequence. Hence, if σ' is the sequence obtained from σ successively applying $\mathbf{R}(\sigma, i)$ for each $i \geqslant 1$, then σ' has the desired property **(☆)**.

Starting with an infinite sequence $\sigma := \vec{m}_1, \vec{m}_2, \ldots, \vec{m}_i, \ldots$, ($\vec{m}_i \in \mathbb{N}$), that satisfies **(☆)**, the Lemma of Dickson shows, that there exists an infinite subsequence $\sigma' := \vec{m}_{i_1}, \vec{m}_{i_2}, \ldots, \vec{m}_{i_j}, \ldots$ of σ where $\vec{m}_{i_r} < \vec{m}_{i_{r+1}}$ for each $r > 1$. and this proves the induction basis.

Induction step: Assuming the Lemma to be true for some fixed $m \in \mathbb{N}$ let $\sigma := \vec{m}_1, \vec{m}_2, \ldots, \vec{m}_i, \ldots$ be an infinite sequence of elements $\vec{m}_i \in N^{m+1}$. By considering the first m components of the elements of σ there exists an infinite subsequence $\sigma' := \vec{m}_{i_1}, \vec{m}_{i_2}, \ldots, \vec{m}_{i_j}, \ldots$ of σ such that the projection onto the first m components yields an infinite subsequence for which the lemma holds. Now consider the last component of the elements forming the sequence σ'. Apply the selection mechanism to the elements of σ' according to the basic step for the one dimensional sequence to finally obtain an infinite subsequence of σ', hence of σ, which satisfies the requirements of the lemma in all coordinates. □

References

[1] B.S. Baker, R.V. Book, Reversal-bounded multipushdown machines, J. Comput. Syst. Sci. 8 (1974) 315–332.
[2] J. Berstel, Transductions and Context-free Languages, Teubner, Stuttgart, 1980.
[3] R.V. Book, S. Greibach, Quasi-realtime languages, Math. Syst. Theory 19 (1970) 97–111.
[4] R.V. Book, S. Greibach, The independence of certain operations on semiAFLs, RAIRO Informatique Théorique 19 (1978) 369–385.

[5] R.V. Book, M. Nivat, M. Paterson, Reversal-bounded acceptors and intersections of linear languages: SIAM J. Comput. 3 (1974) 283–295.
[6] F.J. Brandenburg, Representations of language families by homomorphic equality operations and generalized equality sets, Theoret. Comput. Sci. 55 (1987) 183–263.
[7] J. Engelfriet, Reverse twin shuffles, Bull. EATCS 60 (1996) 144.
[8] J. Engelfriet, G. Rozenberg, Fixed point languages, equality languages and representations of the recursivelz enumerable languages, J. ACM 27 (1980) 499–518.
[9] P.C. Fischer, A.R. Meyer, A.L. Rosenberg, Counter machines and counter languages, Math. Syst. Theory 2 (1968) 265–283.
[10] S. Ginsburg, Algebraic and Automata Theoretic Properties of Formal Languages, North-Holland, Amsterdam, 1975.
[11] S. Ginsburg, S. Greibach, Principal AFL, J. Comput. Syst. Sci. 4 (1970) 308–338.
[12] S. Greibach, Remarks on the complexity of nondeterministic counter languages, Theoret. Comput. Sci. 1 (1976) 269–288.
[13] S. Greibach, Remarks on blind and partially blind one-way multicounter machines, Theoret. Comput. Sci. 7 (1978) 236–311.
[14] M. Jantzen, On the hierarchy of Petri net languages, R.A.I.R.O., Informatique Théorique 13 (1979) 19–30.
[15] M. Jantzen, On twist-closed trios: a new morphic characterization of the r.e. sets, Foundations of Computer Science, Lecture Notes in Computer Science, vol. 1337, Springer-Verlag, Heidelberg, 1997, pp. 135–142.
[16] M. Jantzen, On twist-closed trios, Department of Informatics, University of Hamburg, Technical Report No. FBI-HH-B-204/97, 1997.
[17] M. Jantzen, Hierarchies of principal twist-closed trios, in: Proceedings of the 15th International Symposium on STACS ’98, Paris, Lecture Notes in Computer Science, vol. 1373, Springer, Heidelberg, 1998, pp. 344–355.
[18] M. Jantzen, H. Petersen, Petri net languages and one-sided Dyck-reductions on context-free sets, in: K. Voss, H. Genrich, G. Rozenberg (Eds.), Concurrency and Nets, Springer, Berlin, 1987, pp. 245–252.
[19] M. Jantzen, H. Petersen, Twisting Petri net languages and how to obtain them by reducing linear context-free sets, in: Proceedings of the 12th International Conference on Petri Nets, Gjern, 1991, pp. 228–236.
[20] M. Jantzen, H. Petersen, Cancellation in context-free languages: enrichment by reduction, Theoret. Comput. Sci. 127 (1994) 149–170.
[21] M. Jantzen, A.N. Kurganskyy, Refining the hierarchy of blind multicounter languages, in: Proceedings of the 18th International Symposium on STACS’01, Dresden, Lecture Notes in Computer Science, vol. 2010, Springer, Heidelberg, 2001, pp. 376–387.
[22] M. Jantzen, A.N. Kurganskyy, Refining the hierarchy of blind multicounter languages, Department of Informatics, Univeristy of Hamburg, Technical Report No. FBI-HH-B-229/01, 2001.
[23] J. Kortelainen. Properties of trios and AFLs with bounded or commutative generators, Department of Mathematics, University of Oulu, Finland, Technical Report No. 53, 1980.
[24] S.R. Kosaraju, Decidability of reachability of vector addition systems, in: Proceedings of the 14th Annual ACM Symposium on Theory of Computing, San Francisco, 1982, pp. 267–281.
[25] A.N. Kurganskyy, On the hierarchy of the blind multicounter automata (in Russian), in: Proceedings of the 12th International Conference on Theoretical Questions of Cybernetics, vol. 1, Moskau, 1999, p. 128.
[26] S. Lang, Linear Algebra, Addison Wesley, Reading, MA, 1972.
[27] M. Latteux, Cônes rationnels commutativement clos. R.A.I.R.O., Informatique Théorique 11 (1977) 29–51.
[28] M. Latteux, Languages commutatifs, Thesè Sciences Mathematiques, University of Lille, 1978.
[29] M. Latteux, Cônes rationnels commutatifs, J. Comput. Syst. Sci. 18 (3) (1979) 307–333.
[30] E.W. Mayr, An algorithm for the general Petri net reachability problem, SIAM J. Comput. 13 (1984) 441–459.
[31] A. Salomaa, Jewels of Formal Language Theory, Computer Science Press, Rockville, 1981.