RAIRO-Theor. Inf. Appl. 46 (2012) 511–545
DOI: 10.1051/ita/2012013

Available online at:
www.rairo-ita.org

AFFINE PARIKH AUTOMATA *

Michaël Cadilhac[1], Alain Finkel[2] and Pierre McKenzie[1]

Abstract. The Parikh finite word automaton (PA) was introduced and studied in 2003 by Klaedtke and Rueß. Natural variants of the PA arise from viewing a PA equivalently as an automaton that keeps a count of its transitions and semilinearly constrains their numbers. Here we adopt this view and define the *affine PA*, that extends the PA by having each transition induce an affine transformation on the PA registers, and the *PA on letters*, that restricts the PA by forcing any two transitions on the same letter to affect the registers equally. Then we report on the expressiveness, closure, and decidability properties of such PA variants. We note that deterministic PA are strictly weaker than deterministic reversal-bounded counter machines.

Mathematics Subject Classification. 68Q45.

1. Introduction

Klaedtke and Rueß [17] introduced the *Parikh automaton* as a pair (A, C) where C is a semilinear subset of $\mathbb{N}^d$ and A is a finite automaton over $(\Sigma \times D)$ for Σ a finite alphabet and D a finite subset of $\mathbb{N}^d$. The word $w_1 \dots w_n \in \Sigma^*$ is accepted by

Keywords and phrases. Automata, semilinear sets, affine functions, counter machines.

* *Extended version of the abstract On the expressiveness of Parikh automata and related models. In Proceedings of the 3rd Workshop on Non-Classical Models of Automata and Applications (NCMA'11). books@ocg.at, edited by R. Freund, M. Holzer, C. Mereghetti, F. Otto and B. Palano.* **282**, *103–119.*

[1] DIRO, Université de Montréal, CP 6128 succ. centre-ville, Montréal QC, H3C 3J7, Canada. {cadilhac,mckenzie}@iro.umontreal.ca.
The third author is supported by the Natural Sciences and Engineering Research Council of Canada.

[2] LSV, ENS Cachan, CNRS, 61, avenue du Président Wilson, 94235 Cachan Cedex, France. finkel@lsv.ens-cachan.fr.
Ce travail a bénéficié d'une aide de l'Agence Nationale de la Recherche portant la référence "REACHARD-ANR-11-BS02-001".

Article published by EDP Sciences

(A, C) if A accepts some word $(w_1, \overline{v_1}) \ldots (w_n, \overline{v_n})$ such that $\sum \overline{v_i} \in C$. Motivated by verification issues, Klaedtke and Rueß developed the PA as a tool to probe (weak) monadic second-order logic with successor in which the cardinality $|X|$ of each second-order variable X is available. They proved their logic undecidable but showed decidability of an existential fragment that was successfully applied to verify the specification of actual hardware circuits.

Klaedtke and Rueß also studied decidability properties of the PA and properties of the language classes defined by the PA [16,17]. Karianto [15] took up this study further, elaborating on Klaedtke and Rueß's proofs and considering pushdown automata and constraint sets beyond semilinear. As for tree languages, Klaedtke and Rueß [16] introduced Parikh Tree Automata as top-down tree automata with one global semilinear constraint; at the same time, the related notion of Presburger Tree Automata, which combines bottom-up tree automata and semilinear preconditions about the number of children in a given state, was independently introduced by Zilio and Lugiez [27] and Seidl *et al.* [22].

Our interest in the PA comes both from its role in the area of verification and from the intricate three-way connection known to exist between automata, descriptive complexity and Boolean circuit complexity (see [23,24]). Indeed several circuit-based complexity classes within the class LOGCFL (of languages reducible to a context-free language) can be described in a natural way using first-order logic. In such a logic description, the (generalized) quantifiers reflect the properties of the automaton-based model defining the language while the (numerical) predicates reflect the level of uniformity allowed to the circuit families accepting the language. Since semilinearity arises in the study of LOGCFL (see [21]) and since the circuit depth complexity of regular languages is a major open question in complexity theory, the PA is a very appealing computation model with which to experiment in view of possible future applications to complexity theory.

In this paper we introduce three models closely related to the PA and we carry the study of PA themselves somewhat further. This is our first contribution. Informally, each model involves a finite automaton A and a constraint set $C \subseteq \mathbb{K}^d$ where $\mathbb{K}$ is either $\mathbb{N}$ or $\mathbb{Q}$:

- *Constrained automata* (CA) with d transitions are defined to accept a word $w \in \Sigma^*$ iff C contains the d-tuple that records, for some accepting run of A on w and for each transition t, the number of occurrences of t along that accepting run; we will see that the CA merely provides an alternate view of the PA in that the two models define the same language classes.
- *Affine Parikh automata* (APA) generalize PA by allowing each transition to perform an affine transformation on the d-tuple of PA registers; an APA accepts a word w iff some accepting run of A on w maps the all-zero d-tuple to a d-tuple in C.
- *Parikh automata on letters* (LPA) restrict PA by imposing the condition that any transition on $(a, \overline{u}) \in (\Sigma \times D)$ and any transition on $(b, \overline{v}) \in (\Sigma \times D)$ must satisfy $\overline{u} = \overline{v}$ when $a = b$.

Prop. 40 Prop. 17 Prop. 16

	$\cup$	$\cap$	$-$	$\cdot$	h	$h_{\not\varepsilon}$	h^{-1}	c	$*$	$\emptyset$	Σ^*	fin.	$\subseteq$	reg.
LPA	N	Y	N	N	N	N	Y	Y	N	**D**	**D**	D	D	?
DetPA	**Y**	**Y**	**Y**	N	N	N	**Y**	Y	N	**D**	**D**	D	D	?
PA	**Y**	**Y**	**N**	**Y**	**Y**	**Y**	**Y**	Y	**N**	**D**	**U**	D	U	U
DetAPA	Y	Y	Y	?	N	?	Y	?	?	U	U	U	U	U
APA	Y	Y	?	Y	N	Y	Y	?	?	U	U	U	U	U

Prop. 26 Cor. 33 Cor. 34

FIGURE 1. Closure in the effective sense (Y) or nonclosure (N) of language classes defined by PA variants, under set operations, concatenation, morphisms, nonerasing morphisms, inverse morphisms, commutation, and iteration; decidability (D) or undecidability (U) of emptiness, universality, finiteness, inclusion, and regularity; boldface denotes results known *prior* to this paper.

Our second contribution is the analysis of the closure and decidability properties of these models and their deterministic variants DetPA and DetAPA. We depict the known properties of PA [15–17] together with our new results in Figure 1, where DetLPA is not mentioned because DetLPA and LPA define the same languages.

Our third contribution is the comparison of the language classes that arise. We show that the language $\{a,b\}^* \cdot \{a^n \# a^n \mid n \in \mathbb{N}\}$ belongs to $\mathcal{L}_{\text{PA}} \setminus \mathcal{L}_{\text{DetPA}}$ where these two classes were only proved different in [17]. We show that APA and DetAPA over $\mathbb{Q}$ can be simulated by APA and DetAPA over $\mathbb{N}$ and *vice versa*. Refining [17] slightly, we compare our models with the reversal-bounded counter machines (RBCM) defined by Ibarra [12]. Figure 2 summarizes these and further results.

This paper is organized as follows. Section 2 contains preliminaries and settles notation. Section 3 defines the PA, introduces the equivalent CA, justifies the PA line and DetPA line entries from Figure 1 and compares the PA with Ibarra's RBCM. Sections 4 and 5 investigate the APA and the LPA respectively, completing the proofs of all remaining entries in Figures 1 and 2. Section 6 concludes with a short discussion.

2. PRELIMINARIES

Let $\mathbb{Z}$, $\mathbb{N}$, and $\mathbb{Q}$ denote the integers, the nonnegative integers, and the rational numbers respectively. We write $\mathbb{N}^+$ for $\mathbb{N} \setminus \{0\}$ and $\mathbb{Q}^+$ for the strictly positive rational numbers. We use $\mathbb{K}$ to denote either $\mathbb{N}$ or $\mathbb{Q}$. Let $d, d' \in \mathbb{N}^+$. Vectors in $\mathbb{K}^d$ are noted with a bar on top, *e.g.*, $\overline{v}$ whose elements are $v_1, \ldots, v_d$. For $C \subseteq \mathbb{K}^d$ and $D \subseteq \mathbb{K}^{d'}$, we write $C \times D$ for the set of vectors in $\mathbb{K}^{d+d'}$ which are

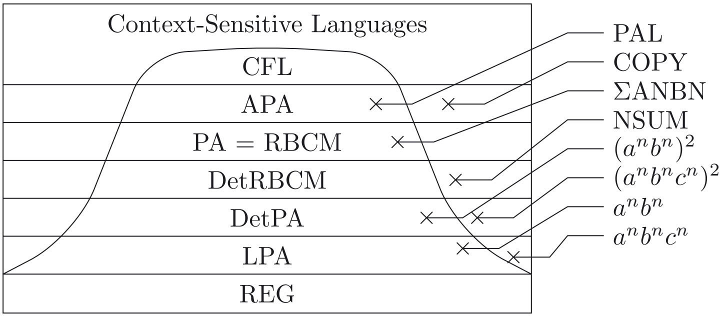


FIGURE 2. Relationships between language classes, sorted vertically by inclusion except for the class CFL of context-free languages delimited by the bell curve; RBCM stands for reversal-bounded counter machine; PAL is the language of pointed palindromes, COPY that of words $w\#w$, ΣANBN that of words wa^nb^n, and NSUM is discussed in Proposition 3.14.

(seen as) the concatenation of a vector of C and a vector of D; we will often use the isomorphism between $\mathbb{K}^{d+d'}$ and $\mathbb{K}^d \times \mathbb{K}^{d'}$. We write 0^d, or $\overline{0}$ when there is no ambiguity, for the vector with d 0-components, equal to $(0,\ldots,0) \in \mathbb{K}^d$, and $\overline{e_i} \in \{0,1\}^d$ for the vector having a 1 only in position i and 0 elsewhere. We view $\mathbb{K}^d$ as the additive monoid $(\mathbb{K}^d,+)$. For a monoid $(M,\cdot)$ and $S \subseteq M$, we write S^* for the monoid generated by S, *i.e.*, the smallest submonoid of $(M,\cdot)$ containing S. A monoid morphism from $(M,\cdot)$ to $(N,\circ)$ is a function $h\colon M \to N$ such that $h(m_1 \cdot m_2) = h(m_1) \circ h(m_2)$, and, with e_M (resp. e_N) the identity element of M (resp. N), $h(e_M) = e_N$. Moreover, if $M = S^*$ for some finite set S (and this will always be the case), then h need only be defined on the elements of S.

A subset $E \subseteq \mathbb{K}^d$ is $\mathbb{K}$-*definable* if it is expressible as a first-order formula which uses the function symbols $+$, λ_c with $c \in \mathbb{K}$ corresponding to the scalar multiplication, the order $<$ and constants. More precisely, a subset E of $\mathbb{K}^d$ is $\mathbb{K}$-definable iff there is such a formula ϕ with d free variables, with $(x_1,\ldots,x_d) \in E \Leftrightarrow \mathbb{K} \models \phi(x_1,\ldots,x_d)$. Let us remark that $\mathbb{N}$-definable sets are the Presburger-definable sets and they coincide with the *semilinear sets* [9], *i.e.*, finite unions of linear sets of the form $\{\overline{a_0} + k_1\overline{a_1} + \ldots + k_n\overline{a_n} \mid k_i \in \mathbb{N}\}$. Moreover, $\mathbb{Q}$-definable sets are the *semialgebraic sets*[3] defined using affine functions, *i.e.*, a set $C \subseteq \mathbb{Q}^d$ is $\mathbb{Q}$-definable iff it is a finite union of sets of the form:

$$\{\overline{x} \mid f_1(\overline{x}) = \ldots = f_p(\overline{x}) = 0 \wedge g_1(\overline{x}) > 0 \wedge \ldots \wedge g_q(\overline{x}) > 0\},$$

where $f_1,\ldots,f_p,g_1,\ldots,g_q\colon \mathbb{Q}^d \to \mathbb{Q}$ are affine functions (see, *e.g.*, [25], Cor. I.7.8); this shows in particular that over $\mathbb{Q}$ the formulas previously described admit quantifier elimination (see also [7]).

[3]Semialgrebraic sets defined using affine functions are sometimes also called semilinear (*e.g.*, [25]). In this paper, we use "semilinear" only for $\mathbb{N}$-definable sets.

Let $\Sigma = \{a_1, \dots, a_n\}$ be an (ordered) alphabet, and write ε for the empty word. The *Parikh image* is the morphism $\Phi\colon \Sigma^* \to \mathbb{N}^n$ defined by $\Phi(a_i) = \overline{e_i}$, for $1 \leq i \leq n$. A language $L \subseteq \Sigma^*$ is said to be *semilinear* if $\Phi(L) = \{\Phi(w) \mid w \in L\}$ is semilinear. The *commutative closure* of a language L is defined as the language $c(L) = \{w \mid \Phi(w) \in \Phi(L)\}$. Two words $u, v \in \Sigma^*$ are *equivalent under the Nerode relation* (of L), if for all $w \in \Sigma^*$, $uw \in L \Leftrightarrow vw \in L$. We then write $u \equiv_L v$ (or $u \equiv v$ when L is understood), and write $[u]_L$ for the equivalence class of u w.r.t. the Nerode relation.

We then fix our notation about automata. An automaton is a quintuple $A = (Q, \Sigma, \delta, q_0, F)$ where Q is the finite set of states, Σ is an alphabet, $\delta \subseteq Q \times \Sigma \times Q$ is the set of transitions, $q_0 \in Q$ is the initial state, and $F \subseteq Q$ are the final states. Note that no transition is labeled by the empty word. For a transition $t \in \delta$, where $t = (q, a, q')$, we define $\mathsf{From}(t) = q$ and $\mathsf{To}(t) = q'$. Moreover, we define $\mu_A\colon \delta^* \to \Sigma^*$ to be the morphism defined by $\mu_A(t) = a$ (where in particular $\mu_A(\varepsilon) = \varepsilon$), and we write μ when A is clear from the context. A *path* on A is a word $\pi = t_1 \dots t_n \in \delta^*$ such that $\mathsf{To}(t_i) = \mathsf{From}(t_{i+1})$ for $1 \leq i < n$; we extend From and To to paths, letting $\mathsf{From}(\pi) = \mathsf{From}(t_1)$ and $\mathsf{To}(\pi) = \mathsf{To}(t_n)$. We say that $\mu(\pi)$ is the *label* of π. A path π is said to be *accepting* if $\mathsf{From}(\pi) = q_0$ and $\mathsf{To}(\pi) \in F$; we let $\mathsf{Run}(A)$ be the language over δ of accepting paths on A. We then define $L(A)$, the *language* of A, as the labels of the accepting paths, *i.e.*, $\mu_A(\mathsf{Run}(A))$.

3. Parikh automata

Let Σ be an alphabet, $d \in \mathbb{N}^+$, and D a finite subset of $\mathbb{N}^d$. Following [17], the monoid morphism from $(\Sigma \times D)^*$ to Σ^* defined by $(a, \overline{v}) \mapsto a$ is called the *projection on* Σ and the monoid morphism from $(\Sigma \times D)^*$ to $\mathbb{N}^d$ defined by $(a, \overline{v}) \mapsto \overline{v}$ is called the *extended Parikh image*.

Remark 3.1. Let $\Sigma = \{a_1, \dots, a_n\}$ and $D \subseteq \mathbb{N}^n$. If a word $\omega \in (\Sigma \times D)^*$ is in $\{(a_i, \overline{e_i}) \mid 1 \leq i \leq n\}^*$, then the extended Parikh image of ω is the Parikh image of its projection on Σ.

Definition 3.2 (Parikh automaton [17])**.** Let Σ be an alphabet, $d \in \mathbb{N}^+$, and D a finite subset of $\mathbb{N}^d$. A *Parikh automaton* (PA) of dimension d over Σ is a pair (A, C) where $A = (Q, \Sigma \times D, \delta, q_0, F)$ is a finite automaton over $\Sigma \times D$, and $C \subseteq \mathbb{N}^d$ is a semilinear set. The PA language, written $L(A, C)$, is the projection on Σ of the words of $L(A)$ whose extended Parikh image is in C. The PA is said to be *deterministic* (DetPA) if for every state $q \in Q$ and every $a \in \Sigma$, there exists at most one pair $(q', \overline{v}) \in Q \times D$ such that $(q, (a, \overline{v}), q') \in \delta$. We write $\mathcal{L}_{\mathrm{PA}}$ (resp. $\mathcal{L}_{\mathrm{DetPA}}$) for the class of languages recognized by PA (resp. DetPA).

We propose an alternative view of the PA which will prove very useful. We note that a PA can be viewed equivalently as an automaton that applies a semilinear constraint on the counts of the individual transitions occurring along its accepting runs.

Definition 3.3 (constrained automaton)**.** A *constrained automaton* (CA) over an alphabet Σ is a pair (A, C) where A is a finite automaton over Σ with d transitions, and $C \subseteq \mathbb{N}^d$ is a semilinear set. Its language is $L(A, C) = \mu_A(\mathsf{Run}(A)\restriction_C)$, where $L\restriction_C = \{w \in L \mid \Phi(w) \in C\}$. The CA is said to be *deterministic* (DetCA) if A is deterministic.

Theorem 3.4. CA *and* PA *define the same classes of languages. The same holds in the deterministic case.*

Proof. Let (A, C) be a PA (resp. DetPA) of dimension d, and let $\delta = \{t_1, \dots, t_n\}$ be the transitions of A. We suppose moreover that A is deterministic – this does not imply the determinism of the PA. Consider the automaton A' which is a copy of A except that the vector part of the transitions is dropped, and note that if (A, C) is a DetPA then A' is deterministic. Suppose that the mapping induced between the transitions of A and A', *i.e.*, $(p, (a, \overline{v}), q) \mapsto (p, a, q)$, is a bijection. The contribution of a transition $t_i = (p, (a, \overline{v_i}), q)$ to the extended Parikh image of the label of a run in which it appears is $\overline{v_i}$; thus, knowing how many times t_i is taken in a path is enough to retrieve the value of the extended Parikh image of the label of a path. More precisely, for a path π in A and the equivalent path π' in A', if we let $\Phi(\pi') = (x_1, \dots, x_n)$ then the extended Parikh image of $\mu(\pi)$, $\tilde{\Phi}(\mu(\pi))$, is $\sum_{i=1}^n x_i \times \tilde{\Phi}(\mu(t_i))$. Thus, we define $C' \subseteq \mathbb{N}^n$ by $C' = \{(x_1, \dots, x_n) \mid \sum_{i=1}^n x_i \times \tilde{\Phi}(\mu(t_i)) \in C\}$, and the PA (A, C) has the same language as the CA (A', C'), and determinism is preserved.

Now note that the aforementioned bijection exists if no two distinct transitions t_i, t_j are such that $t_i = (p, (a, \overline{v_i}), q)$ and $t_j = (p, (a, \overline{v_j}), q)$. So suppose that such t_i and t_j exist, we show how to remove them; iterating this process will lead to a PA with no such pair of transitions. First, we increment the dimension of the PA by adding a 0 component to all the vectors appearing as labels, *i.e.*, each label $(\ell, \overline{v})$ is replaced by $(\ell, (\overline{v}, 0))$. Next, we remove t_i and t_j and add the transition $t = (p, (a, \overline{e_{d+1}}), q)$ where $\overline{e_{d+1}} \in \{0, 1\}^{d+1}$ has a one only in position $d + 1$. Now note that when t is taken in the new automaton, either t_i or t_j could have been taken in the old one. Thus define the semilinear set D to *split* the number of times t is taken – which is stored in the $d + 1$-th component – between t_i and t_j; for $\overline{x} \in \mathbb{N}^d, c \in \mathbb{N}$:

$$(\overline{x}, c) \in D \Leftrightarrow (\exists c_i, c_j \in \mathbb{N})\, [c = c_i + c_j \wedge (\overline{x} + c_i.\overline{v_i} + c_j.\overline{v_j}) \in C]\,.$$

This preserves the language of the PA and does not affect determinism.

For the reverse direction, let (A, C) be a CA (resp. DetCA). Define A' as the automaton A in which each transition $t = (p, a, q)$ is replaced by a transition $(p, (a, \Phi(t)), q)$. Now let π be a path in A and π' be the corresponding path in A', the construction is such that $\tilde{\Phi}(\mu(\pi')) = \Phi(\pi)$, thus (A', C) is a PA with the same language as (A, C), and the determinism of A is preserved. $\square$

3.1. On the expressiveness of Parikh automata

The constrained automaton characterization of PA helps deriving pumping-style necessary conditions for membership in $\mathcal{L}_{\text{PA}}$ and in $\mathcal{L}_{\text{DetPA}}$:

Lemma 3.5. *Let $L \in \mathcal{L}_{\text{PA}}$. There exist $p, \ell \in \mathbb{N}^+$ such that any $w \in L$ with $|w| \geq \ell$ can be written as $w = uvxvz$ where:*

(1) $0 < |v| \leq p$, $|x| > p$, *and* $|uvxv| \leq \ell$;
(2) $uv^2xz \in L$ *and* $uxv^2z \in L$.

Proof. Let (A, C) be a CA of language L. Let p be the number of states in A and m be the number of elementary cycles (*i.e.*, cycles in which no state except the start state occurs twice) in the underlying multigraph of A. Finally, let $\ell = p \times (2m+1)$. Now, let $w \in L$ such that $|w| \geq \ell$ and $\pi \in \mathsf{Run}(A)\restriction_C$ such that $\mu(\pi) = w$. Write π as $\pi_1 \dots \pi_{2m+1}\pi'$ where $|\pi_i| = p$. By the pigeonhole principle, each π_i contains an elementary cycle, and thus, there exist $1 \leq i, j \leq 2m+1$ with $i+1 < j$ such that π_i and π_j share the same elementary cycle η labeled with a word v. Thus π can be written as $\rho_1\eta\rho_2\eta\rho_3$, such that, with $u = \mu(\rho_1)$, $x = \mu(\rho_2)$, and $z = \mu(\rho_3)$, we have condition *(1)*. Moreover, $\rho_1\eta^2\rho_2\rho_3$ and $\rho_1\rho_2\eta^2\rho_3$ are two accepting paths the Parikh images of which are in C, thus their labels, uv^2xz and uxv^2z respectively, are in L, showing condition *(2)*. $\square$

A similar argument leads to a stronger property for the languages of $\mathcal{L}_{\text{DetPA}}$:

Lemma 3.6. *Let $L \in \mathcal{L}_{\text{DetPA}}$. There exist $p, \ell \in \mathbb{N}^+$ such that any w over the alphabet of L with $|w| \geq \ell$ can be written as $w = uvxvz$ where:*

(1) $0 < |v| \leq p$, $|x| > p$, *and* $|uvxv| \leq \ell$;
(2) $uv^2x, uvxv$, *and* uxv^2 *are equivalent under the Nerode relation of* L.

Proof. Let (A, C) be a DetCA of language $L \subseteq \Sigma^*$. We may suppose that A is complete, as $\mathsf{Run}(A)$ is essentially unchanged when adding a sink state to A. Let p be the number of states in A and m be the number of elementary cycles (*i.e.*, cycles in which no state except the start state occurs twice) in the underlying multigraph of A. Finally, let $\ell = p \times (2m+1)$. Now, let $w \in \Sigma^{\geq \ell}$ and let π be the path traced by w in A, which exists as A is complete. Write π as $\pi_1 \dots \pi_{2m+1}\pi'$ where $|\pi_i| = p$. By the pigeonhole principle, each π_i contains an elementary cycle, and thus, there exist $1 \leq i, j \leq 2m+1$ with $i+1 < j$ such that π_i and π_j share the same elementary cycle η labeled with a word v. Thus π can be written as $\rho_1\eta\rho_2\eta\rho_3$, such that, with $u = \mu(\rho_1)$, $x = \mu(\rho_2)$, and $z = \mu(\rho_3)$, we have condition *(1)*. Moreover, $uv^2x, uvxv$, and uxv^2 trace the paths $\rho_1\eta^2\rho_2$, $\rho_1\eta\rho_2\eta$, and $\rho_1\rho_2\eta^2$, respectively, in A. Those paths all go from the initial state to the same state q and have the same Parikh image. Thus let π'' be a path in A from q with some label y, then $uv^2xy \in L(A, C)$ iff π'' ends in a final state and $\Phi(\rho_1\eta^2\rho_2\pi'') \in C$. But since $\Phi(\rho_1\eta^2\rho_2\pi'') = \Phi(\rho_1\eta\rho_2\eta\pi'') = \Phi(\rho_1\rho_2\eta^2\pi'')$, this is the case iff $uvxvy \in L$ and iff $uxv^2y \in L$, showing condition *(2)*. $\square$

We apply Lemma 3.5 to the language COPY, defined as $\{w\#w \mid w \in \{a,b\}^*\}$, as follows:

Proposition 3.7. COPY $\notin \mathcal{L}_{\mathrm{PA}}$.

Proof. Suppose COPY $\in \mathcal{L}_{\mathrm{PA}}$. Let ℓ, p be given by Lemma 3.5, and consider $w = (a^p b)^\ell \# (a^p b)^\ell \in$ COPY. Lemma 3.5 states that $w = uvxvz$ where $uvxv$ lays in the first half of w, and $s = uv^2xz \in$ COPY. Note that x contains at least one b. Suppose $v = a^i$ for $1 \leq i \leq p$, then there is a sequence of a's in the first half of s unmatched in the second half. Likewise, if v contains a b, then s has a sequence of a's between two b's unmatched in the second half. Thus $s \notin$ COPY, a contradiction. Hence COPY $\notin \mathcal{L}_{\mathrm{PA}}$. □

As Klaedtke and Rueß show using closure properties, DetPA are strictly weaker than PA. The thinner grain of Lemma 3.6 suggests explicit languages that witness the separation of $\mathcal{L}_{\mathrm{DetPA}}$ from $\mathcal{L}_{\mathrm{PA}}$. Indeed, let EQUAL $\subseteq \{a, b, \#\}^*$ be the language $\{a,b\}^* \cdot \{a^n \# a^n \mid n \in \mathbb{N}\}$, we have:

Proposition 3.8. EQUAL $\in \mathcal{L}_{\mathrm{PA}} \setminus \mathcal{L}_{\mathrm{DetPA}}$.

Proof. We omit the proof that EQUAL $\in \mathcal{L}_{\mathrm{PA}}$. Now, suppose EQUAL $\in \mathcal{L}_{\mathrm{DetPA}}$, and let ℓ, p be given by Lemma 3.6. Consider $w = (a^p b)^\ell$. Lemma 3.6 then asserts that a prefix of w can be written as $w_1 = uvxv$, and that $w_2 = uv^2x$ verifies $w_1 \equiv w_2$. As $|x| > p$, x contains a b. Let k be the number of a's at the end of w_1. Suppose $v = a^i$ for $1 \leq i \leq p$, then w_2 ends with $k - i < k$ letters a. Thus $w_1 \# a^k \in$ EQUAL and $w_2 \# a^k \notin$ EQUAL, a contradiction. Suppose then that $v = a^i b a^k$, with $0 \leq i + k < p$. Then w_2 ends with $p - i > k$ letters a, and similarly, $w_1 \not\equiv w_2$, a contradiction. Thus EQUAL $\notin \mathcal{L}_{\mathrm{DetPA}}$. □

For comparison, we mention another line of attack for the study of $\mathcal{L}_{\mathrm{DetPA}}$, derived from an argument used by Klaedtke and Rueß to show that PAL $= \{w\#w^{\mathrm{R}} \mid w \in \{a,b\}^+\}$, where w^{R} is the reversal of w, is not in $\mathcal{L}_{\mathrm{PA}}$.

Lemma 3.9. *Let $L \in \mathcal{L}_{\mathrm{DetPA}}$. There exists $c > 0$ such that $|\{[w]_L \mid w \in \Sigma^n\}| \in O(n^c)$.*

Proof. Let (A, C) be a DetCA of language $L \subseteq \Sigma^*$ where we suppose A complete, as this leaves $\mathsf{Run}(A)$ essentially unchanged. For $w \in \Sigma^*$, write $\pi(w)$ for the unique path in A labeled w and starting with the initial state. Let $\sim$ be the equivalence relation on Σ^* defined by $u \sim v$ iff $\Phi(\pi(u)) = \Phi(\pi(v)) \wedge \mathsf{To}(\pi(u)) = \mathsf{To}(\pi(v))$. Then this relation refines $\equiv_L$: let $u, v \in \Sigma^*$ such that $u \sim v$, and let $w \in \Sigma^*$ such that $uw \in L$, then $\pi(uw) \in \mathsf{Run}(A)\restriction_C$, thus the same holds for $\pi(vw)$, implying that $vw \in L$. Moreover, the number of equivalence classes of $\sim$ for a given word length is polynomial in the word length (*e.g.*, [20], p. 41). □

Proposition 3.10. *Let $L = \{w \in \{a,b\}^* \mid w_{|w|_a} = b\}$, where w_i is the i-th letter of w. Then $L \in \mathcal{L}_{\mathrm{PA}} \setminus \mathcal{L}_{\mathrm{DetPA}}$.*

Proof. We omit the proof that $L \in \mathcal{L}_{\mathrm{PA}}$; the main point is simply to guess the position of the b referenced by $|w|_a$. On the other hand, let $n > 0$ and $u, v \in \{a,b\}^n$ such that $|u|_a = |v|_a = \frac{n}{2}$ and there exists $p \in \{\frac{n}{2}, \ldots, n\}$ with $u_p \neq v_p$. Let $w = a^{p-\frac{n}{2}}$, then $(uw)_{|uw|_a} = (uw)_{|u|_a+|w|_a} = (uw)_p = u_p$, and similarly, $(vw)_{|vw|_a} = v_p$. This implies $uw \notin L \leftrightarrow vw \in L$, thus $u \not\equiv v$. Then for $0 \leq i \leq \frac{n}{2}$, define $E_i = \{a^{\frac{n}{2}-i}b^i z \mid z \in \{a,b\}^{\frac{n}{2}} \wedge |z|_a = i\}$. For any $u, v \in \bigcup E_i$ with $u \neq v$, the previous discussion shows that $u \not\equiv v$. Thus $|\{[w]_L \mid w \in \{a,b\}^n\}| \geq |\bigcup_{i=0}^{\frac{n}{2}} E_i| = \sum_{i=0}^{\frac{n}{2}} |E_i| = \sum_{i=0}^{\frac{n}{2}} \binom{\frac{n}{2}}{i} = 2^{\frac{n}{2}} \notin O(n^{O(1)})$. Lemma 3.9 then implies that $L \notin \mathcal{L}_{\mathrm{DetPA}}$. □

Finally, let us recall that a language $L \subseteq \Sigma^*$ is said to be *bounded* if there exist $n > 0$ and $w_1, \ldots, w_n \in \Sigma^+$ such that $L \subseteq w_1^* \ldots w_n^*$. For a given class of languages, we say that it is *Parikh-bounded* if for any L in the class there exists a bounded language L' in the class with $L' \subseteq L$ and $\Phi(L) = \Phi(L')$. This property is known to hold for regular [19] and context-free languages [2] (the latter recently reworked in [8]).

Proposition 3.11. *$\mathcal{L}_{\mathrm{PA}}$ is Parikh-bounded.*

Proof. Let (A, C) be a constrained automaton, where δ is the transition set of A. Note that $\mathsf{Run}(A)$ is regular, thus, as mentioned, we can find a bounded regular language $R \subseteq \mathsf{Run}(A)$ such that $\Phi(R) = \Phi(\mathsf{Run}(A))$. In particular, $\Phi(R\restriction_C) = \Phi(\mathsf{Run}(A)\restriction_C)$. Closure under morphism of $\mathcal{L}_{\mathrm{PA}}$ and of bounded languages implies that $L' = \mu(R\restriction_C)$ is a bounded language of $\mathcal{L}_{\mathrm{PA}}$ included in $L(A, C)$. Moreover, $\Phi(L(A, C)) = \Phi(\mu(R\restriction_C))$, and thus, equals $\Phi(L')$. □

3.2. Parikh automata and reversal-bounded counter machines

Klaedtke and Rueß noticed in [16] that Parikh automata recognize the same languages as reversal-bounded counter machines, a model introduced by Ibarra [12]:

Definition 3.12 (reversal-bounded counter machine [12])**.** A *one-way, k-counter machine* M is a 5-uple $(Q, \Sigma, \delta, q_0, F)$ where Q is a finite set of states, Σ is an alphabet, $\delta \subseteq Q \times (\Sigma \cup \{\sharp\}) \times \{0,1\}^k \times Q \times \{S, R\} \times \{-1, 0, +1\}^k$ is the transition function, $q_0 \in Q$ is the initial state and $F \subseteq Q$ is the set of final states. Moreover, we suppose $\sharp \notin \Sigma$. The machine is *deterministic* if for any $(p, \ell, \overline{x})$, there exists at most one $(q, h, \overline{v})$ such that $(p, \ell, \overline{x}, q, h, \overline{v}) \in \delta$. On input w, the machine starts with a read-only tape containing $w\sharp$, and its head on the first character of w. Let c_i be the value of the i-th counter, then a transition $(p, \ell, \overline{x}, q, h, \overline{v}) \in \delta$ is taken if the machine is in state p, reading character ℓ, and $c_i = 0$ if $x_i = 0$ and $c_i > 0$ if $x_i = 1$, for all i. The machine then enters state q, its head is moved to the right iff $h = R$, and $\overline{v}$ is added to the counters. If the head falls off the tape, or if a counter turns negative, the machine rejects. A word is accepted if an execution leads to a final state. The machine is *reversal-bounded* (RBCM) if there exists an integer r such that any accepting run changes between increments and decrements of the counters a (bounded) number of times less than r. We write DetRBCM for deterministic RBCM. We write $\mathcal{L}_{\mathrm{RBCM}}$ (resp. $\mathcal{L}_{\mathrm{DetRBCM}}$) for the class of languages recognized by RBCM (resp. DetRBCM).

In [16], Section A.3, it is shown that PA have the same expressive power as (nondeterministic) RBCM. Although Fact 30 of [16], on which the authors rely to prove that $\mathcal{L}_{\mathrm{RBCM}} \subseteq \mathcal{L}_{\mathrm{PA}}$, is technically false as stated,[4] the small gap there can be fixed so that:

Proposition 3.13 ([16]). $\mathcal{L}_{\mathrm{PA}} = \mathcal{L}_{\mathrm{RBCM}}$.

Proof. We sketch $\mathcal{L}_{\mathrm{RBCM}} \subseteq \mathcal{L}_{\mathrm{PA}}$ for completeness and reprove $\mathcal{L}_{\mathrm{PA}} \subseteq \mathcal{L}_{\mathrm{RBCM}}$ to extract a more precise structure on the constructed RBCM – this will prove useful when comparing the notion of determinism in both models.

($\mathcal{L}_{\mathrm{RBCM}} \subseteq \mathcal{L}_{\mathrm{PA}}$). First, it is known [12] that any RBCM language can be expressed as an RBCM which makes *at most* one change between increment and decrement on each of its counters. Then a counter can be seen as being in one of three different states: (1) never incremented, (2) incremented but never decremented, (3) decremented. When a counter is in state (1), we may simulate the behavior of the RBCM with the counter set to zero. Similarly, when a counter is in state (2), we may simulate the behavior of the RBCM with this counter set to a nonzero value. Lastly, when in state (3), we may *guess* at some point that the counter reached zero, and act for the rest of the execution as if the counter is actually zero (thus not making any modification to this counter). Now, when in states (1) and (2), the behavior of the RBCM w.r.t. the counter can be simulated using a finite automaton; in state (3), the guess can be taken with a finite automaton, but we must check that the guess was taken at the right moment. Thus we use a PA to count the number of increments and decrements, and we check at the end of the computation that the latter is no greater than the former, and that these are equal iff the counter has been guessed to be zero at some point. Finally, the transitions between the different states can be made knowing only the transitions of the RBCM. This gives the required PA for the RBCM.

($\mathcal{L}_{\mathrm{PA}} \subseteq \mathcal{L}_{\mathrm{RBCM}}$). Let (A, C) be a CA, where $A = (Q, \Sigma, \delta, q_0, F)$ and let $\delta = \{t_1, \ldots, t_k\}$. We define a RBCM of the same language in two steps. (1). First, let M be the k-counter machine $(Q \cup \{q_f\}, \Sigma, \zeta, q_0, q_f)$, where $q_f \notin Q$ and ζ is defined by:

$$\zeta = \bigcup_{\substack{\overline{x} \in \{0,1\}^k \\ 1 \le i \le k}} \{(\mathsf{From}(t_i), \mu(t_i), \overline{x}, \mathsf{To}(t_i), R, \overline{e_i})\} \cup \bigcup_{\substack{\overline{x} \in \{0,1\}^k \\ q \in F}} \{(q, \sharp, \overline{x}, q_f, S, \overline{0})\}.$$

[4] Fact 30 of [16] states the following. Consider a RBCM M which, for any counter, changes between increment and decrement only once. Let M' be M in which negative counter values are allowed and the zero-tests are ignored. Then a word is claimed to be accepted by M iff the run of M' on the same word reaches a final state with all its counters nonnegative. A counter-example is the following. Take A to be the minimal automaton for a^*b, and add a counter for the number of a's that blocks the transition labeled b unless the counter is nonzero. This machine recognizes a^+b. Then by removing this test, the machine now accepts b.

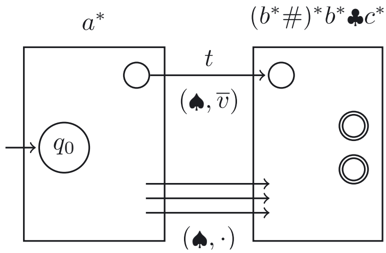
This machine does not make any test, and accepts (in q_f) precisely the words accepted by A. Moreover, the state of the counters in q_f is the Parikh image of the path taken (in A) to recognize the input word. (2) We then refine M to check that the counter values belong to C. We note that we can do that as a direct consequence of the proof of [13], Theorem 3.5, but this proof relied on nontrivial algebraic properties of systems $M\overline{y} = \overline{b}$, where M is a matrix, $\overline{y}$ are unknowns, and $\overline{b}$ is a vector; we present here a proof based on a logical characterization of semilinear sets. Recall that C can be expressed as a quantifier-free first-order formula which uses the function symbol $+$, the congruence relations $\equiv_i$, for $i \geq 2$, and the order relation $<$ (see, *e.g.*, [6]). So let C be given as such formula ϕ_C with k free variables. Let ϕ_C be put in disjunctive normal form. The machine M then tries each and every clause of ϕ_C for acceptance. First, note that a term can be computed deterministically with a number of counters and reversals which depends only on its size. For instance, computing $c_i + c_j$ requires two new counters x, y: c_i is decremented until it reaches 0, while x and y are incremented, so that their value is c_i, now y is decremented until it reaches 0 while c_i is incremented back to its original value, finally the same process is applied with c_j, and as a result x is now $c_i + c_j$. Second, note that any atomic formula ($t_1 < t_2$ or $t_1 \equiv_i t_2$) can be checked by a DetRBCM: for $t_1 < t_2$, compute $x_1 = t_1$ and $x_2 = t_2$, then decrement x_1 and x_2 until one of them reaches 0, if the first one is x_1, then the atomic formula is true, and false otherwise; for $t_1 \equiv_i t_2$, a simple automaton-based construction depending on i can decide if the atomic formula is true. Thus, a DetRBCM can decide, for each clause, if all of its atomic formulas (or negation) are true, and in this case, accept the word. This process does not use the read-only head, and uses a number of counters and a number of reversals that depend only on the length of ϕ_C. $\square$

Further, we study how the notion of determinism compares in the two models. Let NSUM $= \{a^n \spadesuit b^{m_1} \# b^{m_2} \# \ldots \# b^{m_k} \clubsuit c^{m_1 + \ldots + m_n} \mid k \geq n \geq 0 \wedge m_i \in \mathbb{N}\}$: the number of a's is the number of m_i's to sum to get the number of c's. Note that NSUM is not context-free. Then:

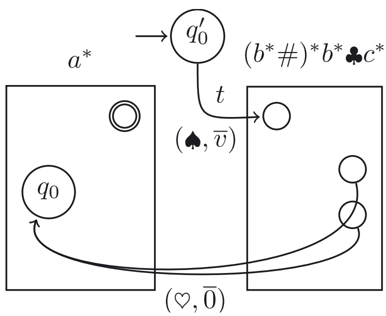
Proposition 3.14. $\mathcal{L}_{\mathrm{DetPA}} \subsetneq \mathcal{L}_{\mathrm{DetRBCM}}$ *and* NSUM $\in \mathcal{L}_{\mathrm{DetRBCM}} \setminus \mathcal{L}_{\mathrm{DetPA}}$.

Proof. The inclusion $\mathcal{L}_{\mathrm{DetPA}} \subseteq \mathcal{L}_{\mathrm{DetRBCM}}$ follows from the Proof of Proposition 3.13, as the resulting RBCM is deterministic if the given CA is.

We now show that NSUM $\in \mathcal{L}_{\mathrm{DetRBCM}} \setminus \mathcal{L}_{\mathrm{DetPA}}$. We omit the fact that NSUM $\in \mathcal{L}_{\mathrm{DetRBCM}}$. Now suppose (A, C) is a DetPA such that $L(A, C) =$ NSUM, with $A = (Q, \Sigma \times D, \delta, q_0, F)$. As (A, C) is a DetPA, A is deterministic – it is indeed already deterministic with respect to the first component of the labels. We may suppose that the projection on Σ of $L(A)$ is a subset of $a^* \spadesuit (b^* \#)^* b^* \clubsuit c^*$, so that there exist $k \geq 0$, $q_1, \ldots, q_k \in Q$, and $j \in \{0, \ldots, k\}$ such that $(q_i, (a, \overline{v_i}), q_{i+1}) \in \delta$, for $0 \leq i < k$ and some $\overline{v_i}$'s, and $(q_k, (a, \overline{v_k}), q_j) \in \delta$. Moreover, we may suppose that no other transition points to one of the q_i's, and that all transitions $t = (q_i, (\ell, \overline{v}), q) \in \delta$ such that $q \notin \{q_0, \ldots, q_k\}$ are with $\ell = \spadesuit$; let T be the set of all such transitions t. Graphically, A looks like:



We define $|T|$ DetPA such that the union of their languages is SUMN $= \{\spadesuit w \heartsuit a^n \mid a^n \spadesuit w \in \text{NSUM}\}$, that is, the strings of NSUM with a^n pushed at the end. For $t \in T$, define A_t as the automaton similar to A but which starts with the transition t and delay the first part of the computation until the very end; graphically, A_t is:



Formally, $A_t = (Q \cup \{q_0'\}, \Sigma \times D, \delta_t, q_0', \{\mathsf{From}(t)\})$ where q_0' is a fresh (*i.e.*, new) state and:

$$\delta_t = (\delta \setminus T) \ \cup \ \{(q_0', \mu(t), \mathsf{To}(t))\} \ \cup \ \{(q_f, (\heartsuit, \overline{0}), q_0) \mid q_f \in F\}.$$

Now for $\omega \in L(A)$, let t be the transition labeled $\spadesuit$ taken when A reads ω, and let $\omega = \omega_1 \mu(t) \omega_2$. Then $\mu(t)\omega_2(\heartsuit, \overline{0})\omega_1 \in L(A_t)$, and this word has the same extended Parikh image as ω. Thus we have that $\bigcup_{t \in T} L(A_t, C) = \text{SUMN}$, and if NSUM $\in \mathcal{L}_{\text{DetPA}}$, then SUMN $\in \mathcal{L}_{\text{DetPA}}$, as $\mathcal{L}_{\text{DetPA}}$ is closed under union (see Fig. 1). We now show that SUMN $\notin \mathcal{L}_{\text{DetPA}}$, thus leading to a contradiction showing the result. Suppose SUMN $\in \mathcal{L}_{\text{DetPA}}$ and let ℓ, p be given by Lemma 3.6 for SUMN. Consider $w = \spadesuit(b^p\#)^\ell$. Lemma 3.6 then asserts that a prefix of w can be written as $w_1 = uvxv$, and that $w_2 = uv^2x$ verifies $w_1 \equiv w_2$. As $|x| > p$ and v is nonempty, x contains a $\#$; moreover, v does not contain $\spadesuit$. Let $s = |w_1|_\# = |w_2|_\#$ and let n_i be the number of b's before the position of the s-th $\#$ in w_i, $i = 1, 2$. Suppose $v \in b^+$, then $n_1 < n_2$, thus $w_1 \clubsuit c^{n_1} \heartsuit a^s \in$ SUMN and $w_2\clubsuit c^{n_1}\heartsuit a^s \notin$ SUMN, a contradiction. Suppose then that $v = b^i\#b^j$, with $0 \leq i+j < p$. Similarly, as $i+j < p$, $n_2 < n_1$, and again, $w_1 \not\equiv w_2$, a contradiction. Thus SUMN $\notin \mathcal{L}_{\text{DetPA}}$. □

The parallel drawn between (Det)PA and (Det)RBCM allows transferring some RBCM and DetRBCM results to PA and DetPA. An example is a consequence of the following lemma proved in 2011 by Chiniforooshan *et al.* [5] for the purpose of showing incomparability results between different models of reversal-bounded counter machines:

Lemma 3.15 ([5])**.** *Let a* DetRBCM *express* $L \subseteq \Sigma^*$*. Then there exists* $w \in \Sigma^*$ *such that* $L \cap w\Sigma^*$ *is a nontrivial regular language.*

Using this lemma, variants of the language EQUAL from Proposition 3.8 can be shown outside $\mathcal{L}_{\text{DetPA}}$. For instance, for $\Sigma = \{a, b\}$, ΣANBN $= \Sigma^* \cdot \{a^n b^n \mid n \in \mathbb{N}^+\}$ is such that any $w \in \Sigma^*$ makes ΣANBN $\cap\, w\Sigma^*$ nonregular. Although Lemma 3.15 thus gives languages in $\mathcal{L}_{\text{PA}} \setminus \mathcal{L}_{\text{DetPA}}$, Lemma 3.15 seemingly does not apply to EQUAL itself since EQUAL $\cap\, \#\{a, b, \#\}^* = \{\#\}$ is regular.

3.3. On decidability and closure properties of Parikh automata

In this section we justify the PA and DetPA line entries in Figure 1. The known decidability results depicted there (in boldface) are from [12,17], and Karianto [15] provided detailed proofs.

Proposition 3.16. *(1) Finiteness is decidable for* PA*; (2) inclusion is decidable for* DetPA *and undecidable for* PA*; (3) regularity is undecidable for* PA*.*

Proof.

(1), (2) These decidability properties follow directly from the same properties for RBCM and DetRBCM [14], the effective equivalence between PA and RBCM (Prop. 3.13), and the effective inclusion of $\mathcal{L}_{\text{DetPA}}$ in $\mathcal{L}_{\text{DetRBCM}}$ (Prop. 3.14).

(3) This follows from a theorem of [11], which states the following. Let $\mathcal{C}$ be a class of languages closed under union and under concatenation with regular languages. Let P be a predicate on languages true of every regular language, false of some languages, preserved by inverse rational transduction, union with $\{\varepsilon\}$ and intersection with regular languages. Then P is undecidable in $\mathcal{C}$. Obviously, $\mathcal{L}_{\text{PA}}$ satisfies the hypothesis for $\mathcal{C}$. Moreover, "being regular in $\mathcal{L}_{\text{PA}}$" is a predicate satisfying the hypothesis for P. Thus, regularity is undecidable for PA. □

We now turn to closure properties:

Proposition 3.17. *(1)* $\mathcal{L}_{\text{DetPA}}$ *is not closed under concatenation; (2)* $\mathcal{L}_{\text{DetPA}}$ *is not closed under nonerasing morphisms; (3) both* $\mathcal{L}_{\text{PA}}$ *and* $\mathcal{L}_{\text{DetPA}}$ *are closed under commutative closure; (4) neither* $\mathcal{L}_{\text{PA}}$ *nor* $\mathcal{L}_{\text{DetPA}}$ *is closed under starring.*

Proof.

(1) The language EQUAL separating $\mathcal{L}_{\text{DetPA}}$ from $\mathcal{L}_{\text{PA}}$ is the concatenation of a regular language and a language of $\mathcal{L}_{\text{DetPA}}$, implying the nonclosure under concatenation.

(2) We note that any language of $\mathcal{L}_{\text{PA}}$ is the image by a nonerasing morphism of a language in $\mathcal{L}_{\text{DetPA}}$. Indeed, say (A, C) is a CA and let B be the deterministic automaton of language $\mathsf{Run}(A)$ defined as a copy of A in which the transition t is relabeled t (*i.e.*, $(p, a, q) \rightsquigarrow (p, (p, a, q), q)$). Then (B, C) is a DetCA such that $L(A, C) = \mu_A(L(B, C))$. This implies the nonclosure of $\mathcal{L}_{\text{DetPA}}$ under nonerasing morphisms.

(3) Let $\Sigma = \{a_1, \ldots, a_n\}$, $L \subseteq \Sigma^*$ a semilinear language, and $C = \Phi(L)$. Define A to be an automaton with one state, initial and final, with n loops, the i-th labeled $(a_i, \overline{e_i}) \in \Sigma \times \{\overline{e_i}\}_{1 \le i \le n}$. Then $c(L) = L(A, C)$. This implies that both $\mathcal{L}_{\text{PA}}$ and $\mathcal{L}_{\text{DetPA}}$ are closed under commutative closure, as both are classes of semilinear languages [17].

(4) We show that the starring of $L = \{a^n b^n \mid n \in \mathbb{N}\}$ is not in $\mathcal{L}_{\text{PA}}$. Suppose $L^* \in \mathcal{L}_{\text{PA}}$, and let $w = (a^p b^p)^\ell$, where ℓ, p are given by Lemma 3.5. The same lemma asserts that $w = uvxvz$, such that, in particular, uv^2xz and uxv^2z are in L^*. Now suppose $v = a^i$ for some $i \le p$. Then uv^2x contains $a^{p+i}b^p$ with no more b's on the right. Thus $uv^2xz \notin L^*$. The case for $v = b^i$ is similar. Now suppose $v = a^i b^j$ with $i, j > 0$. Then uv^2x contains $\ldots a^p b^j a^i b^p \ldots$, but $i < p$, thus $uv^2xz \notin L^*$. The case $v = b^i a^j$ is similar. Thus $L^* \notin \mathcal{L}_{\text{PA}}$. □

Remark 3.18. Baker and Book [1] already note, in different terms, that if $\mathcal{L}_{\text{PA}}$ were closed under starring, it would be an intersection closed full AFL containing $\{a^n b^n \mid n \ge 0\}$, and so would be equal to the class of Turing-recognizable languages. Thus $\mathcal{L}_{\text{PA}}$ is not closed under starring.

4. Affine Parikh automata

A PA of dimension d can be viewed as an automaton in which each transition updates a vector $\overline{x}$ of $\mathbb{N}^d$ using a function $\overline{x} \leftarrow \overline{x} + \overline{v}$ where $\overline{v}$ depends only on the transition. At the end of an accepting computation, the word is accepted if $\overline{x}$ belongs to some semilinear set. We propose to generalize the updating function to an affine function. We start by defining the model, and show that defining it over $\mathbb{N}$ is as general as defining it over $\mathbb{Q}$. We study the expressiveness of this model and show it is strictly more powerful than PA. We then study its (non)closure properties and related decidability problems, leading to the observation that the model lacks some desirable properties – *e.g.*, properties usually needed for any real-world application.

Let $d, d' > 0$. In the following, we consider the vectors in $\mathbb{K}^d$ to be *column* vectors. A function $f\colon \mathbb{K}^d \to \mathbb{K}^{d'}$ is a (total) *affine function* if there exist a matrix $M \in \mathbb{K}^{d' \times d}$ and $\overline{v} \in \mathbb{K}^{d'}$ such that for any $\overline{x} \in \mathbb{K}^d$, $f(\overline{x}) = M.\overline{x} + \overline{v}$; it is *linear* if $\overline{v} = \overline{0}$. We note such a function $(M, \overline{v})$ and abusively write $f = (M, \overline{v})$. We write $\mathcal{F}_d^{\mathbb{K}}$ for the set of affine functions from $\mathbb{K}^d$ to $\mathbb{K}^d$ and view $\mathcal{F}_d^{\mathbb{K}}$ as the monoid $(\mathcal{F}_d^{\mathbb{K}}, \diamond)$ with $(f \diamond g)(\overline{x}) = g(f(\overline{x}))$.

Definition 4.1 (affine Parikh automaton)**.** A $\mathbb{K}$-*affine Parikh automaton* ($\mathbb{K}$-APA) of dimension d is a triple (A, U, C) where $A = (Q, \Sigma, \delta, q_0, F)$, U is a morphism from δ^* to $\mathcal{F}_d^{\mathbb{K}}$ and $C \subseteq \mathbb{K}^d$ is a $\mathbb{K}$-definable set; recall that U need only be defined on δ, and that, in particular, $U(\varepsilon)$ is the identity function. To simplify the notations, we write U_π for $U(\pi)$. The language of the APA is $L(A, U, C) = \{\mu(\pi) \mid \pi \in \mathsf{Run}(A) \wedge U_\pi(\overline{0}) \in C\}$. The $\mathbb{K}$-APA is said to be *deterministic* ($\mathbb{K}$-DetAPA) if A is. We write $\mathcal{L}_{\mathbb{K}\text{-APA}}$ (resp. $\mathcal{L}_{\mathbb{K}\text{-DetAPA}}$) for the class of languages recognized by $\mathbb{K}$-APA (resp. $\mathbb{K}$-DetAPA).

Remark 4.2. It is easily seen that $\mathbb{N}$-APA (resp. $\mathbb{N}$-DetAPA) are a generalization of CA (resp. DetCA). Indeed, let (A, C) be a CA and define, for $t \in \delta$, $U_t = (Id, \Phi(t))$ where Id is the identity matrix of dimension $|\delta| \times |\delta|$. Then $L(A, C) = L(A, U, C)$. We will later see that this containment is strict.

We present a normal form for APA that is similar to a normal form given for PA by Karianto [15]:

Lemma 4.3. *Every* $\mathbb{K}$-APA (A, U, C) *of dimension* d *has the same language as another* $\mathbb{K}$-APA (A', U', C') *of dimension* $d+1$ *with the three following properties:*

(i) the initial state of A' *has no incoming transition;*
(ii) the automaton A' *is complete;*
(iii) every state of A' *is final.*

The same holds for $\mathbb{K}$-DetAPA.

Proof. Let (A, U, C) be a $\mathbb{K}$-APA of dimension d where $A = (Q, \Sigma, \delta, q_0, F)$, $U\colon \delta^* \to \mathcal{F}_d^{\mathbb{K}}$, and $C \subseteq \mathbb{K}^d$. We ensure incrementally the three properties; that is, we assume for each property that the previous ones hold.

Ensuring (i). We define (A', U', C') as follows: $A' = (Q', \Sigma', \delta', q_0', F')$, where $Q' = Q \cup \{q_{\text{fresh}}\}$, with q_{fresh} a fresh state; $\Sigma' = \Sigma$; $\delta' = \delta \cup \delta_{\text{fresh}}$ with $\delta_{\text{fresh}} = \{(q_{\text{fresh}}, a, q) \mid (q_0, a, q) \in \delta\}$; $q_0' = q_{\text{fresh}}$; if $q_0 \in F$, then $F' = F \cup \{q_{\text{fresh}}\}$ and otherwise $F' = F$. Note that A' is deterministic if A is, and that $L(A) = L(A')$. We define $U'\colon \delta' \to \mathcal{F}_d^{\mathbb{K}}$ as follows. For $(q_{\text{fresh}}, a, q) \in \delta_{\text{fresh}}$, $U'_{(q_{\text{fresh}}, a, q)} = U_{(q_0, a, q)}$ and for $t \in \delta$, we have $U'_t = U_t$. Finally, we let $C' = C$. Then $L(A, U, C) = L(A', U', C')$ and (A', U', C') verifies (i).

Ensuring (ii). Now suppose (A, U, C) verifies (i). Let A' be the automaton A in which an additional nonfinal sink state q_{sink} is added – that is, if a state $q \in Q$ has no outgoing transition labeled $a \in \Sigma$, the transition (q, a, q_{sink}) is added to A', and q_{sink} has $|\Sigma|$ self-loops, labeled by each letter of Σ. The new transitions of A' are associated with some function (for instance, the identity or the zero function); the constraint set C is left unchanged. This leaves both the language and the determinism the APA unchanged, and A' now verifies (i) and (ii).

Ensuring (iii). Now suppose (A, U, C) verifies (i) and (ii). We define A' as $(Q, \Sigma, \delta, q_0, Q)$, *i.e.*, the automaton A with all states final. Let us define the $\mathbb{K}$-APA (A', U', C') of dimension $d+1$ where we fix $U' \colon \delta \to \mathcal{F}^{\mathbb{K}}_{d+1}$ such that the last component of the affine functions serves as a *flag*: it is set to 1 if the last state reached is in F, and 2 otherwise – this component takes the value 0 only for the empty path. Formally, for $t \in \delta, \overline{x} \in \mathbb{K}^d$, and $f \in \mathbb{K}$:

$$U'_t(\overline{x}, f) = \left(U_t(\overline{x}), \begin{cases} 1 & \text{if } \mathsf{To}(t) \in F, \\ 2 & \text{otherwise} \end{cases} \right).$$

Let us remark that the ability of APA to use constant functions (and not only translations, as in PA) allows to simplify the construction given by Karianto [15]. Finally, if $\varepsilon \in L(A, U, C)$, we let $C' = C \times \{1\} \cup \{0^{d+1}\}$, and otherwise, we let $C' = C \times \{1\}$. We argue that $L(A, U, C) = L(A', U', C')$. For the empty word, the construction is such that $\varepsilon \in L(A, U, C) \to \varepsilon \in L(A', U', C')$. Now if $\varepsilon \notin L(A, U, C)$ then $U'_\varepsilon(\overline{0}) = \overline{0} \notin C'$, thus $\varepsilon \notin L(A', U', C')$. Now let w be a nonempty word in $L(A, U, C)$ and let π be an accepting path in A such that $\mu(\pi) = w$ and $U_\pi(\overline{0}) \in C$. Then π is also an accepting path in A', and as $\mathsf{To}(\pi) \in F$, we have that $U'_\pi(\overline{0}) = (U_\pi(0^d), 1)$, and as $U_\pi(\overline{0}) \in C$, we have that $U'_\pi(\overline{0}) \in C \times \{1\}$. Hence $\mu(\pi) = w$ is in $L(A', U', C')$. Conversely, suppose w is a nonempty word in $L(A', U', C')$ and let π be an accepting path in A' such that $\mu(\pi) = w$ and $U'_\pi(\overline{0}) \in C'$. Then π is a path in A, and as $U'_\pi(\overline{0}) = (U_\pi(0^d), 1)$, we have that $\mathsf{To}(\pi) \in F$ and $U_\pi(\overline{0}) \in C$, thus $\mu(\pi) = w$ is in $L(A, U, C)$.

We may verify that (A', U', C') satisfies the three properties and that the language $L(A', U', C')$ is equal to the language $L(A, U, C)$. Moreover, A' is deterministic if A is. □

4.1. Affine Parikh automata on $\mathbb{Q}$ and $\mathbb{N}$

In this section, we show that the expressive power of affine Parikh automata is independent from the choice of $\mathbb{K}$. We first show that the constraint set can have a similar form in the two cases. We call *basic formula* a quantifier-free formula which uses the function symbols $+$ for addition and λ_n, $n \in \mathbb{N}$, for scalar multiplication, together with the relation symbol $<$ and constants from $\mathbb{N}$ – equality is expressible, as $t_1 = t_2$ is equivalent to $\neg(t_1 < t_2) \wedge \neg(t_2 < t_1)$. Of course, the scalar multiplication $\lambda_n(t)$ can be replaced by $t + \ldots + t$ where t appears n times, but its inclusion simplifies the proofs slightly. We remark, for future reference, the following property of basic formulas. For $\overline{v}$ a vector of natural numbers and ϕ a basic formula, the fact that ϕ is true of $\overline{v}$ is independent of the underlying model, whether it is $\mathbb{Q}$ or $\mathbb{N}$. In symbols, $\mathbb{Q} \models \phi(\overline{v})$ iff $\mathbb{N} \models \phi(\overline{v})$.

The following lemma shows in particular that the constraint set of $\mathbb{Q}$-APA can be expressed as a basic formula:

Lemma 4.4. *Every $\mathbb{Q}$-definable set can be expressed as a basic formula.*

Proof. Recall that a $\mathbb{Q}$-definable set can be expressed with a quantifier-free formula ϕ. Thus, we need only get rid of the c's not in $\mathbb{N}$ appearing either as λ_c or as a constant in ϕ. First, note that we can suppose that if $\lambda_c(t)$ appears in ϕ, with t a term, then t is some variable: we simply apply the distributivity of λ_c (*i.e.*, replace $\lambda_c(t_1 + t_2)$ by $\lambda_c(t_1) + \lambda_c(t_2)$ and $\lambda_c(\lambda_{c'}(t))$ by $\lambda_{c\times c'}(t)$), then replace $\lambda_c(c')$ with c' a constant by the constant $c \times c'$, neither of those operations changing the set defined. Second, we take care of the negative c's. For any atomic formula $t_1 < t_2$ appearing in ϕ, if the constant $c < 0$ appears in t_1, we remove it from t_1 and add $-c$ to t_2; if $c < 0$ appears as $\lambda_c(x)$ in t_1, with x a variable, we remove it from t_1 and add $\lambda_{-c}(x)$ to t_2 (the same goes with t_1 and t_2 switched). Third and last, we take care of the denominators: let N be the product of all the denominators appearing in the reduced fractions of the c's appearing in ϕ. Then any atomic formula $t_1 > t_2$ is replaced with the atomic formula $t'_1 > t'_2$ where any c (appearing either as a constant or as λ_c) is replaced by $N \times c$: the fact that $c \geq 0$ implies that $(N \times c) \in \mathbb{N}$. Moreover, for any assignment, the value of t'_1 (resp. t'_2) is N times the value of t_1 (resp. t_2), hence, the value of t_1 is greater than the value of t_2 iff the same holds for t'_1 and t'_2. □

Over $\mathbb{N}$, the automaton is needed to incorporate some of the constraint set:

Lemma 4.5. *Every* $\mathbb{N}$-APA (A, U, C) *has the same language as another* $\mathbb{N}$-APA (A, U', C') *where* C' *can be expressed as a basic formula. The same holds for* $\mathbb{N}$-DetAPA.

Proof. Recall that a semilinear set can be expressed as a basic formula with the additional relations $\equiv_p$, expressing congruence (*e.g.*, [6]). Thus we need only get rid of these relations. To do so, we equip the affine functions to compute their own value modulo p.

Let (A, U, C) be an $\mathbb{N}$-APA (resp. $\mathbb{N}$-DetAPA) of dimension d for which we suppose the initial state of A has no incoming transition (Lem. 4.3), and let $\phi(x_1, \ldots, x_d)$ be the formula for C of the form previously mentioned (*i.e.*, a basic formula with the additional relations $\equiv_p$). Suppose ϕ is not a basic formula, then there is a p such that $\equiv_p$ appears in ϕ. We define (A, U', C') of the same language as (A, U, C) with C' expressed by ϕ in which the $\equiv_p$ relation, for this specific p, is replaced by some basic formulas. Applying this process repeatedly gives an $\mathbb{N}$-APA (resp. $\mathbb{N}$-DetAPA) of the same language with its constraint set expressible as a basic formula.

Our goal is to modify U so that for each $\overline{v} \in \{0, \ldots, p-1\}^d$, there is an additional variable $m_{\overline{v}}$ available to ϕ which is set to 1 iff the value of x_i modulo p is v_i, for all $1 \leq i \leq d$ (thus only one of the $m_{\overline{v}}$'s can be set to 1). With this information available, all the atomic formulas of the form $t_1 \equiv_p t_2$, for this specific p, can be rewritten without $\equiv_p$ using a basic formula:

$$t_1 \equiv_p t_2 \qquad \rightsquigarrow \qquad \bigvee_{\substack{\overline{v} \in \{0,\ldots,p-1\}^d \\ t_1(\overline{v}) \equiv_p t_2(\overline{v})}} \qquad (m_{\overline{v}} = 1).$$

Let t be a transition of A; we give $U'_t \in \mathcal{F}^{\mathbb{N}}_{d+p^d}$. In order to do this, we define an additional 0-1-matrix M_t of dimension $p^d \times p^d$, which we index by vectors in $\{0,\ldots,p-1\}^d$ in some natural way (in particular, 0^d is the index of the first row). We let $M_t[\overline{u},\overline{v}] = 1$ iff $\overline{u} = U_t(\overline{v}) \bmod p$, where the modulo is taken component-wise.

We are now ready to define $U'_t(\overline{x},\overline{m})$, for $\overline{x} \in \mathbb{N}^d$ and $\overline{m} \in \mathbb{N}^{p^d}$. If t is an outgoing transition of the initial state of A, then $U'_t(\overline{x},\overline{m}) = (U_t(\overline{x}), M_t.(1,\overline{0}))$, where $(1,\overline{0})$ is the column vector $(1,0,\ldots,0) \in \mathbb{N}^{p^d}$. Otherwise, we let $U'_t(\overline{x},\overline{m}) = (U_t(\overline{x}), M_t.\overline{m})$. Note that in both definitions, U'_t is indeed an affine function. Now with $\overline{m_1}$ (resp. $\overline{m_2}$) the vector in $\{0,1\}^{p^d}$ having a 1 only in position $\overline{x} \bmod p$ (resp. $U_t(\overline{x}) \bmod p$), we have $U'_t(\overline{x},\overline{m_1}) = (U_t(\overline{x}),\overline{m_2})$. Moreover, the initial value of $\overline{x}$ being 0^d, those hypotheses are established at the first transition taken. Thus for a nonempty path π, and with $\overline{m}$ the vector in $\{0,1\}^{p^d}$ having a 1 only in position $U_\pi(0^d) \bmod p$, we have: $U'_\pi(\overline{0}) = (U_\pi(0^d),\overline{m})$.

As previously discussed, ϕ can now be rewritten as ϕ' without the use of $\equiv_p$: ϕ' has access to the usual variables $x_1,\ldots,x_d$ and to variables $m_{\overline{v}}$ for $\overline{v} \in \{0,\ldots,p-1\}^d$. We take care of the empty word by letting ϕ' consider $m_{\overline{0}}$ to be 1 if no other $m_{\overline{v}}$ variable is set. Thus, with C' the set defined by ϕ', we have that $L(A,U,C) = L(A,U',C')$ and ϕ' has one less p appearing as $\equiv_p$ than ϕ. $\square$

Before proving the main result of this section, we show that *affine* functions, in their full generality, are not needed within $\mathbb{K}$-APA or $\mathbb{K}$-DetAPA:

Lemma 4.6. *The language of any* $\mathbb{K}$-APA *is also the language of a* $\mathbb{K}$-APA *in which every affine function is either constant or linear – for both* $\mathbb{K} = \mathbb{Q}$ *and* $\mathbb{K} = \mathbb{N}$. *Moreover, the first transition of a run is always associated with a constant function. The same holds for* $\mathbb{K}$-DetAPA.

Proof. Let (A,U,C) be a $\mathbb{K}$-APA (resp. $\mathbb{K}$-DetAPA) of dimension d and suppose, thanks to Lemma 4.3, that the initial state of A has no incoming transition. We define a $\mathbb{K}$-APA (resp. $\mathbb{K}$-DetAPA) (A',U',C') where the outgoing transitions of the initial state of A initialize the registers with the values of all the constant parts given by U. Specifically, we define the morphism $U' \colon \delta' \to \mathcal{F}^{\mathbb{K}}_{d+dn}$ as follows. Identify the transition set of A with $\{t_1,\ldots,t_n\}$, write $U_{t_i} = (M_i,\overline{v_i})$, for $i \in \{1,\ldots,n\}$, and define $\hat{v} = (\overline{v_1},\ldots,\overline{v_n}) \in \mathbb{K}^{dn}$. Then for t an outgoing transition of the initial state, U'_t is the constant function with value $(U_t(0^d),\hat{v})$; for the other

t_i's, we set $U'_{t_i}(\overline{x}, \overline{y_1}, \ldots, \overline{y_n}) = (M_i.\overline{x} + \overline{y_i}, \overline{y_1}, \ldots, \overline{y_n})$, and in this case, U'_{t_i} is the linear function $(M'_i, \overline{0})$ where here and in the following $\overline{0}$ is of dimension $d + dn$ and:

$$M'_i = \begin{array}{c} \begin{array}{cccccccc} & & & & \overset{(i+1)\text{-th block of width } d}{\downarrow} & & & \end{array} \\ \begin{pmatrix} M_i & \mathbb{O}_d & \cdots & \mathbb{O}_d & Id_d & \mathbb{O}_d & \cdots & \mathbb{O}_d \\ \mathbb{O}_d & & & & & & & \\ \vdots & & & & Id_{dn} & & & \\ \mathbb{O}_d & & & & & & & \end{pmatrix} \end{array}$$

with $\mathbb{O}_k$ (resp. Id_k) the zero (resp. identity) matrix of dimension $k \times k$. Finally, we let $C' = C \times \mathbb{K}^{dn}$.

We now show that $L(A, U, C) = L(A, U', C')$. First, $\varepsilon \in L(A, U, C)$ iff $\varepsilon \in L(A)$ and $U_\varepsilon(\overline{0}) = \overline{0} \in C$, the latter being equivalent to $U'_\varepsilon(\overline{0}) = \overline{0} \in C'$, thus $\varepsilon \in L(A, U, C)$ iff $\varepsilon \in L(A, U', C')$. Now let π be a path in A starting from the initial state. Suppose $|\pi| = 1$ then $U'_\pi(\overline{0}) = (U_\pi(0^d), \hat{v})$. For $|\pi| > 1$, let $\pi = \rho t$, then:

$$U'_\pi(\overline{0}) = U'_t(U'_\rho(\overline{0})) \underset{\text{by induction}}{=} U'_t(U_\rho(0^d), \hat{v}) = (U_\pi(0^d), \hat{v}).$$

Thus let w be a nonempty word in $L(A, U, C)$ and let π be an accepting path in A labeled w and such that $U_\pi(\overline{0}) \in C$. Then $U'_\pi(\overline{0}) = (U_\pi(0^d), \ldots)$ which is in $C \times \mathbb{K}^{dn} = C'$, thus $w \in L(A, U', C')$. Conversely, let w be a nonempty word in $L(A, U', C')$ and let π be an accepting path in A labeled w such that $U'_\pi(\overline{0}) \in C'$. We have that $U'_\pi(\overline{0}) = (U_\pi(0^d), \ldots)$ is in $C' = C \times \mathbb{K}^{dn}$, thus $U_\pi(0^d) \in C$ and $w \in L(A, U, C)$. □

We are now ready to show that the choice of $\mathbb{K}$ in APA does not influence the class of languages defined:

Theorem 4.7. *$\mathcal{L}_{\mathbb{Q}\text{-DetAPA}} = \mathcal{L}_{\mathbb{N}\text{-DetAPA}}$ and $\mathcal{L}_{\mathbb{Q}\text{-APA}} = \mathcal{L}_{\mathbb{N}\text{-APA}}$. Moreover, these correspondences are effective and do not change the underlying automaton.*

Proof. ($\mathcal{L}_{\mathbb{Q}\text{-APA}} \subseteq \mathcal{L}_{\mathbb{N}\text{-APA}}$ and $\mathcal{L}_{\mathbb{Q}\text{-DetAPA}} \subseteq \mathcal{L}_{\mathbb{N}\text{-DetAPA}}$). Let (A, U, C) be a $\mathbb{Q}$-APA (resp. $\mathbb{Q}$-DetAPA) of dimension d, with δ the set of transitions of A. The underlying automaton A will remain the same throughout the proof.

We suppose that the empty word is not in $L(A)$ it is a simple task to add it back at the very end of this construction if needed. Thanks to Lemma 4.6, we assume that all the functions given by U are either linear or constant. Lemma 4.4 then asserts that C is expressible as a basic formula ϕ. We first ensure that no constant appears in ϕ by replacing each of them by a variable (*e.g.*, if $\lambda_3(x) + 8$ is a term in ϕ, we replace it by $\lambda_3(x) + y$ where y is a new variable). Let ϕ' be this modified formula and, for $c_1 < c_2 < \ldots < c_p \in \mathbb{K}$ he increasing sequence of the p constants that appear in ϕ, let $y_1, y_2, \ldots, y_p$ be the associated sequence of new variables. We now update U so that it gives the value c_i to y_i, for all i. Let $\overline{c} = (c_1, c_2, \ldots, c_p)$ and define U' from U as follows. For t such that U_t is constant, set $U'_t(\overline{x_1}, \overline{x_2}) = (U_t(0^d), \overline{c})$, which is still a constant function; and for t such that U_t is linear, set $U'_t(\overline{x_1}, \overline{x_2}) = (U_t(\overline{x_1}), \overline{x_2})$, which is also still linear. We let C' be the set described by ϕ'; it verifies $\{\overline{x} \mid (\overline{x}, \overline{c}) \in C'\} = C$. As the first transition of any run in A is associated, by U, with a constant function, any nonempty run π in A verifies $U'_\pi(\overline{0}) = (U_\pi(0^d), \overline{c})$. Thus the variables $y_1, \ldots, y_p$ of ϕ' are indeed set to $c_1, \ldots, c_p$, implying that $L(A, U, C) = L(A, U', C')$. From now on we denote $d + p$ by n.

We now change U' so that the constants and matrices appearing in the U'_t's are all integer-valued. Let N be the product of all the denominators appearing in the reduced fractions of the entries of the matrices and vectors given by U'. By defining $U''_t = N \times U'_t$, we thus ensure that all the values appearing in the definition of U''_t are integers, hence U'' is a function from δ^* to $\mathcal{F}^{\mathbb{Z}}_n$. Moreover, as all the functions are either linear or constant, this implies that for any path π, there is a $k \leq |\pi|$ such that $U''_\pi = N^k \times U'_\pi$. But as all the atomic formulas of ϕ' are of the form $t_1 < t_2$ where no constant appears and all the λ_c have $c > 0$, we have that $\phi'(\overline{v})$ is true iff $\phi'(K \times \overline{v})$ is true. Thus $L(A, U, C) = L(A, U'', C')$.

Finally, we change U'' and C' so that they are $\mathbb{N}$-valued. We define U''' as U'' where the positive and negative computations are made in different components. Consider U''_t as n affine functions from $\mathbb{Q}^n$ to $\mathbb{Q}$: $U''_t(\overline{x}) = (f_1(\overline{x}), \ldots, f_n(\overline{x}))$. Then let $1 \leq i \leq n$, and write $f_i(\overline{x}) = c + \sum_{j=1}^n v_j \times x_j$.

Let us write $J^+ = \{1 \leq j \leq n \mid v_j \geq 0\}$ and $J^- = \{1 \leq j \leq n \mid v_j < 0\}$. Now, we define f_i^+ and f_i^- by:

$$f_i^+(\overline{x^+}, \overline{x^-}) = \max(c, 0) + \sum_{j \in J^+} |v_j| \times x_j^+, \text{and,}$$
$$f_i^-(\overline{x^+}, \overline{x^-}) = |\min(c, 0)| + \sum_{j \in J^-} |v_j| \times x_j^-.$$

Now define $U'''_t : \mathbb{N}^{2n} \to \mathbb{N}^{2n}$ as $U'''_t(\overline{x^+}, \overline{x^-}) = (f_1^+, f_1^-, \ldots, f_n^+, f_n^-)(\overline{x^+}, \overline{x^-})$, where $\overline{x^+}, \overline{x^-} \in \mathbb{N}^n$. The main property of this construction is that for a path π, we have:

$$U'''_\pi(\overline{0}) = (a_1^+, a_1^-, \ldots, a_n^+, a_n^-) \quad \Rightarrow \quad U''_\pi(\overline{0}) = (a_1^+ - a_1^-, \ldots, a_n^+ - a_n^-).$$

Thus define C'' as:

$$C'' = \{(a_1^+, a_1^-, \ldots, a_n^+, a_n^-) \mid (a_1^+ - a_1^-, \ldots, a_n^+ - a_n^-) \in C'\}.$$

Now C'' is a $\mathbb{Q}$-definable set because C' is $\mathbb{Q}$-definable, thus C'' is expressible as a basic formula. But basic formulas on natural numbers take their truth values regardless of whether $\mathbb{K} = \mathbb{N}$ or $\mathbb{K} = \mathbb{Q}$, thus $C'' \cap \mathbb{N}^{2n}$ is $\mathbb{N}$-definable. Finally, $(A, U''', C'' \cap \mathbb{N}^{2n})$ is an $\mathbb{N}$-APA (resp. $\mathbb{N}$-DetAPA) of the same language as (A, U, C).

($\mathcal{L}_{\mathbb{N}\text{-APA}} \subseteq \mathcal{L}_{\mathbb{Q}\text{-APA}}$ and $\mathcal{L}_{\mathbb{N}\text{-DetAPA}} \subseteq \mathcal{L}_{\mathbb{Q}\text{-DetAPA}}$). This is a consequence of Lemma 4.5. Let (A, U, C) be an $\mathbb{N}$-APA (resp. $\mathbb{N}$-DetAPA). Now, by Lemma 4.5, let (A, U', C') be an $\mathbb{N}$-APA (resp. $\mathbb{N}$-DetAPA) with the same language and with C' expressible as a basic formula. The fact that basic formulas take their truth value on natural numbers regardless of the underlying model implies that there exists a $\mathbb{Q}$-definable set C'' such that $C'' \cap \mathbb{N}^d = C'$ – this is the set described by the basic formula for C' interpreted in $\mathbb{Q}$ – and thus (A, U', C'') is a $\mathbb{Q}$-APA (resp. $\mathbb{Q}$-DetAPA) of the same language as (A, U, C). $\square$

The previous result allows us to write $\mathcal{L}_{\text{DetAPA}}$ for $\mathcal{L}_{\mathbb{Q}\text{-DetAPA}} = \mathcal{L}_{\mathbb{N}\text{-DetAPA}}$ and $\mathcal{L}_{\text{APA}}$ for $\mathcal{L}_{\mathbb{Q}\text{-APA}} = \mathcal{L}_{\mathbb{N}\text{-APA}}$.

4.2. Closure properties of $\mathcal{L}_{\text{APA}}$ and $\mathcal{L}_{\text{DetAPA}}$

The *pointed* concatenation of L and L' is any language of the form $L \cdot \{\#\} \cdot L'$ where # does not appear in a word of L. The arguments used by Klaedtke and Rueß [16] apply equally well to $\mathbb{K}$-APA and $\mathbb{K}$-DetAPA, showing:

Proposition 4.8. *(1) $\mathcal{L}_{\text{APA}}$ is closed under union, intersection, concatenation, nonerasing morphisms, and inverse morphisms; (2) $\mathcal{L}_{\text{DetAPA}}$ is closed under union, intersection, inverse morphisms, complement, and pointed concatenation.*

Proof. (Union and intersection). Let (A', U', C') and (A'', U'', C'') be two $\mathbb{K}$-APA (resp. $\mathbb{K}$-DetAPA) of dimension d' and d'', respectively, and suppose that A' and A'' are complete and with every state final (Lem. 4.3). We suppose moreover, w.l.o.g., that the alphabets of the automata are the same. Let $L' = L(A', U', C')$ and $L'' = L(A'', U'', C'')$. We construct two $\mathbb{K}$-APA (resp. $\mathbb{K}$-DetAPA) $(A, U, C^{\cup})$ and $(A, U, C^{\cap})$ such that their languages are the union and intersection, respectively, of L' and L''. Let $A' = (Q', \Sigma', \delta', q_0', Q')$ and $A'' = (Q'', \Sigma', \delta'', q_0'', Q'')$, and define the Cartesian product of A' and A'' by $A = (Q' \times Q'', \Sigma', \delta, (q_0', q_0''), Q' \times Q'')$ with:

$$\delta = \{((p', p''), a, (q', q'')) \mid (p', a, q') \in \delta' \wedge (p'', a, q'') \in \delta''\}.$$

This automaton is deterministic if both A' and A'' are. Define h' (resp. h''), to be the morphism from δ^* to $(\delta')^*$ (resp. to $(\delta'')^*$) such that $h'((p', p''), a, (q', q'')) = (p', a, q')$ (resp. $h''((p', p''), a, (q', q'')) = (p'', a, q'')$); the fact that A' and A'' are complete implies that for any run π' in A' and π'' in A'' with the same label, there is a run π in A such that $h'(\pi) = \pi'$ and $h''(\pi) = \pi''$. Then we let $U\colon \delta^* \to \mathcal{F}^{\mathbb{K}}_{d'+d''}$

compute the values of U' in the first d' components and the values of U'' in the last d'' components, that is, for $\overline{x'} \in \mathbb{K}^{d'}$, $\overline{x''} \in \mathbb{K}^{d''}$, and $t \in \delta$:

$$U_t(\overline{x'}, \overline{x''}) = (U'_{h'(t)}(\overline{x'}), U''_{h''(t)}(\overline{x''})).$$

Finally, we let $C^{\cup} = C' \times \mathbb{K}^{d''} \cup \mathbb{K}^{d'} \times C''$ and $C^{\cap} = C' \times C''$. We argue that $L(A, U, C^{\cup}) = L' \cup L''$ and $L(A, U, C^{\cap}) = L' \cap L''$.

Let π be a run in A. Then $h'(\pi)$ is a run in A', $h''(\pi)$ is a run in A'', and both have the same label as π. Moreover, $U_\pi(\overline{0}) = (U'_{h'(\pi)}(0^{d'}), U''_{h''(\pi)}(0^{d''}))$. Thus if $U_\pi(\overline{0}) \in C^{\cup}$ then $\mu_A(\pi) \in L(A, U, C^{\cup})$, and $U'_{h'(\pi)}(0^{d'}) \in C'$ or $U''_{h''(\pi)}(0^{d''}) \in C''$, thus $\mu_A(\pi) \in L' \cup L''$. Likewise, if $U_\pi(\overline{0}) \in C^{\cap}$ then $\mu_A(\pi) \in L(A, U, C^{\cap})$, and both $U'_{h'(\pi)}(0^{d'}) \in C'$ and $U''_{h''(\pi)}(0^{d''}) \in C'$ thus $\mu_A(\pi) \in L' \cap L''$.

For the converse, let $w \in L'$ and let π' be a run in A' such that $\mu_{A'}(\pi') = w$ and $(U'(\pi'))(\overline{0}) \in C'$. Then there is a run π in A such that $h'(\pi) = \pi'$. Moreover, $U_\pi(\overline{0}) = ((U'(h'(\pi)))(\overline{0}), (U''(h''(\pi)))(\overline{0}))$ which is in $C' \times \mathbb{K}^{d''}$, thus in $C^{\cup}$, and thus $w \in L(A, U, C^{\cup})$. Likewise, if $w \in L''$, then $w \in L(A, U, C^{\cup})$. Now let $w \in L' \cap L''$, and let π' (resp. π'') be a run in A' (resp. A'') such that $\mu_{A'}(\pi') = w$ and $U'_{\pi'}(0^{d'}) \in C'$ (resp. $\mu_{A''}(\pi'') = w$ and $U''_{\pi''}(0^{d''}) \in C''$). There exists a path π in A such that $h'(\pi) = \pi'$ and $h''(\pi) = \pi''$, and it is such that $U_\pi(\overline{0}) = (U'_{h'(\pi')}(\overline{0}), U''_{h''(\pi')}(\overline{0}))$ which is in $C' \times C''$, that is, $C^{\cap}$, thus $w \in L(A, U, C^{\cap})$.

(Inverse morphisms). We first tackle the $\mathbb{K}$-DetAPA case, which is based on the classical construction on finite automata and followed by the addition of the affine functions. Let (A, U, C) be a $\mathbb{K}$-DetAPA over the alphabet Σ, and let $h\colon \Sigma'^* \to \Sigma^*$ be a morphism; we will give a $\mathbb{K}$-DetAPA (A', U', C) for the language $h^{-1}(L(A, U, C))$. We first construct A' such that its language is $h^{-1}(L(A))$. Let $A = (Q, \Sigma, \delta, q_0, F)$, and write $\mathsf{Path}(q, u, q')$ for the only path in A from q to q' labeled u if it exists, $\perp$ otherwise. Further, we let $\mathsf{Path}(q, \varepsilon, q) = \varepsilon$, *i.e.*, we consider that the empty path is going and ending in any given state. Then $A' = (Q, \Sigma', \delta', q_0, F)$ where:

$$\delta' = \{(q, a, q') \in Q \times \Sigma' \times Q \mid \mathsf{Path}(q, h(a), q') \neq \perp\}.$$

The automaton A' is such that $L(A') = h^{-1}(L(A))$ and is deterministic. In particular, if $h(a) = \varepsilon$, then a loop labeled a appears on each state.

When a word w is read in A' from some state q to a state q', the equivalent action in A is to take the path $\mathsf{Path}(q, h(w), q')$; thus we let $U'_{(q,a,q')} = U_{\mathsf{Path}(q,h(a),q')}$, and in particular, if a transition is labeled with a letter a such that $h(a) = \varepsilon$, then the associated function is the identity. Then for π' a path in A' and π its counterpart in A (that is, $\pi = \mathsf{Path}(\mathsf{From}(\pi'), h(\mu(\pi')), \mathsf{To}(\pi'))$), we have that $U'_{\pi'} = U_\pi$. Now let $w \in L(A')$, π' be the accepting path with label w in A', and π be the accepting path with label $h(w)$ in A. Then $U'_{\pi'}(\overline{0}) = U_\pi(\overline{0})$. Thus we have that $h(w) \in L(A, U, C)$ iff $h(w) \in L(A)$ and the path π for $h(w)$ in A is such that $U_\pi(\overline{0}) \in C$, which is the case iff $w \in L(A')$ and the path π' for w in A' is such that $U'_{\pi'}(\overline{0}) \in C$, that is iff $w \in L(A', U', C)$, concluding this case.

We now focus on the nondeterministic case. Let (A, U, C) be a $\mathbb{K}$-APA over the alphabet Σ and let $h\colon \Sigma'^* \to \Sigma^*$ be a morphism. Here, for some states q, q' of A, we may have several paths from q to q' with the same label – say we have k paths. To circumvent this problem, we use at least k copies of the A' of the deterministic case: we go from the i-th copy of q to the j-th of q' applying the affine functions corresponding to the j-th of the k paths.

Formally, let $A = (Q, \Sigma, \delta, q_0, F)$. Define $\mathsf{Paths}(q, u, q')$ as the *set* of paths in A from q to q' labeled u, and impose an order on this set (say, lexicographical order). Again, we consider the empty path as going and ending in any given state, thus we let $\mathsf{Paths}(q, \varepsilon, q) = \{\varepsilon\}$. Let M be the maximum number of elements in $\mathsf{Paths}(q, h(a), q')$ for $q, q' \in Q$, and $a \in \Sigma'$. We define an A' similar to the deterministic case, but duplicated M times to obtain the $\mathbb{K}$-APA (A', U', C) for $h^{-1}(L(A, U, C))$. For $1 \le i \le M$ and for a state q in A', we write q_i for a fresh copy of q indexed by i (when $q = q_0$ we write $(q_0)_i$ as $q_{0,i}$); we use this notation to define an automaton A' that includes M copies of the deterministic case one. Let $A' = (Q', \Sigma', \delta', q_{0,1}, F')$ where:

- $Q' = \{q_i \mid q \in Q \wedge 1 \le i \le M\}$;
- $\delta' = \{(q_i, a, q'_j) \in Q' \times \Sigma' \times Q' \mid 1 \le i, j \le |\mathsf{Paths}(q, h(a), q')|\}$;
- $F' = \{q_i \mid q \in F \wedge 1 \le i \le M\}$.

Again we have that $L(A') = h^{-1}(L(A))$. Finally, define U' by:

$$U'_{(q_i, a, q'_j)} = U_\pi \text{ where } \pi \text{ is the } j\text{-th path in } \mathsf{Paths}(q, h(a), q').$$

The deterministic case corresponds to $M = 1$, and in this case, the constructed $\mathbb{K}$-APA is the same as in the previous construction. Now suppose $\mathsf{Paths}(q, h(a), q')$ has more than two elements for some $q, q' \in Q$ and $a \in \Sigma'$. In particular, the two transitions (q_1, a, q'_1) and (q_1, a, q'_2) are in A'; the affine functions associated are such that taking the first (resp. second) transition applies the same function as going through the first (resp. second) path of $\mathsf{Paths}(q, h(a), q')$ in A'. Thus, once again, the possible values computed by the affine functions while reading some $h(w)$ in A are the same as those computed while reading w in A'. By the same token as in the deterministic case, $L(A', U', C) = h^{-1}(L(A, U, C))$.

(Concatenation). Let (A', U', C') and (A'', U'', C'') be two $\mathbb{K}$-APA of dimension d' and d'', respectively, and write $L' = L(A', U', C')$ and $L'' = L(A'', U'', C'')$. We construct a $\mathbb{K}$-DetAPA (A, U, C) of dimension $d' + d''$ for $L = L' \cdot L''$. Here, A is the merging of A' and A'', where for all transitions in A'' from the initial state to some state q, a transition from each final state of A' to q with the same label is added. We then compute U' and U'' in parallel.

Formally, let $A' = (Q', \Sigma', \delta', q'_0, F')$ and $A'' = (Q'', \Sigma'', \delta'', q''_0, F'')$, and suppose $Q' \cap Q'' = \emptyset$. We assume that $\varepsilon \notin L(A'')$; otherwise, if $\varepsilon \in L''$, then $L = L' \cdot (L'' \setminus \{\varepsilon\}) \cup L'$, and the closure under union allows us to conclude. Define A as the deterministic automaton $(Q, \Sigma, \delta, q_0, F)$ where:

- $Q = Q' \cup Q''$, $\Sigma = \Sigma' \cup \Sigma''$;
- $\delta = \delta' \cup \delta'' \cup \{(p, a, q) \mid p \in F' \wedge (q_0'', a, q) \in \delta''\}$;
- $q_0 = q_0'$ and $F = F''$.

The language of A is thus $L(A') \cdot L(A'')$. We define $U\colon \delta^* \to \mathbb{K}^{d'+d''}$ so that the d' first components are used for the computations of A', and the d'' last for the computations of A'', *i.e.*, for $\overline{x} \in \mathbb{K}^{d'}$ and $\overline{y} \in \mathbb{K}^{d''}$, we let $U_t(\overline{x}, \overline{y})$ be $(U_t'(\overline{x}), \overline{y})$ if $t \in \delta'$, $(\overline{x}, U_t''(\overline{y}))$ if $t \in \delta''$, and $(U_{(q_0'',a,q)}(\overline{x}), \overline{y})$ if $t = (p, a, q) \notin \delta' \cup \delta''$. Finally, we let C to be the $\mathbb{K}$-definable set $C' \times C''$.

Let $\pi \in \mathsf{Run}(A)$, then π can be written as $\pi'(p, a, q)\pi''$ where $\pi' \in \mathsf{Run}(A')$ and $(q_0'', a, q)\pi'' \in \mathsf{Run}(A'')$. Conversely, for two paths $\pi' \in \mathsf{Run}(A')$, $(q_0'', a, q)\pi'' \in \mathsf{Run}(A'')$, the path $\pi'(\mathsf{To}(\pi'), a, q)\pi''$ is a run in A. Moreover, in both cases, it holds that:

$$U_\pi(\overline{0}) = (U'_{\pi'}(0^{d'}), U''_{\pi''}(0^{d''})).$$

Thus $L = L(A, U, C)$.

(Nonerasing morphisms). Let (A, U, C) be a $\mathbb{K}$-APA over the alphabet Σ and $h\colon \Sigma^* \to \Sigma'^*$ be a nonerasing morphism, that is, for all $a \in \Sigma, h(a) \neq \varepsilon$. We construct a $\mathbb{K}$-APA for $h(L(A, U, C))$ where the main task is the following. For a letter $a \in \Sigma$ and $w = w_1 \dots w_n = h(a)$, a transition $t = (q, a, q')$ of A is replaced by n transitions $(q, w_1, q_{t,1})$, $\dots$, $(q_{t,n-1}, w_n, q')$ where the $q_{t,i}$'s are fresh states named after the transition t. To make the proof concise, we rely on the closure under inverse morphism of $\mathbb{K}$-APA, previously shown. We give a $\mathbb{K}$-APA (A', U', C) for the image of $h(L(A, U, C))$ under the morphism g which maps $a \in \Sigma'$ to $a\#$, for $\# \notin \Sigma'$; we then have that $h(L(A, U, C)) = g^{-1}(L(A', U', C'))$, concluding the proof.

Formally, let $A = (Q, \Sigma, \delta, q_0, F)$ and for $t \in \delta$, write $q_{t,i}^{\perp}$ and $q_{t,i}^{\top}$ to denote some fresh states. Let $\#$ be a symbol not in Σ'. Then $A' = (Q', \Sigma' \cup \{\#\}, \delta', q_0, F)$ where:

- $Q' = Q \cup \{q_{t,i}^{\perp}, q_{t,i}^{\top} \mid t \in \delta \wedge 1 \leq i \leq |h(\mu(t))|\}$;
- $\delta' = \{(q, w_1, q_{t,1}^{\perp}), (q_{t,i}^{\perp}, \#, q_{t,i}^{\top}), (q_{t,i}^{\top}, w_{i+1}, q_{t,i+1}^{\perp}), (q_{t,n}^{\perp}, \#, q') \mid (q, a, q') \in \delta \wedge w_1 \dots w_n = h(a) \wedge 1 \leq i < n\}$.

We now adjust U' so that the computations of the two $\mathbb{K}$-APA are the same. We let U_t' be the identity function for any t with $\mathsf{From}(t) \notin Q$, and for $t' = (q, a, q_{t,1}^{\perp})$, we let $U_{t'}' = U_t$. We argue that this $\mathbb{K}$-APA recognizes $h(L(A, U, C))$ with a $\#$ in every even position. First, $L(A')$ is $h(L(A))$ in which a $\#$ is inserted in every even position. Next, let $w_1\# \dots \#w_n\# \in L(A')$ with $w_i \in \Sigma'$, and let π' be a run with this label in A' such that $U'_{\pi'}(\overline{0}) \in C$. Let π be the corresponding path in A defined by replacing each transition of the form $(q, a, q_{t,1}^{\perp})$ by t and removing the other transitions. Then π is an accepting path, its label is in $h^{-1}(w_1 \dots w_n)$, and $U_\pi(\overline{0}) = U'_{\pi'}(\overline{0})$, thus $w_1 \dots w_n \in h(L(A, U, C))$. Conversely, if $w \in L(A, U, C)$, then let π be a run with label w in A such that $U_\pi(\overline{0}) \in C$. Then the path π' in

A' whose only states of the form $q^{\top}_{t,1}$ are $q^{\top}_{\pi_1,1}, \dots, q^{\top}_{\pi_{|\pi|},1}$ in that order is accepting and such that $U'_{\pi'}(\overline{0}) = U_\pi(\overline{0})$ which is in C. Thus its label, which is $h(w)$ with # inserted in every even position, is in $L(A', U', C)$.

(Complement). Let (A, U, C) be a $\mathbb{K}$-DetAPA. A word is *not* in $L(A, U, C)$ iff it is not in $L(A)$ or, while being in $L(A)$, the path π corresponding to the word is such that $U_\pi(\overline{0}) \notin C$. Thus the complement of $L(A, U, C)$ is $\overline{L(A)} \cup L(A, U, \overline{C})$, which is in $\mathcal{L}_{\mathbb{K}\text{-DetAPA}}$, semilinear sets being closed under complement.

(Pointed concatenation). This is similar to the closure under concatenation for the nondeterministic case. Let (A', U', C') and (A'', U'', C'') be two $\mathbb{K}$-DetAPA and # a symbol not in the alphabet of A'. The main difference with the closure under concatenation of $\mathbb{K}$-APA is that the automaton A is constructed by adding #-labeled transitions from the final states of A' to the initial state of A''. As # is a symbol which is not in the alphabet of A', this preserves the determinism. □

Remark 4.9. These closures are *effective* in the sense that for every operation (*e.g.*, intersection of $\mathbb{K}$-APA), there is an algorithm which computes it (*e.g.*, given two $\mathbb{K}$-APA computes a $\mathbb{K}$-APA whose language is the intersection of the languages of the two). Also, we give the closure of $\mathcal{L}_{\text{DetAPA}}$ under *pointed* concatenation because we were not able to give a construction for the usual concatenation – we even conjecture that $\mathcal{L}_{\text{DetAPA}}$ is not closed under the usual concatenation.

We now give a large class of languages belonging to $\mathcal{L}_{\text{APA}}$ in two steps. First, we show that the language PAL of pointed palindromes, *i.e.*, $\text{PAL} = \{w\#w^{\text{R}} \mid w \in \{a,b\}^*\}$, is recognized by a deterministic APA:

Proposition 4.10. $\text{PAL} \in \mathcal{L}_{\text{DetAPA}}$.

Proof. We sketch an $\mathbb{N}$-DetAPA (A, U, C) for PAL over $\{0,1\}^*$ rather than $\{a, b\}$. The automaton A accepts words of the form $u\#v$, with $u, v \in \{0,1\}^*$. The affine functions compute the value of u (resp. v) seen as a binary number with the most (resp. least) significant bit first. Checking that those values are equal and that $|u| = |v|$ is then the same as checking that $u = v^{\text{R}}$. Formally, $A = (\{q_0, q_1\}, \{0, 1, \#\}, \delta, q_0, \{q_1\})$ where δ is defined, together with the affine functions of U, by:

$$\begin{aligned}
t_1 &= (q_0, 0, q_0) \text{ performs } (x,p,y,\ell) \mapsto (2x, 0, 0, \ell+1),\\
t_2 &= (q_0, 1, q_0) \text{ performs } (x,p,y,\ell) \mapsto (2x+1, 0, 0, \ell+1),\\
t_3 &= (q_0, \#, q_1) \text{ performs } (x,p,y,\ell) \mapsto (x, 1, 0, \ell),\\
t_4 &= (q_1, 0, q_1) \text{ performs } (x,p,y,\ell) \mapsto (x, 2p, y, \ell-1),\\
t_5 &= (q_1, 1, q_1) \text{ performs } (x,p,y,\ell) \mapsto (x, 2p, y+p, \ell-1).
\end{aligned}$$

Now when reading a word $u \in \{0,1\}^*$ from q_0, with x, p, y, and ℓ starting at 0, the final value is $(x, 0, 0, |u|)$ where x is the value of u seen as a binary number with the most significant bit first. Reading u from q_1 with starting value $(x, 1, 0, \ell)$ leads to the value $(x, 2^{|u|}, y, \ell - |u|)$ with y the value of u seen as a binary number with the least significant bit first. Thus, letting C to be the semilinear set

$\{(n, n', n, 0) \mid n, n' \in \mathbb{N}\}$ means that we check, on reading $u\#v$, that $|u| = |v|$ and, in this case, that $u = v^{\mathrm{R}}$, hence $L(A, U, C) = \mathrm{PAL}$. $\square$

Now recall that a semi-AFL is a family of languages closed under nonerasing morphisms, inverse morphisms, intersection with a regular language, and union. Define $\mathcal{M}_\cap(L)$ as the smallest semi-AFL containing L and closed under intersection. The closure properties of $\mathcal{M}_\cap(\mathrm{PAL})$ are implied by those of $\mathcal{L}_{\mathrm{APA}}$ (Prop. 4.8), hence:

Proposition 4.11. *$\mathcal{M}_\cap(\mathrm{PAL}) \subseteq \mathcal{L}_{\mathrm{APA}}$.*

We do not know whether $\mathcal{M}_\cap(\mathrm{PAL}) \subseteq \mathcal{L}_{\mathrm{DetAPA}}$ essentially since we do not know whether $\mathcal{L}_{\mathrm{DetAPA}}$ is closed under nonerasing morphisms, though we conjecture it is not.

The class $\mathcal{M}_\cap(\mathrm{PAL})$ contains a wide range of languages. First, the closure of PAL under nonerasing morphisms, inverse morphisms, and intersection with regular sets is the class of *linear languages* (*e.g.*, [4][5]). In turn, adding closure under intersection permits to express the languages of nondeterministic multipushdown automata where in every computation, each pushdown store makes a bounded number of reversals (that is, going from pushing to popping) [3]; in particular, if there is only one such pushdown store, this corresponds to the *ultralinear languages* [10]. Further, as $\mathcal{M}_\cap(\mathrm{COPY}) \subseteq \mathcal{M}_\cap(\mathrm{PAL})$ (*e.g.*, [4]) this implies that $\mathrm{COPY} \in \mathcal{L}_{\mathrm{APA}}$.

Next, we note that APA express only context-sensitive languages (CSL):

Proposition 4.12. *$\mathcal{L}_{\mathrm{APA}} \subseteq \mathrm{CSL}$.*

Proof. Let (A, U, C) be an $\mathbb{N}$-APA of dimension d, we show that $L(A, U, C) \in \mathrm{NSPACE}[n]$ (which is equal to CSL [18]). Let $A = (Q, \Sigma, \delta, q_0, F)$, and $w = w_1 \dots w_n \in \Sigma^*$. First, initialize $\overline{v} \leftarrow \overline{0}$ and $q \leftarrow q_0$. Iterate through the letters w_i of w: on the i-th letter, choose nondeterministically a transition $t = (q, w_i, q') \in \delta$. Update $\overline{v}$ by setting $\overline{v} \leftarrow U_t(\overline{v})$ and q with $q \leftarrow q'$. Upon reaching the last letter of w, accept w iff $q \in F$ and $\overline{v} \in C$.

We now bound the value of $\overline{v}$. Let c be the greatest value appearing in any of the matrices or vectors in U_t, for any t. For a given $\overline{v}$, let $\max \overline{v}$ be $\max\{v_1, \dots, v_d\}$. Then for any t, $(U_t(\overline{v}))_i \leq d \times (c \times \max \overline{v}) + c$. Let π be a path, we then have that $(U_\pi(\overline{0}))_i \leq (c(d+1))^{n-1}c$, thus the size of $\overline{v}$ at the end of the algorithm is in $O(n)$. Now note that, as C is semilinear, the language of the binary encoding of its elements is regular [26], and thus, checking $\overline{v} \in C$ can be done in, say, logarithmic space. Hence the given algorithm is indeed in $\mathrm{NSPACE}[n]$. $\square$

We now show that $\mathcal{L}_{\mathrm{DetAPA}}$ is not closed under morphisms and we deduce new undecidability results. We rely on the following technical lemma that illustrates the subtle way in which a DetAPA can "perform the conjunction of an

[5]Brandenburg [4] defines PAL as $\{w\overline{w}^{\mathrm{R}} \mid w \in \{a, b\}^*\}$, where $\overline{w}$ is the image of w by the morphism $a \mapsto \overline{a}$ and $b \mapsto \overline{b}$, for $\overline{a}, \overline{b}$ two fresh symbols. We note that the results of [4] carry over with our definition of PAL.

unbounded number of conditions by maintaining a nonzero flag". Let SPACING be the language $\{(a^m\#a^m\#)^n \mid m, n \geq 0\}$; note, for instance, that $a\#a\#a\#a\#$ is in SPACING while $a\#a\#aa\#aa\#$ is not.

Lemma 4.13. SPACING $\in \mathcal{L}_{\mathrm{DetAPA}}$.

Proof. Let $\Sigma = \{a, \#\}$, $L_0 = \{a^m\#a^m\# \mid m \geq 0\}^*$ and $L_1 = L_0 \cdot a^*\#a^*$. Then:

$$\begin{aligned}\text{SPACING} &= \{\varepsilon\} \cup [L_0 \cap a^*\# \cdot L_0 \cdot a^*\#] \\ &= \{\varepsilon\} \cup [L_0 \cap a^*\# \cdot (L_1 \cap \Sigma^*\#)].\end{aligned}$$

We will show that $L_0, L_1 \in \mathcal{L}_{\mathrm{DetAPA}}$. This implies the result as follows. Since $\mathcal{L}_{\mathrm{DetAPA}}$ is closed under intersection (Fig. 1), $L_1 \cap \Sigma^*\# \in \mathcal{L}_{\mathrm{DetAPA}}$. By closure of $\mathcal{L}_{\mathrm{DetAPA}}$ under pointed concatenation (Prop. 4.8), $a^*\# \cdot (L_1 \cap \Sigma^*\#) \in \mathcal{L}_{\mathrm{DetAPA}}$. Applying closure properties again yields SPACING $\in \mathcal{L}_{\mathrm{DetAPA}}$. (Note that L_1 is needed to express SPACING because $L_0 \in \mathcal{L}_{\mathrm{DetAPA}}$ is not known to imply $L_0 \cdot a^*\# \in \mathcal{L}_{\mathrm{DetAPA}}$).

We first construct a $\mathbb{Q}$-DetAPA D_0 on two registers x and y for L_0. As its underlying automaton, D_0 will have a two-state automaton A with initial and final state q_0. The 4 transitions of A, and 4 affine functions $\mathbb{Q}^2 \to \mathbb{Q}^2$ assigned to these transitions, are:

$$\begin{aligned} t_1 &= (q_0, a, q_0) \text{ performs } \tbinom{x}{y} \mapsto \tbinom{x+1}{2y}, \\ t_2 &= (q_0, \#, q_1) \text{ performs } \tbinom{x}{y} \mapsto \tbinom{x}{y}, \\ t_3 &= (q_1, a, q_1) \text{ performs } \tbinom{x}{y} \mapsto \tbinom{x-1}{2y}, \\ t_4 &= (q_1, \#, q_0) \text{ performs } \tbinom{x}{y} \mapsto \tbinom{0}{x+y}. \end{aligned}$$

As usual, $\binom{x}{y}$ is $\binom{0}{0}$ initially. The constraint set C_0 for D_0 will be $\{\binom{0}{0}\}$ which is $\mathbb{Q}$-definable. (Only integers will ever appear in the counters; we use $\mathbb{Q}$ rather than $\mathbb{N}$ only to have access to negative integers). Surprisingly, this works.

We must argue that $L(D_0) = L_0$. We will write $(q, \binom{i}{j})$ for the configuration of D_0 in which the state of A is $q \in \{q_0, q_1\}$ and i and j are the contents of registers x and y. For $w \in \Sigma^*$, we will write $(q, \binom{i}{j})_w$ for the configuration reached when A starts in configuration $(q, \binom{i}{j})$ and reads w. We need to prove two facts:

(i) $\forall w \in L_0,\ (q_0, \binom{0}{0})_w = (q_0, \binom{0}{0})$;
(ii) $\forall w \in (a^*\#a^*\#)^*a^*,\ $ if $(q_0, \binom{0}{0})_w = (q_0, \binom{0}{0})$ then $w \in L_0$.

Fact (i) proves $L_0 \subseteq L(D_0)$ because q_0 is final in A and $\binom{0}{0} \in C_0$. Fact (ii) proves $L(D_0) \subseteq L_0$ because $L(A)$ is seen to be $(a^*\#a^*\#)^*a^*$; hence fact (ii) states that any word that is in $L(A)$ and that further sets $\binom{x}{y}$ to $\binom{0}{0}$ belongs to L_0.

To prove fact (i), let $w = a^{m_1}\#a^{m_1}\#a^{m_2}\#a^{m_2}\# \ldots a^{m_k}\#a^{m_k}\#$ for $k \geq 0$. Any $w \in L_0$ has this form, and an induction on k shows that $(q_0, \binom{0}{0})_w = (q_0, \binom{0}{0})$.

To prove fact (ii), we make the following claim, crucial to the operation of D_0: *Claim:* for any $u, v \in \Sigma^*$, if $(q_0, \binom{0}{0})_u$ sets $y \neq 0$ then $(q_0, \binom{0}{0})_{uv}$ also sets $y \neq 0$.

The claim implies fact (ii) as follows. Let $w \in (a^*\#a^*\#)^*a^i$ satisfy $(q_0, \binom{0}{0})_w = (q_0, \binom{0}{0})$. We must conclude $w \in L_0$. Let $w = a^{m_1}\#a^{m_2}\#\ldots a^{m_{2k-1}}\#a^{m_{2k}}\#a^i$ for some $k \geq 0$. By the Claim, every prefix of w sets $y = 0$. Because reading every second $\#$ returns A to q_0 and resets x to 0, we necessarily have

$$\begin{aligned}
(q_0, \tbinom{0}{0}) &= (q_0, \tbinom{0}{0})_{a^{m_1}\#a^{m_2}\#} \\
&= (q_0, \tbinom{0}{0})_{a^{m_1}\#a^{m_2}\#a^{m_3}\#a^{m_4}\#} \\
&= (q_0, \tbinom{0}{0})_{a^{m_3}\#a^{m_4}\#} \\
&\;\;\vdots \\
&= (q_0, \tbinom{0}{0})_{a^{m_{2k-1}}\#a^{m_{2k}}\#},
\end{aligned}$$

and

$$(q_0, \tbinom{0}{0})_w = (q_0, \tbinom{0}{0})_{a^i} = (q_0, \tbinom{i}{0}).$$

By inspection of A, $(q_0, \binom{0}{0}) = (q_0, \binom{0}{0})_{a^m\#a^{m'}\#}$ implies $m = m'$. Hence $m_1 = m_2, m_3 = m_4, \ldots, m_{2k-1} = m_{2k}$. Moreover, $(q_0, \binom{0}{0})_w = (q_0, \binom{0}{0})$ by assumption. Hence $i = 0$ and $w = a^{m_1}\#a^{m_1}\#\ldots a^{m_{2k-1}}\#a^{m_{2k-1}}\# \in L_0$, concluding fact (ii).

We now prove the Claim. Let $u \in \Sigma^*$ be such that $(q_0, \binom{0}{0})_u$ sets $y \neq 0$. We need to show that for all $v \in \Sigma^*$, $(q_0, \binom{0}{0})_{uv}$ also sets $y \neq 0$. Let $u = u_1u_2$ where u_1 is the shortest prefix of u that sets $y \neq 0$. By inspection of A, $u_1 = u'a^i\#a^j\#$ for some $u' \in (a^*\#a^*\#)^*$ such that $(q_0, \binom{0}{0})_{u_1} = (q_0, \binom{0}{0})_{a^i\#a^j\#} = (q_0, \binom{0}{i-j})$ and $i \neq j$. We will prove by induction on the length of w that for any $w \in \Sigma^*$, $(q_0, \binom{0}{i-j})_w = (q, \binom{x_w}{y_w})$ for some $q \in \{q_0, q_1\}$ and some $x_w, y_w \in \mathbb{Q}$ such that $|y_w| \geq \max\{1, 2|x_w|\}$. This will complete the proof of the Claim since we can pick $w = u_2v$, and conclude that $(q_0, \binom{0}{0})_{uv} = (q_0, \binom{0}{i-j})_{u_2w} = (q, \binom{x_{u_2v}}{y_{u_2v}})$ with $|y_{u_2v}| \geq \max\{1, 2|x_{u_2v}|\} > 0$.

For the basis of the induction, let $w = \varepsilon$. Then $(q_0, \binom{0}{i-j})_w = (q_0, \binom{0}{i-j})$. Now $|i-j| \geq 1 = \max\{1, 2 \times |0|\}$. For the inductive step, let $w \in \Sigma^n$ for some $n > 0$. Then $w = va$ or $w = v\#$ and by induction, $(q_0, \binom{0}{i-j})_v = (q, \binom{x_v}{y_v})$ with $|y_v| \geq \max\{1, 2|x_v|\}$.

Case 1: $w = va$. Then $(q, \binom{0}{i-j})_{va} = (q, \binom{x_v \pm 1}{2y_v})$. If $x_v = 0$, then $|y_{va}| = |2y_v| \geq 2\max\{1, 2|x_v|\} = 2 = \max\{1, 2 \times |\pm 1|\} = \max\{1, 2|x_{va}|\}$. If $x_v \neq 0$, then $|y_{va}| = |2y_v| \geq 2\max\{1, 2|x_v|\} = 2(|x_v| + |x_v|) \geq 2(|x_v| + 1) \geq \max\{1, 2|x_v \pm 1|\} = \max\{1, 2|x_{va}|\}$.

Case 2: $w = v\#$. If t_2 is the transition that consumed the last $\#$, then $x_v = x_{v\#}$ and $y_v = y_{v\#}$ so the induction hypothesis immediately yields $|y_w| \geq \max\{1, 2|x_w|\}$. So let t_4 be the transition that consumed the last $\#$.

Then $(q, \binom{0}{i-j})_{v\#} = (q, \binom{0}{x_v+y_v})$. Now $|y_{va}| = |x_v + y_v| \geq |y_v| - |x_v| \geq \max\{1, 2|x_v|\} - |x_v| \geq \max\{1, |x_v|\} \geq 1 = \max\{1, 2 \times |0|\} = \max\{1, 2 \times |x_{v\#}|\}$. This concludes the proof of the Claim and the proof that $L(D_0) = L_0$.

We have yet to construct a $\mathbb{Q}$-DetAPA D_1 for L_1. The automaton underlying D_1 will be A, as above, except that the final state will be q_1 rather than q_0. The affine functions associated with the transitions remain the same. The constraint set C_1 will be $\{\binom{r}{0} : r \in \mathbb{Q}\}$ and it is $\mathbb{Q}$-definable. We need to prove the following facts:

(iii) $\forall w \in L_1,\ (q_0, \binom{0}{0})_w = (q_1, \binom{i}{0})$ for some $i \in \mathbb{Q}$;
(iv) $\forall w \in (a^*\#a^*\#)^*a^*\#a^*,\ \ \text{if } (q_0, \binom{0}{0})_w = (q_1, \binom{i}{0})$ then $w \in L_1$.

Fact (iii) follows from fact (i) since any $w \in L_1$ is of the form $w = ua^i\#a^j$ with $u \in L_0$, so that $(q_0, \binom{0}{0})_w = (q_0, \binom{0}{0})_{a^i\#a^j} = (q_1, \binom{i-j}{0})$. To prove fact (iv), let $w \in (a^*\#a^*\#)^*a^*\#a^*$ satisfy $(q_0, \binom{0}{0})_w = (q_1, \binom{x_w}{0})$. By the Claim above, every prefix of w sets $y = 0$. By inspection of A, some suffix $a^i\#a^j$ of w must have sent A to state q_1, that is, $w = ua^i\#a^j$ with $(q_0, \binom{0}{0})_u = (q_0, \binom{0}{0})$ and $x_w = i - j$. But then, $u \in L_0$ by fact (ii) and thus $w \in L_1$. This concludes the proof that $L(D_1) = L_1$ and proves the lemma. □

Lemma 4.14. *Given a Turing machine M, we can construct a morphism h and a* DetAPA *D such that $L(M) = h(L(D))$.*

Proof. We adapt [1], Theorem 1. With no loss of generality, we assume that M is a one-tape Turing machine that accepts by halting and makes an odd number of moves on any accepting computation. Let L_1 (resp. L_2) be the set of strings

$$ID_0\#ID_2\#\ldots\#ID_{2k}\$(ID_{2k+1})^{\mathrm{R}}\#\ldots\#(ID_3)^{\mathrm{R}}\#(ID_1)^{\mathrm{R}}\# \tag{4.1}$$

such that ID_i, $0 \leq i \leq 2k+1$, are instantaneous descriptions of M padded with the blank symbol b to a common length ℓ, $ID_0 = \left[\begin{smallmatrix}w_1\\q_0\end{smallmatrix}\right]w_2\ldots w_nb^{\ell-n}$ codes the initial configuration of M ($\left[\begin{smallmatrix}w_1\\q_0\end{smallmatrix}\right]$ is considered as a single letter, and, $w_1 = b$ when the word $w = w_1w_2\ldots w_n \in \Sigma^*$ input to M is ε), ID_{2k+1} codes an accepting configuration and for $0 \leq i \leq k$, ID_{2i+1} (resp. for $0 < i \leq k$, ID_{2i}) codes the configuration which would be reached in one step from configuration ID_{2i} (resp. ID_{2i-1}). Each ID_i other than ID_0 is coded using an alphabet Γ disjoint from $\Sigma \cup \{\left[\begin{smallmatrix}\sigma\\q_0\end{smallmatrix}\right] \mid \sigma \in \Sigma\} \cup \{\left[\begin{smallmatrix}b\\q_0\end{smallmatrix}\right]\}$. It should be clear that $w \in L(M)$ iff $w \in h(L_1 \cap L_2)$ where for every $\sigma \in \Sigma$ and every $\gamma \in \Gamma$,

$$h(\left[\begin{smallmatrix}\sigma\\q_0\end{smallmatrix}\right]) = h(\sigma) = \sigma \text{ and } h(\left[\begin{smallmatrix}b\\q_0\end{smallmatrix}\right]) = h(b) = h(\#) = h(\$) = h(\gamma) = \varepsilon.$$

To complete the proof, we claim that $L_1 \cap L_2 \in \mathcal{L}_{\mathrm{DetAPA}}$ in the effective sense. Since $\mathcal{L}_{\mathrm{DetAPA}}$ is closed under intersection in that sense (Fig. 1), it suffices to show that $L_1 \in \mathcal{L}_{\mathrm{DetAPA}}$ and $L_2 \in \mathcal{L}_{\mathrm{DetAPA}}$. We first show how to construct a DetAPA recognizing L_1. We will construct an $\mathbb{N}$-DetAPA D_1 able to handle only the words of the form (4.1) in which the distance between any two consecutive symbols # or \$ is $|ID_0|$. Handling only those words will be sufficient because the language L_1 can then be expressed as $L_1 = g^{-1}(\text{SPACING}) \cap L(D_1)$ where g is the morphism mapping both # and \$ to # and mapping every other symbol to the letter a. Since Lemma 4.13 shows SPACING $\in \mathcal{L}_{\text{DetAPA}}$ and since $\mathcal{L}_{\text{DetAPA}}$ is closed under intersection and inverse morphisms in the effective sense (Fig. 1), a DetAPA for L_1 can be constructed from D_1.

So we now describe D_1. Let m be the size of the alphabet Γ. We argue as if ID_0 in (4.1) were coded over the same alphabet Γ used to code ID_i for $0 < i \leq 2k+1$, since a finite automaton can easily adjust for this. Our strategy will extend the strategy used to construct an $\mathbb{N}$-DetAPA for pointed palindromes (Prop. 4.10) as follows. As D_1 reads the prefix $ID_0\#ID_2\#\ldots\#ID_{2k}$ of (4.1), D_1 will internally translate that prefix into $ID_1ID_3\ldots ID_{2k+1}$ and will treat the latter as the prefix u of a pointed palindrome $u\$u^{\mathrm{R}}$. As D_1 processes u, D_1 builds in a register the natural number having u as its m-ary representation with the most significant bit first (as in Prop. 4.10, where m was 2). Then D_1 encounters \$ and begins to do the matchup with the suffix $(ID_{2k+1})^{\mathrm{R}}\#\ldots\#(ID_3)^{\mathrm{R}}\#(ID_1)^{\mathrm{R}}\#$ of (4.1). D_1 does this matchup by internally translating this suffix into $(ID_{2k+1})^{\mathrm{R}}\ldots(ID_3)^{\mathrm{R}}(ID_1)^{\mathrm{R}} = (ID_1ID_3\ldots ID_{2k+1})^{\mathrm{R}} = u^{\mathrm{R}}$. As D_1 processes this suffix, D_1 computes in a register the natural number having the suffix as its m-ary representation, with the least significant bit first this time (again as in Prop. 4.10, now with m rather than 2). We set D_1 to accept iff reading (4.1) indeed leads to $u\$u^{\mathrm{R}}$ with ID_{2k+1} final. Two subtleties are worth mentioning concerning processing the prefix. First, when processing ID_{2i}, D_1 always reads one symbol ahead of position p to determine the proper symbol at position p in ID_{2i+1}, to account for the input head of M possibly moving left from position $p+1$ to position p. Second, D_1 rejects immediately if ID_0 is not a legal coding of an initial configuration of M or if another ID_i in the prefix contains two input head symbols. This completes the operational description of D_1. The formal definition (A_1, U_1, C_1) of D_1 thus needs to implement these operations. The $\mathbb{N}$-definable set C_1 is the set $C \subseteq \mathbb{N}^4$ given in the Proof of Proposition 4.10 but adapted to handle m-ary representation rather than binary. The affine functions assigned to the transitions of A_1 are the identity function together with the five (adapted) functions assigned to the transitions t_1, t_2, t_3, t_4, t_5 from the Proof of Proposition 4.10: transitions performing the bookkeeping operations of A_1 (such as when A_1 processes the symbol #) will be assigned the identity function, and transitions that discover the next PAL symbol are assigned the affine transformation prescribed by Proposition 4.10 on reading that symbol (with the understanding that reading \$ here corresponds to reading # there).

The strategy to construct an $\mathbb{N}$-DetAPA D_2 recognizing L_2 is of course similar. But now, since ID_2 (for example) does not uniquely determine ID_1, the prefix of (4.1) is handled by D_2 as the suffix of (4.1) was handled by D_1. Specifically, D_2 internally translates the prefix $ID_0\#ID_2\#\ldots\#ID_{2k}$ into $u = ID_2ID_4\ldots ID_{2k}$ and stores the m-ary number u in a register, with the most significant bit first. Then D_2 encounters \$, discards $(ID_{2k+1})^{\mathrm{R}}$ and internally translates the remainder $(ID_{2k-1})^{\mathrm{R}}\#\ldots\#(ID_3)^{\mathrm{R}}\#(ID_1)^{\mathrm{R}}\#$ of the suffix into

$(ID_{2k})^{\mathrm{R}} \dots (ID_4)^{\mathrm{R}}(ID_2)^{\mathrm{R}} = (ID_2 ID_4 \dots ID_{2k})^{\mathrm{R}} = u^{\mathrm{R}}$. As did D_1 when reading the prefix, D_2 needs to look ahead by one symbol while processing the suffix. The matchup with u^{R} is otherwise done by D_2 just as the matchup was done by D_1. This completes the description of D_2 and proves the lemma. $\square$

Corollary 4.15. *Neither* $\mathcal{L}_{\mathrm{APA}}$ *nor* $\mathcal{L}_{\mathrm{DetAPA}}$ *is closed under morphisms.*

Proof. Given a Turing machine M, we can construct a morphism h and a DetAPA (and a fortiori an APA) D such that $L(M) = h(L(D))$. If either $\mathcal{L}_{\mathrm{APA}}$ or $\mathcal{L}_{\mathrm{DetAPA}}$ were closed under morphisms, then the language $h(L(D))$ would be the language of an APA. But the language of any APA is context-sensitive (Lem. 4.12), thus decidable, so we could decide $L(M)$. $\square$

Corollary 4.16. *The emptiness, universality, inclusion, finiteness, and regularity problems are undecidable for* DetAPA.

Proof. (Emptiness, universality, and inclusion). Given a Turing machine M with $L(M) \subseteq \Sigma^*$, let h be the morphism and D the DetAPA provided by Lemma 4.14. For any $x \in \Sigma^*$, $x \in L(M)$ iff $x = h(y)$ for some $y \in L(D)$ iff $L(D) \cap h^{-1}(x)$ is nonempty. Now $\{x\} \in \mathcal{L}_{\mathrm{DetAPA}}$ and $\mathcal{L}_{\mathrm{DetAPA}}$ is closed (in the effective sense) under inverse morphisms and intersection (Fig. 1). Hence we can construct a DetAPA for $L(D) \cap h^{-1}(x)$ and deciding its emptiness would decide $x \in L(M)$. Moreover, $\mathbb{K}$-DetAPA being closed under complement (in the effective sense), the emptiness problem reduces to the universality problem. Finally, the undecidability of emptiness implies that we cannot decide if the language of a $\mathbb{K}$-DetAPA is included in the empty set.

(Finiteness and regularity [pointed out by Andreas Krebs]). Let $L \subseteq \Sigma^*$ be a language of $\mathcal{L}_{\mathrm{DetAPA}}$, and let $\# \notin \Sigma$. Then $L \cdot \{\#a^n b^n \mid n \in \mathbb{N}\}$ is in $\mathcal{L}_{\mathrm{DetAPA}}$ and effectively constructible from the given DetAPA, as a pointed concatenation (Prop. 4.8). Its language is finite iff it is regular iff L is empty. $\square$

Remark 4.17. None of these results allow us to conclude that $\mathcal{L}_{\mathrm{DetAPA}}$ and $\mathcal{L}_{\mathrm{APA}}$ are different, though we conjecture they are. One argument supporting this conjecture is the fact that DetAPA do not need their automaton: we can encode the transition function of an automaton *within* the affine functions, showing that any language of $\mathcal{L}_{\mathrm{DetAPA}}$ can be expressed using a two-state DetAPA.

5. Parikh automata on letters

The PA *on letters* requires that the "weight" of a transition depends only on the input letter from Σ triggering the transition. In a way similar to the CA characterization of PA, we characterize PA on letters solely in terms of automata over Σ and semilinear sets. We further give expressiveness and closure properties of the classes of languages that arise.

Definition 5.1 (Parikh automaton on letters)**.** A *Parikh automaton on letters* (LPA) is a PA (A, C) where whenever $(a, \overline{v_1})$ and $(a, \overline{v_2})$ are labels of some transitions in A, then $\overline{v_1} = \overline{v_2}$. We write $\mathcal{L}_{\text{LPA}}$ (resp. $\mathcal{L}_{\text{DetLPA}}$) for the class of languages recognized by LPA (resp. LPA which are DetPA).

First, we prove that $\mathcal{L}_{\text{LPA}}$ and $\mathcal{L}_{\text{DetLPA}}$ coincide:

Theorem 5.2. $\mathcal{L}_{\text{LPA}} = \mathcal{L}_{\text{DetLPA}}$.

Proof. Let (A, C) be a LPA. Without loss of generality, we can consider $A = (Q, \Sigma \times D, \delta, q_0, F)$ to be deterministic (this does not imply that the PA is deterministic). Now let $t, t' \in \delta$ with $t = (p, (a, \overline{v_1}), q)$ and $t' = (p, (a, \overline{v_2}), q')$. The fact that (A, C) is a LPA implies that $\overline{v_1} = \overline{v_2}$, and A being deterministic, this implies that $q = q'$, and in turn that $t = t'$. Thus (A, C) is a DetPA. $\square$

For $L \subseteq \Sigma^*$ and $C \subseteq \mathbb{N}^{|\Sigma|}$, recall that $L{\restriction_C} = \{w \in L \mid \Phi(w) \in C\}$. Then:

Proposition 5.3. *Let $L \subseteq \Sigma^*$ be a language. The following are equivalent:*

(i) $L \in \mathcal{L}_{\text{LPA}}$*;*
(ii) there exist a regular language $R \subseteq \Sigma^*$ and a semilinear set $C \subseteq \mathbb{N}^{|\Sigma|}$ such that $R{\restriction_C} = L$.

Proof.
(i) $\rightarrow$ (ii) Let (A, C) be a LPA which is a DetPA over the alphabet $\{a_1, \ldots, a_n\}$. For $1 \leq i \leq n$, let $\overline{v_i}$ be the only vector appearing as the label $(a_i, \overline{v_i})$ of a transition in A. Define $C' \subseteq \mathbb{N}^n$ by $(x_1, \ldots, x_n) \in C' \Leftrightarrow \sum_i x_i \times \overline{v_i} \in C$. Then let $w \in \Sigma^*$ and ω be the word which can be read from the initial state of A with $\Psi(\omega) = w$. We have that $\sum_i \Phi(w)_i \times \overline{v_i} = \tilde{\Phi}(\omega)$, and thus $w \in L(A, C)$ iff $w \in \Psi(L(A)){\restriction_{C'}}$.
(ii) $\rightarrow$ (i) Let $R \subseteq \{a_1, \ldots, a_n\}^*$ be a regular language and $C \subseteq \mathbb{N}^n$ be a semilinear set. Let A be an automaton for R, and change each transition label a_i in A by $(a_i, \overline{e_i})$. Now for $\omega \in L(A)$, $\tilde{\Phi}(\omega) = \Phi(\Psi(\omega))$ and thus (A, C) is a LPA with language $R{\restriction_C}$. $\square$

The following property will be our central tool for showing nonclosure results:

Lemma 5.4. *Let $L \in \mathcal{L}_{\text{LPA}}$. For any regular language E:*

$$L \cap E \text{ is not regular} \quad \Rightarrow \quad (\exists w \in E)[c(w) \cap L = \emptyset].$$

Proof. Let $R \subseteq \Sigma^*$ be a regular language and $C \subseteq \mathbb{N}^{|\Sigma|}$ be a semilinear set. Define $L = R{\restriction_C}$. Let E be a regular language such that $L \cap E$ is not regular. As $L \subseteq R$, we have $(L \cap E) \subseteq (R \cap E)$. The left hand side being non regular, those two sets differ. Thus, let $w \in (R \cap E)$ such that $w \notin L \cap E$, we have $w \notin L$. Hence, $w \in (R \setminus L)$, which implies that $\Phi(w) \notin C$, and in turn, $c(w) \cap L = \emptyset$. $\square$

Proposition 5.5. *(1) $\mathcal{L}_{\text{LPA}}$ is not closed under union, complement, concatenation, nonerasing morphisms, and starring; (2) $\mathcal{L}_{\text{LPA}}$ is closed under intersection, commutative closure, and inverse morphisms.*

Proof.

(1) Let $L = \{a^m b^n \mid m, n \in \mathbb{N} \wedge m \neq n\}$ be a language of LPA. (Union). Suppose $L' = L \cup \overline{a^*b^*} \in \mathcal{L}_{\text{LPA}}$. Let E be the regular language (a^+b^+). By the pumping lemma, $L' \cap E$ is not regular, thus Lemma 5.4 states there exists $w \in E$ such that $c(w) \cap L' = \emptyset$. But $u = b^{|w|_b} a^{|w|_a} \in c(w)$ and $u \in L'$, a contradiction, thus $L' \notin \mathcal{L}_{\text{LPA}}$. (Complement). We note that L' is the complement in $\{a, b\}^*$ of $\{a^n b^n \mid n \in \mathbb{N}\}$, which is the language of an LPA. (Concatenation). Suppose $L^2 \in \mathcal{L}_{\text{LPA}}$. Again, as $L^2 \cap E^2$ is not regular, Lemma 5.4 asserts that there exists $w \in E^2$ such that $c(w) \cap L^2 = \emptyset$. But $a^{|w|_a} b^0 a^0 b^{|w|_b} \in c(w) \cap L^2$, a contradiction, thus $L^2 \notin \mathcal{L}_{\text{LPA}}$. (Nonerasing morphism). We note that L^2 is the image of the LPA language $\{a_1^m b_1^n a_2^r b_2^s \mid m \neq n \wedge r \neq s\}$ by the nonerasing morphism $h(a_i) = a, h(b_i) = b$, $i \in \{1, 2\}$. Also, by the very definition of constrained automata (Def. 3.3), each language of $\mathcal{L}_{\text{PA}}$ is the image by a nonerasing morphism of a language of $\mathcal{L}_{\text{LPA}}$, but the two classes are different. (Starring). The proof of the nonclosure under starring of $\mathcal{L}_{\text{PA}}$ (Prop. 3.17) shows that the starring of $\{a^n b^n \mid n \in \mathbb{N}\}$ is not in $\mathcal{L}_{\text{PA}}$, thus not in $\mathcal{L}_{\text{LPA}}$.

(2) Let $R, R' \subseteq \Sigma^*$ be two regular languages and let $C, C' \subseteq \mathbb{N}^{|\Sigma|}$ be two semilinear sets. (Intersection). Note that $(R \restriction_C) \cap (R' \restriction_{C'}) = (R \cap R') \restriction_{C \cap C'}$, the latter being a language of $\mathcal{L}_{\text{LPA}}$. (Commutative closure). Likewise, note that $c(R \restriction_C) = \Sigma^* \restriction_{C \cap \Phi(R)}$, which is in $\mathcal{L}_{\text{LPA}}$ since $\Phi(R)$ is effectively semilinear by Parikh's theorem. (Inverse morphism). Let $h\colon \{a_1, \ldots, a_n\}^* \to \Sigma^*$ be a morphism, and let $C^h = \{\overline{x} \in \mathbb{N}^n \mid \sum_i x_i \times \Phi(h(a_i)) \in C\}$. Then we claim that $h^{-1}(R \restriction_C) = (h^{-1}(R)) \restriction_{C^h}$, which concludes the proof as $h^{-1}(R)$ is regular and C^h is semilinear. Indeed, let $w \in h^{-1}(R \restriction_C)$, then $w \in h^{-1}(R)$ and $\Phi(h(w)) \in C$, the latter implying that $\sum_i |w|_{a_i} \times \Phi(h(a_i)) \in C$, and thus, $\Phi(w) \in C^h$; in particular, if a letter a is such that $h(a) = \varepsilon$, it is discarded when looking at the Parikh image of $h(w)$. Conversely, if $w \in h^{-1}(R) \restriction_{C^h}$ then $h(w) \in R$ and $\Phi(h(w)) = \sum_i |w|_{a_i} \times \Phi(h(a_i)) \in C$, thus $h(w) \in R \restriction_C$, implying that $w \in h^{-1}(R \restriction_C)$. □

6. Conclusion

Figures 1 and 2 in our introductory section summarize the current state of knowledge concerning the PA and its variants studied here.

An intriguing question is whether there are context-free or context-sensitive languages outside $\mathcal{L}_{\text{APA}}$. How difficult is that question? How about $\mathcal{L}_{\text{DetAPA}}$? We have been unable to locate the latter class meaningfully. In particular, can $\mathcal{L}_{\text{DetAPA}}$ be separated from $\mathcal{L}_{\text{APA}}$?

Several questions thus remain open concerning the poorly understood (and possibly overly powerful) APA model. But surely we expect testing a LPA or a DetPA for regularity to be decidable. How can regularity be tested for these models? One avenue for future research towards this goal might be characterizing $\mathcal{L}_{\mathrm{DetPA}}$ along the lines of algebraic automata theory.

Acknowledgements. The first author thanks L. Beaudou, M. Kaplan, and A. Lemaître for stimulating discussions and comments on early versions of this paper. We further thank the anonymous referees for their critical comments and especially for pointing out a flawed claim made in an earlier version of this work concerning DetAPA.

References

[1] B.S. Baker and R.V. Book, Reversal-bounded multipushdown machines. *J. Comput. Syst. Sci.* **8** (1974) 315–332.
[2] M. Blattner and M. Latteux, Parikh-bounded languages, in ICALP. *Lect. Notes Comput. Sci.* **115** (1981) 316–323.
[3] R. Book, M. Nivat and M. Paterson, Reversal-bounded acceptors and intersections of linear languages. *SIAM J. Comput.* **3** (1974) 283.
[4] F. Brandenburg, Analogies of PAL and COPY, in Fundamentals of Computation Theory. *Lect. Notes in Comput. Sci.* **117** (1981) 61–70.
[5] E. Chiniforooshan, M. Daley, O.H. Ibarra, L. Kari and S. Seki, One-reversal counter machines and multihead automata: revisited, in Proc. of *SOFSEM*. ACM (2011) 166–177.
[6] H.B. Enderton, *A Mathematical Introduction to Logic*. Academic Press (1972).
[7] J. Ferrante and C. Rackoff, A decision procedure for the first order theory of real addition with order. *SIAM J. Comput.* **4** (1975) 69–76.
[8] P. Ganty, R. Majumdar and B. Monmege, Bounded underapproximations. *Form. Methods Syst. Des.* **40** (2012) 206–231.
[9] S. Ginsburg and E.H. Spanier, Semigroups, Presburger formulas and languages. *Pacific J. Math.* **16** (1966) 285–296.
[10] S. Ginsburg and E. Spanier, Finite-turn pushdown automata. *SIAM J. Control Optim.* **4** (1966) 429.
[11] S.A. Greibach, A note on undecidable properties of formal languages. *Math. Syst. Theor.* **2** (1968) 1–6.
[12] O.H. Ibarra, Reversal-bounded multicounter machines and their decision problems. *J. ACM* **25** (1978) 116–133.
[13] O.H. Ibarra and J. Su, A technique for proving decidability of containment and equivalence of linear constraint queries. *J. Comput. Syst. Sci.* **59** (1999) 1–28.
[14] O.H. Ibarra, J. Su, Z. Dang, T. Bultan and R.A. Kemmerer, Counter machines and verification problems. *Theor. Comput. Sci.* **289** (2002) 165–189.
[15] W. Karianto, *Parikh automata with pushdown stack*. Diploma thesis, RWTH Aachen (2004).
[16] F. Klaedtke and H. Rueß, *Parikh automata and monadic second-order logics with linear cardinality constraints*. Universität Freiburg, Tech. Rep. 177 (2002).
[17] F. Klaedtke and H. Rueß, Monadic second-order logics with cardinalities, in Proc. of ICALP. *Lect. Notes Comput. Sci.* **2719** (2003) 681–696.
[18] S.Y. Kuroda, Classes of languages and linear bounded automata. *Inform. Control* **7** (1964) 207–223.
[19] M. Latteux, Mots infinis et langages commutatifs. *RAIRO Inform. Théor. Appl.* **12** (1978) 185–192.
[20] D.R. Mazur, *Combinatorics: A Guided Tour*. Mathematical Association of Mathematics (2010).
[21] P. McKenzie, M. Thomas and H. Vollmer, Extensional uniformity for boolean circuits. *SIAM J. Comput.* **39** (2010) 3186–3206.

[22] H. Seidl, T. Schwentick and A. Muscholl, Numerical document queries, in *Principles of Database Systems*. ACM, San Diego, CA, USA (2003) 155–166.
[23] H. Straubing, *Finite Automata, Formal Logic, and Circuit Complexity*. Birkhäuser, Boston (1994).
[24] P. Tesson and D. Thérien, Logic meets algebra: the case of regular languages. *Log. Meth. Comput. Sci.* **3** (2007) 1–37.
[25] L.P.D. van den Dries, *Tame Topology and O-minimal Structures*. Cambridge Univ. Press (1998).
[26] P. Wolper and B. Boigelot, An automata-theoretic approach to Presburger arithmetic constraints, in Static Analysis (SAS'95). *Lect. Notes Comput. Sci.* **983** (1995) 21–32.
[27] S.D. Zilio and D. Lugiez, Xml schema, tree logic and sheaves automata, in *Rewriting Techniques and Applications* (2003) 246–263.

Communicated by M. Holzer.
Received November 4, 2011. Accepted April 30, 2012.