Languages of Boundedly-Ambiguous Vector Addition Systems with States

Wojciech Czerwiński
University of Warsaw, Poland

Łukasz Orlikowski
University of Warsaw, Poland

Abstract

The aim of this paper is to deliver broad understanding of a class of languages of boundedly-ambiguous VASSs, that is k-ambiguous VASSs for some natural k. These are languages of Vector Addition Systems with States with the acceptance condition defined by the set of accepting states such that each accepted word has at most k accepting runs. We develop tools for proving that a given language is not accepted by any k-ambiguous VASS. Using them we show a few negative results: lack of some closure properties of languages of k-ambiguous VASSs and undecidability of the k-ambiguity problem, namely the question whether a given VASS language is a language of some k-ambiguous VASS. In fact we show an even more general undecidability result stating that for any class containing all regular languages and only k-ambiguous VASS languages for some $k \in \mathbb{N}$ it is undecidable whether a language of a given 1-dimensional VASS belongs to this class. Finally, we show that the regularity problem is decidable for k-ambiguous VASSs.

2012 ACM Subject Classification Theory of computation → Parallel computing models

Keywords and phrases vector addition systems, Petri nets, unambiguity, bounded-ambiguity, languages

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2025.13

Related Version *Full Version*: `https://arxiv.org/abs/2504.07669`

Funding Both authors are supported by the ERC grant INFSYS, agreement no. 950398.

Acknowledgements We would like to thank David Purser for sharing with us a conjecture about language not recognised by an unambiguous VASS. We thank also Sławomir Lasota for suggesting us to reduce from 0-finite-reach instead of reducing from the halting problem for 2CM in the proof of Theorem 1. Additionally, we would like to thank Piotr Hofman for inspiring discussions and for reviewing the Master's Thesis [35], which served as the foundation for the results presented in this paper.

1 Introduction

Determinism is a central notion in computer science. Deterministic systems often allow for more efficient algorithms. On the other hand, usually deterministic systems are less expressive, so in many cases for a non-deterministic system there is no equivalent deterministic system and one cannot use more efficient techniques. For those reasons, there is recently a lot of research devoted to various notions restricting nondeterminism in a milder way than determinism. The hope is that systems having the considered properties are more expressive than the deterministic ones, but still allow for robust algorithms design. One prominent example of such a notion is unambiguity; a system is unambiguous if for every word there is at most one accepting run over this word. In the last decade unambiguous systems were intensely studied and for various classes of infinite-state systems the unambiguous case turns out to be much more tractable [8, 9, 18, 25, 29]. Similar notions were also investigated recently,

© Wojciech Czerwiński and Łukasz Orlikowski;
licensed under Creative Commons License CC-BY 4.0

like k-ambiguity (each word is accepted by at most k runs) and history-determinism (a weakened version of determinism), in both cases one can design more efficient algorithms in some cases [7, 10].

In this paper we focus on studying milder version of determinism for Vector Addition Systems with States (VASS). VASSs and related Petri nets are popular and fundamental models of concurrency with many applications both in theory and in practical modelling [30, Section 5]. Languages of VASSs with restricted nondeterminism were already studied for several years, mostly with the acceptance condition being the set of accepting states. In [9] it was shown that the universality problem for unambiguous VASSs is decidable in ExpSpace, in contrast to Ackermann-completeness of the problem for VASSs without that restriction. In [10, 11] the language equivalence problem was considered for unambiguous VASS and more generally for k-ambiguous VASSs and it was shown to be decidable and Ackermann-complete, in contrast to undecidability in general [2], even in dimension one [21, Thm. 20]. The choice of universality and equivalence problems is deliberate here. The complexity of the seemingly more natural language emptiness problem (or equivalently of the reachability problem) does not depend on the unambiguity assumption. Indeed, one can always transform a system to a deterministic one by assigning all the transitions unique labels without affecting nonemptiness. Therefore, in order to observe a difference in complexity or decidability after restricting nondeterminism, one should consider problems, which do not simply ask about the existence of some object. Some natural problems concerning languages of that type are the universality problem and the language equivalence problem and these two problems were extensively studied for many models with the unambiguity assumption or some modification of it. First of all, the universality and equivalence problems are solvable in PTime [32] (and even in NC^2 [33]) for unambiguous finite automata (UFA), in contrast to PSpace-hardness of both problems for NFA. The universality problem was also shown decidable for unambiguous register automata in a sequence of papers [3, 14, 25] with improving complexity, which contrasts undecidability in the case without restricted nondeterminism [26, Thm 5.1]. This line of research culminated in a very elegant contribution [5, 6], which showed in particular that not only the universality problem, but also the equivalence problem is solvable in ExpTime for unambiguous register automata, and in PTime in the case of fixed number of registers. Another popular restriction of nondeterminism was also studied recently: history-determinism. A system is history-deterministic if its nondeterminism can be resolved on the fly, based on the history of a particular run. The equivalence and inclusion problems were shown to be decidable for one-dimensional history-deterministic VASSs [27], but undecidable for two-dimensional history-deterministic VASSs [7].

Despite a lot of research on algorithms for unambiguous and boundedly-ambiguous (k-ambiguous for some $k \in \mathbb{N}$) VASS not much is known about the class of languages recognisable by such models. In particular, to our best knowledge, till now it was even not known whether there exist any VASS language (we consider acceptance by states), which is not a language of an unambiguous VASS. The reason behind this lack of knowledge was absence of any technique, which can show that a given language of a VASS cannot be recognised by an unambiguous VASS. The quest for such a technique is natural and the question deserves investigation. Analogous problems were considered for other models of computation. For finite automata the question trivialises, as deterministic automata recognise all the regular languages. However, already for weighted automata over a field the problems are highly nontrivial and were recently studied in-depth [4, 28], in particular it is decidable whether a given weighted automaton is unambiguisable, so equivalent to some unambiguous weighted automaton [4]. The problem whether a given context-free language is unambiguisable is known to be undecidable since the 60s [17, 19]. The aim of this paper is deepening the understanding of the class of languages recognisable by boundedly-ambiguous VASS and its subclasses.

Our contribution. In the paper we deliver several results, which help understanding unambiguous, k-ambiguous and boundedly-ambiguous VASS languages. Our first main tool is Lemma 5, which delivers the first example of a VASS language, which is known to be not a k-ambiguous VASS language. The second main tool is Lemma 8, which formulates a condition, which needs to be satisfied by all k-ambiguous VASS languages. Using these two lemmas we can rather straightforwardly inspect closure properties of k-ambiguous VASS languages in Lemma 10 and show several expressivity results in Section 4.2. Further building on Lemma 5 we obtain our main contribution.

▶ **Theorem 1.** *For any class $\mathcal{C}$ of languages containing all regular languages and contained in the class of all boundedly-ambiguous VASS languages it is undecidable to check whether the language of a given* 1*-VASS accepting by states belongs to $\mathcal{C}$.*

Consequences of Theorem 1 are broad. It reproves undecidability of regularity of 1-dimensional VASSs (in short 1-VASSs) [12, Section 8] and undecidability of determinisability of 1-VASSs considered in [1]. It is important to emphasise that Theorem 1 gives us extensive flexibility wrt. the undecidability results. For example in [1] it was shown that for a given 1-VASS it is undecidable whether there is an equivalent deterministic 1-VASS. One can argue that possibly asking about equivalent deterministic VASS, without a bounding dimension, is a more natural question, which deserves independent research. Theorem 1 answers negatively all questions of that kind in one shot. We formulate below its corollary to illustrate variety of consequences we obtain. In particular we know now that for many classical restrictions of nondeterminism it is undecidable whether for a given 1-VASS there exists some equivalent one with nondeterminism restricted in that way.

▶ **Corollary 2.** *It is undecidable whether the language of a given* 1*-VASS accepting by states is recognisable by some*

- *unambiguous VASS,*
- *k-ambiguous VASS for given $k \in \mathbb{N}$,*
- *boundedly ambiguous VASS,*
- *deterministic VASS,*
- *k-ambiguous* 1*-VASS for given $k \in \mathbb{N}$.*

Our last main contribution is Theorem 22, which states that the regularity problem is decidable for boundedly-ambiguous VASSs, which contrasts undecidability without that assumption [2,21]. One can see this result as an intuitive indication that boundedly-ambiguous VASSs are closer to deterministic VASSs rather than to general nondeterministic VASSs.

Organisation of the paper. In Section 2 we introduce preliminary notions and recall useful lemmas. Section 3 is devoted to showing the two main technical tools, namely Lemmas 5 and 8. In Section 4 we present closure properties and expressivity results for the classes of k-ambiguous VASSs. Next, in Section 5 we show our main result, namely Theorem 1. Theorem 22 is proved in Section 6. Finally, in Section 7 we discuss interesting future research directions.

2 Preliminaries

Basic notions. For $a, b \in \mathbb{N}$ such that $a \leq b$ we write $[a, b]$ for the set of integers $\{a, a + 1, \ldots, b\}$. For $a \in \mathbb{N}$ we write $[a]$ for the set $[1, a]$. For a vector $v \in \mathbb{Z}^d$ we write v_i for the i-th entry of v. For a vector $v \in \mathbb{Z}^d$ we write $\mathrm{support}(v)$ for the set $\{i \mid v_i > 0\}$. For two vectors $u, v \in \mathbb{N}^d$ we write $u \leq v$ if for all $i \in [d]$ we have $u_i \leq v_i$. We also extend this order to $(\mathbb{N} \cup \{\omega\})^d$ where ω is bigger than any natural number. We write $\mathbb{N}_\omega$ for the set $\mathbb{N} \cup \{\omega\}$. Whenever we speak about the norm of vector $v \in \mathbb{Z}^d$ we mean $||v|| = \max_{1 \leq i \leq d} |v_i|$. For a word w we denote by $\#_a(w)$ the number of letters a in the word w.

Downward-closed sets. A set $S \subseteq N^d$ is downward-closed if for each $u, v \in \mathbb{N}^d$ it holds that $u \in S$ and $v \leq u$ implies $v \in S$. For $u \in \mathbb{N}^d$ we write $u\downarrow$ for the set $\{v \mid v \leq u\}$. If a downward-closed set is of the form $u\downarrow$ we call it a down-atom. Observe, that a one-dimensional set $S \subseteq \mathbb{N}$ is downward-closed if either $S = \mathbb{N}$ or $S = [0, n]$ for some $n \in \mathbb{N}$. Thus we have either $S = \omega\downarrow$ in the first case or $S = n\downarrow$ in the second case. So equivalently a downward-closed set $D \subseteq \mathbb{N}^d$ is a down-atom if it is of a form $D = D_1 \times \ldots \times D_d$, where for all $i \in [d]$ we have that D_i is a downward-closed one dimensional set. For simplicity we write $D = (n_1, n_2, \ldots, n_d)\downarrow$ if $D = n_1\downarrow \times n_2\downarrow \ldots n_d\downarrow$. Therefore each down-atom is of the form $u\downarrow$ where $u \in (\mathbb{N} \cup \omega)^d$. The following proposition will be helpful in our considerations.

▶ **Proposition 3** (Lemma 17 in [13], [16]). *Each downward-closed set in $\mathbb{N}^d$ is a finite union of down-atoms.*

Vector Addition Systems with States. A d-dimensional Vector Addition System with States (d-VASS) is a nondeterministic finite automaton with d non-negative integer counters. Transitions of the VASS manipulate these counters. Formally, we define VASS V as $V = (\Sigma, Q, \delta, I, F)$ where Σ is a finite alphabet, Q is a finite set of automaton states, $\delta \subseteq Q \times (\Sigma \cup \varepsilon) \times \mathbb{Z}^d \times Q$ is a transition relation, $I \subseteq Q \times \mathbb{N}^d$ is a finite set of initial configurations and $F \subseteq Q \times \mathbb{N}^d$ is a finite set of final configurations. For a transition $t = (s, a, v, s') \in \delta$ we say that the transition is over a or the transition reads letter a. We also write $\textsc{eff}(t)$ for the effect of transition, which is v. We define the norm of transition t as the norm of v.

A d-VASS can be seen as an infinite-state labelled transition system in which each configuration is a pair $(s, u) \in Q \times \mathbb{N}^d$. We denote such configuration as $s(u)$. We define the norm of the configuration as the norm of u. A transition $t = (s, a, v, s') \in \delta$ can be fired in a configuration $q(u)$ if and only if $q = s$ and $u' \geq 0$ where $u' = u + v$. After firing the transition configuration is changed to $s'(u')$. We also define run as a sequence of transitions, which can be fired one after another from some configuration. For a run $\rho = t_1 t_2 \ldots t_n$ we write $\textsc{eff}(\rho)$ for $\Sigma_{i=1}^{n} \textsc{eff}(t_i)$. If we want to say something only about jth (for $j \in [d]$) entry of the $\textsc{eff}(\rho)$ we write $\textsc{eff}_j(\rho)$. We also define $\mathrm{support}(\rho)$ as $\mathrm{support}(\textsc{eff}(\rho))$. We say that a run ρ is from configuration $q(u)$ to $p(u')$ if the sequence of transitions can be fired from $q(u)$ and the final configuration is $p(u')$. We say that a run is a loop if the state of its initial configuration is the same as the state of the final configuration. For two runs $\rho_1 = \alpha_1 \ldots \alpha_n$ and $\rho_2 = \beta_1 \ldots \beta_k$ if the sequence $\alpha_1 \ldots \alpha_n \beta_1 \ldots \beta_k$ is a run ρ then we can write $\rho = \rho_1 \rho_2$. We say, that a run $\rho = t_1 \ldots t_n$ is over $w = \lambda_1 \ldots \lambda_n \in (\Sigma \cup \{\varepsilon\})^*$ (or reads w) if and only if for each $i \in [n]$ transition t_i is over λ_i. We denote by $w(\rho)$ the word read by ρ. We say that the length of a run ρ is equal to n if it consists of n transitions. For the length of a run, we write $|\rho|$.

We say, that VASS is ε-free if there is no transition over ε. In this work, unless stated otherwise, we work with ε-free VASSs.

Languages of VASSs. A run of a VASS is accepting if it starts in an initial configuration $c_0 \in I$ and ends in an accepting configuration. A configuration is accepting if it covers some configuration from the set of final configurations F. In other words configuration $q(v)$ is accepting if there exists a configuration $q(v') \in F$ such that $v \geq v'$. For a VASS V we define its language as the set of all words read by accepting runs and we denote it by $L(V)$. Languages defined in this way are called coverability languages. Sometimes we consider languages with other acceptance condition (reachability languages), in that case we indicate it clearly.

We sometimes consider reachability languages in which run ending in configuration $q(v)$ is accepting if and only if $q(v) \in F$. Sometimes, we consider VASSs, where the set of accepting configurations is infinite, but has a specific form. For instance we consider downward-VASSs, where the set of accepting configurations is possibly infinite, but it is downward-closed. In this type of VASSs a run ending in configuration $q(v)$ if and only if $q(v) \in F$ where F is a downward-closed set of accepting configurations. We say, that VASS V is deterministic if it has only one initial configuration, it is ε-free and for each state q and each letter $a \in \Sigma$ there is at most one transition over a leaving the state q. We say, that VASS V is k-ambiguous if and only if for every $w \in L(V)$ we have at most k accepting runs over w. We also say then, that $L(V)$ is a k-ambiguous language. For a d-VASS consider a function $r : (Q \times \mathbb{N}^d \times \delta)^* \times (Q \times \mathbb{N}^d) \times \Sigma \to \delta^*$ that, given a history of the run (configurations and taken transitions), current configuration $q(v)$ and a next letter $\lambda \in \Sigma$, returns a sequence of possibly many transitions. One of these transitions is over λ, all the other are over ε, and the sequence can be fired from the current configuration $q(v)$. Let us call r a resolver. We say, that d-VASS V is history-deterministic if and only if it has one initial configuration and there exists a resolver r such that for each $w \in L(V)$ run ρ over w from the initial configuration given by the resolver is accepting.

We denote by Det, Hist, k-Amb, BAmb and NonDet the class of languages of respectively deterministic, history-deterministic, k-ambiguous , all boundedly-ambiguous and fully non-deterministic VASSs languages. We sometimes denote by 1-Amb the class of unambiguous VASSs languages. By d-NonDet we denote the class of languages of d-VASSs.

3 Tools for separating BAmb and NonDet

In this section we develop two techniques for showing that a language is not recognised by a k-ambiguous VASS: Lemma 5 and Lemma 8.

3.1 Dominating block

The first tool is based on the following observation about a language not recognised by a k-ambiguous VASS.

▶ **Lemma 4.** *For every $k \in \mathbb{N}_+$ the language*

$$L_k = \{a^{n_1}ba^{n_2}ba^{n_3}b \ldots a^{n_{k+2}} \mid \exists_{i \in [k+1]}\ n_i \geq n_{i+1}\}$$

is not recognised by a k-ambiguous VASS.

For showing various properties of boundedly-ambiguous VASS Lemma 4 is sufficient. In one case, for proving Theorem 1, we need a strengthening of Lemma 4, which is presented in the following lemma:

▶ **Lemma 5.** *Let Σ be an alphabet such that $b \notin \Sigma$ and let L be a language over Σ. For each function $f : L \to \mathbb{N}_\omega$ such that $\sup f = \omega$ (recall that $\sup f = \sup\{f(x) \mid x \in L\}$) language $L_1 = \{a^{n_1}ba^{n_2}ba^{n_3}b \ldots a^{n_{k+2}}bw \mid w \in L, \exists_{i\in[k+1]}\ n_i \geq n_{i+1} \vee n_{k+2} \geq f(w)\}$ is not recognised by a k-ambiguous VASS.*

Before proving Lemma 5 we show how it implies Lemma 4.

Proof of Lemma 4. Let us fix $k \in \mathbb{N}_+$ and let $L = \{\varepsilon\}$. Let $f : L \to \mathbb{N}_\omega$ be defined as $f(\varepsilon) = \omega$. Hence by Lemma 5 we get that L_k is not recognised by a k-ambiguous VASS. ◀

Below we sketch the idea behind the proof of Lemma 5. We assume, for the sake of contradiction, that L_1 is recognised by a k-ambiguous VASS and aim at a contradiction by showing $k+1$ different runs over the same word. We first consider $k+1$ words $w_1, \ldots, w_{k+1} \in L_1$, where $u \in L$ is a particularly chosen word and $N_0 < N_1 < \ldots < N_{k+2} \in \mathbb{N}$ are particularly chosen constants:

$$w_i = a^{N_1!}ba^{N_2!}b \ldots a^{N_i!}ba^{N_0!}ba^{N_{i+2}!}ba^{N_{i+3}!}b \ldots a^{N_{k+2}!}bu.$$

Then we dive into combinatorics of VASS runs ρ_i over words w_i and conclude that there are specific pumping cycles in ρ_i. We show two main lemmas, which analyse the structure of the runs. The first one gives conditions for finding a specific loop in a run. Observe, that this lemma is general and is not restricted to k-ambiguous VASSs.

▶ **Lemma 6.** *For each d-dimensional VASS V, each subset of counters $S \subseteq [d]$ and each $n \in \mathbb{N}$ there exists a constant $M := M(V, S, n) \geq 1$ such that in every run in V, starting from a configuration in which values of counters from S are at most n, which is of length at least M there exists a loop, which has non-negative effect on the counters from S.*

The proof of Lemma 6 uses rather standard techniques similar to the ones in the Karp-Miller construction solving the coverability problem for VASS [22]. Lemma 6 is used to find loops in words w_i. For example one can show that if N_1 is big enough then one can find a loop as in Lemma 6 in the first block of letters a. Similarly, if N_2 is big enough wrt. to N_1, one can find an appropriate loop in the second block of letters a. Using this approach one can find loops in blocks of letters a, if the block is much longer than the whole prefix before it. This is however not true for the block of length N_0. Therefore we need a more sophisticated tool in order to be able to modify runs over all the words w_i to runs over the same word.

This requires the more challenging part of the proof, which provides a detailed characterisation of runs of k-ambiguous VASS. We formulate below one of the used lemmas in order to illustrate the kind of arguments we consider and, as it is also used in Section 3.2.

▶ **Lemma 7.** *Let L be a language over $\Sigma = \{a, b\}$ recognised by some k-ambiguous d-VASS V. For each $n \in \mathbb{N}$ there exists a constant C such that each run ρ in V such that:*

- *Run ρ is a prefix of an accepting run.*
- *Run ρ is reading $a^{m_1}ba^{m_2}b \ldots a^{m_{n-1}}ba^{m_n}$.*

can be decomposed as:

1. *$\rho = \alpha_1\beta_1^{a_1}\alpha'_1\alpha_2\beta_2^{a_2}\alpha'_2 \ldots \alpha_n\beta_n^{a_n}\alpha'_n$ for some $a_1, a_2, \ldots, a_n \geq 1$ and for each $i \in [1, n]$ we have $|\alpha_i|, |\beta_i|, |\alpha'_i| \leq C$.*
2. *For each $j < n$ we have $w(\alpha'_j) \in L(a^*b)$ and $w(\alpha'_n) \in L(a^*)$*
3. *For each $j \in [n]$ we have that $w(\alpha_j), w(\beta_j) \in L(a^*)$*
4. *For each $j \in [n]$ we have that β_j is either a loop or ε. Moreover if $m_j \geq 2 \cdot C + 1$ then $\beta_j \neq \varepsilon$.*

5. *For each $j \in [n]$ let $A_j = \bigcup_{1 \le i < j} \mathrm{support}(\beta_i)$ then β_j is nonnegative on counters from $[d] \setminus A_j$*
6. *For each $j \in [n]$ there is no λ, δ and λ' such that δ is a nonnegative loop on counters from $[d] \setminus A_j$, $\lambda' \neq \varepsilon$ and $\lambda\delta\lambda' = \alpha_j\beta_j$*

The proof of Lemma 7 uses the Lemma 6 and a lot of combinatorial observations. It is moved to the Appendix.

By appropriate use of Lemma 7 and other auxiliary lemmas we show that ρ_i can be modified a bit into runs ρ'_i, which are all different, all accepting and all over the same word w, where

$$w = a^{n_1} b a^{n_2} b \dots a^{n_{k+2}} b u$$

and $n_j = 2m^{k+3-j} \Pi_{l=j}^{k+2} N_l!$ (where m is the maximal norm of a transition). More concretely, ρ'_i is obtained by pumping the loops β_i in ρ_i more times. A challenge it to show that the resulting ρ'_i are indeed different, which we achieve by more careful, but technical investigation of the runs. Existence of $k+1$ different runs over the same word w is a contradiction with the assumption that L_1 is recognised by a k-ambiguous VASS and finishes the proof.

3.2 Semilinear image

Now, we develop the second tool for showing that language is not recognised by a k-ambiguous VASS. Before formulating the tool we have to provide a few definitions. For any language $L \subseteq \{a, b\}^*$ such that for each $w \in L$ we have $\#_b(w) = l$ for some $l \in \mathbb{N}$ we define

$$im(L) = \{(a_1, a_2, \dots, a_{l+1}) \mid a_1, a_2, \dots, a_{l+1} \in \mathbb{N}, a^{a_1} b a^{a_2} b \dots b a^{a_{l+1}} \in L\}$$

Given a base vector $b \in \mathbb{Z}^d$ and a finite set of period vectors $P = \{p_1, \dots, p_n\} \subseteq \mathbb{Z}^d$, the linear set $L(b, P)$ is defined as

$$L(b, P) = \{b + a_1 p_1 + \dots + a_n p_n \mid a_i \in \mathbb{N}, 1 \le i \le n\}$$

A semi-linear set is a finite union of linear sets. Now we are ready to formulate the second tool.

▶ **Lemma 8.** *Let $L \subseteq \{a, b\}^*$ be a language satisfying:*
- *L is recognised by k-ambiguous VASS V.*
- *There exists $n \in \mathbb{N}$ such that for each $w \in L$ we have $\#_b(w) = n$.*

Then $im(L)$ is a semilinear set.

Proof. The proof is based on Lemma 7 and the fact that set of solution of a system of linear Diophantine inequalities is semilinear. Notice first, that L satisfies conditions of Lemma 7. Therefore we can apply Lemma 7 to L and $n+1$ and decompose each accepting run in V in the way presented in Lemma 7. Let us fix some $\alpha_1, \dots, \alpha_{n+1}$, $\alpha'_1, \dots, \alpha'_{n+1}$ and $\beta_1, \dots, \beta_{n+1}$, initial configuration $q(c)$ and accepting configuration $p(f)$. We call a run ρ accepting with respect to $q(c)$ and $p(f)$ if ρ is an accepting run of the VASS starting in $q(c)$ and ending in configuration $p(c')$ such that $c' \ge f$. Let K be the following language:

$$K = \{w \in L \mid \text{there exist } a_1, a_2, \dots, a_{n+1} \in \mathbb{N} \text{ such that } \alpha_1 \beta_1^{a_1} \alpha'_1 \alpha_2 \beta_2^{a_2} \alpha'_2 \dots \alpha_{n+1} \beta_{n+1}^{a_{n+1}} \alpha'_{n+1}$$
$$\text{is an accepting run with respect to } q(c) \text{ and } p(f) \text{ and reads } w\}$$

Observe, that K depends on chosen α_i, β_i and α'_i. Moreover, observe, that because of the constant given by Lemma 7 we have only a finite number of possibilities of $\alpha_1, \dots, \alpha_{k+1}$,

$\alpha'_1, \ldots, \alpha'_{k+1}$, $\beta_1, \ldots, \beta_{k+1}$, initial configuration $q(c)$ and accepting configuration $p(f)$. Moreover, semilinear sets are closed under a finite union. Therefore to conclude, that $im(L)$ is a semilinear set it is enough to show that $im(K)$ is a semilinear set. Notice, that from conditions of Lemma 7, we know, that only α'_i (for $i \in [n]$) contain letter b, each exactly one letter at the last position. Hence:

$$im(K) = \{(|\beta_1|a_1 + |\alpha_1| + |\alpha'_1| - 1, |\beta_2|a_2 + |\alpha_2| + |\alpha'_2| - 1, \ldots,$$

$$, |\beta_n|a_n + |\alpha_n| + |\alpha'_n| - 1, |\beta_{k+1}|a_{n+1} + |\alpha_{n+1}| + |\alpha'_{n+1}|) \mid$$

$$\text{such that } \alpha_1\beta_1^{a_1}\alpha'_1\alpha_2\beta_2^{a_2}\alpha'_2 \ldots \alpha_{n+1}\beta_{n+1}^{a_{k+1}}\alpha'_{n+1} \text{ is an accepting run with respect to } q(c) \text{ and } p(f)\}$$

Therefore it is enough to show, that:

$$A = \{(a_1, a_2, \ldots, a_{n+1}) \mid \text{such that}$$

$$\alpha_1\beta_1^{a_1}\alpha'_1\alpha_2\beta_2^{a_2}\alpha'_2 \ldots \alpha_{n+1}\beta_{n+1}^{a_{n+1}}\alpha'_{n+1}$$

$$\text{is an accepting run with respect to } q(c) \text{ and } p(f)\}$$

is a semilinear set. We have two cases. Either $A = \emptyset$, hence semilinear. This case occurs if for any $a_1, a_2, \ldots, a_{n+1} \geq 1$ we do not have an accepting run with respect to $q(c)$ and $p(f)$. Otherwise, we will show semilinearity, by providing a system of linear inequalities for $a_1, a_2, \ldots, a_{n+1}$. It is enough because in [20] it was shown, that the set of solutions of a system of linear inequalities is a semilinear set. The goal of this system of linear inequalities is to express, that after each prefix of a run, we are non-negative on all of the counters. Moreover, we want also to express the acceptance condition. Therefore for each counter i we write the following inequalities:

- For each transition t and each α_j such that there exist u and v such that $\alpha_j = utv$:

$$c_i + \text{EFF}_i(\alpha_1\beta_1^{a_1}\alpha'_1\alpha_2\beta_2^{a_2}\alpha'_2 \ldots \alpha_{j-1}\beta_{j-1}^{a_{j-1}}\alpha'_{j-1}) + \text{EFF}_i(ut) \geq 0$$

- For each transition t and each α'_j such that there exist u and v such that $\alpha'_j = utv$:

$$c_i + \text{EFF}_i(\alpha_1\beta_1^{a_1}\alpha'_1\alpha_2\beta_2^{a_2}\alpha'_2 \ldots \alpha_{j-1}\beta_{j-1}^{a_{j-1}}\alpha'_{j-1}\alpha_j\beta_j^{a_j}) + \text{EFF}_i(ut) \geq 0$$

- For each transition t and each β_j such that there exist u and v such that $\beta_j = utv$:
  - If $\text{EFF}_i(\beta_j) \leq 0$:

$$c_i + \text{EFF}_i(\alpha_1\beta_1^{a_1}\alpha'_1\alpha_2\beta_2^{a_2}\alpha'_2 \ldots \alpha_{j-1}\beta_{j-1}^{a_{j-1}}\alpha'_{j-1}\alpha_j\beta_j^{a_j-1}) + \text{EFF}_i(ut) \geq 0$$

  - Otherwise:

$$c_i + \text{EFF}_i(\alpha_1\beta_1^{a_1}\alpha'_1\alpha_2\beta_2^{a_2}\alpha'_2 \ldots \alpha_{j-1}\beta_{j-1}^{a_{j-1}}\alpha'_{j-1}\alpha_j) + \text{EFF}_i(ut) \geq 0$$

- Acceptance condition:

$$c_i + \text{EFF}_i(\alpha_1\beta_1^{a_1}\alpha'_1\alpha_2\beta_2^{a_2}\alpha'_2 \ldots \alpha_{k+1}\beta_{n+1}^{a_{n+1}}\alpha'_{n+1}) \geq f_i$$

- Condition, that each a_i is positive (this is needed because of conditions of Lemma 7): $a_i \geq 1$

In other words, this system of inequalities ensures, that each transition in the sequence can be fired, check the acceptance condition and ensures that each a_i is positive. We have shown, that the set A is semilinear and hence $im(K)$ is a semilinear set. Therefore, because semilinar sets are closed under a finite union we have that $im(L)$ is a semilinear set. ◀

Then, as shown in Lemma 10, k-ambiguous VASS languages are closed under intersection with regular languages. As mentioned below languages K_n are regular for each $n \in \mathbb{N}$, the following corollary holds:

▶ **Corollary 9.** *Let $a, b \in \Sigma$, $L \subseteq \Sigma^*$ be a language recognised by a k-ambiguous VASS and for $n \in \mathbb{N}$ let $K_n \subseteq \{a, b\}^*$ be the language of words containing exactly n letters b. Then for any $n \in N$ we have that $im(L \cap K_n)$ is a semilinear set.*

4 Properties

In this section we present several properties of languages of boundedly-ambiguous Vector Addition Systems with States.

4.1 Closure properties

First, we investigate the closure properties of boundedly-ambiguous languages.

▶ **Lemma 10.** *If L_1 and L_2 are recognised by a k_1-ambiguous and k_2-ambiguous VASS respectively then:*

- *$L_1 \cap L_2$ is recognised by a $(k_1 \cdot k_2)$-ambiguous VASS;*
- *$L_1 \cup L_2$ is recognised by a $(k_1 + k_2)$-ambiguous VASS.*

Moreover, class of languages of boundedly-ambiguous VASSs is not closed under complementation and commutative closure.

Proof. We split the proof into several parts, each corresponding to the closure properties under one operation.

Intersection. Let L_1 and L_2 be languages recognised respectively by d_1-dimensional k_1-ambiguous VASS A_1 and d_2-dimensional k_2-ambiguous VASS A_2. Language $L_1 \cap L_2$ can be recognised by the standard synchronised product of A_1 and A_2. It is easy to observe that the product is a $(k_1 \cdot k_2)$-ambiguous $(d_1 + d_2)$-VASS.

Union. Let L_1 and L_2 be languages recognised by d_1-dimensional k_1-ambiguous VASS A_1 and d_2-dimensional k_2-ambiguous VASS A_2. The idea is to recognise $L_1 \cup L_2$ by taking union VASS $A_1 \cup A_2$, which is clearly $k_1 + k_2$-ambiguous and $\max(d_1, d_2)$-dimensional.

Complementation. This comes from a general fact, that coverability languages are not closed under complementation. Language $L = a^n b^{\leq n}$ is recognised even by a deterministic VASS (hence also by one with bounded-ambiguity). On the other hand, in [13] it was shown, that every two disjoint coverability VASS[1] languages L_1 and L_2 are regular separable, which means there exists a regular language L_3 such that $L_1 \subseteq L_3$ and $L_2 \cap L_3 = \emptyset$. Because L is a coverability VASS language its complement can be a coverability VASS language if and only if L is a regular language. Clearly L_1 is not and this can be shown using the Pumping Lemma for regular languages.

Commutative closure. Boundedly-ambiguous languages are not closed under commutative closure. Let us consider language $L = a^n b^{\leq n}$, which is recognised by a deterministic VASS. Its commutative closure is equal to $L_1 = \{w \mid \#_a(w) \geq \#_b(w)\}$. Assume, towards contradiction, that L_1 is recognised by a k-ambiguous VASS for $k \in \mathbb{N}_{\geq 0}$. Because boundedly-ambiguous languages are closed under intersection with regular languages also language

[1] In fact this result was shown for a wider class of Well-structured transition system (WSTS)

$L_1 \cap L(b^* a^*) = b^n a^{\geq n}$ is recognised by a k-ambiguous VASS. Let V be k-ambiguous VASS recognising $b^n a^{\geq n}$. By Lemma 6 we can take N such, that while accepting $b^N a^N$ VASS V will fire a non-negative loop on all of the counters. This loop reads b^l for some $l \in \mathbb{N}$. Because VASS V is k-ambiguous $l \geq 1$. Hence we can fire this loop one more time and accept $b^{N+l} a^N$, which is not in the language $b^n a^{\geq n}$. Therefore we reached a contradiction and boundedly-ambiguous languages are not closed under commutative closure. ◀

Using the fact that regular languages are unambiguous (i.e. 1-ambiguous) VASS languages and Lemma 10 we can formulate the following remark about closure of boundedly-ambiguous languages under intersection with unambiguous (hence also regular) languages.

▶ Remark 11. For each $k \in \mathbb{N}_+$ the class of k-ambiguous VASS languages is closed under intersection with unambiguous VASS languages. Hence, the same holds for the intersection with regular languages.

We complement Lemma 10 with Lemma 12 and Conjecture 13.

▶ **Lemma 12.** *For each $k_1, k_2 \in \mathbb{N}_+$ there exist languages L_1, L_2, which are respectively recognised by a k_1 and k_2 ambiguous VASS such that language $L_1 \cup L_2$ is not recognised by an n-ambiguous VASS for $n \in [k_1 + k_2 - 1]$.*

Proof. Let $k = k_1 + k_2$. For $i \in [k]$ let us define language $U_i = \{a^{n_1} b a^{n_2} b a^{n_3} b \ldots a^{n_{k+1}} \mid n_i \geq n_{i+1}\}$. Observe, that U_i is a language of an unambiguous VASS. Let $L_1 = \bigcup_{i=1}^{k_1} U_i$ and $L_2 = \bigcup_{i=k_1+1}^{k}$. By Lemma 10 languages L_1 and L_2 are respectively recognised by a k_1 and k_2 ambiguous VASS. By applying Lemma 4 to $k-1$ we get that language $L_1 \cup L_2$ is not recognised by a n-ambiguous VASS for $n \in [k-1]$. ◀

▶ **Conjecture 13.** *For each $k_1, k_2 \in \mathbb{N}_+$ there exists languages L_1, L_2, which are respectively recognised by a k_1 and k_2 ambiguous VASS such that language $L_1 \cap L_2$ is not recognised by a n-ambiguous VASS for $n \in [k_1 \cdot k_2 - 1]$.*

4.2 Expressiveness

Firstly, we show that the hierarchy of k-ambiguous VASS is strict.

▶ **Lemma 14.** *For every $k \in \mathbb{N}_+$ there exists language $L \in$ (k+1)-Amb \ k-Amb.*

Proof. Let $L = \{a^{n_1} b a^{n_2} b a^{n_3} b \ldots a^{n_{k+2}} \mid \exists_{i \in [k+1]}\, n_i \geq n_{i+1}\}$. Because of Lemma 4 language L is not recognised by a k-ambiguous VASS. On the other hand it is a union of $k+1$ unambiguous VASS languages. For $i \in [k+1]$ let us define $L_i = \{a^{n_1} b a^{n_2} b a^{n_3} b \ldots a^{n_{k+2}} \mid n_i \geq n_{i+1}\}$. Observe, that $L = \bigcup_{i=1}^{k+1} L_i$. Hence, by Lemma 10, L is recognised by a $(k+1)$-ambiguous VASS. ◀

In terms of the classes of languages they define, boundedly-ambiguous VASS can express strictly more than deterministic ones. Moreover, they are strictly less expressive than non-deterministic ones. We prove this in Lemmas 15 and 17.

▶ **Lemma 15.** *There exists language $L \in$ 1-Amb \ Det.*

We show an even stronger Lemma 16. It is stronger because we have Det $\subseteq$ Hist.

▶ **Lemma 16.** *There exists language $L \in$ 1-Amb \ Hist.*

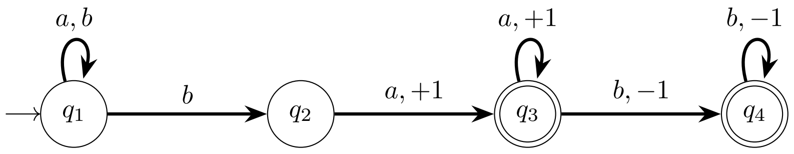

Figure 1 Unambiguous VASS recognising (starting from zero) $\{a,b\}^*ba^{n>0}b^{\leq n}$.

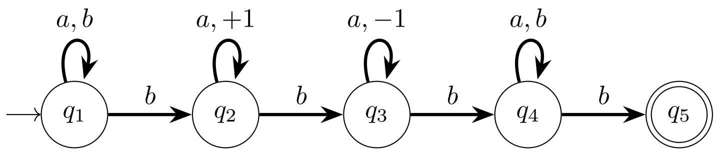

Figure 2 VASS recognising (starting from zero) $\{a^nba^mba^k \mid n,m,k \in \mathbb{N}, (n \geq m \vee n \geq k)\}$.

Proof. Let $L = \{a,b\}^*ba^{n>0}b^{\leq n}$. Observe, that L is recognized by an unambiguous VASS depicted in Figure 1. The only point of nondeterminism is in state q_1 and there is only one way to guess when the last block, ending the word in the form $a^{n>0}b^{\leq n}$ comes.

On the other hand, assume, towards contradiction, that L is recognised by a history-deterministic VASS. It is easy to see, that $L_1 = a^{n>0}b^{\leq n}$ is a language of a deterministic VASS. Hence it is also history-deterministic. In [7] it was shown, that history-deterministic VASS languages are closed under union. Hence $L_2 = L \cup L_1 = \{a,b\}^*a^{n>0}b^{\leq n}$ is also history-deterministic VASS language. However, in [7] it was also shown, that L_2 is not a history-deterministic VASS language. Hence we get, that L is not a history-deterministic VASS language. ◀

▶ **Lemma 17.** *There exists language $L \in$ 1-NonDet $\setminus$ BAmb.*

Proof. Let $L = \{ua^nba^mbv \mid u, v \in L((a^*b)^*), n \geq m\}$. Let us fix $k \in \mathbb{N}_+$. We show, that L is not recognised by a k-ambiguous VASS. Assume, towards contradiction, that it is. Hence language $L_k = L \cap L((a^*b)^{k+2})$ is also recognised by a k-ambiguous VASS. Let f be a function such that $f : \{\epsilon\} \to \mathbb{N}_\omega$ and $f(\epsilon) = \omega$. Therefore, by Lemma 5, we get, that L_k is not recognised by a k-ambiguous VASS, contradiction. Hence L is not recognised by a k-ambiguous VASS. On the other hand, it can be recognised by a 1-dimensional VASS, which is presented in Figure 2. ◀

Observe, that because each word is read by only finite number of accepting runs, one can see bounded-ambiguity as an extension of the notion of determinism. A second extension of the determinism is history-determinism. In [7] it was shown, that history-deterministic VASSs can express more than deterministic ones and less than nondeterministic ones. Up to now, there has been no comparison of the expressive power of history-deterministic and bounded-ambiguous VASSs. Now, we show, that these language classes are incomparable, which means that there exists a language recognised by an unambiguous VASS, which is not a language of history-deterministic VASS and language which is recognised by a history-deterministic VASS and it is not recognised by a k-ambiguous VASS for any $k \in \mathbb{N}_+$. Recall, that in Lemma 16 we have shown that there exists $L \in$ 1-Amb $\setminus$ Hist. Hence it is enough to show the following Lemma:

▶ **Lemma 18.** *There exists language $L \in$ Hist $\setminus$ BAmb.*

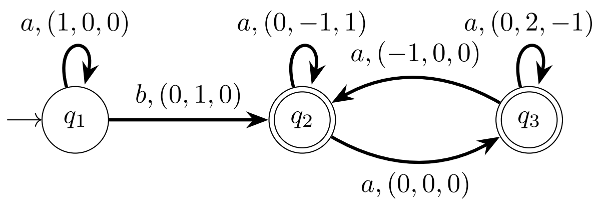

Figure 3 VASS recognising (starting from zero) $\{a^n b a^m \mid n, m \in \mathbb{N}, (m \leq 2n + 2^{n+2} - 1)\}$.

Proof. Let $L = a^n b a^{2n+2^{n+2}-1}$. Observe, that $im(L)$ is not a semilinear set. Hence, due to Lemma 8, for every $k \in \mathbb{N}_+$ we have that L is not recognised by a k-ambiguous VASS. Hence $L \notin$ BAmb. On the other hand it is recognised by a history-deterministic VASS presented in the Figure 3. It is history-deterministic, because the best option is to fire loops in states q_2 and q_3 as long as possible. In this way one can read words $a^n b a^k$ for each $k \in \mathbb{N}$ such that

$$k \leq 2 \cdot \Sigma_{i=1}^{n-1}(2^i + 1) + 2^n + 1 + 2^n = 2 \cdot (2^n - 1) + 2 \cdot n + 2^{n+1} + 1 = 2 \cdot n + 2^{n+2} - 1 \blacktriangleleft$$

5 Proof of Theorem 1

In this section we prove Theorem 1. We start by introducing *Lossy counter machines* (LCMs) [24, 31]. Formally, an LCM is $M = \langle Q, Z, \Delta \rangle$ where $Q = \{l_1, \ldots, l_m\}$ is a finite set of states, $Z = (z_1, \ldots, z_n)$ are n counters, and $\Delta \subseteq Q \times OP(Z) \times Q$, where $OP(Z) = \{\text{inc}, \text{dec}, \text{ztest}, \text{skip}\}^n$. A configuration of M is $q(a)$ where $q \in Q$ and $a = (a_1, ..., a_n) \in \mathbb{N}^n$. There is a transition $q(a) \xrightarrow{t} q'(b)$ if there exists $t \in \Delta$ such that $t = (q, op, q')$ and for each $i \in [1, n]$:

- If $op_i = \text{inc}$ then $b_i \leq a_i + 1$
- If $op_i = \text{dec}$ then $b_i \leq a_i - 1$
- If $op_i = \text{skip}$ then $b_i \leq a_i$
- If $op_i = \text{ztest}$ then $a_i = b_i = 0$

Observe, that counters can nondeterministically decrease at each step. A run of M is a finite sequence $q_1(a_1) \xrightarrow{t_1} q_2(a_2) \xrightarrow{t_2} \ldots \xrightarrow{t_{n-1}} q_n(a_n)$. Given a configuration $q(u)$, the reachability set of $q(u)$ is the set of all configurations reachable from $q(u)$ via runs of M. We denote this set as $\textsc{Reach}(q(u))$. For simplicity we denote by $\textsc{Reach}(q)$ the set of configurations reachable from $q(\vec{0})$. It was shown in [31] that the problem of deciding whether for a configuration $q(u)$ and LCM M set $\textsc{Reach}(q(u))$ configuration is finite, is undecidable. Due to [1] even if u is always equal to $\vec{0}$ the problem is still undecidable. We call this problem **0-finite reach**. We prove Theorem 1 by reducing from 0-finite reach. The proof is similar to the proofs of undecidability of regularity [34] and determinization [1]. The rest of the section is devoted to the proof of Theorem 1.

Firstly, we present an overview of the proof. For each LCM M_1 with an initial state we create another LCM M and initial state l_0 with the same answer to the 0-finite reach problem. Then, we define a language L_{M,l_0}, which intuitively encodes the correct runs of M. For technical reasons, namely because coverability VASSs are well-suited for recognising languages similar to $a^n b^{\leq n}$ we encode the correct runs in reverse, that means from the final configuration to the initial one. Moreover, because we work with LCMs, it is better for us to work with the complement of L_{M,l_0}, that is $\overline{L_{M,l_0}}$. In such a way we get at the end a language for which Lemma 5 is useful. Then, the proof of Theorem 1 is split into three claims. Claim 19 states, that $\overline{L_{M,l_0}}$ is recognised by an effectively constructible 1-dimensional VASS

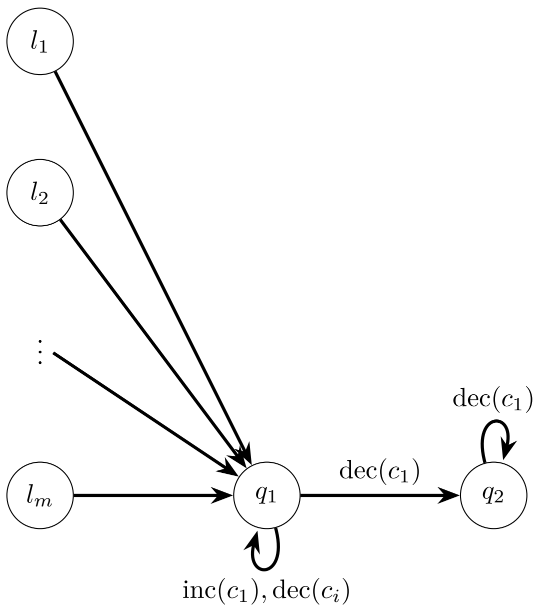

Figure 4 Sketch of the construction of LCM M.

A. Claim 20 provides, that if $\textsc{Reach}(l_0)$ is finite then L_{M,l_0} is a regular language. Finally, Claim 21 states, that if $\textsc{Reach}(l_0)$ is not finite then L_{M,l_0} is not recognised by a VASS from the class of all boundedly-ambiguous VASS. All these claims give a direct reduction from 0-finite reach to deciding whether a language of a 1-VASS belongs to class C of languages, which contains all regular languages and is contained in the class of all boundedly-ambiguous VASS languages. Thus they conclude the proof of Theorem 1.

Let us fix LCM $M_1 = \langle Q_1, Z_1, \Delta_1 \rangle$ with $Q = \{l_1, l_2, \ldots l_m\}$ and $Z = \{z_1, z_2, \ldots, z_n\}$. Let l_0 be the initial state of M_1. We add to M_1 two states: q_1 and q_2 and for $i \in [1, m]$ transitions t_i from l_i to q_1 with no effect on the counters. In addition, for each $i \in [2, n]$ we add a transition from q_1 to q_1 decrementing the ith counter and incrementing the first counter. We also add two transitions decrementing the first counter. The first one goes from q_1 to q_2. The second one goes from q_2 to q_2. In such a way we obtain LCM $M = \langle Q, Z, \Delta \rangle$. The sketch of the construction of M is presented in the Figure 4. Observe, that because each of the added transitions does not increase the sum of the counters, from q_1 we can go only to q_2 and later we can only stay in q_2 the answer for 0-finite reach is the same for both: M_1 and M. Hence, we proceed later with M. We encode each configuration $q(a_1, a_2, \ldots, a_n)$ as a word over $\Sigma = Q \cup Z$ as $qz_1^{a_1} z_2^{a_2} \ldots z_n^{a_n}$. We use encodings of configurations to obtain an encoding of a run by concatenating encodings of its configurations. Finally, we define language $L_{M,l_0} = \{w^r \mid w \text{ is an encoding of a run in } M \text{ from } l_0(0, 0, \ldots, 0)\}$.

▷ Claim 19. One can construct 1-dimensional VASS recognising $\overline{L_{M,l_0}}$.

Proof. W.l.o.g. we assume, that there is at most one transition between each pair of states (otherwise we may split a state into several states, one for each incoming letter). We construct A such that it accepts w if and only if w^r does not represent a valid run of M from $l_0(0, 0, \ldots, 0)$. The idea is, that A guesses the violation in the run represented by the word w. We have three types of violations. The first type is a control state violation, that is the run uses nonexisting transition or does not start in $l_0(0, 0, \ldots, 0)$. The second type is a counter violation, that we have invalid counter values between two consecutive configurations.

The third violation is that we have invalid encoding of a configuration, that is we have two consecutive letters $z_i z_j$ such that $j > i$. To spot control violation for nonexisting transitions we will have gadget for each pair of states p and q such that there is no transition from p to q spotting infix of the form $z_n^* z_{n-1}^* \dots z_1^* q z_n^* z_{n-1}^* \dots z_1^* p$. Such gadget is just an NFA. Observe, that if a run start in $l_0(0,0,\dots,0)$ then either the whole encoding of the run is equal to l_0 or suffix is of the form pl_0 for some $p \in Q$. Therefore to spot control violation, that the run does not start in $l_0(0,0,\dots,0)$ we will have an NFA recognising words such that suffix is not of the form pl_0 for some $p \in Q$ and the encoding is not equal to l_0. To spot counter violation we will have a 1-dimensional VASS for each transition and each counter spotting violation when firing this transition on this counter. Let us fix transition t and counter z_i. Let transition t be from state p to state q. We have four possibilities for the operation performed by t on the counter z_i. Therefore we need to spot infix of the form:

1. $z_n^* z_{n-1}^* \dots z_i^a \dots z_1^* q z_n^* z_{n-1}^* \dots z_i^b \dots z_1^* p$ where $a > b-1$ (equivalently $a \geq b$) if transition t decrements counter z_i.
2. $z_n^* z_{n-1}^* \dots z_i^a \dots z_1^* q z_n^* z_{n-1}^* \dots z_i^b \dots z_1^* p$ where $a > b+1$ if transition t increments counter z_i.
3. $z_n^* z_{n-1}^* \dots z_i^a \dots z_1^* q z_n^* z_{n-1}^* \dots z_i^b \dots z_1^* p$ where $a > b$ if transition t has no effect on the counter z_i.
4. $z_n^* z_{n-1}^* \dots z_i^a \dots z_1^* q z_n^* z_{n-1}^* \dots z_i^b \dots z_1^* p$ where $a > 0$ or $b > 0$ if transition t zero-tests counter z_i.

All of the above can be done with 1-dimensional VASS. To spot invalid encoding of a configuration for each $1 \leq i < j \leq n$ we will have an NFA recognising words with infix $z_i z_j$. As all the gadgets have one initial state in which we "ignore" some prefix of the word, we can join them and obtain one dimensional VASS with single initial configuration. ◁

▷ Claim 20. If Reach(l_0) is finite then $\overline{L_{M,l_0}}$ is a regular language.

Proof. Observe, that in the proof of Claim 19 only gadgets spotting counter violations were not an NFA. Therefore it is enough to replace them by some NFAs. As Reach(l_0) is finite there exists a bound $B \in \mathbb{N}$ on the possible values of the counters. Hence we can implement each gadget spotting counter violation using the fact, that $B \geq a, b$. For instance for transition t from state p to q having no effect on the counter z_i we can have an NFA being a union of NFAs for each $0 \leq b < a \leq B$ spotting infix of the form $z_n^* z_{n-1}^* \dots z_i^a \dots z_1^* q z_n^* z_{n-1}^* \dots z_i^b \dots z_1^* p$. In this way we will not detect counter violation increasing the counter to the value above B. Therefore we add another gadget spotting, that one counter is above B hence then the word does not encode a correct run. This can be done by spotting for each $i \in [1,n]$ infix z_i^{B+1}, which can be done by an NFA. Because every gadget is now an NFA we showed that $\overline{L_{M,l_0}}$ is a regular language when Reach(l_0) is finite. ◁

▷ Claim 21. If Reach(l_0) is infinite then $\overline{L_{M,l_0}}$ is not recognised by a boundedly-ambiguous VASS.

Proof. Assume, towards contradiction, that $\overline{L_{M,l_0}}$ is recognized by a k-ambiguous VASS for some $k \in \mathbb{N}_+$. Recall, that there exist $q_1, q_2 \in Q$ such that from each $q_3 \in Q$ such that $q_3 \neq q_2$ and $q_3 \neq q_1$ there exists a transition from q_3 to q_1 with no effect on the counters. Moreover, for each $i \in [2,n]$ there exists a transition from q_1 to q_1 decrementing the ith counter and increasing the first counter. Additionally, there is a single transition leaving q_1 decrementing the first counter to q_2 and there is exactly one transition leaving q_2, which goes to q_2 and decrements the first counter.

As k-ambiguous VASS languages are closed under intersection with a regular language also $L_1 = \overline{L_{M,l_0}} \cap (z_1^* q_2)^{k+2} Z^* q_1 (Q \cup Z \setminus q_2)^*$ is recognised by a k-ambiguous VASS. The idea of this intersection is to get a language similar to the language presented in Lemma 5. Observe, that each $w \in L_1$ can be uniquely decomposed into $w = z_1^{n_1} q_2 z_1^{n_2} q_2 \dots z_1^{n_{k+2}} q_2 v$. We denote v by $\mathrm{suff}(w)$. We define $L = \{\mathrm{suff}(w) \mid w \in L_1\}$. We define function f on words from L by setting $f(w) = 0$ if w^r encodes an incorrect run of M from $l_0(0, 0, \dots, 0)$ and otherwise $f(w) = n$ where n is the maximal number such that $w = z_1^n v$ for some word v. Observe, that $L_1 = \{z_1^{n_1} q_2 z_1^{n_2} q_2 z_1^{n_3} q_2 \dots z_1^{n_{k+2}} q_2 w \mid w \in L, \exists_{1 \le i \le k+1} n_i \ge n_{i+1} \vee n_{k+2} \ge f(w)\}$. This is because both transition from q_1 to q_2 and loop from q_2 to q_2 decrements the first counter and therefore for each $v \in L_1$ such that $v = z_1^{n_1} q_2 z_1^{n_2} q_2 z_1^{n_3} q_2 \dots z_1^{n_{k+2}} q_2 w$ we have that v^r encodes invalid run if and only if either $\mathrm{suff}(v)^r = w^r$ encodes an incorrect run (that is $f(w) = 0 \le n_{k+2}$) or v^r encodes a correct run but $f(w) \le n_{k+2}$ (recall the definition of $f(w)$ in this case) or there exists $i \in [k+1]$ such that $n_i \ge n_{i+1}$. Moreover, notice that because $\textsc{Reach}(l_0)$ is infinite and we have transitions moving values to the first counter from all the other counters in the state q_1 for each each $B \in \mathbb{N}$ there exist $B' \in \mathbb{N}$ such that $B \le B'$ and a correct run ρ_B of M from $l_0(0, 0, \dots, 0)$ to $q_1(B', 0, \dots, 0)$. Let w_B be the encoding of this run. Observe, that $(z_1^{B'} q_2)^{k+2} w_B^r \in L_1$. Hence $w_B^r \in L$ and $f(w_B^r) = B' \ge B$. Hence $\sup(f) = \omega$ and hence, by Lemma 5, L_1 is not recognised by a k-ambiguous VASS. Contradiction. ◁

6 Deciding regularity

In this section we present a proof of the following theorem, that deciding regularity of a language of a k-ambigous VASS is decidable. This is in contrast to the general case where regularity is undecidable [34].

▶ **Theorem 22.** *For every $k \in \mathbb{N}$ it is decidable whether for a given k-ambigous VASS A language $L(A)$ is regular.*

The proof uses two, already present in the literature, results. The first result is the following Theorem from [10].

▶ **Theorem 23** (Theorem 28 [10]). *For any $k \in \mathbb{N}$ and a k-ambiguous VASS one can build in elementary time a downward-VASS which recognises the complement of its language.*

The second result is the decidability of regular separability of coverability VASS language and reachability VASS language. Regular separability problem asks whether for two VASSs A and B there exists a regular language L such that $L(A) \subseteq L$ and $L(B) \cap L = \emptyset$.

▶ **Theorem 24** (Theorem 7 [15]). *Regular separability is decidable if one VASS is a coverability VASS and the second VASS is a reachability VASS.* [2]

We devote the rest of this section to the proof of Theorem 22. Let us fix a k-ambiguous VASS V. We show how to decide regularity of a language of a k-ambiguous VASS. Using Theorem 23 we get a downward-VASS $\hat{V}$ recognising the complement of $L(V)$. Observe, that $L(V)$ is regular if and only if $L(V)$ and $L(\hat{V})$ are regular separable. Now we will use the following claim about downward-VASSs.

[2] Recently even stronger result about decidability of regular separability for reachability VASSs was proven in [23].

▷ Claim 25. For every downward-VASS one can construct reachability VASS recognising the same language.

Proof. Let us fix d-dimensional downward-VASS V and let Q be the states of V. We know, that the set of accepting configurations F is downward-closed. Hence, due to Proposition 3, we have a finite set $D \subseteq Q \times (\mathbb{N} \cup \omega)^d$ such that:

$$F = \bigcup_{q(u) \in D} \{q\} \times u \downarrow$$

We obtain reachability VASS V' recognising the same language as V by taking VASS V and for each $q(u) \in D$ adding state $q_{q(u)}$, ε transition from with no effect from q to $q_{q(u)}$. Moreover, for each $i \in [1, d]$ if $u_i \in \mathbb{N}$ we add ε transition from $q_{q(u)}$ to $q_{q(u)}$ incrementing by one the ith counter and otherwise if $u_i = \omega$ we add an ε-transition from $q_{q(u)}$ to $q_{q(u)}$ decrementing by one ith counter. Let the set of added states be equal to Q'. We set the initial configurations of V' to be the initial conditions of V. For vector $u \in (\mathbb{N} \cup \omega)^d$ let us denote by $\hat{u} \in \mathbb{N}^d$ vector such that for all $i \in [1, d]$ we have $u_i = \hat{u}_i$ if $u_i \in \mathbb{N}$ and $\hat{u}_i = 0$ otherwise. We set the accepting configurations of V' to the following set of configurations $F' = \{q_{q(u)}(\hat{u}) \mid q_{q(u)} \in Q'\}$. Now we have to prove, that $L(V) = L(V')$. For $L(V) \subseteq L(V')$ observe, that for each $w \in L(V)$ there exists configuration $q(v) \in F$ such that w is read by an accepting run ρ ending in $q(v)$. As F is a downward-closed set there is u such that $v \preceq u$ such that $q_{q(v)}(\hat{v}) \in F'$. Moreover, observe that $q_{q(v)}(\hat{v})$ is reachable from $q(u)$ in V using ε-transitions. Hence $w \in L(V')$.

For $L(V') \subseteq L(V)$ observe that for each $w \in L(V')$ we have an accepting run ρ ending in configuration $q_{q(u)}(\hat{u}) \in F'$. Due to construction of V' we know, that there exists prefix of ρ, such that it also reads w and reaches configuration $q(v)$ such that $v \preceq u$ and hence $q(v) \in F$ and $w \in L(V)$. ◁

Now we invoke Claim 25 on $\hat{V}$ to get reachability VASS $\hat{V}'$ recognising the same language. Finally, we use algorithm from Theorem 24 for V and $\hat{V}'$ knowing that they are regular separable if and only if $L(V)$ is regular. This concludes the proof of Theorem 22.

7 Future research

Other problems for boundedly-ambiguous VASSs. We have shown in Section 6 that the regularity problem is decidable for baVASSs (boundedly-ambiguous VASSs), in contrast to general VASSs. It is also known that the language equivalence problem is decidable for baVASSs [11], while being undecidable for general VASSs [2, 21]. It is natural to ask whether other classical problems are decidable for baVASSs, for example deciding whether there exists an equivalent deterministic or unambiguous VASS. These problems are undecidable for general VASS due to Theorem 1, but can possibly be decidable for baVASSs. One can also ask whether given k-ambiguous VASS has an equivalent $(k-1)$-ambiguous VASS, our techniques from Section 5 used for showing undecidability does not seem to work in that case. Other further research for baVASSs would be to ask about complexity of the mentioned problems, for example to understand complexity of the regularity problem for baVASSs, since we already know it is decidable.

Languages of VASSs accepting by configuration. In this paper we have considered VASSs accepting by set of accepting states, it is natural to ask what happens if we modify the acceptance set to be a finite set of accepting configurations or even a single configuration. For

VASSs accepting by configuration already the language universality problem is undecidable. We are not aware of any citation, but the proof is rather straightforward and uses the classical technique. For a given two-counter automaton one can construct 1-VASS accepting by configuration, which recognises all the words beside correct encodings of the runs of the two-counter automaton. Therefore the reachability problem for two-counter automaton, which is undecidable, can be reduced to the universality problem for 1-VASSs accepting by a configuration. Since the universality problem is undecidable for VASSs accepting by configuration, there is not much hope that the other nontrivial problems (beside emptiness) are decidable.

However, one can ask what about VASSs accepting by configuration with some restriction on nondeterminism, for example unambiguous or boundedly-ambiguous VASSs accepting by configuration. It is natural to ask whether universality, language equivalence, regularity or determinisability problems are decidable for that models and what is its complexity.

References

1 Shaull Almagor and Asaf Yeshurun. Determinization of One-Counter Nets. In *Proceedings of CONCUR 2022*, volume 243 of *LIPIcs*, pages 18:1–18:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICS.CONCUR.2022.18.

2 Toshiro Araki and Tadao Kasami. Some Decision Problems Related to the Reachability Problem for Petri Nets. *Theor. Comput. Sci.*, 3(1):85–104, 1976. doi:10.1016/0304-3975(76)90067-0.

3 Corentin Barloy and Lorenzo Clemente. Bidimensional linear recursive sequences and universality of unambiguous register automata. In *Proceedings of STACS 2021)*, volume 187 of *LIPIcs*, pages 8:1–8:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.STACS.2021.8.

4 Jason P. Bell and Daniel Smertnig. Computing the linear hull: Deciding Deterministic? and Unambiguous? for weighted automata over fields. In *Proceedings of LICS 2023*, pages 1–13. IEEE, 2023. doi:10.1109/LICS56636.2023.10175691.

5 Mikolaj Bojanczyk, Joanna Fijalkow, Bartek Klin, and Joshua Moerman. Orbit-Finite-Dimensional Vector Spaces and Weighted Register Automata. *TheoretiCS*, 3, 2024. doi:10.46298/THEORETICS.24.13.

6 Mikolaj Bojanczyk, Bartek Klin, and Joshua Moerman. Orbit-finite-dimensional vector spaces and weighted register automata. In *Proceedings of LICS 2021*, pages 1–13. IEEE, 2021. doi:10.1109/LICS52264.2021.9470634.

7 Sougata Bose, David Purser, and Patrick Totzke. History-Deterministic Vector Addition Systems. In *Proceedings of CONCUR 2023*, volume 279 of *LIPIcs*, pages 18:1–18:17, 2023. doi:10.4230/LIPICS.CONCUR.2023.18.

8 Thomas Colcombet. Unambiguity in automata theory. In *Proceedings of DCFS 2015*, pages 3–18, 2015. doi:10.1007/978-3-319-19225-3_1.

9 Wojciech Czerwinski, Diego Figueira, and Piotr Hofman. Universality Problem for Unambiguous VASS. In *Proceedings of CONCUR 2020*, volume 171 of *LIPIcs*, pages 36:1–36:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPICS.CONCUR.2020.36.

10 Wojciech Czerwinski and Piotr Hofman. Language Inclusion for Boundedly-Ambiguous Vector Addition Systems Is Decidable. In *Proceedings of CONCUR 2022*, volume 243 of *LIPIcs*, pages 16:1–16:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICS.CONCUR.2022.16.

11 Wojciech Czerwinski and Piotr Hofman. Language Inclusion for Boundedly-Ambiguous Vector Addition Systems is Decidable. *Log. Methods Comput. Sci.*, 21(1), 2025. doi:10.46298/LMCS-21(1:27)2025.

12 Wojciech Czerwinski and Slawomir Lasota. Regular Separability of One Counter Automata. *Log. Methods Comput. Sci.*, 15(2), 2019. doi:10.23638/LMCS-15(2:20)2019.

13 Wojciech Czerwiński, Sławomir Lasota, Roland Meyer, Sebastian Muskalla, K. Narayan Kumar, and Prakash Saivasan. Regular separability of well-structured transition systems. In *29th International Conference on Concurrency Theory, CONCUR 2018*, volume 118 of *LIPIcs*, pages 35:1–35:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.CONCUR.2018.35.

14 Wojciech Czerwinski, Antoine Mottet, and Karin Quaas. New techniques for universality in unambiguous register automata. In *Proceedings of ICALP 2021*, volume 198 of *LIPIcs*, pages 129:1–129:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.ICALP.2021.129.

15 Wojciech Czerwinski and Georg Zetzsche. An Approach to Regular Separability in Vector Addition Systems. In *LICS '20: 35th Annual ACM/IEEE Symposium on Logic in Computer Science, Saarbrücken, Germany, July 8-11, 2020*, pages 341–354. ACM, 2020. doi:10.1145/3373718.3394776.

16 L.E. Dickson. Finiteness of the odd perfect and primitive abundant numbers with n distinct prime factors. *American Journal of Mathematics*, 35((4)):413–422, 1913.

17 Seymour Ginsburg and Joseph Ullian. Ambiguity in Context Free Languages. *J. ACM*, 13(1):62–89, 1966. doi:10.1145/321312.321318.

18 Mika Göös, Stefan Kiefer, and Weiqiang Yuan. Lower Bounds for Unambiguous Automata via Communication Complexity. In *Proceedings of ICALP 2022*, volume 229 of *LIPIcs*, pages 126:1–126:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICS.ICALP.2022.126.

19 Sheila A. Greibach. A Note on Undecidable Properties of Formal Languages. *Math. Syst. Theory*, 2(1):1–6, 1968. doi:10.1007/BF01691341.

20 Christoph Haase. A survival Guide to Presburger Arithmetic. *ACM SIGLOG News*, 5(3):67–82, 2018. doi:10.1145/3242953.3242964.

21 Piotr Hofman, Richard Mayr, and Patrick Totzke. Decidability of weak simulation on one-counter nets. In *Proceedings of LICS 2013*, pages 203–212. IEEE Computer Society, 2013. doi:10.1109/LICS.2013.26.

22 Richard M. Karp and Raymond E. Miller. Parallel program schemata. *J. Comput. Syst. Sci.*, 3(2):147–195, 1969. doi:10.1016/S0022-0000(69)80011-5.

23 Eren Keskin and Roland Meyer. On the separability problem of VASS reachability languages. In *Proceedings of the 39th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2024, Tallinn, Estonia, July 8-11, 2024*, pages 49:1–49:14. ACM, 2024. doi:10.1145/3661814.3662116.

24 Richard Mayr. Undecidable problems in unreliable computations. *Theor. Comput. Sci.*, 297(1-3):337–354, 2003. doi:10.1016/S0304-3975(02)00646-1.

25 Antoine Mottet and Karin Quaas. The containment problem for unambiguous register automata. In *Proceedings of STACS 2019*, pages 53:1–53:15, 2019. doi:10.4230/LIPICS.STACS.2019.53.

26 Frank Neven, Thomas Schwentick, and Victor Vianu. Finite state machines for strings over infinite alphabets. *ACM Trans. Comput. Log.*, 5(3):403–435, 2004. doi:10.1145/1013560.1013562.

27 Aditya Prakash and K. S. Thejaswini. On History-Deterministic One-Counter Nets. In *Proceedings of FoSSaCS 2023*, volume 13992 of *Lecture Notes in Computer Science*, pages 218–239. Springer, 2023. doi:10.1007/978-3-031-30829-1_11.

28 Antoni Puch and Daniel Smertnig. Factoring through monomial representations: arithmetic characterizations and ambiguity of weighted automata. *CoRR*, abs/2410.03444, 2024. doi:10.48550/arXiv.2410.03444.

29 Mikhail Raskin. A superpolynomial lower bound for the size of non-deterministic complement of an unambiguous automaton. In *Proceedings of ICALP 2018*, pages 138:1–138:11, 2018. doi:10.4230/LIPICS.ICALP.2018.138.

30 Sylvain Schmitz. The complexity of reachability in vector addition systems. *ACM SIGLOG News*, 3(1):4–21, 2016. doi:10.1145/2893582.2893585.

31 Philippe Schnoebelen. Lossy Counter Machines Decidability Cheat Sheet. In *Reachability Problems, 4th International Workshop, RP 2010, Brno, Czech Republic, August 28-29, 2010. Proceedings*, volume 6227 of *Lecture Notes in Computer Science*, pages 51–75. Springer, 2010. doi:10.1007/978-3-642-15349-5_4.

32 Richard Edwin Stearns and Harry B. Hunt III. On the equivalence and containment problems for unambiguous regular expressions, regular grammars and finite automata. *SIAM J. Comput.*, 14(3):598–611, 1985. doi:10.1137/0214044.

33 Wen-Guey Tzeng. On path equivalence of nondeterministic finite automata. *Inf. Process. Lett.*, 58(1):43–46, 1996. doi:10.1016/0020-0190(96)00039-7.

34 Rüdiger Valk and Guy Vidal-Naquet. Petri nets and regular languages. *J. Comput. Syst. Sci.*, 23(3):299–325, 1981. doi:10.1016/0022-0000(81)90067-2.

35 Łukasz Orlikowski. Languages of Unambiguous Vector Addition Systems with States. Master's thesis, University of Warsaw, 2024. URL: https://apd.uw.edu.pl/diplomas/225085/.

A Proofs for Section 3 (Tools for separating BAmb and NonDet)

Proof of Lemma 7. We prove this lemma by induction on n. For $n = 0$ we have $\rho = \varepsilon$ and hence all decomposition conditions are satisfied. Now we assume, that this Lemma is true for $n - 1$ and we show that this implies Lemma 7 for n.

Let π_1 be the prefix of ρ reading $a^{m_1}ba^{m_2}b\ldots a^{m_{n-1}}$ and let t be the next transition of ρ after π_1. Hence $\rho = \pi_1 t \pi_2$ for some π_2. By induction assumption we can apply Lemma 7 to π_1 and get constant C and decomposition:

$$\pi_1 = \alpha_1\beta_1^{a_1}\alpha_1'\alpha_2\beta_2^{a_2}\alpha_2'\ldots\alpha_{n-1}\beta_{n-1}^{a_{n-1}}\lambda$$

Observe, that we changed α_{n-1}' to λ as we will redefine α_{n-1}'. Now let us set $\alpha_{n-1}' = \lambda t$. Hence

$$\rho = \alpha_1\beta_1^{a_1}\alpha_1'\alpha_2\beta_2^{a_2}\alpha_2'\ldots\alpha_{n-1}\beta_{n-1}^{a_{n-1}}\alpha_{n-1}'\pi_2$$

Recall the definition of set A_n from condition 5, let m be the maximal norm of a transition, s be the maximal norm of an initial configuration and let us define constants $T = s + 2(n-1)mC$ and $C_n = M(V, A_n, T)$ given by Lemma 6. Let us also define constant $K_n = \max_{S\subseteq[d]} M(V, S, T + mC_n)$. Let also $C' = \max(C + 1, C_n, K_n)$. This will be a constant for Lemma 7 and n. We have two cases. The first case is that $\pi_2 = \alpha_n\beta_n^{a_n}\alpha_n'$ for some loop β_n, which is nonnegative on counters from $[d] \setminus A_n$ and $|\alpha_n\beta_n| \leq C_n$ and β_n is not a prefix of α_n'. The second case is that such a decomposition of π_2 does not exist. We start the proof with the second case.

Case 2. Observe, that for each $i \in [n-1]$ and $l \in [d] \setminus A_n \subseteq [d] \setminus A_i$:

$$\textsc{eff}_l(\beta_i) = 0$$

Hence:

$$\textsc{eff}_l(\alpha_1\beta_1^{a_1}\alpha_1'\alpha_2\beta_2^{a_2}\alpha_2'\ldots\alpha_{n-1}\beta_{n-1}^{a_{n-1}}\alpha_{n-1}') = \Sigma_{i=1}^{n-1}(\textsc{eff}_l(\alpha_i) + \textsc{eff}_l(\alpha_i')) \leq$$
$$m\Sigma_{i=1}^{n-1}(|\alpha_i| + |\alpha_i'|) \leq 2(n-1)mC = T$$

Hence, because of the definition of C_n and the fact that in this case, the decomposition of π_2 does not exist, we have that $|\pi_2| \leq C_n \leq C'$. Then we set $\alpha_n = \pi_2$ and $\beta_n = \alpha_n' = \varepsilon$. Then clearly from the induction assumption and the definition of $\alpha_{n-1}', \alpha_n, \beta_n, \alpha_n'$ conditions 1-3 and 5 are satisfied. Condition 4 is satisfied for $j < n$ by induction assumption and for $j = n$ we have $m_j = |\alpha_n\beta_n\alpha_n'| \leq C_n \leq C$ hence it is also satisfied. Condition 6 is satisfied for $j < n$ by induction assumption. It is satisfied for $j = n$ because otherwise π_2 could have been decomposed and we are in case 1.

Case 1. We know, that π_2 can be decomposed as described above. If there are multiple decompositions we choose any of the ones minimizing $|\alpha_n\beta_n|$. Now, we show that decomposition of ρ:

$$\rho = \alpha_1\beta_1^{a_1}\alpha_1'\alpha_2\beta_2^{a_2}\alpha_2' \dots \alpha_n\beta_n^{a_n}\alpha_n'$$

satisfies all the conditions. We start with conditions 2-6 as Condition 1 is the most challenging to prove.

Condition 2. For $j < n-1$ this condition follows from the induction assumption. For $j = n-1$ we have that $\alpha_{n-1}' = \lambda t$. Recall that t was a transition reading letter b and from the induction assumption λ does not read letter b. Moreover, α_n' is a suffix of π_2, which does not read the letter b. Hence the condition follows.

Condition 3. For $j < n$ the condition follows from the induction assumption and for $j = n$ we have that $\alpha_n\beta_n$ is a prefix of π_2, which does not read the letter b.

Condition 4. For $j < n$ we have this condition from induction assumption and for $j = n$ we have that $\beta_n \neq \varepsilon$ is a loop.

Condition 5. This condition for $j < n$ follows from inductions assumption and for $j = n$ follows from the fact that β_n is a nonnegative loop on counters from $[d] \setminus A_n$ as required.

Condition 6. For $j < n$ we have it from induction assumption. For $j = n$ we have it from minimality of $|\alpha_n\beta_n|$.

Condition 1. Observe, that for $i \in [n-2]$ from induction assumption we have $|\alpha_i|, |\beta_i|, |\alpha_i'| \leq C \leq C'$ and moreover $|\alpha_{n-1}|, |\beta_{n-1}| \leq C \leq C'$. Moreover, from induction assumption, $|\alpha_{n-1}|, |\beta_{n-1}| \leq C \leq C'$ and $|\alpha_{n-1}'| = |\lambda t| \leq C + 1 \leq C'$. Additionally $|\alpha_n\beta_n| \leq C_n \leq C'$. Therefore it is enough to prove that $|\alpha_n'| \leq K_n \leq C'$. Let

$$S = [d] \setminus \bigcup_{i=1}^{n} \text{support}(\beta_i)$$

Observe, that for each $l \in S$ we have:

$$\text{EFF}_l(\alpha_1\beta_1^{a_1}\alpha_1'\alpha_2\beta_2^{a_2}\alpha_2' \dots \alpha_n\beta_n^{a_n}) = \text{EFF}_l(\alpha_n) + \Sigma_{i=1}^{n-1}(\text{EFF}_l(\alpha_i) + \text{EFF}_l(\alpha_i')) \leq mC_n + 2(n-1)mC$$

Recall that $K_n = M(V, S, T + mC_n)$ and $T = 2(n-1)mC$, so if $|\alpha_n'| > K_n$ then α_n' contains a nonnegative loop on counters from S. Hence it is enough to prove, that α_n' does not contain such loop. Assume, for the sake of contradiction, that $\alpha_n' = \lambda\gamma\lambda'$ such that γ is a nonnegative loop on counters from S. We define accepting runs $\phi_1, \phi_2, \dots, \phi_{k+1}$, which will read the same word, but will be different and hence this will create a contradiction with the fact that V is k-ambiguous VASS. Intuitively, due to loops β_n and γ we have more degrees of freedom in the last block and we can create $k+1$ different runs over the same word. As ρ is a prefix of an accepting run we know, that there exists ρ_1 such that $\rho\rho_1$ is an accepting run. Firstly, we define runs ψ_1, ψ_2 and ψ_3, which will be parts of $\phi_1, \phi_2, \dots, \phi_{k+1}$:

$$\psi_1 = \alpha_1\beta_1^{a_1+b_1}\alpha_1'\alpha_2\beta_2^{a_2+b_2}\alpha_2' \dots \alpha_n\beta_n^{a_n+b_n}$$

$$\psi_2 = \lambda\gamma$$

$$\psi_3 = \lambda'\rho_1$$

where for $i \in [n]$ we define $b_i = (k+1)m|\gamma||\beta_n| + \Sigma_{j=i+1}^{n} b_j m|\beta_j|$. The intuition is, that b_i is chosen in such a way to compensate the possible decrease of counters because of additional executions of loops $\beta_{i+1}, \beta_{i+2}, \ldots, \beta_n$ and γ. Finally we define ϕ_i

$$\phi_i = \psi_1 \beta_n^{i|\gamma|} \psi_2 \gamma^{(k+1-i)|\beta_n|} \psi_3$$

where for $i \in [n]$ we define $b_i = (k+1)m|\gamma||\beta_n| + \Sigma_{j=i+1}^{n} b_j m|\beta_j|$. Now it is enough to prove Claims 26, 27 and 28.

▷ Claim 26. For each $i \in [k+1]$ we have that ϕ_i is an accepting run.

Proof. Let us fix $i \in [k+1]$. Because $\rho\rho_1$ is an accepting run it is enough to prove that:

1. For each $j \in [n-1]$ and $l \in [d]$ we have

$$\text{EFF}_l(\alpha_1 \beta_1^{a_1+b_1} \alpha_1' \alpha_2 \beta_2^{a_2+b_2} \alpha_2' \ldots \alpha_j \beta_j^{a_j+b_j}) \geq \text{EFF}_l(\alpha_1 \beta_1^{a_1} \alpha_1' \alpha_2 \beta_2^{a_2} \alpha_2' \ldots \alpha_j \beta_j^{a_j})$$

2. For each $l \in [d]$ we have

$$\text{EFF}_l(\psi_1 \beta_n^{i|\gamma|}) \geq \text{EFF}_l(\alpha_1 \beta_1^{a_1} \alpha_1' \alpha_2 \beta_2^{a_2} \alpha_2' \ldots \alpha_n \beta_n^{a_n})$$

3. For each $l \in [d]$ we have

$$\text{EFF}_l(\psi_1 \beta_n^{b_n+i|\gamma|} \psi_2 \gamma^{(k+1-i)|\beta_n|}) \geq \text{EFF}_l(\alpha_1 \beta_1^{a_1} \alpha_1' \alpha_2 \beta_2^{a_2} \alpha_2' \ldots \alpha_n \beta_n^{a_n} \lambda \gamma)$$

Point 1. We prove this by induction on j. For $j = 1$ the inequality follows from the fact that for each $l \in [d]$ we have $\text{EFF}_l(\beta_1) \geq 0$. Now we show the induction step. If $\text{EFF}_l(\beta_j) \geq 0$ we get the inequality from the induction assumption. Hence now we assume $\text{EFF}_l(\beta_j) < 0$. Therefore there has to be $u < j$ such that $\text{EFF}_l(\beta_u) > 0$. Let u be the maximal such. Observe, that from induction assumption:

$$\text{EFF}_l(\alpha_1 \beta_1^{a_1+b_1} \alpha_1' \alpha_2 \beta_2^{a_2+b_2} \alpha_2' \ldots \alpha_{s-1} \beta_{u-1}^{a_{u-1}+b_{u-1}} \alpha_u \beta_u^{a_u} \alpha_u' \ldots \alpha_j \beta_j^{a_j} \alpha_j') \geq \\ \text{EFF}_l(\alpha_1 \beta_1^{a_1} \alpha_1' \alpha_2 \beta_2^{a_2} \alpha_2' \ldots \alpha_j \beta_j^{a_j} \alpha_j')$$

And moreover:

$$b_u \text{EFF}_l(\beta_u) \geq b_u$$

and for $t \in [u+1, j]$ we have:

$$b_t \text{EFF}_l(\beta_t) \geq -b_t m |\beta_t|$$

Hence:

$$\text{EFF}_l(\alpha_1 \beta_1^{a_1+b_1} \alpha_1' \alpha_2 \beta_2^{a_2+b_2} \alpha_2' \ldots \alpha_j \beta_j^{a_j+b_j}) = \\ \text{EFF}_l(\alpha_1 \beta_1^{a_1+b_1} \alpha_1' \alpha_2 \beta_2^{a_2+b_2} \alpha_2' \ldots \alpha_{s-1} \beta_{u-1}^{a_{u-1}+b_{u-1}} \alpha_u \beta_u^{a_u} \alpha_u' \ldots \alpha_j \beta_j^{a_j} \alpha_j') + \Sigma_{t=u}^{j} b_t \text{EFF}_l(\beta_t) \geq \\ \text{EFF}_l(\alpha_1 \beta_1^{a_1} \alpha_1' \alpha_2 \beta_2^{a_2} \alpha_2' \ldots \alpha_j \beta_j^{a_j} \alpha_j') + b_u - \Sigma_{t=u+1}^{j} b_t m |\beta_t| \geq \text{EFF}_l(\alpha_1 \beta_1^{a_1} \alpha_1' \alpha_2 \beta_2^{a_2} \alpha_2' \ldots \alpha_j \beta_j^{a_j} \alpha_j')$$

Point 2. If $\text{EFF}_l(\beta_n) \geq 0$ than because of point 1 we have:

$$\text{EFF}_l(\alpha_1\beta_1^{a_1+b_1}\alpha_1'\alpha_2\beta_2^{a_2+b_2}\alpha_2'\dots\alpha_n\beta_n^{a_n+b_n+i|\gamma|}) \geq \text{EFF}_l(\alpha_1\beta_1^{a_1}\alpha_1'\alpha_2\beta_2^{a_2}\alpha_2'\dots\alpha_n\beta_n^{a_n})$$

Hence, we can assume $\text{EFF}_l(\beta_n) < 0$. Therefore there has to be $j < n$ such that $\text{EFF}_l(\beta_j) > 0$. Let j be the maximal such. Observe, that:

$$\text{EFF}_l(\alpha_1\beta_1^{a_1+b_1}\alpha_1'\alpha_2\beta_2^{a_2+b_2}\alpha_2'\dots\alpha_j\beta_j^{a_j}\alpha_j'\beta_{j+1}^{a_{j+1}}\alpha_{j+1}'\dots\alpha_n\beta_n^{a_n}) \geq \text{EFF}_l(\alpha_1\beta_1^{a_1}\alpha_1'\alpha_2\beta_2^{a_2}\alpha_2'\dots\alpha_n\beta_n^{a_n})$$

$$b_j\text{EFF}_l(\beta_j) \geq b_j$$

And for each $s \in [j+1, n]$

$$b_s\text{EFF}_l(\beta_s) \geq b_s m|\beta_s|$$

Hence:

$$\begin{aligned}&\text{EFF}_l(\alpha_1\beta_1^{a_1+b_1}\alpha_1'\alpha_2\beta_2^{a_2+b_2}\alpha_2'\dots\alpha_n\beta_n^{a_n+b_n+i|\gamma|}) =\\ &\text{EFF}_l(\alpha_1\beta_1^{a_1+b_1}\alpha_1'\alpha_2\beta_2^{a_2+b_2}\alpha_2'\dots\alpha_j\beta_j^{a_j}\alpha_j'\beta_{j+1}^{a_{j+1}}\alpha_{j+1}'\dots\alpha_n\beta_n^{a_n}) + \Sigma_{s=j}^n b_s\text{EFF}_l(\beta_s) + i|\gamma|\text{EFF}_l(\beta_n) \geq\\ &\text{EFF}_l(\alpha_1\beta_1^{a_1}\alpha_1'\alpha_2\beta_2^{a_2}\alpha_2'\dots\alpha_n\beta_n^{a_n}) + b_j - \Sigma_{s=j+1}^n b_s m|\beta_s| - mi|\lambda||\beta_n| \geq \text{EFF}_l(\alpha_1\beta_1^{a_1}\alpha_1'\alpha_2\beta_2^{a_2}\alpha_2'\dots\alpha_n\beta_n^{a_n})\end{aligned}$$

Point 3. If $\text{EFF}_l(\gamma) \geq 0$ than the inequality follows from Point 2. Hence now we assume $\text{EFF}_l(\gamma) < 0$. Therefore there has to be $j \in [n]$ such that $\text{EFF}_l(\beta_j) > 0$. Let j be the maximal such. Hence: Observe, that:

$$\text{EFF}_l(\alpha_1\beta_1^{a_1+b_1}\alpha_1'\alpha_2\beta_2^{a_2+b_2}\alpha_2'\dots\alpha_j\beta_j^{a_j}\alpha_j'\dots\alpha_n\beta_n^{a_n}\lambda\gamma) \geq \text{EFF}_l(\alpha_1\beta_1^{a_1}\alpha_1'\alpha_2\beta_2^{a_2}\alpha_2'\dots\alpha_j\beta_j^{a_j}\alpha_j'\dots\alpha_n\beta_n^{a_n}\lambda\gamma)$$

$$b_j\text{EFF}_l(\beta_j) \geq b_j$$

And for $s \in [j+1, n]$ we have:

$$b_s\text{EFF}_l(\beta_s) \geq b_s m|\beta_s|$$

$$\begin{aligned}&\text{EFF}_l(\alpha_1\beta_1^{a_1+b_1}\alpha_1'\alpha_2\beta_2^{a_2+b_2}\alpha_2'\dots\alpha_n\beta_n^{a_n+b_n+i|\gamma|}\lambda\gamma^{1+(k+1-i)|\beta_n|}) =\\ &\text{EFF}_l(\alpha_1\beta_1^{a_1+b_1}\alpha_1'\alpha_2\beta_2^{a_2+b_2}\alpha_2'\dots\alpha_j\beta_j^{a_j}\alpha_j'\dots\alpha_n\beta_n^{a_n}\lambda\gamma) + b_j\text{EFF}_l(\beta_j)+\\ &\Sigma_{s=j+1}^n b_s\text{EFF}_l(\beta_s) + i|\gamma|\text{EFF}_l(\beta_n) + (k+1-i)|\beta_n|\text{EFF}_l(\gamma) \geq\\ &\text{EFF}_l(\alpha_1\beta_1^{a_1}\alpha_1'\alpha_2\beta_2^{a_2}\alpha_2'\dots\alpha_j\beta_j^{a_j}\alpha_j'\dots\alpha_n\beta_n^{a_n}\lambda\gamma) + b_j - \Sigma_{s=j+1}^n b_s m|\beta_s| - (k+1)m|\beta_n||\gamma| =\\ &\text{EFF}_l(\alpha_1\beta_1^{a_1}\alpha_1'\alpha_2\beta_2^{a_2}\alpha_2'\dots\alpha_j\beta_j^{a_j}\alpha_j'\dots\alpha_n\beta_n^{a_n}\lambda\gamma)\end{aligned}$$

◁

▷ Claim 27. For each $i \in [k+1]$ run ϕ_i reads the same word.

Proof. Let us fix $i, j \in [k+1]$ such that $i < j$. Observe, that $w(\phi_i) = w(\phi_j)$ if and only if $w(\beta_n^{(j-i)|\gamma|}\lambda)$ and $w(\lambda\gamma^{(j-i)|\beta_n|})$. We have, that $w(\beta_n), w(\alpha_n') \in L(a^*)$. Hence also $\gamma, \lambda \in L(a^*)$, as they are parts of α_n'. Therefore these two runs read the same word because

$$|\beta_n^{(j-i)|\gamma|}\lambda| = |\lambda\gamma^{(j-i)|\beta_n|}|$$ ◁

▷ Claim 28. For each $i, j \in [k+1]$ such that $i \neq j$ we have that $\phi_i \neq \phi_j$.

Proof. W.l.o.g. assume $i < j$. Observe, that it is enough to show, that

$$\beta_n^{(j-i)|\gamma|}\lambda \neq \lambda\gamma^{(j-i)|\beta_n|}$$

Assume, for the sake of contradiction that:

$$\beta_n^{(j-i)|\gamma|}\lambda = \lambda\gamma^{(j-i)|\beta_n|}$$

Observe, that β_n is not a prefix of $\lambda\gamma$, because then β_n would be a prefix of α'_n, which is not possible because of properties of π_2 (recall, that $\pi_2 = \alpha_n \beta_n^{a_n} \alpha'_n$ and β_n is not a prefix of α'_n). Hence $\lambda\gamma$ is a prefix of β_n.

Hence for all $l \in [d]$ we have:

$$\text{EFF}_l(\beta_n^{(j-i)|\gamma|}\lambda) = \text{EFF}_l(\lambda\gamma^{(j-i)|\beta_n|})$$

Hence $\text{support}(\beta_n) = \text{support}(\gamma)$. Because $\lambda\gamma$ is a prefix of β_n and $\lambda\gamma \neq \beta_n$ this contradicts the minimality of $|\alpha_n\beta_n|$. ◁

Since we have proven Claims 26, 27 and 28 we know that runs ϕ_i are all accepting, different and read the same word. This concludes the proof of Lemma 7. ◀