Regular Separability of Parikh Automata

Lorenzo Clemente[*1], Wojciech Czerwiński[†1], Sławomir Lasota[‡1], and Charles Paperman[2]

1 **University of Warsaw**
{l.clemente, wczerwin, sl}@mimuw.edu.pl
2 **University of Tübingen**
charles.paperman@gmail.com

Abstract

We investigate a subclass of languages recognized by vector addition systems, namely languages of nondeterministic Parikh automata. While the regularity problem (is the language of a given automaton regular?) is undecidable for this model, we surprisingly show decidability of the regular separability problem: given two Parikh automata, is there a regular language that contains one of them and is disjoint from the other? We supplement this result by proving undecidability of the same problem already for languages of visibly one counter automata.

1998 ACM Subject Classification D.2.2 [*Design Tools and Techniques*]: Petri nets; F.1.1 [*Theory of Computation*]: Models of Computation; F.2.2 [*Nonnumerical Algorithms and Problems*]: Computations on discrete structures; F.3.1 [*Specifying and Verifying and Reasoning about Programs*]: Mechanical verification.

Keywords and phrases Regular separability problem, Parikh automata, integer vector addition systems, visible one counter automata, decidability, undecidability.

Digital Object Identifier 10.4230/LIPIcs...

1 Introduction

We investigate separability problems for languages of finite words. We say that a language U is *separated from* a language V *by* S if $U \subseteq S$ and $V \cap S = \emptyset$. In the sequel we also often say that U *and* V are *separated by* S. For two families of languages $\mathcal{F}$ and $\mathcal{G}$, the $\mathcal{F}$ *separability problem for* $\mathcal{G}$ asks for two given languages $U, V \in \mathcal{G}$ whether U is separated from V by some language from $\mathcal{F}$. The same notion of separability makes clearly sense if $\mathcal{F}$ and $\mathcal{G}$ are classes of sets of vectors instead of classes of languages. In this paper, we consider the case where $\mathcal{F}$ are regular languages and $\mathcal{G}$ languages recognized by Parikh automata, and the case where $\mathcal{F}$ are the unary sets and $\mathcal{G}$ the semilinear sets.

Motivation. Separability is a classical problem in theoretical computer science. It was investigated most extensively in the area of formal languages, for $\mathcal{G}$ being the family of all regular word languages. Since regular languages are effectively closed under complement, the $\mathcal{F}$ separability problem is a generalization of the $\mathcal{F}$ characterization problem, which asks whether a given language belongs to $\mathcal{F}$. Indeed, $L \in \mathcal{F}$ if and only if L is separated from its complement by some language from $\mathcal{F}$. Separability problems for regular languages attracted

[*] Partially supported by the Polish National Science Centre grant 2016/21/B/ST6/01505.
[†] Partially supported by the Polish National Science Centre grant 2016/21/D/ST6/01376.
[‡] Partially supported by the Polish National Science Centre grant 2016/21/B/ST6/01505.

recently a lot of attention, which resulted in establishing the decidability of $\mathcal{F}$ separability for various families $\mathcal{F}$ such as the piecewise testable languages [7, 15] (recently generalized to finite ranked trees [9]), the locally and locally threshold testable languages [14], the languages definable in first order logic [17], and the languages of certain higher levels of the first order hierarchy [16], among others.

Separability of nonregular languages attracted little attention till now. The reasons for this may be twofold. First, for regular languages one can use standard algebraic tools, like syntactic monoids, and indeed most of the results have been obtained with the help of such techniques. Second, some strong intractability results have been known already since 70's, when Szymanski and Williams proved that regular separability of context-free languages is undecidable [18]. Later Hunt [10] generalized this result for every class $\mathcal{F}$ closed under finite boolean combinations and containing all languages of the form $w\Sigma^*$ for $w \in \Sigma^*$. This is a very weak condition, so it seemed that nothing nontrivial can be done outside regular languages with respect to separability problems. Furthermore, Szymanski and Williams's negative result has recently been strengthened by considering two incomparable subclasses of pushdown automata. First, Kopczyński has shown that regular separability is undecidable for languages of visibly pushdown automata [13], and then Czerwiński and Lasota have shown that the same problem is undecidable for one counter automata [6].

On the positive side, piecewise testable separability has been shown decidable for context-free languages, languages of vector addition systems (VAS languages), and some other classes of languages [8]. Another surprising result has been recently obtained by Czerwiński and Lasota [6] who show that regular separability is decidable (and PSPACE-complete) for languages recognized by one counter nets (i.e., one counter automata without zero test). Notice that in all these examples regularity (resp. piecewise testability) is undecidable, but regular (resp. piecewise testable) separability *is* decidable, and until recently there were not many results of this kind.

Finally, in [5] we have shown decidability of *unary separability* of reachability sets of vector addition systems (VASes). By *unary sets* we mean Parikh images of commutative regular languages, and thus the latter problem is equivalent to commutative regular separability of (commutative closures of) VAS languages. The decidability status of the regular separability problem for the whole class of VAS languages remains open.

Our contribution. This paper is a continuation of the line of research trying to understand the regular separability problem for language classes beyond regular languages. We report a further progress towards solving the open problem mentioned above by providing a positive decidability result and a new negative undecidability result: As our first (positive) result, we show decidability of the regular separability problem for the subclass of VAS languages where we allow negative counter values during a run. This class of languages is also known as languages of *integer VASSes*, and it admits many different characterizations; for instance, it coincides with languages of *one-way reversal-bounded counter machines* [11], *Parikh automata* [12] (cf. also [2, Proposition 11]), which in turn are equivalent to the very similar model of *constrained automata* [3]. In this paper, we present our results in terms of constrained automata, but given the similarity with Parikh automata (and in light of their equivalence), we overload the name Parikh automata for both models.

Notice that PA languages are not closed under complement, and thus our decidability result about regular separability does not imply decidability of the regularity problem (is the language of a given Parikh automaton regular?). Moreover, the regularity problem for

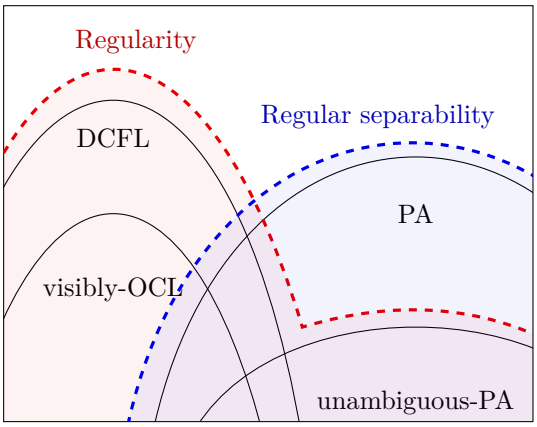

Figure 1 The regularity and the regular separability problems.

PA languages is actually *undecidable* [2][1], which makes our decidability result one of few instances where regularity is undecidable but regular separability is decidable; cf. Fig 1.

Parikh automata are finite nondeterministic automata where accepting runs are further restricted to satisfy a semilinear condition on the multiset of transitions appearing in the run. Our decidability result is actually stated in the more general setting of $\mathcal{C}$-*Parikh automata*, where $\mathcal{C} \subseteq \bigcup_{d\in\mathbb{N}} \mathcal{P}(\mathbb{N}^d)$ is a class of sets of vectors used as an acceptance condition. We prove that the regular separability problem for languages of $\mathcal{C}$-Parikh automata effectively reduces to the *unary* separability problem for the class $\mathcal{C}$ itself, provided that $\mathcal{C}$ is effectively closed under inverse images of affine functions. Two prototypical classes $\mathcal{C}$ satisfying the latter closure condition are semilinear sets and VAS reachability sets. Moreover, unary separability of semilinear sets is known to be decidable [4], and as recalled before the same result has recently been extended to VAS reachability sets [5]. As a consequence of our reduction, we deduce decidability of regular separability of $\mathcal{C}$-Parikh automata languages where the acceptance condition $\mathcal{C}$ can be instantiated to either the semilinear sets, or the VAS reachability sets.

We complement our decidability result by a new negative undecidability result subsuming simultaneously Kopczyński's undecidability for visibly pushdown languages [13] and Czerwiński and Lasota's undecidability of one counter languages [6]: We show that regular separability is undecidable for (deterministic[2]) visibly one counter languages. Inside the proof we use the result from [6], but actually in order to only reprove [13] it would be sufficient to use the old work by Szymanski and Williams [18].

2 Preliminaries

Vectors. A set $S \subseteq \mathbb{N}^d$ is *linear* if there exist a *base* $b \in \mathbb{N}^d$ and *periods* $p_1, \ldots, p_k \in \mathbb{N}^d$ s.t. $S = \{b + n_1p_1 + \ldots + n_kp_k \mid n_1, \ldots, n_k \in \mathbb{N}\}$, and it is *semilinear* if it is a finite union of linear sets. For a vector $v \in \mathbb{N}^d$ and $i \in \{1, \ldots, d\}$, let $v[i]$ denote its i-th coordinate. For $n \in \mathbb{N}$, we say that two vectors $x, y \in \mathbb{N}^d$ are n-*unary equivalent*, written $x \equiv_n y$, if for every coordinate $i \in \{1, \ldots, d\}$ it holds $x[i] \equiv y[i] \mod n$ and moreover $x[i] \leq n \iff y[i] \leq n$. A

[1] Later shown decidable for unambiguous PA [3].
[2] Determinism here is irrelevant because this class can be determinized.

set $S \subseteq \mathbb{N}^d$ is *unary* if for some n, S is a union of equivalence classes of $\equiv_n$. Intuitively, to decide membership in a unary set S it is enough to count on every coordinate exactly up to some threshold n, and modulo n for values larger than n. Unary sets are semilinear.

Let $\Sigma = \{a_1, \ldots, a_k\}$ be a totally ordered alphabet. For a word $w \in \Sigma^*$ and a letter $a_i \in \Sigma$, by $\#_{a_i}(w)$ we denote the number of letters a_i in w. The *Parikh image* of a word $w \in \Sigma^*$ is the vector $\Pi(w) = (\#_{a_1}(w), \ldots, \#_{a_k}(w)) \in \mathbb{N}^k$. The *Parikh image* of a language $L \subseteq \Sigma^*$ is $\Pi(L) = \{\Pi(w) \mid w \in L\}$, the set of Parikh images of all words belonging to L.

Parikh automata. A *nondeterministic finite automaton with ε-transitions* (ε-NFA) $\mathcal{A} = (Q, I, F, T)$ over a finite alphabet Σ consists of a finite set of states Q, two distinguished subsets of initial and final states $I, F \subseteq Q$, and a set of transitions $T \subseteq Q \times \Sigma_\varepsilon \times Q$, where $\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$. A *nondeterministic Parikh automaton*[3] is a pair $(\mathcal{A}, S)$ consisting of an ε-NFA $\mathcal{A}$ and a semilinear set $S \subseteq \mathbb{N}^d$, where $d = |T|$ is the number of transitions of $\mathcal{A}$. Notice that we allow ε-transitions in the definition of Parikh automata. A *run* of a Parikh automaton over a word $w = a_1 \ldots a_n \in \Sigma^*$ is a sequence of transitions $\rho = t_1 \ldots t_n \in T^*$, where $t_i = (q_{i-1}, a_i, q_i)$, starting in an initial state q_0. A run ρ is *accepting* if its ending state q_n is final and $\Pi(\rho) \in S$ (we assume here that the set of transitions T is totally ordered). The language of a Parikh automaton, denoted $L(\mathcal{A}, S)$, contains all words w admitting an accepting run; it is thus a subset of the language $L(\mathcal{A})$ of the underlying ε-NFA.

One can generalize Parikh automata by using some other family of vector sets in the place of semilinear sets. For a class $\mathcal{C} \subseteq \bigcup_{d \in \mathbb{N}} \mathcal{P}(\mathbb{N}^d)$ of vector sets, a *$\mathcal{C}$-Parikh automaton* is a pair $(\mathcal{A}, S)$, where $\mathcal{A}$ is an ε-NFA and $S \in \mathcal{C}$. The language $L(\mathcal{A}, S)$ is defined as above.

A $\mathcal{C}$-Parikh automaton $(\mathcal{A}, S)$ is *deterministic* if the underlying automaton $\mathcal{A}$ is so; here, we assume that a deterministic automaton does not have ε-transitions. The languages of (non)deterministic $\mathcal{C}$-Parikh automata are shortly called (non)deterministic $\mathcal{C}$-Parikh languages below.

3 Main results

We call a function $f : \mathbb{N}^k \to \mathbb{N}^\ell$ *affine* if it is of the form $f(v) = Mv + u$ for an integer non-negative matrix M of dimension $\ell \times k$ and a vector $u \in \mathbb{N}^\ell$. In a special case when $u = 0$ we call the function f *linear*. A class of vector sets $\mathcal{C} \subseteq \bigcup_{d \in \mathbb{N}} \mathcal{P}(\mathbb{N}^d)$ is called *robust* if it fulfills the following two conditions:

- $\mathcal{C}$ is effectively closed under inverse images of affine functions,
- the unary separability problem is decidable for $\mathcal{C}$.

Our first main result is decidability of the regular separability problem for $\mathcal{C}$-Parikh automata.

▶ **Theorem 1.** *The regular separability problem is decidable for $\mathcal{C}$-Parikh automata, for every robust class $\mathcal{C}$ of vector sets.*

The proof of Theorem 1 is split into two parts. In Section 4 we provide a reduction of the regular separability problem for *nondeterministic* $\mathcal{C}$-Parikh automata to the same problem for *deterministic* ones; this step is crucial for understanding how the regular separability problem differs from the regularity problem, which does not admit a similar reduction. Then in Section 5 we reduce the regular separability problem for deterministic $\mathcal{C}$-Parikh automata to the unary separability problem for vector sets in $\mathcal{C}$.

[3] This is the same as *constrained automata* from [3].

In Section 6 we consider two instantiations of the class $\mathcal{C}$. First, taking $\mathcal{C}$ to be the semilinear sets we derive decidability for (ordinary) Parikh automata. Second, we consider the class $\mathcal{C}_{\text{SEC-VAS}}$ of *sections of* reachability sets of VASes (detailed definitions are deferred to Section 6), which allows us to obtain decidability for $\mathcal{C}_{\text{SEC-VAS}}$-Parikh automata. Note that the latter model properly extends Parikh automata.

Before proceeding with the rest of the proof for our decidability result, we present a generic reduction of the regular separability problem from which we can immediately derive a new undecidability result, which is our second main contribution.

3.1 A generic reduction

We observe that regular separability of homomorphic images of a class of languages $\mathcal{G}$ reduces to regular separability for $\mathcal{G}$ itself (cf. Lemma 2 below). Since nondeterministic Parikh automata are the homomorphic image of deterministic ones, we reduce regular separability for the nondeterministic class to the deterministic one (shown in Sec. 4); together with the decidability result for the deterministic class presented in Sec. 5, this proves Theorem 1.

Our reduction can also be used to derive undecidability results. Since context-free languages are the homomorphic image of (deterministic) visibly pushdown languages (cf. [1, Theorem 5.2]), and since regular separability is undecidable for the former class [18], we concisely reprove the recent LICS'16 result by Kopczyński [13] about undecidability of regular separability of visibly pushdown languages. Moreover, this result can be further strengthened to (deterministic) visibly one counter languages using the same observation and the recent result by Czerwiński and Lasota [6] about undecidability of regular separability for one counter automata (cf. Sec. 3.2).

We now present our generic reduction. Given two alphabets Σ and Γ, a *homomorphism* is a function $h : \Sigma \to \Gamma^*$ which extends homomorphically to a function from Σ^* to Γ^*, and thus to languages. For $\mathcal{G}$ a class of languages and $\mathcal{H}$ a class of homomorphisms, let $\mathcal{H}(\mathcal{G})$ be the class of languages obtained by applying some homomorphism from $\mathcal{H}$ to some language in $\mathcal{G}$.

▶ **Lemma 2.** *If $\mathcal{G}$ and $\mathcal{H}(\mathcal{G})$ are effectively closed under inverse images of homomorphisms from $\mathcal{H}$, then the regular separability problem in $\mathcal{H}(\mathcal{G})$ reduces to the same problem in $\mathcal{G}$.*

In statements of this form, "effective" means that for given finite computational model representing $L \in \mathcal{G}$ and $h \in \mathcal{H}$, one can effectively find a representation with a finite computational model for $h^{-1}(L) \in \mathcal{G}$, and similarly for $\mathcal{H}(\mathcal{G})$. The reduction above is a consequence of the following fundamental relationship between separators and (inverse) images of functions. (We do not exploit the further structure of homomorphisms here.)

▶ **Lemma 3.** *Let $L \subseteq \Sigma^*, K \subseteq \Gamma^*$ be two languages, and let $h : \Sigma^* \to \Gamma^*$ be a function.*

1. *If R separates $h(L)$ and K, then $h^{-1}(R)$ separates L and $h^{-1}(K)$.*
2. *If R separates L and $h^{-1}(K)$, then $h(R)$ separates $h(L)$ and K.*

Proof. The proof is elementary and it is given for completeness. For the first point, $L \subseteq h^{-1}(R)$ follows from the inclusion $h(L) \subseteq R$ since $L \subseteq h^{-1}(h(L))$, and the disjointness of $h^{-1}(R)$ and $h^{-1}(K)$ follows from disjointness of R and K. For the second point, the inclusion $h(L) \subseteq h(R)$ follows by the inclusion $L \subseteq R$, and the disjointness of $h(R)$ and K follows from the disjointness of R and $h^{-1}(K)$. ◀

Since regular languages are closed under images and inverse images of homomorphisms, we immediately obtain the following corollary.

▶ **Corollary 4.** *Let h be a homomorphism. Languages $h(L), K$ are regular separable if, and only if, $L, h^{-1}(K)$ are so.*

Since regular languages are closed under complement, the regular separability problem is in fact symmetric. Combining this observation with the corollary above, we can now prove correctness of our generic reduction.

Proof of Lemma 2. Let $h(L), K$ be two languages in $\mathcal{H}(\mathcal{G})$. By Corollary 4, regular separability for $h(L), K$ is the same as for $L, h^{-1}(K)$. Since $\mathcal{H}(\mathcal{G})$ is closed under inverse images by assumption, $h^{-1}(K)$ equals the image $g(K_1)$ of language K_1 in $\mathcal{G}$ for some g from $\mathcal{H}$. We have thus reduced to regular separability for $L, g(K_1)$, where now both L and K_1 are in $\mathcal{G}$. Since regular languages are closed under complement, regular separability for $L, g(K_1)$ is the same for $g(K_1), L$. Applying once more Corollary 4, the latter statement is equivalent to regular separability for $K_1, g^{-1}(L)$. Since $\mathcal{G}$ closed under inverse images by assumption, $g^{-1}(L)$ is itself in $\mathcal{G}$. Since every step was effective, this concludes the proof. ◀

3.2 A new undecidability result

A *one counter automaton* is a finite-state device manipulating a single natural counter, which can be incremented, decremented, and tested for zero; it is *visible* if the input symbol uniquely determines which counter operation will be performed. Therefore, languages recognized by visible one counter automata are a strict subclass of visibly pushdown languages [1]. It was recently proved that regular separability for one counter automata is undecidable [6], which is incomparable with undecidability for visibly pushdown languages [13].

As a consequence of Lemma 2 we obtain undecidability of regular separability for visible one counter automata, which is our second main result, strengthening both [6] and [13].

▶ **Theorem 5.** *Regular separability of languages recognised by (deterministic) visible one counter automata is undecidable.*

Let $\mathcal{G}$ be the class of languages recognized by visible one counter automata, and let $\mathcal{H}$ be the class of letter-to-letter (non-erasing) homomorphisms, i.e., functions of the form $h : \Sigma \to \Gamma$. In order to apply Lemma 2, it suffices to show that languages recognized by one counter automata are the effective homomorphic image of those recognized by the visible subclass, and that both classes are effectively closed under inverse images of letter-to-letter homomorphisms. We begin with the second result.

▶ **Lemma 6.** *One counter languages and visibly one counter languages are effectively closed under inverse images of letter-to-letter homomorphisms.*

Proof. Given one counter automaton $\mathcal{A}$ over Σ and a letter-to-letter homomorphism $h : \Gamma \to \Sigma$, one computes an automaton $\mathcal{B}$ over Γ of the same kind s.t. $L(\mathcal{B}) = h^{-1}(L(\mathcal{A}))$ as follows. The automaton $\mathcal{B}$ is obtained by replacing a transition reading $a \in \Sigma$ in $\mathcal{A}$ by corresponding transitions reading $b \in \Gamma$ *performing the same counter operation*, for every $b \in h^{-1}(a)$. Is it easy to check that $L(\mathcal{B}) = h^{-1}(L(\mathcal{A}))$, as required. Moreover, if $\mathcal{A}$ was visible, since the counter operation is preserved, $\mathcal{B}$ will be visible too. ◀

Proof of Theorem 5. It remains to show that one counter languages are the effective homomorphic image of visible one counter languages. This is easy to show. Let $L \subseteq \Sigma^*$ be a one counter language. Each symbol $a \in \Sigma$ is split into three symbols $a_{\mathsf{inc}}, a_{\mathsf{dec}}, a_{=0?}$. The corresponding homomorphism h just forgets the new annotation, i.e., $h(a_{\mathsf{inc}}) = h(a_{\mathsf{dec}}) = h(a_{=0?}) = a$; notice that h is letter-to-letter and non-erasing. Counter operations for the

new automaton are made visible by replacing an increment operation over a by the same operation over a_{inc}, and similarly for decrements and tests. Clearly, we obtain a visible one counter automaton recognizing a language M s.t. $L = h(M)$. Thus, by Lemma 2, Lemma 6, and the undecidability of regular separability of one counter languages [6, Theorem 2], we obtain that regular separability for visibly one counter languages is undecidable. ◀

4 From nondeterministic to deterministic PA

The aim of this section is to prove the following lemma:

▶ **Lemma 7.** *If $\mathcal{C}$ is effectively closed under inverse images of linear mappings, then the regular separability problem of nondeterministic $\mathcal{C}$-Parikh automata effectively reduces to the same problem for deterministic ones.*

As a consequence of Lemma 7, we can focus on separability of deterministic PA languages in the rest of the paper. Let $\mathcal{G}$ be the class of deterministic $\mathcal{C}$-Parikh automata languages, and let $\mathcal{H}$ be the class of *letter-to-letter erasing homomorphism*, i.e., functions of the form $h : \Sigma \to (\Gamma \cup \{\varepsilon\})$ extended homomorphically to $\Sigma^* \to \Gamma^*$. The proof of the lemma follows immediately from Lemma 2 once we prove that nondeterministic languages are the effective images of deterministic ones (cf. Lemma 8), and that both classes are closed under inverse images (cf. Lemma 9). In the rest of the section, we assume that the class $\mathcal{C}$ is closed under inverse images of linear mappings, which is the case for a robust class $\mathcal{C}$ (cf. Sec. 3).

▶ **Lemma 8.** *Every nondeterministic $\mathcal{C}$-Parikh language is the effective image of a letter-to-letter erasing homomorphism of a deterministic $\mathcal{C}$-Parikh language.*

Proof. Fix a nondeterministic $\mathcal{C}$-Parikh automaton $(\mathcal{A}, S)$ over the alphabet Σ, and let T be the set of transitions of $\mathcal{A}$. Consider the letter-to-letter erasing homomorphism $h : T \to (\Sigma \cup \{\varepsilon\})$ that maps a transition (p, a, q) to a. Let $(\mathcal{B}, S)$ be the deterministic $\mathcal{C}$-Parikh automaton over the alphabet T which is obtained from $\mathcal{A}$ by relabelling every transition of $t = (p, a, q) \in T$ of $\mathcal{A}$ as a (unique) transition (p, t, q) of $\mathcal{B}$. Notice that the acceptance condition of $\mathcal{B}$ is the same as that for $\mathcal{A}$, since we only relabelled transitions. One easily verifies that $L(\mathcal{A}, S) = h(L(\mathcal{B}, S))$, as required. ◀

▶ **Lemma 9.** *Deterministic and nondeterministic $\mathcal{C}$-Parikh languages are effectively closed under inverse images of letter-to-letter erasing homomorphisms.*

Proof. Given a deterministic $\mathcal{C}$-Parikh automaton $(\mathcal{A}, S)$ over Σ and a letter-to-letter erasing homomorphism $h : \Gamma \to (\Sigma \cup \{\varepsilon\})$, one computes a deterministic $\mathcal{C}$-Parikh automaton $(\mathcal{B}, T)$ s.t. $L(\mathcal{B}, T) = h^{-1}(L(\mathcal{A}, S))$ as follows. The automaton $\mathcal{B}$ is obtained by replacing every transition (p, a, q) in $\mathcal{A}$ by transitions (p, b, q), one for every $b \in h^{-1}(a)$. Moreover, each state p in the automaton $\mathcal{B}$ has a self-loop (p, b, p) for every $b \in h^{-1}(\varepsilon)$. The constraint $T \in \mathcal{C}$ is the inverse image of S under the linear function obtained by counting a transition (p, b, q) as a transition $(p, h(b), q)$ if $h(b) \neq \varepsilon$, and by counting (p, b, q) as zero (i.e., ignoring it) otherwise. Finally, the constraint T, and hence also the automaton $(\mathcal{B}, T)$ can be computed. Is it easy to check that $L(\mathcal{B}, T) = h^{-1}(L(\mathcal{A}, S))$, as required. Moreover, if $\mathcal{A}$ is deterministic, and h is a function, then the resulting automaton $\mathcal{B}$ is also deterministic. ◀

5 Regular separability reduces to unary separability

In this section we reduce regular separability of deterministic $\mathcal{C}$-Parikh languages to unary separability of vector sets in $\mathcal{C}$.

▶ **Lemma 10.** *Let $\mathcal{C}$ be a class of vectors effectively closed under inverse images of affine mappings. The regular separability problem for deterministic $\mathcal{C}$-Parikh automata reduces to the unary separability problem for vector sets in $\mathcal{C}$.*

The rest of this section is devoted to the proof of the lemma. Let $L_1, L_2 \subseteq \Sigma^*$ be languages of deterministic $\mathcal{C}$-Parikh automata $(\mathcal{A}_1, S_1)$ and $(\mathcal{A}_2, S_2)$, respectively. There are three steps:

1. As the first step, we show that w.l.o.g. we may assume $\mathcal{A}_1 = \mathcal{A}_2$.
2. In the second step, we partition Σ^* into finitely many regular languages $K_1, \ldots, K_m$ and we reduce regular separability of L_1 and L_2 to regular separability of $L_1 \cap K_i$ and $L_2 \cap K_i$ for every $i \in \{1, \ldots, m\}$. These subproblems turn out to be easier than the general one, due to the additional structural information encoded in the languages K_i's.
3. In the last step, we reduce regular separability of $L_1 \cap K_i$ and $L_2 \cap K_i$ to unary separability of vector sets in $\mathcal{C}$.

Step 1: Unifying the underlying automaton. As the input languages are subsets of regular languages recognised by their underlying finite automata, $L_1 = L(\mathcal{A}_1, S_1) \subseteq L(\mathcal{A}_1)$ and $L_2 = L(\mathcal{A}_2, S_2) \subseteq L(\mathcal{A}_2)$, it is enough to consider separability of L_1 and L_2 *inside* the intersection of $L(\mathcal{A}_1)$ and $L(\mathcal{A}_2)$:

▶ Proposition 1. The languages L_1 and L_2 are regular separable if, and only if, the languages $L_1 \cap L(\mathcal{A}_2)$ and $L_2 \cap L(\mathcal{A}_1)$ are so.

Proof. The "only if" direction is trivial as every language separating L_1 and L_2 separates $L_1 \cap L(\mathcal{A}_2)$ and $L_2 \cap L(\mathcal{A}_1)$ as well. For the opposite direction, we observe that if a regular language S separates $L_1 \cap L(\mathcal{A}_2)$ and $L_2 \cap L(\mathcal{A}_1)$, then $S' = (S \cap L(\mathcal{A}_1)) \cup \overline{L(\mathcal{A}_2)}$ is a regular language separating L_1 and L_2. ◀

Let $\mathcal{A}$ be the product automaton of $\mathcal{A}_1$ and $\mathcal{A}_2$, and thus $L(\mathcal{A}) = L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$. It is deterministic since both $\mathcal{A}_1$ and $\mathcal{A}_2$ are so. We claim that one can compute sets $U_1, U_2 \in \mathcal{C}$ such that $L_1 \cap L(\mathcal{A}_2) = L(\mathcal{A}, U_1)$ and $L_2 \cap L(\mathcal{A}_1) = L(\mathcal{A}, U_2)$. The set T of transitions of $\mathcal{A}$ is a subset of the product $T_1 \times T_2$ of transitions of $\mathcal{A}_1$ and $\mathcal{A}_2$, and thus there are obvious projections functions $\pi_1 : T \to T_1$ and $\pi_2 : T \to T_2$. If we enumerate the transition sets, say $T_1 = \{t_1^1, \ldots, t_1^m\}$, $T_2 = \{t_2^1, \ldots, t_2^n\}$, and $T = \{t_1, \ldots, t_\ell\}$ with $\ell \leq m \cdot n$, we obtain $\pi_1 : \{1, \ldots, \ell\} \to \{1, \ldots, m\}$ and $\pi_2 : \{1, \ldots, \ell\} \to \{1, \ldots, n\}$. We use these projections to define two linear (and in particular, affine) functions $\psi_1 : \mathbb{N}^\ell \to \mathbb{N}^m$ and $\psi_2 : \mathbb{N}^\ell \to \mathbb{N}^n$ which instead of counting transitions in T, count the corresponding transitions in T_1 or in T_2, respectively; formally,

$$\psi_1(v)[j] = \sum_{i:\pi_1(i)=j} v[i] \qquad \psi_2(v)[j] = \sum_{i:\pi_2(i)=j} v[i].$$

Finally, we set $U_1 := \psi_1^{-1}(S_1)$ and $U_2 := \psi_2^{-1}(S_2)$. Intuitively, U_1 and U_2 are as S_1 and S_2, except that instead of single transitions of $\mathcal{A}_1$ or $\mathcal{A}_2$ they are seeing pairs of transitions, and simply ignore one of them. Since $\mathcal{C}$ is closed under inverse images of affine mappings by assumption, $U_1, U_2 \in \mathcal{C}$. For the rest of the proof we may thus assume that the input automata are $(\mathcal{A}, U_1)$ and $(\mathcal{A}, U_2)$.

Step 2: Regular partitioning using skeletons. We finitely partition Σ^* s.t. words belonging to the same partition behave similarly with respect to automaton $\mathcal{A}$. We use the notion of *skeleton* of a run, defined already in [3], where it was used to solve the regularity problem

of *unambiguous* Parikh automata. The idea is to traverse a run from left to right while removing (and counting) simple cycles visiting states that have already appeared.

A *simple cycle* is a sequence of transitions $c = t_1 \dots t_n \in T^*$, where $t_i = (q_{i-1}, a_i, q_i)$, starting and ending in the same state $q_0 = q_n$ where $q_1, \dots, q_n$ are pairwise distinct. Two simple cycles c, d are *equivalent* if one is a cyclic permutation of the other. Let $[c]$ denote the equivalence class of c, and let $[c_1], \dots, [c_m]$ be a fixed enumeration of all such equivalence classes. (Since a simple cycle cannot visit the same state twice, except the initial state, it has length at most n, and thus the number of simple cycles, and also of equivalence classes thereof, is $m \leq d^n$, where d is the number of transitions of the automaton.) The skeleton is an inductively defined function from runs to pairs consisting of a run and a vector $v \in \mathbb{N}^m$. In the base case, $\textsc{skel}(\varepsilon) = (\varepsilon, 0)$. For the induction step, suppose that $\textsc{skel}(t_1 \dots t_{k-1}) = (u_1 \dots u_\ell, v)$ is already defined, and let q be the ending state of the new transition t_k. If q does not appear in the run $u_1 \dots u_\ell$, then we put $\textsc{skel}(t_1 \dots t_k) = (u_1 \dots u_\ell t_k, v)$. Otherwise, let u_h, for $h < \ell$, be the last transition that ends in state q, and consider the cycle $c = u_{h+1} \dots u_\ell t_k \in [c_j]$ (for some $1 \leq j \leq m$). We have two cases to consider. If all states visited by this cycle appeared before in $u_1 \dots u_h$, then we call this cycle *absorbed* and we remove it by putting $\textsc{skel}(t_1 \dots t_k) = (u_1 \dots u_h, v + e_j)$, where e_j is the vector which is 1 in coordinate j, and 0 everywhere else. Otherwise, we just put $\textsc{skel}(t_1 \dots t_k) = (u_1 \dots u_\ell t_k, v)$.

We remove only simple cycles visiting states that have already appeared before in order to have the following useful property.

▶ Proposition 2. If $\textsc{skel}(\rho) = (\hat{\rho}, v)$, then ρ and $\hat{\rho}$ visit the same set of states.

By abusing nomenclature, we call a run ρ a *skeleton* if $\textsc{skel}(\rho) = (\rho, v)$, for some $v \in \mathbb{N}^m$. It is easy to see that the length of a skeleton is at most n^2, where n is the number of states in the automaton $\mathcal{A}$. (Assume towards a contradiction that the length of the skeleton is longer than n^2. By the pigeonhole principle, some state is thus visited more than n times, so there are at least n cycles in between two consecutive occurrences of this state in the skeleton. Therefore it is impossible that each loop contains some new state not present in all the previous loops, and thus one of these loops should be removed during the process of creating the skeleton, a contradiction.) Consequently, if d is the total number of transitions of $\mathcal{A}$, then there are at most d^{n^2} skeletons. Let $\rho_1, \dots, \rho_h$ be all the skeletons, with $h \leq d^{n^2}$. We define K_i to be the set of all words w having an accepting run ρ in automaton $\mathcal{A}$ with $\textsc{skel}(\rho) = \rho_i$. Since $\mathcal{A}$ is deterministic we know that $K_i \cap K_j = \emptyset$ for $i \neq j$. Therefore $K_1, \dots, K_h$ and $K_{h+1} = \Sigma^* \setminus (\bigcup_{1 \leq i \leq h} K_i)$ form a partition of Σ^*. All languages K_i are necessarily regular, since the skeleton can be computed by a finite automaton. The following lemma can be seen as a generalization of Proposition 1 and it is immediate to prove.

▶ **Lemma 11.** *Let Σ^* be partitioned into regular languages $K_1, \dots, K_k$. Two languages $L_1, L_2 \subseteq \Sigma^*$ are regular separable if, and only if, $L_1 \cap K_i$ and $L_2 \cap K_i$ are regular separable for every $i \in \{1, \dots, k\}$.*

It remains to decide regular separability for the languages $L(\mathcal{A}, U_1) \cap K_i$ and $L(\mathcal{A}, U_2) \cap K_i$. In the following, fix a skeleton ρ and the set of words K with skeleton ρ. Since we have fixed a skeleton, we assume w.l.o.g. that the acceptance conditions U_1, U_2 are included in $\Pi(K)$.

Step 3: Reduction to unary separability for $\mathcal{C}$. Let the set of transitions of the automaton be $T = \{t_1, \dots, t_d\}$ (thus $\rho \in T^*$), and let $\mu : \mathbb{N}^m \to \mathbb{N}^d$ be the following affine function that

transforms counting cycles into counting transitions:

$$\mu(x_1, \ldots, x_m) = \Pi(\rho) + \sum_{1 \le j \le m} \Pi([c_j]) \cdot x_j.$$

Since $\Pi(c) = \Pi(d)$ for $c, d \in [c_j]$, $\Pi([c_j])$ is well-defined. Notice that μ is affine, and not linear, since we must take into account the initial cost of the skeleton $\Pi(\rho)$. Let $V_1 = \mu^{-1}(U_1)$ and $V_2 = \mu^{-1}(U_2)$ be the corresponding sets counting cycles instead of transitions. Since we assumed $U_1, U_2 \subseteq \Pi(K)$, every vector $v \in V_1$ is realizable by an accepting run $\hat{\rho}$ s.t. $\textsc{skel}(\hat{\rho}) = (\rho, v)$. Since $\mathcal{C}$ is closed under the inverse image of affine mappings, $V_1, V_2 \in \mathcal{C}$.

▶ **Lemma 12.** *The following two conditions are equivalent:*

1. *The two languages $L(\mathcal{A}, U_1) \cap K, L(\mathcal{A}, U_2) \cap K \subseteq \Sigma^*$ are regular separable.*
2. *The two sets of vectors $V_1, V_2 \subseteq \mathbb{N}^m$ are unary separable.*

Proof. For the implication 1) $\Rightarrow$ 2), suppose R is a regular language separating $L(\mathcal{A}, U_1) \cap K$ and $L(\mathcal{A}, U_2) \cap K$. For two words $x, y \in \Sigma^*$, define $x \equiv_R y$ if $x \in R \iff y \in R$. Fix $\omega \in \mathbb{N}$ such that for all words $x, y, z \in \Sigma^*$,

$$xy^{\omega}z \;\equiv_R\; xy^{2\omega}z. \tag{1}$$

It is easy to see that for every regular language R such ω exists. The simplest way of showing this is to consider the syntactic monoid M of R and to let ω be its idempotent power, i.e., a number such that $m^{\omega} = (m^{\omega})^2$ for every $m \in M$.

Recall n-unary equivalence: $u \equiv_n v$ if $u[i] \equiv v[i] \mod n$ and moreover $u[i] \le n \iff v[i] \le n$ for every coordinate $1 \le i \le m$. It is enough to show that for all $v_1 \in V_1, v_2 \in V_2$ it holds $v_1 \not\equiv_{\omega} v_2$. Indeed, if this is the case, the unary set $S = \{v \in \mathbb{N}^m \mid \exists_{v_1 \in V_1} v \equiv_{\omega} v_1\}$ separates V_1 and V_2.

Suppose, towards a contradiction, that there are some $v_1 \in V_1, v_2 \in V_2$ such that $v_1 \equiv_{\omega} v_2$. There are runs ρ_1, ρ_2 s.t. $\textsc{skel}(\rho_1) = (\rho, v_1)$ and $\textsc{skel}(\rho_2) = (\rho, v_2)$. We extend the equivalence $\equiv_R$ on runs by saying that $\rho_1 \equiv_R \rho_2$ if their two labellings are $\equiv_R$-equivalent. Since the labelling of ρ_1 is in $L(\mathcal{A}, U_1) \cap K$, and similarly for ρ_2, if $\rho_1 \equiv_R \rho_2$, then we derive a contradiction since R was supposed to separate $L(\mathcal{A}, U_1) \cap K$ and $L(\mathcal{A}, U_2) \cap K$. While in general $\rho_1 \equiv_R \rho_2$ does not hold, we can construct two *canonical runs* $\hat{\rho}_1$ and $\hat{\rho}_2$ s.t. 1) $\textsc{skel}(\hat{\rho}_1) = (\rho, v_1)$ (thus the labelling of $\hat{\rho}_1$ is also in $L(\mathcal{A}, U_1) \cap K$), 2) $\textsc{skel}(\hat{\rho}_2) = (\rho, v_2)$ (similarly), and 3) $\hat{\rho}_1 \equiv_R \hat{\rho}_2$ (thus bringing the contradiction). We show the construction for $\hat{\rho}_1$; the one for $\hat{\rho}_2$ is similar. By Proposition 2, states visited by the run ρ_1 are among those visited by the skeleton ρ, and in particular every absorbed cycle (i.e., $v_1[j] \neq 0$) also has this property. While in general a simple cycle $d \in [c_j]$ starting (and ending) at some state q cannot be reintroduced in the skeleton ρ at any position labelled by q (because not all states in d need to have appeared before this position), there always is a position i_j in the skeleton labelled by some state q_j and a simple cycle $\hat{c}_j \in [c_j]$ *starting (and ending) at* q_j s.t. all states in $\hat{c}_j$ have appeared already before position i_j in the skeleton. Assume w.l.o.g. that $i_1 \le \cdots \le i_m$. It is possible to split the skeleton ρ as $\alpha_0 \cdots \alpha_m$ s.t., for every $0 \le j \le m$, $\alpha_j \in T^*$ is a sequence of transitions, and the prefix $\alpha_0 \cdots \alpha_{j-1}$ has length i_j (thus ends in q_j). Then, define $\hat{\rho}_1$ and similarly $\hat{\rho}_2$ as

$$\hat{\rho}_1 := \alpha_0 \hat{c}_1^{v_1[1]} \alpha_1 \cdots \alpha_{m-1} \hat{c}_m^{v_1[m]} \alpha_m \quad \text{and} \quad \hat{\rho}_2 := \alpha_0 \hat{c}_1^{v_2[1]} \alpha_1 \cdots \alpha_{m-1} \hat{c}_m^{v_2[m]} \alpha_m.$$

(This is well-defined since $v_1 \equiv_{\omega} v_2$ implies $v_1[j] \neq 0$ iff $v_2[j] \neq 0$.) Properties 1) and 2) above are guaranteed by construction. For Property 3), since $v_1 \equiv_{\omega} v_2$, by repetitive use of Equation (1) we have $\hat{\rho}_1 \equiv_R \hat{\rho}_2$. This concludes the proof of the first implication.

For proving the implication 2) $\Rightarrow$ 1), suppose that a unary set S separates V_1 and V_2. We claim that the language $R = L(\mathcal{A}, \mu(S)) \cap K$ is regular and separates $L(\mathcal{A}, U_1) \cap K$ and $L(\mathcal{A}, U_2) \cap K$. We first verify that R separates these two languages. Clearly, $U_1 \subseteq \mu(V_1) \subseteq \mu(S)$, so $L(\mathcal{A}, U_1) \cap K \subseteq L(\mathcal{A}, \mu(S)) \cap K = R$. The disjointness of $L(\mathcal{A}, U_2) \cap K$ and R is shown by contradiction. Suppose that there is a word $w \in K$ belonging both to $L(\mathcal{A}, \mu(S))$ and to $L(\mathcal{A}, U_2)$, let ρ be the run of $\mathcal{A}$ over w and let $v = \Pi(\rho)$. We have $v \in \mu(S) \cap U_2$, which implies $v = \mu(s)$ for some $s \in S \cap \mu^{-1}(U_2) = S \cap V_2$. In consequence $S \cap V_2$ is nonempty, thus contradicting the assumption that S separates V_1 and V_2.

In order to prove that R is regular it suffices to prove that $L(\mathcal{A}, \mu(S))$ is regular. The finite nondeterministic automaton recognizing this language simulates a run $\rho = t_{i_1} \dots t_{i_\ell}$ of $\mathcal{A}$, and accepts when $\Pi(\rho) \in \mu(S)$. Since S is unary, the automaton can evaluate this condition using finite memory. For every cycle c_j, the automaton stores a vector $x_j < \Pi(c_j)$, and a number n_j up to the unary equivalence $\equiv_n$, with the following meaning: the vector $\Pi(c_i)$ has been already executed n_j times, and x_i is the current "remainder". Additionally, the automaton stores a vector $x \leq \Pi(\rho)$ which is counting those transitions on the skeleton which have not been counted as cycles. At every input letter the automaton guesses nondeterministically one of cycles c_i or the skeleton and updates x_j, n_j and x accordingly. The automaton accepts when $x = \Pi(\rho_i)$, $x_j = 0$ for all j, and $(n_1, \dots, n_m) \in S$. ◀

6 Applications

We derive two corollaries of Theorem 1. By a *projection* we mean a function $\pi_{k,I} : \mathbb{N}^k \to \mathbb{N}^{|I|}$, for $I \subseteq \{1 \dots k\}$, that drops coordinates not in I. We start with a simple but useful lemma:

▶ **Lemma 13.** *If a class $\mathcal{C} \subseteq \bigcup_{d \in \mathbb{N}} \mathcal{P}(\mathbb{N}^d)$ contains all semilinear sets and is effectively closed under intersections, projections, and inverse images of projections, then it is effectively closed under inverse images of affine maps.*

Proof. Let S be a set in $\mathcal{C}$ and $f : \mathbb{N}^k \to \mathbb{N}^\ell$ be an affine map defined by $f(u) = Mu + v$ for $M = (m_{i,j})$ a matrix of dimension $\ell \times k$ and v a vector of dimension ℓ. Let $e_j \in \mathbb{N}^k$ be the vector s.t. $e_j[j] = 1$ and 0 otherwise, and let $m_j = (m_{1,j}, m_{2,j}, \dots, m_{\ell,j})$ be the (transpose of) the j-th column of M. First remark that the set $E_1 = \{(x, f(x)) \mid x \in \mathbb{N}^k\} \subseteq \mathbb{N}^{k+\ell}$ is linear with base $(0^k, v)$ and periods $\{p_1, \dots, p_k\}$, where $p_j = (e_j, m_j) \in \mathbb{N}^{k+\ell}$. Thus, $E_1 \in \mathcal{C}$. Therefore the set $E_2 = E_1 \cap \pi^{-1}_{k+\ell,I}(S)$ is also in $\mathcal{C}$, for $I = \{k+1, \dots, k+\ell\}$. Finally, we conclude since $\pi_{k+\ell,J}(E_2) = f^{-1}(S)$ with $J = \{1, \dots, k\}$. ◀

▶ **Corollary 14.** *Regular separability is decidable for nondeterministic Parikh automata.*

Proof. In order to apply Theorem 1 for $\mathcal{C}$ being semilinear sets, we need to know that the class of semilinear sets is robust. First, Lemma 13 yields effective closure under inverse images of affine maps, as semilinear sets are effectively closed under boolean combinations, images, and inverse images of projections. Second, decidability of the unary separability problem for semilinear sets is a corollary of the main result in [4]. This theorem states that separability of rational relations in $\Sigma^* \times \mathbb{N}^m$ by recognizable relations is decidable. If we ignore the Σ^* component we get the same result for rational and recognizable relations in $\mathbb{N}^m$, which are exactly semilinear sets and unary sets, respectively. ◀

For the second corollary we have to introduce vector addition systems and sections thereof. A d-dimensional *vector addition system* (VAS) is a pair $V = (s, T)$, where $s \in \mathbb{N}^d$ is a *source* configuration and $T \subseteq_{\text{FIN}} \mathbb{Z}^d$ is a finite set of *transitions*. A *run* is a sequence

$$(v_0, t_0, v_1), (v_1, t_1, v_2), \dots, (v_{n-1}, t_{n-1}, v_n) \in \mathbb{N}^d \times T \times \mathbb{N}^d$$

such that for all $i \in \{0, \ldots, n-1\}$ we have $v_i + t_i = v_{i+1}$ and $v_0 = s$. The *target* of this run is the configuration v_n. The *reachability set* of a VAS V is the set of targets of all its runs.

In order to ensure robustness, we slightly enlarge the family of VAS reachability sets to *sections* thereof. The intuition about a section is that we fix values on a subset of coordinates in vectors, and collect all the values that can occur on the other coordinates. For a subset $I \subseteq \{1, \ldots, d\}$, the projection $\pi_{d,I}$ extends element-wise to sets of vectors $S \subseteq \mathbb{N}^d$, denoted $\pi_{d,I}(S)$. For a vector $u \in \mathbb{N}^{d-|I|}$, the *section* of S w.r.t. I and u is the set

$$\pi_{d,I}(\{v \in S \mid \pi_{d,\{1,\ldots,d\}\setminus I}(v) = u\}) \subseteq \mathbb{N}^{|I|} \ .$$

We denote by $\mathcal{C}_{\text{SEC-VAS}}$ the family of all sections of VAS reachability sets.

▶ **Corollary 15.** *Regular separability is decidable for nondet. $\mathcal{C}_{\text{SEC-VAS}}$-Parikh automata.*

Proof. We apply Theorem 1 for $\mathcal{C} = \mathcal{C}_{\text{SEC-VAS}}$; we thus need to show that class $\mathcal{C}_{\text{SEC-VAS}}$ is robust. Decidability of unary separability of sets from $\mathcal{C}_{\text{SEC-VAS}}$ is shown in Theorem 9 in [5]. Effective closure of $\mathcal{C}$ under inverse images of affine functions will follow by Lemma 13 once we prove all its assumptions. First, $\mathcal{C}_{\text{SEC-VAS}}$ contains all semilinear sets. Effective closure under intersections is shown in Proposition 7 in [5]. Effective closure under inverse images of projections is easy: extend the VAS with additional coordinates, and allow it to arbitrarily increase these coordinates. Finally, to see that $\mathcal{C}_{\text{SEC-VAS}}$ is effectively closed under projections consider a section $S \subseteq \mathbb{N}^d$ of the reachability set of a VAS V, and a subset of coordinates $I \subseteq \{1, \ldots, d\}$. We construct a VAS V' which is like V, but additionally allows to decrease every coordinate from $\{1, \ldots, d\} \setminus I$. The projection $\pi_{d,I}(S)$ of S onto I is a section of the reachability set of V' defined similarly as S, but with an additional requirement that all coordinates from $\{1, \ldots, d\} \setminus I$ have value 0. ◀

7 Conclusions

We have shown that the regular separability problem for $\mathcal{C}$-Parikh automata is decidable, for every class $\mathcal{C}$ of acceptance conditions satisfying mild assumptions. In particular, we have shown decidability for $\mathcal{C}$ being either the semilinear sets or sections of VAS reachability sets. We have complemented our positive result by proving undecidability for visibly one counter languages, which sharpens two existing undecidability results.

The complexity of our algorithm depends on two factors: the complexity of unary separability for $\mathcal{C}$, and the complexity of computing inverse images of sets in $\mathcal{C}$ under affine mappings. The first factor is the dominant one here, since computing inverse images can be shown to be in PTIME for both semilinear sets and sections of VAS reachability sets (cf. [5]). Under this assumption, it can be seen that our reduction from nondeterministic to deterministic automata in Sec. 4 is also PTIME. Moreover, when reducing to unary separability in Sec. 5, since there are at most exponentially many skeletons and simple cycles, we obtain exponentially many unary separability instances, each of exponential size. Therefore, all together our reduction can be performed in exponential space.

On the other hand, the complexity of unary separability for semilinear sets and VAS sections is much higher. For semilinear sets, no upper bounds have been published for this problem, but an analysis of the algorithm of [4] yields an elementary bound[4]. For VAS sections, one can easily see that unary separability is at least as hard as the VAS reachability problem [5], which is hard for exponential space and not known to be primitive recursive.

[4] The problem becomes PTIME for the subclass of *diagonal* linear sets, in which case unary separability becomes the same as *modular separability*, and the latter problem is PTIME [4].

References

1 Rajeev Alur and P. Madhusudan. Adding nesting structure to words. *J. ACM*, 56(3):16:1–16:43, May 2009. URL: http://doi.acm.org/10.1145/1516512.1516518, doi:10.1145/1516512.1516518.

2 Michaël Cadilhac, Alain Finkel, and Pierre McKenzie. On the Expressiveness of Parikh Automata and Related Models. In *Proc. of NCMA'11*, pages 103–119, 2011.

3 Michaël Cadilhac, Alain Finkel, and Pierre McKenzie. Unambiguous constrained automata. *Int. J. Found. Comput. Sci.*, 24(7):1099–1116, 2013. URL: http://dx.doi.org/10.1142/S0129054113400339, doi:10.1142/S0129054113400339.

4 Christian Choffrut and Serge Grigorieff. Separability of rational relations in $A^* \times \mathbb{N}^m$ by recognizable relations is decidable. *Inf. Process. Lett.*, 99(1):27–32, 2006.

5 Lorenzo Clemente, Wojciech Czerwinski, Slawomir Lasota, and Charles Paperman. Separability of Reachability Sets of Vector Addition Systems. In *Proc. of STACS'17*, volume 66 of *LIPICs*, pages 24:1–24:14, 2017. URL: http://drops.dagstuhl.de/opus/volltexte/2017/7009, doi:10.4230/LIPIcs.STACS.2017.24.

6 Wojciech Czerwinski and Slawomir Lasota. Regular separability of one counter automata. In *Proc. of LICS'17*. To appear.

7 Wojciech Czerwiński, Wim Martens, and Tomás Masopust. Efficient separability of regular languages by subsequences and suffixes. In *Proc. of ICALP'13*, pages 150–161, 2013.

8 Wojciech Czerwiński, Wim Martens, Lorijn van Rooijen, and Marc Zeitoun. A note on decidable separability by piecewise testable languages. In *Proc. of FCT'15*, pages 173–185, 2015.

9 Jean Goubault-Larrecq and Sylvain Schmitz. Deciding piecewise testable separability for regular tree languages. In *Proc. of ICALP'16*, pages 97:1–97:15, 2016. URL: http://dx.doi.org/10.4230/LIPIcs.ICALP.2016.97, doi:10.4230/LIPIcs.ICALP.2016.97.

10 Harry B. Hunt III. On the decidability of grammar problems. *J. ACM*, 29(2):429–447, 1982.

11 Oscar H. Ibarra. Reversal-bounded multicounter machines and their decision problems. *J. ACM*, 25(1):116–133, 1978.

12 Felix Klaedtke and Harald Rueß. Monadic second-order logics with cardinalities. In *Proc. of ICALP'03*, pages 681–696, 2003. doi:10.1007/3-540-45061-0_54.

13 Eryk Kopczynski. Invisible pushdown languages. In *Proc. of LICS'16*, pages 867–872, 2016. URL: http://doi.acm.org/10.1145/2933575.2933579, doi:10.1145/2933575.2933579.

14 Thomas Place, Lorijn van Rooijen, and Marc Zeitoun. Separating regular languages by locally testable and locally threshold testable languages. In *Proc. of FSTTCS'13*, pages 363–375, 2013.

15 Thomas Place, Lorijn van Rooijen, and Marc Zeitoun. Separating regular languages by piecewise testable and unambiguous languages. In *Proc. of MFCS'13*, pages 729–740, 2013.

16 Thomas Place and Marc Zeitoun. Going higher in the first-order quantifier alternation hierarchy on words. In *Proc. of ICALP'14*, pages 342–353, 2014.

17 Thomas Place and Marc Zeitoun. Separating regular languages with first-order logic. *Log. Methods Comput. Sci.*, 12(1), 2016.

18 Thomas G. Szymanski and John H. Williams. Noncanonical extensions of bottom-up parsing techniques. *SIAM Journal on Computing*, 5(2):231–250, 1976.