Unboundedness problems for languages of vector addition systems

Wojciech Czerwiński[1]
University of Warsaw, Poland
https://orcid.org/0000-0002-6169-868X

Piotr Hofman[2]
University of Warsaw, Poland
https://orcid.org/0000-0001-9866-3723

Georg Zetzsche[3]
IRIF (Université Paris-Diderot & CNRS), France
https://orcid.org/0000-0002-6421-4388

Abstract

A vector addition system (VAS) with an initial and a final marking and transition labels induces a language. In part because the reachability problem in VAS remains far from being well-understood, it is difficult to devise decision procedures for such languages. This is especially true for checking properties that state the existence of infinitely many words of a particular shape. Informally, we call these *unboundedness properties*.

We present a simple set of axioms for predicates that can express unboundedness properties. Our main result is that such a predicate is decidable for VAS languages as soon as it is decidable for regular languages. Among other results, this allows us to show decidability of (i) separability by bounded regular languages, (ii) unboundedness of occurring factors from a language K with mild conditions on K, and (iii) universality of the set of factors.

2012 ACM Subject Classification Theory of computation → Concurrency, Theory of computation → Formal languages and automata theory

Keywords and phrases vector addition systems, decision problems, unboundedness, separability

Digital Object Identifier 10.4230/LIPIcs.ICALP.2018.191

Acknowledgements We are indebted to Mohamed Faouzi Atig for suggesting to study separability by bounded languages, which was the starting point for this work. We are also grateful to Sławomir Lasota and Sylvain Schmitz for important discussions. This collaboration started at the Gregynog 71717 workshop organized by Ranko Lazić and Patrick Totzke.

1 Introduction

Vector addition systems (VAS) and, essentially equivalent, Petri nets are among the most widely used models of concurrent systems. Although they are used extensively in practice, there are still fundamental questions that are far from being well understood.

[1] Partially supported by the Polish National Science Centre grant 2016/21/D/ST6/01376
[2] Partially supported by the Polish National Science Centre grant 2016/21/B/ST6/01505
[3] Supported by a fellowship of the Fondation Sciences Mathématiques de Paris and partially funded by the DeLTA project (ANR-16-CE40-0007). Part of the research was conducted when the author was supported by a fellowship within the Postdoc-Program of the German Academic Exchange Service (DAAD) and by Labex DigiCosme, Univ. Paris-Saclay, project VERICONISS.

This is reflected in what we know about decidability questions regarding the most expressive class of languages associated to VAS: The languages of (arbitrarily) labeled VAS with a given initial and final configuration, which we call *VAS languages*. In the 1970s, this class has been characterized in terms of closure properties and Dyck languages by Greibach [27] and Jantzen [36]. Almost all decidability results about these languages use a combination of these closure properties and the decidability of the reachability problem for VAS [43] (or for Reinhardt's extension [47], such as in [1, 51]). Of course, this method is confined to procedures that somehow reduce to the existence of one or finitely many runs of VAS.

There are two notable exceptions to this and they both rely on an extensions of decision procedures for reachability in VAS. The first is Hauschildt and Jantzen's result [33] from 1994 that finiteness of VAS languages is decidable, which employs Hauschildt's algorithm to decide semilinearity of reachability sets [32]. The second is the much more recent result of Habermehl, Meyer, and Wimmel from 2010 [28]. It shows computability of downward closures for VAS languages (which significantly generalizes decidability of finiteness) and involves a careful inspection of marked graph-transition sequences (MGTS) of Lambert's algorithm. (As observed in [15, 52], computability of downward closures can also be derived from [6].) This sparsity of results is due to the fact that the algorithms for the reachability problem are still quite unwieldy and have been digested by few researchers.

In particular, it currently seems difficult to decide whether there exist infinitely many words of some shape in a given language—unless the problem reduces to computing downward closures. Informally, we call problems of this type *unboundedness problems*. Such problems are important for two reasons. The first concerns *separability problems*, which have attracted attention in recent years [5, 11, 26, 44, 45]. Here, instead of deciding whether two languages are disjoint, we are looking for a (typically finite-state) certificate for disjointness, namely a set that includes one language and is disjoint from the other. For general topological reasons, inseparability is usually witnessed by a common pattern, whose presence in a language is an unboundedness property. The second reason is that unboundedness problems tend to be *decidable where exact queries are not*. This phenomenon also occurs in the theory of regular cost functions [12]. Moreover, as it turns out in this work, this is true for VAS languages as well.

Contribution We present a simple notion of an *unboundedness predicate* on languages and show that such predicates are decidable for VAS languages as soon as they are decidable for regular languages. On the one hand, this provides an easy and general way to obtain new decidability results for VAS languages without the need to understand the details of the KLMST decomposition. On the other hand, we apply this framework to prove:

(i) Boundedness in the sense of Ginsburg and Spanier [24] is decidable. A language $L \subseteq \Sigma^*$ is *bounded* if $L \subseteq w_1^* \cdots w_n^*$ for some $w_1, \ldots, w_n \in \Sigma^*$. Moreover, it is decidable whether two given VAS languages are separable by a bounded regular language.

(ii) Computability of downward closures can be recovered as well.

(iii) Suppose that $K \subseteq \Sigma^*$ is chosen so that it is decidable whether K intersects a given regular language. Then, it is decidable for a given VAS language L whether L contains words with arbitrarily many factors from K. Moreover, in case the number of factor occurrences in L is bounded, we can even compute an upper bound.

(iv) Under the same assumptions as above on $K \subseteq \Sigma^*$, one can decide if every word from K^* appears as a factor of a given VAS language $L \subseteq \Sigma^*$. In particular, it is decidable whether L contains every word from Σ^* as a factor.

It should be stressed that results (iii) and (iv) came deeply unexpected to the authors for two reasons. First, the assumptions are already satisfied when K is induced by a system model as powerful as well-structured transition systems (WSTS) or higher-order recursion schemes. In these cases, it is in general *undecidable* whether a given VAS language contains a factor from K *at least once*, because intersection emptiness easily reduces to this problem (see the remarks after Theorem 4.8). We therefore believe that these results might lead to new approaches to verifying systems with concurrency and (higher-order) recursion, where the latter undecidability (or the unknown status in the case of simple recursion [40]) is usually a barrier for decision procedures. Secondly, the problems in (iii) and (iv) are undecidable as soon as L is just slightly beyond the realm of VAS: Already for one-counter languages L, both (iii) and (iv) become undecidable. Thus, compared to other infinite-state systems, VAS languages turn out to be extraordinarily amenable to unboundedness problems.

Related work Other authors have investigated general notions of unboundedness properties for VAS [2, 4, 16, 50], usually with the goal of obtaining EXPSPACE upper bounds. However, those properties a priori concern the state space itself. While they can sometimes be used to reason about languages [4, 16], this has been confined to coverability languages, which are significantly less expressive than the reachability languages studied here. Specifically, every problem we consider here is hard for the reachability problem (see Remark 3.2).

An early attempt was Yen's work [50], which claimed an EXPSPACE upper bound for a powerful logic concerning paths in VAS. Unfortunately, a serious flaw in the latter was discovered by Atig and Habermehl [2], who presented a corrected proof for a restricted version of Yen's logic. Demri [16] then introduced a notion of *generalized unboundedness properties*, which covers more properties from Yen's logic and proved an EXPSPACE procedure to check them. Examples include reversal-boundedness, place boundedness, and regularity of firing sequences of unlabeled VAS. Finally, Blockelet and Schmitz [4] introduce an extension of computation tree logic (CTL) that can express "coverability-like properties" of VAS. The authors prove an EXPSPACE upper bound for model checking this logic on VAS.

2 Preliminaries

Let Σ be a finite alphabet. For $w \in \Sigma^*$, we denote its length by $|w|$. The i-th letter of w, for $i \in [1, |w|]$ is denoted $w[i]$. Moreover, we write $\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$. A *(d-dimensional) vector addition system* (VAS) V consists of finite set of *transitions* $T \subseteq \mathbb{Z}^d$, *source* and *target* vectors $s, t \in \mathbb{N}^d$ and a *labeling* $h\colon T \to \Sigma_\varepsilon$, whose extension to a morphism $T^* \to \Sigma^*$ is also denoted h. Vectors $v \in \mathbb{N}^d$ are also called *configurations*. A transition $t \in T$ can be *fired* in a configuration $v \in \mathbb{N}^d$ if $v + t \in \mathbb{N}^d$. Then, the result of firing t is the configuration $v + t$ and we write $v \xrightarrow{h(t)} v'$ for $v' = v + t$. For $w \in \Sigma^*$, we write $v \xrightarrow{w} v'$ if there exist $v_1, \ldots, v_k \in \mathbb{N}^d$ such that $v = v_0 \xrightarrow{x_1} v_1 \xrightarrow{x_2} \ldots \xrightarrow{x_k} v_k \xrightarrow{x_{k+1}} v_{k+1} = v'$, where $w = x_1 \cdots x_{k+1}$ for some $x_1, \ldots, x_{k+1} \in \Sigma_\varepsilon$. The *language* of V, denoted $L(V)$, is the set of all labels of runs from source to target, i.e. $L(V) = \{w \in \Sigma^* \mid s \xrightarrow{w} t\}$. The languages of the form $L(V)$ for VAS V are called *VAS languages*. A word $u = a_1 \cdots a_n$ with $a_i \in \Sigma$ is a *subword* of a word $v \in \Sigma^*$ if $v \in \Sigma^* a_1 \Sigma^* \cdots \Sigma^* a_n \Sigma^*$, which is denoted $u \preceq v$. For a language $L \subseteq \Sigma^*$ its *downward closure* is the language $L{\downarrow} = \{u \in \Sigma^* \mid \exists v \in L\colon u \preceq v\}$. It is known that $L{\downarrow}$ is regular for every $L \subseteq \Sigma^*$ [34, 31]. A *language class* is a collection of languages, together with some way of finitely describing these languages (such as by grammars, automata, etc.). If $\mathcal{C}$ is a language class so that given a description of a language L from $\mathcal{C}$, we can compute an automaton for $L{\downarrow}$, we say that *downward closures are computable* for $\mathcal{C}$.

A *full trio* is a language class that is effectively closed under rational transductions [3], which are relations defined by nondeterministic two-tape automata. Equivalently, a full trio is a class that is effectively closed under morphisms, inverse morphisms, and regular intersection [3]. Full trios are abundant among infinite-state models: If a nondeterministic machine model involves a finite-state control, the resulting language class is a full trio. Examples include *VAS languages* [36], *coverability languages of WSTS* [23], *one-counter languages* (which are accepted by one-counter automata *with zero tests*) [35], and languages of *higher-order pushdown automata* [42] and *higher-order recursion schemes* [30]. The context-sensitive languages do not constitute a full trio, as they are not closed under erasing morphisms.

3 Main result

Here, we introduce our notion of unboundedness predicates and present our main result.

For didactic purposes, we begin our exposition of unboundedness predicates with a simplified (but already useful) version. An important aspect of the definition is that technically, an unboundedness predicates is not a property of the language $L \subseteq \Sigma^*$ we want to analyze, but of the set of its factors. In other words, we have a unary predicate $\mathfrak{p}$ on languages and we want to decide whether $\mathfrak{p}(F(L))$, where $F(L) = \{w \in \Sigma^* \mid L \cap \Sigma^* w \Sigma^* \neq \emptyset\}$ is the set of factors of L. For the definition, it is helpful to keep in mind the simplest example of an unboundedness predicate, the *infinity predicate* $\mathfrak{p}_{\mathsf{inf}}$, where $\mathfrak{p}_{\mathsf{inf}}(K)$ if and only if K is infinite. Then, $\mathfrak{p}_{\mathsf{inf}}(F(L))$ if and only if L is infinite. A unary predicate $\mathfrak{p}$ on languages over Σ^* is called *1-dimensional unboundedness predicate* if for every $K, L \subseteq \Sigma^*$, we have:

(i*) if $\mathfrak{p}(K)$ and $K \subseteq L$, then $\mathfrak{p}(L)$.
(ii*) if $\mathfrak{p}(K \cup L)$, then either $\mathfrak{p}(L)$ or $\mathfrak{p}(K)$.
(iii*) if $\mathfrak{p}(F(KL))$, then either $\mathfrak{p}(F(K))$ or $\mathfrak{p}(F(L))$.

Part of our result will be that for such predicates, if we can decide whether $\mathfrak{p}(F(R))$ for regular languages R, we can decide whether $\mathfrak{p}(F(L))$ for VAS languages L. Before we come to that, we want to generalize a bit. There are predicates we want to decide that fail to satisfy axiom (iii*), such as the one stating $a^*b^* \subseteq L{\downarrow}$ for $L \subseteq \Sigma^*$: It is satisfied for a^*b^*, but neither for a^* nor for b^*. (Deciding such predicates is useful for computing downward closures [52] and separability by piecewise testable languages [15]) To capture such predicates, which intuitively ask for several quantities being unbounded simultaneously, we present a more general set of axioms. Here, the idea is to formulate predicates over simultaneously occurring factors. For a language $L \subseteq \Sigma^*$ and $n \in \mathbb{N}$, let

$$F_n(L) = \{(w_1, \ldots, w_n) \in (\Sigma^*)^n \mid \Sigma^* w_1 \Sigma^* \cdots w_n \Sigma^* \cap L \neq \emptyset\}.$$

We will speak of *n-dimensional* predicates, i.e., predicates $\mathfrak{p}$ on subsets of $(\Sigma^*)^n$, and we want to decide whether $\mathfrak{p}(F_n(L))$ for a given language L. The following are axioms referring to all subsets $S, T \subseteq (\Sigma^*)^n$, languages $L_i \subseteq \Sigma^*$, and all $k \in \mathbb{N}$. We call $\mathfrak{p}$ an *(n-dimensional) unboundedness predicate* if

(i) if $\mathfrak{p}(S)$ and $S \subseteq T$, then $\mathfrak{p}(T)$.
(ii) if $\mathfrak{p}(S \cup T)$, then $\mathfrak{p}(S)$ or $\mathfrak{p}(T)$.
(iii) if $\mathfrak{p}(F_n(L_1 \cdots L_k))$, then $n = n_1 + \cdots + n_k$ such that $\mathfrak{p}(F_{n_1}(L_1) \times \cdots \times F_{n_k}(L_k))$.

Intuitively, the last axiom says that if a concatenation satisfies the predicate, then this is already witnessed by factors in at most n participants of the concatenation. Note that for $n = 1$, the axioms coincide with the simplified axioms (i*) to (iii*) above. An n-dimensional

unboundedness predicate $\mathfrak{p}$ is *decidable for a language class* $\mathcal{C}$ if, given a language L from $\mathcal{C}$, it is decidable whether $\mathfrak{p}(F_n(L))$. The following is our main result.

▶ **Theorem 3.1.** *Given a VAS language $L \subseteq \Sigma^*$, one can compute a regular $R \subseteq \Sigma^*$ such that $L \subseteq R$ and for every n-dim. unboundedness predicate $\mathfrak{p}$, we have $\mathfrak{p}(F_n(L))$ iff $\mathfrak{p}(F_n(R))$.*

Note that this implies that decidability of $\mathfrak{p}$ for regular languages implies decidability of $\mathfrak{p}$ for VAS languages for any n-dim. unboundedness predicate $\mathfrak{p}$. In addition, when our unboundedness predicate expresses that a certain quantity is unbounded, then in the bounded case, Theorem 3.1 sometimes allows us to compute an upper bound (see, e.g. Theorem 4.8).

▶ Remark 3.2. Let us comment on the complexity of deciding whether $\mathfrak{p}(F_n(L))$ for a VAS language L. Call $\mathfrak{p}$ *non-trivial* if there is at least one $S \subseteq (\Sigma^*)^n$ that satisfies $\mathfrak{p}$ and least one $S' \subseteq (\Sigma^*)^n$ for which $\mathfrak{p}$ is not satisfied. Then, deciding whether $\mathfrak{p}(F_n(L))$ is at least as hard as the reachability problem. Indeed, in this case axiom (i) implies that $F_n(\Sigma^*) = (\Sigma^*)^n$ satisfies $\mathfrak{p}$, but $F_n(\emptyset) = \emptyset$ does not. Given a VAS V and two vectors μ_1 and μ_2, it is easy to construct a VAS V' so that $L(V') = \Sigma^*$ if V can reach μ_2 from μ_1 and $L(V') = \emptyset$ otherwise.

4 Applications

Bounded languages Our first application concerns bounded languages. A language $L \subseteq \Sigma^*$ is *bounded* if there exist words $w_1, \ldots, w_n \in \Sigma^*$ such that $L \subseteq w_1^* \cdots w_n^*$. This notion was introduced by Ginsburg and Spanier [24]. Since a bounded language as above can be characterized by the set of vectors $(x_1, \ldots, x_n) \in \mathbb{N}^n$ for which $w_1^{x_1} \cdots w_n^{x_n} \in L$, bounded languages are quite amenable to analysis. This has led to a number of applications to concurrent recursive programs [18, 19, 20, 21, 41], but also counter systems [17] and WSTS [9].

Boundedness has been shown decidable for *context-free languages* by Ginsburg and Spanier [24] (PTIME-completeness by Gawrychowski et al. [22]) and hence also for *regular languages* (NL-completeness also in [22]), for *equal matrix languages* by Siromoney [49], and for trace languages of *complete deterministic well-structured transition systems* by Chambart et al. [9]. The latter implies that boundedness is decidable for coverability languages of deterministic vector addition systems, in which case EXPSPACE-completeness was shown by Chambart et al. [9] (the upper bound had been established by Blockelet and Schmitz [4]).

We use Theorem 3.1 to show the following.

▶ **Theorem 4.1.** *Given a VAS, it is decidable whether its language is bounded.*

The rest of this section is devoted to the proof of Theorem 4.1. Let $\mathfrak{p}_{\mathsf{notb}}$ be the 1-dimensional predicate that holds for a language $K \subseteq \Sigma^*$ if and only if K it is not bounded. We plan to apply Theorem 3.1 to $\mathfrak{p}_{\mathsf{notb}}$, but it allows us to decide only whether $\mathfrak{p}_{\mathsf{notb}}(F(L))$ for a given VAS language L. Thus we need the following fact, which we prove in a moment.

▶ **Fact 4.2.** *A language $L \subseteq \Sigma^*$ is bounded if and only if $F(L)$ is bounded.*

We also need to show that $\mathfrak{p}_{\mathsf{notb}}$ is indeed an unboundedness predicate, meaning that it satisfies axioms (i*) to (iii*). By definition of boundedness, $\mathfrak{p}_{\mathsf{notb}}$ clearly fulfills axiom (i*). Axioms (ii*) and (iii*) are implied by Fact 4.2 and the following.

▶ **Fact 4.3.** *If K and L are bounded then both $K \cup L$ and KL are bounded as well.*

Let us prove Facts 4.2 and 4.3 and begin with Fact 4.3. If K and L are bounded, then $K \subseteq w_1^* \cdots w_n^*$ and $L \subseteq w_{n+1}^* \cdots w_m^*$ for some $w_1, \ldots, w_m \in \Sigma^*$. Then we have $K \cup L, KL \subseteq w_1^* \cdots w_m^*$, which shows Fact 4.3. In order to show Fact 4.2, observe first

that for each individual word $w \in \Sigma^*$, the language $F(w)$ is bounded because it is finite. Thus, if $L \subseteq w_1^* \cdots w_n^*$, then $F(L)$ is included in $F(w_1)w_1^*F(w_1) \cdots F(w_n)w_n^*F(w_n)$, which is bounded as a concatenation of bounded languages by Fact 4.3. Thus, $F(L)$ is bounded as well. Conversely, L inherits boundedness from its superset $F(L)$.

To conclude Theorem 4.1, we need to show that given regular language $R \subseteq \Sigma^*$, it is decidable whether $\mathfrak{p}_{\mathsf{notb}}(F(R))$. By Fact 4.2, this amounts to checking whether R is bounded. This is decidable even for context-free languages [24] (and in NL for regular ones [22]).

Separability We can also use our results to decide whether two VAS languages are separable by a bounded regular language. Very generally, if $\mathcal{S}$ is a class of sets, we say that a set K is *separable from* a set L *by* a set from $\mathcal{S}$ if there is a set S in $\mathcal{S}$ so that $K \subseteq S$ and $L \cap S = \emptyset$.

The separability problem was recently investigated for VAS languages and several subclasses. In [15] it is shown that separability of VAS languages by piecewise testable languages (a subclass of regular languages) is decidable. Decidability of separability of VAS languages by regular languages is still open, but it is known for several subclasses thereof [10, 11, 13]. In [14] it is shown that any two disjoint VAS coverability languages are separable by a regular language. Here, using Theorem 4.1 we are able to show the following.

▶ **Theorem 4.4.** *Given two VAS languages K and L, it is decidable whether K is separable from L by a bounded regular language.*

Clearly, in order for that to hold, K has to be bounded, which we can decide. Moreover, by enumerating expressions $w_1^* \cdots w_n^*$, we can find one with $K \subseteq w_1^* \cdots w_n^*$. Since the bounded regular languages (BRL) are closed under intersection (recall that a subset of a bounded language is again bounded), K and L are separable by a BRL if and only if $L_0 = K$ and $L_1 = L \cap w_1^* \cdots w_n^*$ are separable by a BRL. Since now both input languages are included in $w_1^* \cdots w_n^*$, we can reformulate the problem into one over vector sets.

▶ **Lemma 4.5.** *Let $L_0, L_1 \subseteq w_1^* \cdots w_n^*$ and $U_i = \{(x_1, \ldots, x_n) \in \mathbb{N}^n \mid w_1^{x_1} \cdots w_n^{x_n} \in L_i\}$ for $i \in \{0, 1\}$. Then, L_0 is separable from L_1 by a BRL if and only if U_0 is separable from U_1 by a recognizable subset of $\mathbb{N}^n$.*

Recall that a subset $S \subseteq \mathbb{N}^n$ is *recognizable* if there is a morphism $\varphi\colon \mathbb{N}^n \to F$ into a finite monoid F with $S = \varphi^{-1}(\varphi(S))$. Lemma 4.5 is a straightforward application of Ginsburg and Spanier's characterization of BRL [25].

Since in our case, L_0 and L_1 are VAS languages, a standard construction shows that U_0 and U_1 are (effectively computable) sections of VAS reachability sets. Here, sections are defined as follows. For a subset $I \subseteq [1, n]$, let $\pi_I\colon \mathbb{N}^n \to \mathbb{N}^{|I|}$ be the projection onto the coordinates in I. Then, every set of the form $\pi_{[1,n]\setminus I}(S \cap \pi_I^{-1}(x))$ for some $I \subseteq [1, n]$ and $x \in \mathbb{N}^{|I|}$ is called a *section* of $S \subseteq \mathbb{N}^n$. Thus, the following result by Clemente et al. [11] allows us to decide separability by BRL.

▶ **Theorem 4.6** ([11])**.** *Given two sections $S_0, S_1 \subseteq \mathbb{N}^n$ of reachability sets of VAS, it is decidable whether S_0 is separable from S_1 by a recognizable subset of $\mathbb{N}^n$.*

Downward closures and simultaneous unboundedness We now illustrate how to compute downward closures using our results. First of all, computability of downward closures for VAS languages follows directly from Theorem 3.1 because it implies $R{\downarrow} = L{\downarrow}$: For each word $w = a_1 \cdots a_n$ with $a_1, \ldots, a_n \in \Sigma$, consider the n-dimensional predicate $\mathfrak{p}_w$ which is satisfied for $S \subseteq (\Sigma^*)^n$ iff $(a_1, \ldots, a_n) \in S$. Then $\mathfrak{p}_w(F_n(L))$ if and only if $w \in L{\downarrow}$. It is easy to check that this is an unboundedness predicate. Hence, $R{\downarrow} = L{\downarrow}$.

However, in order to illustrate how to apply unboundedness predicates, we present an alternative approach. In [52], it was shown that if a language class $\mathcal{C}$ is closed under rational transductions (which is the case for VAS languages), then downward closures are computable for $\mathcal{C}$ if and only if, given a language L from $\mathcal{C}$ and letters $a_1, \ldots, a_n$, it is decidable whether $a_1^* \cdots a_n^* \subseteq L{\downarrow}$. Let us show how to decide the latter using unboundedness predicates.

For this, we use an n-dimensional predicate. For a subset $S \subseteq (\Sigma^*)^n$, let $S{\downarrow}$ be the set of all tuples $(u_1, \ldots, u_n) \in (\Sigma^*)^n$ such that there is some $(v_1, \ldots, v_n) \in S$ with $u_i \preceq v_i$ for $i \in [1, n]$. Our predicate $\mathfrak{p}_{\mathsf{sup}}$ is satisfied for $S \subseteq (\Sigma^*)^n$ if and only if $a_1^* \times \cdots \times a_n^* \subseteq S$. Then clearly $\mathfrak{p}_{\mathsf{sup}}(F_n(L))$ if and only if $a_1^* \cdots a_n^* \subseteq L{\downarrow}$. It is easy to check that $\mathfrak{p}_{\mathsf{sup}}$ fulfills axiom (i) and axiom (ii). For the latter, note that $a_1^* \times \cdots \times a_n^* \subseteq (S_1 \cup S_2){\downarrow}$ implies that for some $j \in \{1, 2\}$, there are infinitely many $\ell \in \mathbb{N}$, with $(a_1^\ell, \ldots, a_n^\ell) \in S_j$ and hence $a_1^* \times \cdots \times a_n^* \subseteq S_j{\downarrow}$. For axiom (iii), we need a simple combinatorial argument:

▶ **Lemma 4.7.** *If $a_1^* \times \cdots \times a_n^* \subseteq F_n(L_1 \cdots L_k){\downarrow}$, then $n = n_1 + \cdots + n_k$ with $a_1^* \times \cdots \times a_n^* \subseteq (F_{n_1}(L_1) \times \cdots \times F_{n_k}(L_k)){\downarrow}$.*

It remains to show that for a regular language R, it is decidable whether $a_1^* \cdots a_n^* \subseteq R{\downarrow}$. Since it is easy to construct an automaton for $R{\downarrow}$, this amounts to a simple inclusion check.

Non-overlapping factors Our next example shows that under very mild assumptions on a language K, one can decide whether the words in a VAS language L contain arbitrarily many factors from K. For $w \in \Sigma^*$ and $K \subseteq \Sigma^+$, let $|w|_K$ be the largest number m such that there are $w_1, \ldots, w_m \in K$ with $(w_1, \ldots, w_m) \in F_m(w)$. Note that since $\varepsilon \notin K$, there is always a maximal such m. Consider the function $f_K \colon \Sigma^* \to \mathbb{N}$, $w \mapsto |w|_K$. A function $f \colon \Sigma^* \to \mathbb{N}$ is *unbounded* on $L \subseteq \Sigma^*$ if for every $k \in \mathbb{N}$, we have $f(w) \geq k$ for some $w \in L$.

▶ **Theorem 4.8.** *If $\mathcal{C}$ is a full trio with decidable emptiness problem, then given a VAS language L and a language $K \subseteq \Sigma^+$ from $\mathcal{C}$, it is decidable whether f_K is unbounded on L. If f_K is bounded on L, we can compute an upper bound.*

Theorem 4.8 is quite unexpected because very slight variations lead to undecidability. If we ask whether f_K is non-zero on a given VAS language (as opposed to unbounded), then this is in general undecidable. Indeed, suppose $\mathcal{C}$ is a full trio for which intersection with VAS languages is undecidable (such as languages of lossy channel systems[4] or higher-order pushdown languages [29, 52]). Then given a language $K \subseteq \Sigma^*$ from $\mathcal{C}$, a VAS language L and some $c \notin \Sigma$, the function f_{cKc} is non-zero on cLc if and only if $K \cap L \neq \emptyset$.

Furthermore, the same problem becomes undecidable in general if instead of VAS languages, we want to decide the problem for a language class as simple as one-counter languages (OCL). Indeed, suppose $\mathcal{C}$ is a full trio for which intersection with OCL is undecidable (such as the class of OCL). For a given $K \subseteq \Sigma^*$ from $\mathcal{C}$, an OCL $L \subseteq \Sigma^*$, and some $c \notin \Sigma$, the set $c(Lc)^*$ is effectively an OCL and f_{cKc} is unbounded on $c(Lc)^*$ if and only if $K \cap L \neq \emptyset$.

Let us prove Theorem 4.8. Fix a language $K \subseteq \Sigma^*$ from $\mathcal{C}$. Our predicate $\mathfrak{p}_{\mathsf{nof}}$ is one-dimensional and is satisfied on a set $L \subseteq \Sigma^*$ if and only if f_K is unbounded on L. Then clearly, $\mathfrak{p}_{\mathsf{nof}}(F(L))$ if and only if f_K is unbounded on L. It is immediate that axioms (i*) and (ii*) are satisfied. Furthermore, axiom (iii*) follows by contraposition: If neither $\mathfrak{p}_{\mathsf{nof}}(F(L_0))$ nor $\mathfrak{p}_{\mathsf{nof}}(F(L_1))$, then there are $B_0, B_1 \in \mathbb{N}$ such that f_K is bounded by

[4] It seems to be folklore that intersection between languages of lossy channel systems and languages of one-dimensional VAS is undecidable (the additional counter can be used to ensure that no letter is dropped). The only reference we could find is [46].

B_i on L_i for $i = 0, 1$. That implies that f_K is bounded by $B_0 + B_1 + 1$ on L_0L_1. This rules out $\mathfrak{p}_{\mathsf{nof}}(F(L_0L_1))$, which establishes axiom (iii*). The following uses standard arguments.

▶ **Lemma 4.9.** *Let $\mathcal{C}$ be a full trio with decidable emptiness problem. Given a language K from $\mathcal{C}$ and a regular language R, it is decidable whether f_K is unbounded on R. Moreover, if f_K is bounded on R, we can compute an upper bound.*

We can deduce Theorem 4.8 from Lemma 4.9 as follows. Using Theorem 3.1, we compute the language R. Then, f_K is unbounded on R iff it is unbounded on L. Moreover, an upper bound for f_K on R is also an upper bound for f_K on L because $L \subseteq R$.

Counting automata To illustrate how these results can be used, we formulate an extension of Theorem 4.8 in terms of automata that can count. Let $\mathcal{C}$ be a full trio. Intuitively, a $\mathcal{C}$-counting automaton can read a word produced by a VAS and can use machines corresponding to $\mathcal{C}$ as oracles. Just like the intersection of two languages that describe threads in a concurrent system signals a safety violation [7, 8, 41], a successful oracle call would signal a particular undesirable event. In such a model, it would be undecidable whether any oracle call can be successful if, for example, $\mathcal{C}$ is the class of higher-order pushdown languages. However, we show that it *is* decidable whether such an automaton can make an unbounded number of successful oracle calls and if not, compute an upper bound. Hence, we can decide if the number of undesirable events is bounded and, if so, provide a bound.

A *$\mathcal{C}$-counting automaton* is a tuple $\mathcal{A} = (Q, \Sigma, \Gamma, C, q_0, E, Q_f)$, where Q is a finite set of *states*, Σ is its *input alphabet*, Γ is its *(oracle) tape alphabet*, C is a finite set of *counters*, $q_0 \in Q$ is its *initial state*, $Q_f \subseteq Q$ is its set of *final states*, and $E \subseteq Q \times \Sigma^* \times (\Omega \cup \{\varepsilon\}) \times Q$ is a finite set of *edges*, where Ω is a set of *operations* of the following form. First, we have an operation $\mathsf{push}(a)$ for each $a \in \Gamma$, which appends a to the oracle tape. Moreover, we have $\mathsf{check}(K, c)$ for each $K \subseteq \Gamma^*$ from $\mathcal{C}$ and each $c \in C$, which first checks whether the current tape content belongs to K and if so, increments the counter c. After the oracle query, it empties the oracle tape, regardless of whether the oracle answers positively or negatively.

A *configuration* of $\mathcal{A}$ is a triple (q, u, μ), where $q \in Q$ is the current state, $u \in \Gamma^*$ is the oracle tape content, and $\mu \in \mathbb{N}^C$ describes the counter values. For a label $x \in \Sigma \cup \{\varepsilon\}$, and configurations $(q, u, \mu), (q', u', \mu')$, we write $(q, u, \mu) \xrightarrow{x} (q', u', \mu')$ if (q', u', μ') results from (q, u, μ) as described above. In the general case $w \in \Sigma^*$, $(q, u, \mu) \xrightarrow{w} (q', u', \mu')$ has the obvious meaning. We extend the set of natural numbers $\mathbb{N}$ by setting $\overline{\mathbb{N}} = \mathbb{N} \cup \{\omega\}$, where ω is the first infinite ordinal number and represents the infinity. $\mathcal{A}$ defines a function $\Sigma^* \to \overline{\mathbb{N}}$ with $\mathcal{A}(w) = \sup \left\{ \inf_{c \in C} \mu(c) \mid \mu \in \mathbb{N}^C,\ (q_0, \varepsilon, 0) \xrightarrow{w} (q, u, \mu) \text{ for some } q \in Q_f,\ u \in \Gamma^* \right\}$, where 0 is the zero vector in $\mathbb{N}^C$. Hence, $\mathcal{A}$ is unbounded on L if for every $k \in \mathbb{N}$, there is a $w \in L$ and a run of $\mathcal{A}$ on w in which for each $c \in C$, at least k of the oracle queries for c are successful. The following can be shown similarly to Theorem 4.8, but using a multi-dimensional unboundedness predicate.

▶ **Theorem 4.10.** *Let $\mathcal{C}$ be a full trio with decidable emptiness. Given a VAS language L and a $\mathcal{C}$-counting automaton $\mathcal{A}$, it is decidable whether $\mathcal{A}$ is unbounded on L. Moreover, if $\mathcal{A}$ is bounded on L, then one can compute an upper bound $B \in \mathbb{N}$ for $\mathcal{A}$ on L.*

Factor inclusion As a last example, we show how our results can be used to decide inclusion problems. Specifically, given a VAS language $L \subseteq \Sigma^*$, it is decidable whether $\Sigma^* \subseteq F(L)$. In fact, we show a more general result:

▶ **Theorem 4.11.** *If $\mathcal{C}$ is a full trio with decidable emptiness problem, then given a VAS language L and a language K from $\mathcal{C}$, it is decidable whether $K^* \subseteq F(L)$.*

Here, $\Sigma^* \subseteq F(L)$ is the special case where $K = \Sigma$. Recall that is is undecidable whether $L = \Sigma^*$ for VAS languages and for one-counter languages (OCL) (e.g. [15, Lemma 6.1]).

Similar to Theorem 4.8, deciding whether $\Sigma^* \subseteq F(L)$ is already undecidable for OCL L: For a given OCL $L \subseteq \Sigma^*$, pick a letter $c \notin \Sigma$ and note that $L' = c(Lc)^* \subseteq (\Sigma \cup \{c\})^*$ is effectively an OCL and $(\Sigma \cup \{c\})^* \subseteq F(L')$ if and only if $L = \Sigma^*$. Also, under the assumptions of the theorem, it is undecidable whether $K \subseteq F(L)$: If $L \subseteq \Sigma^*$ and $c \notin \Sigma$, then $c\Sigma^* c \subseteq F(cLc)$ if and only if $L = \Sigma^*$ (every full trio contains the regular set $c\Sigma^* c$).

Let us see how Theorem 4.11 follows from Theorem 3.1. Fix a language K from $\mathcal{C}$. We use the 1-dim. predicate $\mathfrak{p}_{\mathsf{fu}}$, which is satisfied on a set $L \subseteq \Sigma^*$ if and only if $K^* \subseteq F(L)$. Of course, axiom (i) holds by definition. Axiom (iii) follows by contraposition: Suppose that $K^* \subseteq F(L_1 L_2)$ and $K^* \not\subseteq F(L_1)$ with some $u \in K^* \setminus F(L_1)$. Let $v \in K^*$ be arbitrary. Then, since $K^* \subseteq F(L_1 L_2)$, we have $uv \in F(L_1 L_2)$. This means, there are $x, y \in \Sigma^*$ with $xuvy \in L_1 L_2$. Hence, we have $xuvy = w_1 w_2$ for some $w_i \in L_i$ for $i = 1, 2$. Then $|w_1| < |xu|$, because otherwise u would belong to $F(L_1)$. Therefore, v is a factor of w_2 and thus $v \in F(L_2)$. Hence, $K^* \subseteq F(L_2)$. Of course, a similar argument works if $K^* \subseteq F(L_1 L_2)$ and $K^* \not\subseteq F(L_2)$. This proves axiom (iii). Axiom (ii) can be shown the same way. Thus, by Theorem 3.1, it suffices to decide whether $K^* \subseteq F(R)$ for regular R, which follows from $\mathcal{C}$ being a full trio and having decidable emptiness (see Lemma E.1).

5 Proof of the main result

We prove our decidability result using the KLMST decomposition. More specifically, we show a consequence that might be interesting in its own right.

▶ **Theorem 5.1.** *Given a VAS language $L \subseteq \Sigma^*$, one can compute $m, k \in \mathbb{N}$ and regular languages $R_{i,j} \subseteq \Sigma^*$, for $i \in [1, m]$, $j \in [1, k]$ so that*

$$L \subseteq \bigcup_{i=1}^{m} R_{i,1} \cdots R_{i,k} \qquad \text{and} \qquad R_{i,1} \times \cdots \times R_{i,k} \subseteq F_k(L) \text{ for every } i \in [1, m]. \tag{1}$$

We first show how to derive Theorem 3.1 from Theorem 5.1 and then proceed with the proof of Theorem 5.1, as it is much more technically complicated.

Proof of Theorem 3.1 Suppose Theorem 5.1 holds. Then, given a VAS language L, we compute $m, k \in \mathbb{N}$ and the regular languages $R_{i,j}$ for $i \in [1, m], j \in [1, k]$. We choose $R = \bigcup_{i=1}^{m} R_{i,1} \cdots R_{i,k}$. Then we have $L \subseteq R$. Let us show that $\mathfrak{p}(F_n(L))$ if and only if $\mathfrak{p}(F_n(R))$. If $\mathfrak{p}(F_n(L))$, then clearly $\mathfrak{p}(F_n(R))$, because $L \subseteq R$ implies $F_n(L) \subseteq F_n(R)$ and by axiom (i), this implies $\mathfrak{p}(F_n(R))$. Conversely, suppose $\mathfrak{p}(F_n(R))$. Then by axiom (ii), there is an $i \in [1, m]$ such that $\mathfrak{p}(F_n(R_i))$, where $R_i = R_{i,1} \cdots R_{i,k}$. According to axiom (iii), we can write $n = n_1 + \cdots + n_k$ such that $\mathfrak{p}$ holds for $S := F_{n_1}(R_{i,1}) \times \cdots \times F_{n_k}(R_{i,k})$. Note that by the choice of $R_{i,j}$, we have $R_{i,1} \times \cdots \times R_{i,k} \subseteq F_k(L)$ and therefore $S \subseteq F_n(L)$. This implies $\mathfrak{p}(F_n(L))$ by axiom (i).

Proof of Theorem 5.1 The remainder of this section is devoted to the proof of Theorem 5.1. Like the method for computing downward closures by Habermehl, Meyer, and Wimmel [28], the construction of the sets $R_{i,j}$ is based on Lambert's proof [38] of the decidability of the reachability problem for Petri nets. In order to be compatible with Lambert's exposition, we phrase our proof in terms of Petri nets instead of vector addition systems.

A *Petri net* $N = (P, T, \textsc{Pre}, \textsc{Post})$ consists of a finite set P of *places*, a finite set T of *transitions* and two mappings $\textsc{Pre}, \textsc{Post} \colon T \to \mathbb{N}^P$. Configurations of Petri net are elements

of $\mathbb{N}^P$, called *markings*. For two markings M, M' we say that M' *dominates* M, denoted $M \leq M'$, if for every place $p \in P$, we have $M[p] \leq M'[p]$. The *effect* of a transition $t \in T$ is $\textsc{Post}(t) - \textsc{Pre}(t) \in \mathbb{Z}^P$, denoted $\Delta(t)$. If a marking M dominates $\textsc{Pre}(t)$ for a transition $t \in T$ then t is *fireable* in M and the result of firing t in marking M is $M' = M + \Delta(t)$, we write $M \xrightarrow{t} M'$. We extend notions of fireability and firing naturally to sequences of transitions, we also write $M \xrightarrow{w} M'$ for $w \in T^*$. The *effect* of $w \in T^*$ is sum of the effects of its letters, $\Delta(w) = \sum_{i=1}^{|w|} \Delta(w[i])$. For a Petri net $N = (P, T, \textsc{Pre}, \textsc{Post})$ and markings M_0, M_1, we define the language $L(N, M_0, M_1) = \{w \in T^* \mid M_0 \xrightarrow{w} M_1\}$. Hence, $L(N, M_0, M_1)$ is the set of transition sequences leading from M_0 to M_1. Moreover, let $L(N, M_0) = \bigcup_{M \in \mathbb{N}^P} L(N, M_0, M)$, i.e. the set of all the transition sequences fireable in M_0. A *labeled Petri net* is a Petri net $N = (P, T, \textsc{Pre}, \textsc{Post})$ together with an *initial marking* M_I, a *final marking* M_F, and a *labeling*, i.e. a homomorphism $h : T^* \to \Sigma^*$. The language *recognized by* the labeled Petri net is then defined as $L_h(N, M_I, M_F) = h(L(N, M_I, M_F))$.

It is folklore (and easy to see) that a language is a VAS language if and only if it is recognized by a labeled Petri net (and the translation is effective). Thus, it suffices to show Theorem 5.1 for languages of the form $L = h(L(N, M_I, M_F))$. Moreover, it is already enough to prove Theorem 5.1 for languages of the form $L(N, M_I, M_F)$. Indeed, observe that if we have constructed $R_{i,j}$ so that Eq. (1) is satisfied, then with $S_{i,j} = h(R_{i,j})$, we have $h(L) \subseteq \bigcup_{i=1}^{m} S_{i,1} \cdots S_{i,k}$ and $S_{i,1} \times \cdots \times S_{i,k} \subseteq F_k(h(L))$ for every $i \in [1, m]$. Thus from now on, we assume $L = L(N, M_I, M_F)$ for a fixed Petri net $N = (P, T, \textsc{Pre}, \textsc{Post})$.

The KLMST decomposition Lambert's decision procedure [38] is a refinement of the previous ones by Mayr [43] and Kosaraju [37]. Leroux and Schmitz [39] recast it as an algorithm using WQO ideals and dubbed it *KLMST decomposition* after its inventors [37, 38, 43, 48].

The idea is the following. We disregard for a moment that a transition sequence has to keep all intermediate markings non-negative and only look for a sequence that may go negative on the way. It is standard technique to express the existence of such a sequence as a linear equation system $Ax = b$. As expected, solvability of this system is not sufficient for the existence of an actual run. However, if we are in the situation that we can find (a) runs that pump up all coordinates arbitrarily high and also (b) counterpart runs that remove those excess tokens again, then solvability of the equation system is also sufficient: We first increase all coordinates high enough, then we execute our positivity-ignoring sequence, and then we pump down again. Roughly speaking, the achievement of the KLMST decomposition is to put us in the latter situation, which we informally call *perfect circumstances*.

To this end, one uses a data structure, in Lambert's version called *marked graph-transition sequence (MGTS)*, which restricts the possible runs of the Petri net. If the MGTS satisfies a condition that realizes the above perfect circumstances, then it is called *perfect*. Unsurprisingly, not every MGTS is perfect. However, part of the procedure is a decomposition of an imperfect MGTS into finitely many MGTS that are less imperfect. Moreover, this decomposition terminates in a finite set of perfect MGTS. Thus, applied to an MGTS whose restriction is merely to start in M_I and end in M_F, then the decomposition yields finitely many perfect MGTS $\mathcal{N}_1, \ldots, \mathcal{N}_n$ such that the runs from M_I to M_F are precisely those conforming to at least one of the MGTS. Moreover, checking whether $\mathcal{N}_i$ admits a run amounts to solving a linear equation system.

Basic notions Let us introduce some notions used in Lambert's proof. We extend the set of configurations $\mathbb{N}^d$ into $\overline{\mathbb{N}}^d$. We extend the notion of transition firing into $\overline{\mathbb{N}}^d$ naturally, by defining $\omega - k = \omega = \omega + k$ for every $k \in \mathbb{N}$. For $u, v \in \overline{\mathbb{N}}^d$ we write $u \leq_\omega v$ if $u[i] = v[i]$ or

$v[i] = \omega$. Intuitively reaching a configuration with ω at some places means that it is possible to reach configurations with values ω substituted by arbitrarily high values.

A key notion in [38] is that of MGTS, which formulate restrictions on paths in Petri nets. A *marked graph-transition sequence (MGTS)* for our Petri net $N = (P, T, \text{Pre}, \text{Post})$ is a finite sequence $C_0, t_1, C_1 \dots C_{n-1}, t_n, C_n$, where t_i are transitions from T and C_i are precovering graphs, which are defined next. A *precovering graph* is a quadruple $C = (G, m, m^{\text{init}}, m^{\text{fin}})$, where $G = (V, E, h)$ is a finite, strongly connected, directed graph with $V \subseteq \overline{\mathbb{N}}^P$ and labeling $h\colon E \to T$, and three vectors: a *distinguished* vector $m \in V$, an *initial* vector $m^{\text{init}} \in \overline{\mathbb{N}}^P$, and a *final* vector $m^{\text{fin}} \in \overline{\mathbb{N}}^P$. A precovering graph has to meet two conditions: First, for every edge $e = (m_1, m_2) \in E$, there is an $m_3 \in \overline{\mathbb{N}}^P$ with $m_1 \xrightarrow{h(e)} m_3 \leq_\omega m_2$. Second, we have $m^{\text{init}}, m^{\text{fin}} \leq_\omega m$. Additionally we impose the restriction on MGTS that the initial vector of C_0 equals M_I and the final vector of C_n equals M_F.

Languages of MGTS Each precovering graph can be treated as a finite automaton. For $m_1, m_2 \in V$, we denote by $L(C, m_1, m_2)$ the set of all $w \in T^*$ read on a path from m_1 to m_2. Moreover, let $L(C) = L(C, m, m)$. MGTS have associated languages as well. Let $\mathcal{N} = C_0, t_1, C_1 \dots C_{n-1}, t_n, C_n$ be an MGTS of a Petri net N, where $C_i = (G_i, m_i, m_i^{\text{init}}, m_i^{\text{fin}})$. Its language $L(\mathcal{N})$ is the set of all words of the form $w = w_0 t_1 w_1 \cdots w_{n-1} t_n w_n \in T^*$ where: $w_i \in L(C_i)$ for each $i \in [0, n]$ and (ii) there exist markings $\mu_0, \mu_0', \mu_1, \mu_1', \dots, \mu_n, \mu_n' \in \mathbb{N}^P$ such that $\mu_i \leq_\omega m_i^{\text{init}}$ and $\mu_i' \leq_\omega m_i^{\text{fin}}$ and $\mu_0 \xrightarrow{w_0} \mu_0' \xrightarrow{t_1} \mu_1 \xrightarrow{w_1} \dots \xrightarrow{w_{n-1}} \mu_{n-1}' \xrightarrow{t_n} \mu_n \xrightarrow{w_n} \mu_n'$. Notice that by (ii) and the restriction that $m_0^{\text{init}} = M_I$ and $m_n^{\text{fin}} = M_F$, we have $L(\mathcal{N}) \subseteq L(N, M_I, M_F)$ for any MGTS $\mathcal{N}$.

Hence roughly speaking, $L(\mathcal{N})$ is the set of runs that contain the transitions $t_1, \dots, t_n$ and additionally markings before and after firing these transitions are prescribed on some places: this is exactly what the restrictions $\mu_i \leq_\omega m_i^{\text{init}}$, $\mu_i' \leq_\omega m_i^{\text{fin}}$ impose.

Notice that at the moment we do not expect that values ω occurring at $m_i, m_i^{\text{init}}, m_i^{\text{fin}}$ impose any restriction on the form of accepted runs. Meaning of ω values is reflected in the notion of *perfect* MGTS described later. As an immediate consequence of the definition, we observe that for every MGTS $\mathcal{N} = C_0, t_1, C_1 \dots C_{n-1}, t_n, C_n$ we have

$$L(\mathcal{N}) \subseteq L(C_0) \cdot \{t_1\} \cdot L(C_1) \cdots L(C_{n-1}) \cdot \{t_n\} \cdot L(C_n). \tag{2}$$

Perfect MGTS As announced above, Lambert calls MGTS with a paricular property *perfect* [38]. Since the precise definition is involved and we do not need all the details, it is enough for us to mention a selection of properties of perfect MGTS. Intuitively, in perfect MGTSes, the value ω on place p in m_i means that inside of the component C_i, the token count in place p can be made arbitrarily high. In [38] it is shown (Theorem 4.2 (page 94) together with the preceding definition) that

▶ **Theorem 5.2** ([38]). *For a Petri net N one can compute finitely many perfect MGTS $\mathcal{N}_1, \dots, \mathcal{N}_m$ such that $L(N, M_I, M_F) = \bigcup_{i=1}^m L(\mathcal{N}_i)$.*

Moreover, by Corollary 4.1 in [38] (page 93), given a perfect MGTS $\mathcal{N}$, it is decidable whether $L(\mathcal{N}) \neq \emptyset$. Therefore, our task reduces to the following. We have a perfect MGTS $\mathcal{N}$ with $L(\mathcal{N}) \neq \emptyset$ and want to compute regular languages $R_1, \dots, R_k$ such that $L(\mathcal{N}) \subseteq R_1 \cdots R_k$ and $R_1 \times \cdots \times R_k \subseteq F_k(L(\mathcal{N}))$. (Note that if the MGTS have different lengths, we can always fill up with $\{\varepsilon\}$). We choose $R_1, \dots, R_k$ to be the sequence $L(C_0), \{t_1\}, L(C_1), \dots, L(C_{n-1}), \{t_n\}, L(C_n)$. Then Eq. (2) tells us that this achieves

$L(\mathcal{N}) \subseteq R_1 \cdots R_k$ and all that remains to be shown is

$$L(C_0) \times \{t_1\} \times L(C_1) \times \cdots \times L(C_{n-1}) \times \{t_n\} \times L(C_n) \subseteq F_{2n+1}(L(\mathcal{N})). \tag{3}$$

Constructing runs In order to show Eq. (3), we employ a simplified version of Lambert's iteration lemma, which involves covering sequences. Let C be a precovering graph for a Petri net $N = (P, T, \text{Pre}, \text{Post})$ with a distinguished vector $m \in \overline{\mathbb{N}}^P$ and initial vector $m^{\text{init}} \in \overline{\mathbb{N}}^P$. A sequence $u \in L(C) \cap L(N, m^{\text{init}})$ is called a *covering sequence for* C if for every place $p \in P$ we have either 1) $m^{\text{init}}[p] = \omega$, or 2) $m[p] = m^{\text{init}}[p]$ and $\Delta(u)[p] = 0$, or 3) $m[p] = \omega$ and $\Delta(u)[p] > 0$. This corresponds intuitively to the three possible cases for the set of runs in N crossing the component C in a place p: (i) runs that can have arbitrarily high value on p when entering C, (ii) runs where, when entering C, p has a fixed value, and the tokens in p cannot be pumped inside of C, or (iii) runs where, when entering C, p has a fixed value, but it can be pumped up inside of C.

Let $\mathcal{N} = C_0, t_1, C_1 \dots C_{n-1}, t_n, C_n$ be an MGTS, where $C_i = (V_i, E_i, h_i)$ is a precovering graph, and let the distinguished vertex be m_i and initial vertex be m_i^{init}. If $\mathcal{N}$ is a perfect MGTS then according to the definition from [38] (page 92), for every $i \in [0, n]$ there exists a covering sequence $u_i \in L(C_i) \cap L(N, m_i^{\text{init}})$. This corresponds to the mentioned intuition that ω values imply arbitrarily high values. As a direct consequence of Lemma 4.1 in [38] (page 92), Lambert's iteration lemma, we obtain:

▶ **Lemma 5.3.** *Let $\mathcal{N} = C_0, t_1, C_1 \dots C_{n-1}, t_n, C_n$ be a perfect MGTS and let x_i be a covering sequences for C_i for $i \in [0, n]$. Then there exist words $y_i \in T^*$ for $i \in [0, n]$ such that $x_0 y_0 \cdot t_1 \cdot x_1 y_1 \cdots x_{n-1} y_{n-1} \cdot t_n \cdot x_n y_n \in L(\mathcal{N})$.*

Lemma 5.3 is obtained from Lemma 4.1 in [38] as follows. The word u_i there is our x_i and v_i there is an arbitrary covering sequence of C_i reversed. Then, our y_i is set to $u_i^{k-1} \beta_i(w_i)^k (v_i)^k$ for some $k \geq k_0$. The only technical part of the proof of Theorem 5.1 is the following lemma.

▶ **Lemma 5.4.** *Let C be a precovering graph for a Petri net $N = (P, T, \text{Pre}, \text{Post})$ with a distinguished vector $m \in \overline{\mathbb{N}}^P$ and initial vector $m^{\text{init}} \in \overline{\mathbb{N}}^P$ such that $s \in L(C) \cap L(N, m^{\text{init}})$ is a covering sequence. Then for every $v \in L(C)$ there is a covering sequence for C of the form uv, for some $u \in T^*$.*

Proof. Intuitively, we do the following. The existence of a covering sequence means that one can obtain arbitrarily high values on places p where $m[p] = \omega$. Thus, in order to construct a covering sequence containing v as a suffix, we first go very high on the ω places, so high that adding v as a suffix later will still result in a sequence with positive effect.

Let us make this precise. Executing the sequence v might have a negative effect in a place $p \in P$ with $m[p] = \omega$. Let $k \in \mathbb{N}$ be the largest possible negative effect a prefix of v can have in any coordinate. Note that since s is a covering sequence, s^k is a covering sequence as well. We claim that $s^k v$ is also a covering sequence. It is contained in $L(C)$ and fireable at m^{init}. Moreover, by choice of k, the sequence $s^k v$ has a positive effect on each p with $m[p] = \omega$. If $m[p] < \omega$, then $\Delta(s)[p] = 0 = \Delta(v)[p]$ and hence $\Delta(s^k v)[p] = 0$. ◀

Using Lemma 5.3 and Lemma 5.4, it is now easy to show Eq. (3). Given words $v_i \in T^*$ with $v_i \in L(C_i)$ for $i \in [0, n]$, we use Lemma 5.4 to choose $x_i \in T^*$ such that $x_i v_i$ is a covering sequence of C_i for $i \in [0, n]$. By Lemma 5.3, we can find $w_1, \dots, w_n$ so that the word $x_0 v_0 w_0 \cdot t_1 \cdot x_1 v_1 w_1 \cdots x_{n-1} v_{n-1} w_{n-1} \cdot t_n \cdot x_n v_n w_n$ belongs to $L(\mathcal{N})$, and thus $(v_0, t_1, v_1, \dots, v_{n-1}, t_n, v_n) \in F_{2n+1}(L(\mathcal{N}))$, which proves Eq. (3).

References

1 Mohamed Faouzi Atig and Pierre Ganty. Approximating Petri net reachability along context-free traces. In *FSTTCS 2011*, volume 13, pages 152–163, Dagstuhl, Germany, 2011.

2 Mohamed Faouzi Atig and Peter Habermehl. On Yen's path logic for Petri nets. *Int. J. Found. Comput. Sci.*, 22(4):783–799, 2011.

3 Jean Berstel. *Transductions and Context-Free Languages*. Teubner, 1979.

4 Michel Blockelet and Sylvain Schmitz. Model checking coverability graphs of vector addition systems. In *MFCS 2011*, pages 108–119, Berlin, Heidelberg, 2011. Springer.

5 Mikołaj Bojańczyk. It is undecidable if two regular tree languages can be separated by a deterministic tree-walking automaton. *Fundam. Inform.*, 154(1-4):37–46, 2017.

6 Rémi Bonnet, Alain Finkel, Jérôme Leroux, and Marc Zeitoun. Place-Boundedness for Vector Addition Systems with one zero-test. In *FSTTCS 2010*, pages 192–203, 2010.

7 Ahmed Bouajjani, Javier Esparza, and Tayssir Touili. A generic approach to the static analysis of concurrent programs with procedures. *International Journal of Foundations of Computer Science*, 14(04):551–582, 2003.

8 Sagar Chaki, Edmund M. Clarke, Nicholas A. Kidd, Thomas W. Reps, and Tayssir Touili. *Verifying Concurrent Message-Passing C Programs with Recursive Calls*, pages 334–349. Springer-Verlag, Berlin Heidelberg, 2006.

9 Pierre Chambart, Alain Finkel, and Sylvain Schmitz. Forward analysis and model checking for trace bounded WSTS. *Theoretical Computer Science*, 637:1–29, 2016.

10 Lorenzo Clemente, Wojciech Czerwiński, Sławomir Lasota, and Charles Paperman. Regular separability of Parikh automata. In *ICALP 2017, July 10-14, 2017, Warsaw, Poland*, pages 117:1–117:13, 2017.

11 Lorenzo Clemente, Wojciech Czerwiński, Sławomir Lasota, and Charles Paperman. Separability of reachability sets of vector addition systems. In *STACS 2017, March 8-11, 2017, Hannover, Germany*, pages 24:1–24:14, 2017.

12 Thomas Colcombet. Regular cost functions, part I: logic and algebra over words. *Logical Methods in Computer Science*, 9(3), 2013.

13 Wojciech Czerwiński and Sławomir Lasota. Regular separability of one counter automata. In *LICS 2017, Reykjavik, Iceland, June 20-23, 2017*, pages 1–12, 2017.

14 Wojciech Czerwiński, Sławomir Lasota, Roland Meyer, Sebastian Muskalla, K Narayan Kumar, and Prakash Saivasan. Regular separability of well structured transition systems. *CoRR*, abs/1702.05334, 2018.

15 Wojciech Czerwiński, Wim Martens, Lorijn van Rooijen, Marc Zeitoun, and Georg Zetzsche. A characterization for decidable separability by piecewise testable languages. *Discrete Mathematics & Theoretical Computer Science*, 19(4), 2017.

16 Stéphane Demri. On selective unboundedness of VASS. In *Proceedings of the 12th International Workshops on Verification of Infinite State Systems (INFINITY'10)*, volume 39 of *Electronic Proceedings in Theoretical Computer Science*, pages 1–15, Singapore, 2010.

17 Stéphane Demri, Alain Finkel, Valentin Goranko, and Govert van Drimmelen. Model-checking CTL* over flat Presburger counter systems. *Journal of Applied Non-Classical Logics*, 20(4):313–344, 2010.

18 Javier Esparza and Pierre Ganty. Complexity of pattern-based verification for multithreaded programs. In *POPL 2011*, pages 499–510, 2011.

19 Javier Esparza, Pierre Ganty, and Rupak Majumdar. A perfect model for bounded verification. In *LICS 2012*, pages 285–294, 2012.

20 Javier Esparza, Pierre Ganty, and Tomás Poch. Pattern-based verification for multithreaded programs. *ACM Trans. Program. Lang. Syst.*, 36(3):9:1–9:29, 2014.

21 Pierre Ganty, Rupak Majumdar, and Benjamin Monmege. Bounded underapproximations. *Formal Methods in System Design*, 40(2):206–231, 2012.

22 Paweł Gawrychowski, Dalia Krieger, Narad Rampersad, and Jeffrey Shallit. Finding the growth rate of a regular or context-free language in polynomial time. *International Journal of Foundations of Computer Science*, 21(04):597–618, 2010.

23 Gilles Geeraerts, Jean-François Raskin, and Laurent Van Begin. Well-structured languages. *Acta Informatica*, 44(3–4):249–288, 2007.

24 Seymour Ginsburg and Edwin H Spanier. Bounded algol-like languages. *Transactions of the American Mathematical Society*, 113(2):333–368, 1964.

25 Seymour Ginsburg and Edwin H. Spanier. Bounded regular sets. *Proceedings of the American Mathematical Society*, 17(5):1043–1049, 1966.

26 Jean Goubault-Larrecq and Sylvain Schmitz. Deciding piecewise testable separability for regular tree languages. In *ICALP 2016, July 11-15, 2016, Rome, Italy*, pages 97:1–97:15, 2016.

27 Sheila A. Greibach. Remarks on blind and partially blind one-way multicounter machines. *Theoretical Computer Science*, 7(3):311 – 324, 1978.

28 Peter Habermehl, Roland Meyer, and Harro Wimmel. The downward-closure of Petri net languages. In *ICALP 2010*, volume 6199 of *Lecture Notes in Computer Science*, pages 466–477. Springer-Verlag, 2010.

29 Matthew Hague and Anthony Widjaja Lin. Model checking recursive programs with numeric data types. In *CAV 2011*, volume 6806 of *Lecture Notes in Computer Science*, pages 743–759. Springer-Verlag, 2011.

30 Matthew Hague, Andrzej S. Murawski, C.-H. Luke Ong, and Olivier Serre. Collapsible pushdown automata and recursion schemes. In *LICS 2008*, pages 452–461, 2008.

31 Leonard H. Haines. On free monoids partially ordered by embedding. *Journal of Combinatorial Theory*, 6(1):94–98, 1969.

32 Dirk Hauschildt. *Semilinearity of the reachability set is decidable for Petri nets*. PhD thesis, Fachbereich Informatik, Universität Hamburg, 1990. Also available as Technical Report No. FBI-HH-B-146/90, Fachbereich Informatik, University of Hamburg.

33 Dirk Hauschildt and Matthias Jantzen. Petri net algorithms in the theory of matrix grammars. *Acta Informatica*, 31(8):719–728, Aug 1994.

34 Graham Higman. Ordering by divisibility in abstract algebras. *Proceedings of the London Mathematical Society. Third Series*, 2:326–336, 1952.

35 John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, Reading, Massachusetts, 1979.

36 Matthias Jantzen. On the hierarchy of petri net languages. *RAIRO - Theoretical Informatics and Applications - Informatique Théorique et Applications*, 13(1):19–30, 1979.

37 S. Rao Kosaraju. Decidability of reachability in vector addition systems (preliminary version). In *STOC 1982, May 5-7, 1982, San Francisco, California, USA*, pages 267–281, 1982.

38 Jean-Luc Lambert. A structure to decide reachability in Petri nets. *Theor. Comput. Sci.*, 99(1):79–104, 1992.

39 Jérôme Leroux and Sylvain Schmitz. Reachability in vector addition systems demystified. In *LICS 2015*, 2015.

40 Jérôme Leroux, Grégoire Sutre, and Patrick Totzke. On the coverability problem for pushdown vector addition systems in one dimension. In *ICALP 2015*, pages 324–336, Berlin Heidelberg, 2015. Springer-Verlag.

41 Zhenyue Long, Georgel Calin, Rupak Majumdar, and Roland Meyer. Language-theoretic abstraction refinement. In *FASE 2012*, pages 362–376. Springer, 2012.

42 A. N. Maslov. Multilevel stack automata. *Problems of Information Transmission*, 12(1):38–42, 1976.

43 Ernst W. Mayr. An algorithm for the general Petri net reachability problem. In *STOC 1981, May 11-13, 1981, Milwaukee, Wisconsin, USA*, pages 238–246, 1981.

44 Thomas Place and Marc Zeitoun. Concatenation hierarchies: New bottle, old wine. In *CSR 2017, Kazan, Russia, June 8-12, 2017, Proceedings*, pages 25–37, 2017.

45 Thomas Place and Marc Zeitoun. Separation for dot-depth two. In *LICS 2017, Reykjavik, Iceland, June 20-23, 2017*, pages 1–12, 2017.

46 Klaus Reinhardt. The "trio-zoo"—classes of formal languages generated from one language by rational transduction. Unpublished manuscript.

47 Klaus Reinhardt. Reachability in Petri nets with inhibitor arcs. *Electronic Notes in Theoretical Computer Science*, 223:239–264, 2008. Proceedings of the Second Workshop on Reachability Problems in Computational Models (RP 2008).

48 George S. Sacerdote and Richard L. Tenney. The decidability of the reachability problem for vector addition systems (preliminary version). In *STOC 1977*, pages 61–76. ACM, 1977.

49 Rani Siromoney. A characterization of semilinear sets. *Proceedings of the American Mathematical Society*, 21(3):689–694, 1969.

50 Hsu-Chun Yen. A unified approach for deciding the existence of certain Petri net paths. *Information and Computation*, 96(1):119–137, 1992.

51 Georg Zetzsche. The emptiness problem for valence automata over graph monoids. To appear in Information and Computation.

52 Georg Zetzsche. An approach to computing downward closures. In *ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part II*, pages 440–451, 2015.

A Separability by bounded regular languages

This section contains the omitted proofs concerning separability by bounded regular languages.

Proof of Lemma 4.5. First, if L_0 and L_1 are separable by a regular $R \subseteq w_1^* \cdots w_n^*$, then the set

$$S = \{(x_1, \ldots, x_n) \mid w_1^{x_1} \cdots w_n^{x_n} \in R\}$$

is recognizable. This is a classical result by Ginsburg and Spanier [25]. Moreover, S clearly separates U_0 from U_1.

Conversely, if $S \subseteq \mathbb{N}^n$ is recognizable and separates U_0 from U_1, then the set

$$R = \{w_1^{x_1} \cdots w_n^{x_n} \mid (x_1, \ldots, x_n) \in S\}$$

is regular. Let us show that it separates L_0 and L_1. If $w \in L_0$, then we can write $w = w_1^{x_1} \cdots w_n^{x_n}$, which implies $(x_1, \ldots, x_n) \in U_0$. Therefore, we have $(x_1, \ldots, x_n) \in S$ and thus $w = w_1^{x_1} \cdots w_n^{x_n} \in R$. Thus, $L_0 \subseteq R$. Now suppose $w \in R$. Then we can write $w = w_1^{x_1} \cdots w_n^{x_n}$ with $(x_1, \ldots, x_n) \in S$. That implies $(x_1, \ldots, x_n) \notin U_1$ and hence $w_1^{x_1} \cdots w_n^{x_n} \notin L_1$. Hence, $R \cap L_1 = \emptyset$. ◀

In the proof, we also use the following fact:

▶ **Proposition A.1.** *If $L \subseteq w_1^* \cdots w_n^*$ is a VAS language, then the set*

$$U = \{(x_1, \ldots, x_n) \in \mathbb{N}^n \mid w_1^{x_1} \cdots w_n^{x_n} \in L\}$$

is a effectively a section of a VAS reachability set.

Proof. First recall the notion of a section. For a subset $I \subseteq [1,n]$, let $\pi_I \colon \mathbb{N}^n \to \mathbb{N}^{|I|}$ be the projection onto the coordinates in I. Then, every set of the form $\pi_{[1,n]\setminus I}(S \cap \pi_I^{-1}(x))$ for some $I \subseteq [1,n]$ and $x \in \mathbb{N}^{|I|}$ is called a *section* of $S \subseteq \mathbb{N}^n$. Intuitively, we fix a vector $x \in \mathbb{N}^{|I|}$ on coordinates from I and take into the section all the vectors $y \in \mathbb{N}^{n-|I|}$, which together with x form an n-dimensional vector from S.

Assume that L is a language of d-dimensional VAS V. In order to show that U is a section of a VAS reachability set we construct another VAS V' in the following way. VAS V' simulates V on d coordinates and has n additional coordinates, on which it counts number of occurrences of words $w_1, \ldots, w_n$. It is easy to see that VAS indeed can count such occurrences by keeping some additional finite information, like the suffix of current run, which has not been yet counted into any w_i and the information which w_i has recently appeared. Section of V' leaving only these n counting coordinates is exactly the set U. ◀

B Downward closures

Proof of Lemma 4.7. Since $a_1^* \times \cdots \times a_n^* \subseteq F_n(L_1 \cdots L_k)\downarrow$, we know that for every $\ell \in \mathbb{N}$, we can find words $w_i \in L_i$ for $i \in [1,k]$ so that $a_1^{\ell \cdot k} \cdots a_n^{\ell \cdot k} \preceq w_1 \cdots w_k$. Then, in particular, there is a monotone map $\pi_\ell \colon [1,n] \to [1,k]$ so that $a_i^\ell \preceq w_{\pi(i)}$. Since there are only finitely many maps $[1,n] \to [1,k]$, there is one monotone map $\pi \colon [1,n] \to [1,k]$ that occurs infinitely often in the sequence $\pi_1, \pi_2, \ldots$. We can decompose $[1,n] = \pi^{-1}(1) \cup \cdots \cup \pi^{-1}(k)$ and since π is monotone, each $\pi^{-1}(i)$ is convex. This give rise to a decomposition $n = n_1 + \cdots + n_k$ so that $\pi^{-1}(1) \subseteq [1,n_1]$, $\pi^{-1}(2) \subseteq [n_1+1, n_2]$, etc. Now, by choice of π, for each $\ell \in \mathbb{N}$, we can find $w_i \in L_i$ so that $a_i^\ell \preceq w_{\pi(i)}$, which means $(a_1^\ell, \ldots, a_n^\ell) \in (F_{n_1}(L_1) \times \cdots \times F_{n_k}(L_k))\downarrow$. This implies $a_1^* \times \cdots \times a_n^* \subseteq (F_{n_1}(L_1) \times \cdots \times F_{n_k}(L_k))\downarrow$. ◀

C Non-overlapping factors

Proof of Lemma 4.9. Suppose $K \subseteq \Sigma^*$ and let $\mathcal{A}$ be a finite automaton for $R \subseteq \Sigma^*$. Pick a symbol $c \notin \Sigma$. We obtain a finite automaton $\mathcal{B}$ from $\mathcal{A}$ as follows. In the first step, for each pair p, q of states, we check whether there is a word in K that labels a path p to q in $\mathcal{A}$: This is decidable because we can effectively intersect languages in $\mathcal{C}$ with regular languages and emptiness is decidable for $\mathcal{C}$. If such a word exists, we add an edge labeled c from p to q. In the second step, for each edge with a label $\neq c$, we replace the label by ε. This completes the construction of $\mathcal{B}$.

Clearly, f_K is unbounded on R if and only if $\{c\}^* \subseteq L(\mathcal{B})$. Moreover, if f_K is bounded on R, then $L(\mathcal{B})$ is finite and we can compute the maximal length ℓ of a word in $L(\mathcal{B})$. This ℓ is then an upper bound for f_K on L. ◀

D Counting automata

We begin with a formal definition of the step relation in counting automata. For a label $x \in \Sigma \cup \{\varepsilon\}$, and configurations $(q,u,\mu), (q',u',\mu')$, we write $(q,u,\mu) \xrightarrow{x} (q',u',\mu')$ if there is an edge $(q,x,o,q) \in E$ such that one of the following holds:

- We have $o = \mathsf{push}(a)$ for some $a \in \Gamma$ and $u' = ua$ and $\mu' = \mu$.
- We have $o = \mathsf{check}(K,c)$ for some $K \subseteq \Gamma^*$ from $\mathcal{C}$ and $c \in C$ and $u' = \varepsilon$ and either (a) $u \in K$ and $\mu' = \mu + 1_c$ or (b) $u \notin K$ and $\mu' = \mu$. Here, $1_c \in \mathbb{N}^C$ is the vector with $1_c(c) = 1$ and $1_c(c') = 0$ for $c' \in C \setminus c$.

Moreover, for $w \in \Sigma^*$, we write $(q,u,\mu) \xrightarrow{w} (q',u',\mu')$ if

$$(q,u,\mu) = (q_1,u_1,\mu_1) \xrightarrow{x_0} \cdots \xrightarrow{x_n} (q_n,u_n,\mu_n) = (q',u',\mu'),$$

for some configurations (q_i,u_i,μ_i) and $w = x_0 \cdots x_n$, where $x_0,\ldots,x_n \in \Sigma_\varepsilon$.

In our proof of Theorem 4.10, we will use Theorem 3.1 and hence decidability of a multidimensional predicate. Suppose $t = (K_1,\ldots,K_n)$ is a tuple of languages $K_i \subseteq \Sigma^+$. We define a function $f_t\colon \Sigma^* \to \mathbb{N}$ as follows. Intuitively, $f_t(w)$ is the largest number k so that we can pick a set of non-overlapping factors of w among whom there are at least k members of K_i for each $i \in [1,n]$.

Formally, for a word $w \in \Sigma^*$, let $f_t(w)$ be the largest number ℓ such that there is a tuple $(w_1,\ldots,w_m) \in (\Sigma^+)^m$ with $(w_1,\ldots,w_m) \in F_m(w)$ such that for each $i \in [1,n]$, we have $|\{j \in [1,m] \mid w_j \in K_i\}| \geq \ell$. Using an n-dimensional predicate and Theorem 3.1, we can show the following.

▶ **Lemma D.1.** *Let $\mathcal{C}$ be a full trio with decidable emptiness. Given a tuple $t = (K_1,\ldots,K_n)$ of languages from $\mathcal{C}$ and a VAS language L, it is decidable whether f_t is unbounded on L. Moreover, if f_t is bounded on L, one can compute an upper bound $B \in \mathbb{N}$ for f_t on L.*

Proof. Let $t = (K_1,\ldots,K_n)$ be a tuple of languages with $K_i \subseteq \Sigma^+$ for $i \in [1,n]$. For a word $w \in \Sigma^*$, let $\Delta(w) \subseteq \mathbb{N}^n$ be the set of all $(x_1,\ldots,x_n) \in \mathbb{N}^n$ such that there is a tuple $(w_1,\ldots,w_m) \in (\Sigma^+)^m$ with $(w_1,\ldots,w_m) \in F_m(w)$ and $x_i = |\{j \in [1,m] \mid w_j \in K_i\}|$.

Let us now define the predicate $\mathfrak{p}$. For $S \subseteq (\Sigma^*)^n$, let $\mathfrak{p}(S)$ express that for every $\ell \in \mathbb{N}$, there is a tuple $(w_1,\ldots,w_n) \in S$ and a vector $(x_1,\ldots,x_n) \in \sum_{i=1}^n \Delta(w_i)$ such that $x_i \geq \ell$ for each $i \in [1,n]$. Here, the sum on subsets of $\mathbb{N}^n$ is to be read as the Minkowski sum: $X + Y = \{x + y \mid x \in X,\ y \in Y\}$. Note that then indeed $\mathfrak{p}(F_n(L))$ if and only if f_t is unbounded on L.

The predicate $\mathfrak{p}$ clearly satisfies axioms (i) and (ii), so let us prove axiom (iii) and suppose $\mathfrak{p}(F_n(L_1 \cdots L_k))$. A *profile* is a map $\pi\colon [1,n] \to [1,k]$. Intuitively, a profile records for each $i \in [1,n]$ which of the factors $L_1,\ldots,L_k$ can be chosen to find a particular number of factors from K_i.

Let $\ell \in \mathbb{N}$. Since $\mathfrak{p}(F_n(L_1 \cdots L_k))$, we know that there is a $(w_1,\ldots,w_n) \in F_n(L_1 \cdots L_k)$ such that there is a $(x_1,\ldots,x_n) \in \sum_{i=1}^n \Delta(w_i)$ with $x_i \geq k \cdot \ell + k$. Since $(w_1,\ldots,w_n) \in F_n(L_1 \cdots L_k)$, there is a word $u \in L_1 \cdots L_k$ with $(w_1,\ldots,w_n) \in F_n(u)$. Thus, we have a $(y_1,\ldots,y_n) \in \Delta(u)$ with $y_i \geq k \cdot \ell + k$ for each $i \in [1,n]$. Since $u \in L_1 \cdots L_k$, we can write $u = u_1 \cdots u_k$ with $u_i \in L_i$ for $i \in [1,k]$.

Observe that then there is a $(z_1,\ldots,z_n) \in \sum_{i=1}^k \Delta(u_i)$ with $z_i \geq y_i - k$ for $i \in [1,n]$: From the set of factors that witnesses $(y_1,\ldots,y_n) \in \Delta(u)$, we can select those that are confined to a single u_i; then we lose at most those that fall on the border of two u_i's, hence at most k. Since $y_i \geq k \cdot \ell + k$, we have $z_i \geq k \cdot \ell$ for $i \in [1,n]$. Write $(z_1,\ldots,z_n) = \sum_{i=1}^k (z_{i,1},\ldots,z_{i,n})$ with $(z_{i,1},\ldots,z_{i,n}) \in \Delta(u_i)$. Since $z_{1,i} + \cdots + z_{k,i} = z_i \geq k \cdot \ell$, we can find for each $i \in [1,n]$, an index $j \in [1,k]$ so that $z_{j,i} \geq \ell$. This defines a profile π_ℓ: Let $\pi_\ell(i) = j$.

To summarize, we have defined for each $\ell \in \mathbb{N}$ a profile π_ℓ so that the following holds. For each $\ell \in \mathbb{N}$, there are words $u_1,\ldots,u_k$ with $u_j \in L_j$ for $j \in [1,k]$ so that for each $i \in [1,n]$, the set $\Delta(u_{\pi_\ell(i)})$ contains a vector $(z_1,\ldots,z_n)$ with $z_i \geq \ell$.

Since there are only finitely many profiles, the sequence $\pi_1,\pi_2,\ldots$ must contain one profile π infinitely often. This profile has thus the following property. For each $\ell \in \mathbb{N}$, there are words $u_1,\ldots,u_k$ with $u_j \in L_j$ for $j \in [1,k]$ so that for each $i \in [1,n]$, the set $\Delta(u_{\pi(i)})$ contains a vector $(z_1,\ldots,z_n)$ with $z_i \geq \ell$.

This allows us to define the decomposition $n = n_1 + \cdots + n_k$: For each $j \in [1,k]$, let $n_j = |\{i \in [1,n] \mid \pi(i) = j\}|$. We claim that then $\mathfrak{p}(F_{n_1}(L_1) \times \cdots F_{n_k}(L_k))$ holds. Let $j \in \mathbb{N}$.

We can choose words $u_1, \dots, u_k$ with $u_j \in L_j$ for $j \in [1,k]$ so that for each $i \in [1,n]$, the set $\Delta(u_{\pi(i)})$ contains a vector $(z_1, \dots, z_n)$ with $z_i \geq \ell$.

Let us construct the tuple $(v_1, \dots, v_n)$ successively from left to right. For each $j = 1, \dots, k$, we do the following. If $n_j = 0$, then we add no new component. If $n_j > 0$, then we include u_j and then $(n_j - 1)$ entries containing just the empty word ε. This clearly yields a tuple with $n = n_1 + \cdots + n_k$ entries. Moreover, we have $(v_1, \dots, v_n) \in F_{n_1}(L_1) \times \cdots \times F_{n_k}(L_k)$. Finally, for each $i \in [1,n]$, we have $n_{\pi(i)} > 0$ and hence $u_{\pi(i)}$ occurs in the tuple $(v_1, \dots, v_n)$. Therefore, some $\Delta(v_i)$ contains a vector $(z_1, \dots, z_n)$ with $z_i \geq \ell$. Therefore, the sum $\sum_{i=1}^{n} \Delta(v_i)$ contains a tuple $(z_1, \dots, z_n)$ with $z_i \geq \ell$ for every $i \in [1,n]$. This proves our claim and hence that $\mathfrak{p}$ satisfies axiom (iii). This shows that $\mathfrak{p}$ is in fact an unboundedness predicate.

According to Theorem 3.1, we can compute a regular language $R \supseteq L$ such that $\mathfrak{p}(F_n(L))$ if and only if $\mathfrak{p}(F_n(R))$. This means f_t is unbounded on L if and only if it is unbounded on R. Moreover, since $L \subseteq R$, an upper bound of f_t on R is also an upper bound of f_t on L. Thus, it remains to show that we can decide whether f_t is bounded on R and, if so, we can compute an upper bound of f_t on R.

Take a finite automaton $\mathcal{A}$ for R. From $\mathcal{A}$, we obtain a finite automaton $\mathcal{B}$ over the alphabet $\Gamma = \{a_1, \dots, a_n\}$ as follows. First, we remove all edges. Then, for each pair p, q of states and each $i \in [1,n]$, we check whether there is a word K_i that is read on a path from p to q in $\mathcal{A}$: This can be checked because K_i belongs to $\mathcal{C}$, $\mathcal{C}$ is effectively closed under intersecion with regular languages, and emptiness is decidable for $\mathcal{C}$. If that is the case, then we draw a new edge labeled a_i from p to q. Then, clearly, f_t is unbounded on R if and only if for every $\ell \in \mathbb{N}$, there is a word w accepted by $\mathcal{B}$ that contains a_i at least ℓ times, for each $i \in [1,n]$. Consider the set

$$S = \{\ell \in \mathbb{N} \mid \exists w \in L(\mathcal{B}) \colon \forall i \in [1,n] \colon |w|_{a_i} \geq \ell\}.$$

It is easy to see that S is effectively semilinear: the Parikh image of $L(\mathcal{B})$ is semilinear and hence S is definable in Presburger arithmetic. Furthermore, f_t is unbounded on R if and only if S is infinite, which is easy to check. Finally, if f_t is bounded on R, then S is finite and we can compute the maximal element of S, which is an upper bound of f_t on R. ◀

In the proof of Theorem 4.10, we will use the concept of a transducer. A *(finite-state) transducer* is a tuple $\mathcal{A} = (Q, \Sigma, \Gamma, E, q_0, Q_f)$, where Q is a finite set of *states*, Σ is its *input alphabet*, Γ is its *output alphabet*, $E \subseteq Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \times Q$ is its set of *edges*, $q_0 \in Q$ is its *initial state*, and $Q_f \subseteq Q$ is its set of *final states*. A *configuration* of $\mathcal{A}$ is a triple $(q,u,v) \in Q \times \Sigma^* \times \Gamma^*$ and we write $(q,u,v) \to (q',u',v')$ if there is an edge (q,x,y,q') with $u' = ux$ and $v' = vy$. Let $\to^*$ denote the reflexive transitive closure of $\to$.

Subsets of $\Sigma^* \times \Gamma^*$ for alphabets Σ, Γ are called *transductions*. A transducer induces a transduction as follows:

$$T(\mathcal{A}) = \{(u,v) \in \Sigma^* \times \Gamma^* \mid (q_0, \varepsilon, \varepsilon) \to^* (q,u,v) \text{ for some } q \in Q_f\}.$$

Then, $T(\mathcal{A})$ is called the transduction *induced by* $\mathcal{A}$. A transduction of the form $T(\mathcal{A})$ is called a *rational transduction*. In general, for a transduction $T \subseteq \Sigma^* \times \Gamma^*$ and a language $L \subseteq \Sigma^*$, we define

$$T(L) = \{v \in \Gamma^* \mid \exists u \in L \colon (u,v) \in T\}.$$

It is well known that a language class $\mathcal{C}$ is a full trio if and only if it is effectively closed under rational transductions, meaning given a description of L, we can effectively compute a description of $T(L)$ in $\mathcal{C}$.

We are now ready to prove Theorem 4.10.

Proof of Theorem 4.10. Given $\mathcal{A}$, we can transform $\mathcal{A}$ into a transducer $\mathcal{B}$ as follows. Let $K_1, \ldots, K_n$ be the languages occurring in edges $\mathsf{check}(K, c)$ in $\mathcal{A}$ and pick letters $d, e_{i,c} \notin \Gamma$ for each $i \in [1, n]$ and $c \in C$. The transducer $\mathcal{B}$ operates like $\mathcal{A}$, but instead of performing operations $\mathsf{push}(a)$ or $\mathsf{check}(K_i, c)$, it outputs symbols from the alphabet $\Lambda = \Gamma \cup \{d, e_{i,c} \mid i \in [1, n], c \in C\}$: When $\mathcal{A}$ performs $\mathsf{push}(a)$, $\mathcal{B}$ outputs a. When $\mathcal{A}$ performs $\mathsf{check}(K_i, c)$, then $\mathcal{B}$ outputs $e_{i,c}d$. Moreover, in the beginning of a run, $\mathcal{B}$ outputs a single d before it starts operating like $\mathcal{A}$. Now let T be the transduction induced by $\mathcal{B}$ and let $L' = T(L)$. Then L' is again a VAS language and consists of precisely those words $du_1e_{i_1,c_1}du_2e_{i_2,c_2}\cdots du_me_{i_m,c_m}u$ such that $u \in \Gamma^*$ and $\mathcal{A}$ has a run on a member of L that performs for each $j \in [1, m]$ the operation $\mathsf{check}(K_{i_j}, c_j)$ while u_j is on the work tape.

Consider the language class $\bar{\mathcal{C}}$, which consists of all finite unions of languages in $\mathcal{C}$. Then $\bar{\mathcal{C}}$ is again a full trio and has a decidable emptiness problem. For each $c \in C$, let $\bar{K}_c = \bigcup_{i \in [1,n]} dK_ie_{i,c}$. Then clearly $\bar{K}_c$ belongs to $\bar{\mathcal{C}}$. Let $C = \{c_1, \ldots, c_k\}$ and consider the language tuple $t = (\bar{K}_{c_1}, \ldots, \bar{K}_{c_k})$. Then f_t is unbounded on L' if and only if $\mathcal{A}$ is unbounded on L. Moreover, an upper bound $B \in \mathbb{N}$ for f_t on L' is also an upper bound for $\mathcal{A}$ on L. Thus, an application of Lemma D.1 completes the proof. ◀

E Factor inclusion

Detailed proof of axiom (ii) First, let us verify axiom (ii) in detail. Suppose that $L_1 \cup L_2$ is K-factor universal and that L_1 is not K-factor universal. The latter means there is some $u \in K^*$ with $u \notin F(L_1)$. Now let $v \in K^*$ be arbitrary. Since $uv \in K^*$ and by K-factor universality of $L_1 \cup L_2$, we know that $uv \in F(L_1 \cup L_2) = F(L_1) \cup F(L_2)$. Since $uv \in F(L_1)$ is impossible, this only leaves $uv \in F(L_2)$ and in particular $v \in F(L_2)$. This proves that L_2 is K-factor universal and hence axiom (ii).

It remains to show decidability of whether $K^* \subseteq F(R)$.

▶ **Lemma E.1.** *Let $\mathcal{C}$ be a full trio with decidable emptiness. Given a language K from $\mathcal{C}$ and a regular language R, it is decidable whether $K^* \subseteq F(R)$.*

Proof. Suppose $K \subseteq \Sigma^*$ and let $\mathcal{A}$ be a finite automaton for the regular language $\Sigma^* \setminus F(R)$. We have to decide whether $K^* \cap L(\mathcal{A}) = \emptyset$. Pick a symbol $c \notin \Sigma$. We obtain a finite automaton $\mathcal{B}$ from $\mathcal{A}$ as follows. For each pair p, q of states, we check whether there is a word in K that labels a path p to q in $\mathcal{A}$: This is decidable because we can effectively intersect languages in $\mathcal{C}$ with regular languages and emptiness is decidable for $\mathcal{C}$. If such a word exists, we add an edge labeled c from p to q. In the second step, we remove all edges except for those labeled c. This finishes the construction of $\mathcal{B}$. Then we have $L(\mathcal{B}) \subseteq \{c\}^*$. Furthermore, K^* intersects $L(\mathcal{A})$ if and only if $L(\mathcal{B}) \neq \emptyset$. ◀