The covering and boundedness problems for branching vector addition systems[☆]

Stéphane Demri[a], Marcin Jurdziński[b], Oded Lachish[b,1], Ranko Lazić[b,*]

[a] *LSV, ENS de Cachan & CNRS & INRIA, 61, avenue du Président Wilson, 94235 Cachan Cedex, France*
[b] *DIMAP, Department of Computer Science, University of Warwick, Gibbet Hill Road, Coventry CV4 7AL, UK*

Abstract

The covering and boundedness problems for branching vector addition systems are shown complete for doubly-exponential time.

Keywords: Petri nets, branching vector addition systems, covering, boundedness, computational complexity

1. Introduction

Vector addition systems (shortly, VAS), or equivalently Petri nets (e.g., [2]), are a fundamental model of computation, which is more expressive than finite-state machines and less than Turing-powerful. Decidability and complexity of a variety of problems have been extensively studied ([3] is a comprehensive survey).

A k-dimensional VAS consists of an initial vector of non-negative integers, and a finite set of vectors of integers, all of dimension k. Let us call the initial vector *axiom*, and the other vectors *rules*. A computation can then be thought of as a *derivation*: it starts with the axiom, and at each step, the next vector is derived from the current one by adding a rule. The vectors of interest are the ones derived *admissibly*, i.e. at the end of a derivation which is such that none of the vectors derived during it contains a negative entry.

Covering and boundedness are two central decision problems for VAS. The former asks whether a vector that is pointwise greater than or equal to a given vector can be admissibly derived, and the latter asks whether the set of all admissibly derived vectors is finite. In a landmark article [4], Rackoff showed that covering and boundedness for VAS are in ExpSpace, matching Lipton's lower bound of ExpSpace-hardness [5].[2] Considering the expressively equivalent VAS with states (shortly, VASS), Rosier and Yen refined the proofs of Lipton and Rackoff to obtain almost matching lower and upper bounds in terms of three

[☆] A preliminary and shorter version of this work was published in the proceedings of FSTTCS 2009 [1].

[*] Corresponding author. Tel: +44 24 7652 3193. Fax: +44 24 7657 3024.
Email addresses: demri@lsv.ens-cachan.fr (Stéphane Demri), mju@dcs.warwick.ac.uk (Marcin Jurdziński), oded@dcs.bbk.ac.uk (Oded Lachish), lazic@dcs.warwick.ac.uk (Ranko Lazić)
[1] Present address: Computer Science and Information Systems, Birkbeck (University of London), Malet Street, London WC1E 7HX, UK.

[2] We recommend http://rjlipton.wordpress.com/2009/04/08/an-expspace-lower-bound/.

parameters: the dimension, the binary size of the maximum absolute value of an entry in a rule, and the number of states [6]. Lipton's result was also extended by Mayr and Meyer to reversible Petri nets, which are equivalent to commutative semigroups [7]. Building on Rosier and Yen's work, Habermehl showed that space exponential in the size of the system and polynomial in the size of the formula suffices for model checking the propositional linear-time μ-calculus on VASS, and he obtained a matching lower bound already for LTL on BPP [8]. Further related developments include the identification by Atig and Habermehl of another path logic for Petri nets whose model checking problem is ExpSpace-complete [9], and Demri's proof of ExpSpace-membership of a generalised boundedness problem which subsumes reversal boundedness, place boundedness, regularity and several other interesting problems for VASS [10].

The following is a natural extension of VAS: instead of linearly, computation proceeds from the leaves to the root of a tree. For each node which is not a leaf, its vector is derived by summing the vectors derived at its children and adding a rule vector.[3] The same condition of admissibility applies, i.e. no derived vector may contain a negative entry. This model of computation is branching VAS (shortly, BVAS).

In recent years, it has turned out that BVAS have interesting connections to a number of formalisms:

- BVAS correspond to a class of linear index grammars in computational linguistics [12, 13];
- reachability (i.e. admissible derivability) for BVAS is decidable iff provability in multiplicative exponential linear logic is decidable [14];
- Verma and Goubault-Larrecq have extended the computation of Karp and Miller trees [15] to BVAS, and used it to draw conclusions about a class of equational tree automata which are useful for analysing cryptographic protocols [16];
- if first-order logic with 2 variables on finite data trees (which has applications to the XPath query language for XML) is decidable, then so is reachability for BVAS [17].

Covering and boundedness for BVAS are decidable easily using the branching extension of Karp and Miller's procedure [16]. However, the resulting algorithms do not operate in primitive recursive time or space, even in the linear case [18].

The main results we report are that, by switching from VAS to BVAS, covering and boundedness move two notches up the complexity hierarchy, to 2ExpTime-complete.

For the 2ExpTime-memberships, consider the following simple-minded idea for transferring knowledge about VAS derivations to the branching case:

Every simple path from a leaf to the root in a BVAS derivation is a VAS derivation.

[3] A different branching extension of VAS was used by Urquhart [11].

We show that the idea can give us mileage, but only after the following new insight, which is needed because the subderivations that grow off the simple path and hence contribute summands to it make the resulting VAS contain rules with unbounded positive entries.

> *For VAS, we can obtain similar upper bounds to Rackoff's, but which depend only on the dimension and the minimum negative entry in a rule, i.e. not on the maximum positive entry in a rule.*

The insight is at the centre of our proofs. In the case of covering, we show it essentially by inspecting carefully a proof of Rackoff, but in the case of boundedness, it relies on proving a new result on small solutions of integer programming problems, which extends a classical theorem of Borosh and Treybig and may also be a contribution of wider interest. To complete the proofs of the 2ExpTime-memberships, we provide arguments for reducing the heights of appropriate BVAS derivations to at most doubly-exponential, and for why resulting small witnesses can be guessed and verified by alternating Turing machines in exponential space.

To obtain 2ExpTime-hardness for covering and boundedness for BVAS, we extend the proof of Lipton to show that computations of alternating machines of size N with counters bounded by 2^{2^N} can be simulated in reverse by BVAS of size $O(N^2)$. Although universal branchings of alternating counter machines copy counter valutations whereas BVAS sum vectors derived at children nodes, the inner workings of Lipton's construction enable us to add a bit of machinery by which the BVAS can simulate the copying. We remark that, as is the case with Lipton's result, the lower bound is shown already for BVAS whose rules contain only entries -1, 0 or 1.

After fixing notations and making some preliminary observations in the next section, that covering and boundedness are in 2ExpTime is shown in Sections 3 and 4, respectively. We then argue in Section 5 that both problems are 2ExpTime-hard.

2. Preliminaries

Numbers, vectors and matrices. We write $\mathbb{N}_+$, $\mathbb{N}$ and $\mathbb{Z}$ for the sets of all positive, non-negative and arbitrary integers, respectively. Since we shall only work with integers, let the open interval (a, b) denote $(a, b) \cap \mathbb{Z}$, and analogously for half-open and closed intervals.

Given a dimension $k \in \mathbb{N}$, let $\mathbf{0}$ denote the zero vector and, for each $i \in [1, k]$, $\mathbf{e}_i$ denote the ith unit vector. For $\mathbf{v}, \mathbf{w} \in \mathbb{Z}^k$ and $B \in \mathbb{Z}$, we write:

- $\mathbf{v}(1), \ldots, \mathbf{v}(k)$ for the entries of $\mathbf{v}$;
- $\mathrm{supp}(\mathbf{v})$ for the set of all $i \in [1, k]$ such that $\mathbf{v}(i) \neq 0$;

- $\mathbf{v} \leq \mathbf{w}$ iff $\mathbf{v}(i) \leq \mathbf{w}(i)$ for all $i \in [1, k]$, and $\mathbf{v} < \mathbf{w}$ iff $\mathbf{v} \leq \mathbf{w}$ and $\mathbf{v} \neq \mathbf{w}$;
- $\min(B, \mathbf{v})$ for the vector $\langle \min\{B, \mathbf{v}(1)\}, \ldots, \min\{B, \mathbf{v}(k)\}\rangle$, and analogously for max;
- $\mathbf{v}^-$ for the vector $-\min(0, \mathbf{v})$, and $\mathbf{v}^+$ for the vector $\max(0, \mathbf{v})$.

For $\mathbf{v} \in \mathbb{N}^k$, let $\max(\mathbf{v}) = \max\{\mathbf{v}(1), \ldots, \mathbf{v}(k)\}$, where in case $k = 0$, we have $\max(\langle\rangle) = \max \emptyset = 0$. For finite $R \subseteq \mathbb{Z}^k$, let $\max(R^{-/+})$ denote $\max\{\max(\mathbf{r}^{-/+}) : \mathbf{r} \in R\}$, respectively.

Let $S^{k \times n}$ denote the set of all matrices with k rows, n columns and entries from S. Conveniently albeit slightly eccentrically, we use $-i$ for an index i to denote all rows or columns other than the ith, and $\bullet$ to denote all rows or columns. For example, $\mathbf{A}_{i\bullet}$ is row i of $\mathbf{A}$, and $\mathbf{A}_{\bullet(-j)}$ is $\mathbf{A}$ with column j removed.

Trees. A finite binary tree $\mathcal{T}$, which may contain nodes with one child, is a non-empty finite subset of $\{1, 2\}^*$ such that, for all $n \in \{1, 2\}^*$ and $i \in \{1, 2\}$, $n \cdot 2 \in \mathcal{T}$ implies $n \cdot 1 \in \mathcal{T}$, and $n \cdot i \in \mathcal{T}$ implies $n \in \mathcal{T}$. The nodes of $\mathcal{T}$ are its elements. The root of $\mathcal{T}$ is ε, the empty word. All notions such as parent, first child, second child, subtree and leaf, have their standard meanings. The height of $\mathcal{T}$ is the length, i.e. the number of nodes, of the longest simple path from the root to a leaf.

BVAS. The systems we define are equivalent to the branching vector addition systems with states [16] and the vector addition tree automata [14, 17]. To simplify our technical life, we work with stateless systems. In the linear case, it is well-known that states can be eliminated in logarithmic space, e.g. by adding the number of states to the dimension. For branching systems, the same is true, but computation steps that join two vectors by addition need to be generalised so that a vector from a fixed finite set (which may contain negative entries) is added also. Since we are not studying the systems as recognisers of languages, we do not have to work with alphabets either. Another simplification which costs only a logarithmic amount of space is in relation to the VATA [14], where branching up to a fixed finite arity was permitted. Hence, adopting a proof-theoretic terminology like that of Verma and Goubault-Larrecq [16], a system will consist of finite sets of axioms, unary rules and binary rules, all of which are simply integral vectors. The unary rules are present for easy compatibility with the linear case.

Let a *branching vector addition system* (*BVAS*) be a tuple $\mathcal{B} = \langle k, A_0, R_1, R_2\rangle$, where:

- $k \in \mathbb{N}$ is the dimension;
- $A_0 \subseteq \mathbb{N}^k$ is a non-empty finite set of axioms;
- $R_1, R_2 \subseteq \mathbb{Z}^k$ are finite sets of unary and binary rules, respectively.

A derivation starts with a number of integral vectors, proceeds by applying the rules, and finishes with a single vector. Applying a unary rule means adding it to a derived vector, and applying a binary rule means

adding it to the sum of two derived vectors. For a vector to be considered produced by the system, it needs to be derived by a derivation which starts with the axioms and whose derived vectors are all non-negative.

Formally, a *derivation* of $\mathcal{B}$ is a labelling $\mathcal{D} : \mathcal{T} \to \mathbb{Z}^k$ such that:

- $\mathcal{T}$ is a finite binary tree;
- if n has one child in $\mathcal{T}$, then $\mathcal{D}(n) \in R_1$;
- if n has two children in $\mathcal{T}$, then $\mathcal{D}(n) \in R_2$.

The vectors that are derived at every node are obtained recursively as follows:

- if n is a leaf in $\mathcal{T}$, then $\widehat{\mathcal{D}}(n) = \mathcal{D}(n)$;
- if n has one child n' in $\mathcal{T}$, then $\widehat{\mathcal{D}}(n) = \mathcal{D}(n) + \widehat{\mathcal{D}}(n')$;
- if n has two children n' and n'' in $\mathcal{T}$, then $\widehat{\mathcal{D}}(n) = \mathcal{D}(n) + \widehat{\mathcal{D}}(n') + \widehat{\mathcal{D}}(n'')$.

Now, we say that $\mathcal{D}$:

- is *initialised* iff, for each leaf n of $\mathcal{T}$, we have $\mathcal{D}(n) \in A_0$;
- is *admissible* iff, for each node n of $\mathcal{T}$, we have $\widehat{\mathcal{D}}(n) \in \mathbb{N}^k$;
- *derives* $\widehat{\mathcal{D}}(\varepsilon)$, which is the vector derived at the root.

For $\mathbf{v} \in \mathbb{N}^k$, we say that $\mathcal{B}$ *produces* $\mathbf{v}$ iff some initialised admissible derivation of $\mathcal{B}$ derives $\mathbf{v}$.

Substitutions and contractions. For finite binary trees $\mathcal{T}$ and $\mathcal{T}'$, and a node n of $\mathcal{T}$, let $\mathcal{T}[n \leftarrow \mathcal{T}']$ denote the tree obtained by replacing with $\mathcal{T}'$ the subtree of $\mathcal{T}$ rooted at n. To extend the notation to derivations, for $\mathcal{D} : \mathcal{T} \to \mathbb{Z}^k$ and $\mathcal{D}' : \mathcal{T}' \to \mathbb{Z}^k$, and a node n of $\mathcal{T}$, let $\mathcal{D}[n \leftarrow \mathcal{D}'] : \mathcal{T}[n \leftarrow \mathcal{T}'] \to \mathbb{Z}^k$ denote the derivation obtained by replacing with $\mathcal{D}'$ the subderivation of $\mathcal{D}$ rooted at n. Observe that the vector derived at node $n^\dagger$ in $\mathcal{D}[n \leftarrow \mathcal{D}']$ is:

- $\widehat{\mathcal{D}'}(n')$, if $n^\dagger$ corresponds to the node n' of $\mathcal{D}'$;
- $\widehat{\mathcal{D}}(n^\dagger) - \widehat{\mathcal{D}}(n) + \widehat{\mathcal{D}'}(\varepsilon)$, if $n^\dagger$ is an ancestor of n;
- $\widehat{\mathcal{D}}(n^\dagger)$, otherwise.

When $\mathcal{D}'$ has only one leaf n, we write $\mathcal{D}; \mathcal{D}'$ instead of $\mathcal{D}'[n \leftarrow \mathcal{D}]$.

For a derivation $\mathcal{D}$ and its nodes n and n' such that n is an ancestor of n', we write $\mathcal{D}[n \leftarrow n']$ instead of $\mathcal{D}[n \leftarrow \mathcal{D}']$, where $\mathcal{D}'$ is the subderivation of $\mathcal{D}$ rooted at n'. We call such substitutions *contracting*. For two derivations $\mathcal{D}^\dagger$ and $\mathcal{D}^\ddagger$, we say that $\mathcal{D}^\ddagger$ is a *contraction* of $\mathcal{D}^\dagger$ iff $\mathcal{D}^\ddagger$ is obtained from $\mathcal{D}^\dagger$ by a finite sequence of contracting substitutions.

VAS. The classical vector addition systems can be defined as BVAS of the form $\mathcal{V} = \langle k, \{\mathbf{a}\}, R, \emptyset\rangle$, i.e. with one axiom and no binary rules. We may write them as just $\langle k, \mathbf{a}, R\rangle$.

All the definitions for BVAS apply to VAS, but they simplify. For each derivation $\mathcal{D} : \mathcal{T} \to \mathbb{Z}^k$, its underlying tree $\mathcal{T}$ is a sequence.

Restrictions and bounds. For k-dimensional X, and $I \subseteq [1, k]$, we write $X(I)$ for the "restriction of X to the set of places I", e.g.: $\mathbf{v}(I)$ is the vector obtained from $\mathbf{v}$ by removing the entries in places outside of I; $\langle k, \mathbf{a}, R\rangle(I)$ is the $|I|$-dimensional VAS obtained from $\langle k, \mathbf{a}, R\rangle$ by replacing $\mathbf{a}$ with $\mathbf{a}(I)$, and by replacing every rule $\mathbf{r} \in R$ with $\mathbf{r}(I)$; and $\mathcal{D}(I)$ is the derivation obtained from $\mathcal{D}$ by replacing, for every node n, the label $\mathcal{D}(n)$ of n with $\mathcal{D}(n)(I)$.

For $\mathbf{v} \in \mathbb{Z}^k$ and $B \in \mathbb{N}$, we say that $\mathbf{v}$ is *B-bounded* iff $\mathbf{v} \in [0, B-1]^k$. We regard a derivation *B-bounded* iff all the vectors derived at its nodes are B-bounded. Thus, B-boundedness implies admissibility.

For a k-dimensional vector or derivation X, and $I \subseteq [1, k]$, we say that X is *I-B-bounded* iff $X(I)$ is B-bounded.

Decision problems. We study the complexity of the following problems. As is standard, the input sizes are with respect to binary representations of integers.

Covering Given a BVAS $\mathcal{B}$ and a non-negative vector $\mathbf{t}$ of the same dimension, does $\mathcal{B}$ produce some $\mathbf{v}$ such that $\mathbf{v} \geq \mathbf{t}$?

Boundedness Given a BVAS, is the set of all vectors that it produces finite?

Theorem 1. *[5, 4] Covering and boundedness for VAS are* ExpSpace*-complete.*

Theorem 2. *[16] Covering and boundedness for BVAS are decidable.*

3. Upper bound for the covering problem

We say that a derivation $\mathcal{D}$ of a BVAS $\mathcal{B}$ is a *covering* of a vector $\mathbf{t}$ iff the vector that $\mathcal{D}$ derives is at least $\mathbf{t}$, i.e. $\widehat{\mathcal{D}}(\varepsilon) \geq \mathbf{t}$. Thus, the covering problem asks whether there exists an initialised admissible covering.

For VAS, Rackoff [4] established ExpSpace-membership of the covering problem by showing that, if an initialised admissible covering exists, then there must exist one of at most doubly-exponential length. Such a "short" covering can be guessed and verified in non-deterministic exponential space, and determinism is regained by Savitch's Theorem.

More precisely, Rackoff proved:

Lemma 3. *[4, Section 3] If a VAS $\langle k, \mathbf{a}, R\rangle$ has an initialised admissible covering of $\mathbf{t} \in \mathbb{N}^k$, then it has one whose length is at most $2^{(3L)^{k+1}}$, where $L = \max\{\text{size}(R), \text{size}(\mathbf{t})\}$.*

Now, consider the following proof scheme for showing that, if a k-dimensional BVAS $\mathcal{B}$ has an initialised admissible covering $\mathcal{D}$ of $\mathbf{t}$, then it has one of at most doubly-exponential height:

(i) If $\mathcal{D}$ has an excessively high leaf n, let $\mathcal{V}$ be the VAS whose axiom is $\mathcal{D}(n)$ and whose rules R are all the vectors:

- $\mathcal{D}(n')$, such that n' is on the path π from n to the root, and has one child;
- $\mathcal{D}(n') + \widehat{\mathcal{D}}(n'')$, such that n' is on π, and n'' is a child of n' not on π.

Hence, the sequence obtained from π by relabelling the nodes with two children as specified is a derivation $\mathcal{D}^\dagger$ of $\mathcal{V}$. The vectors derived along $\mathcal{D}^\dagger$ are the same as the vectors derived along π in $\mathcal{D}$, so $\mathcal{D}^\dagger$ is an initialised admissible covering of $\mathbf{t}$.

(ii) By Lemma 3, $\mathcal{V}$ has an initialised admissible covering $\mathcal{D}^\ddagger$ of $\mathbf{t}$ with length at most $2^{(3L)^{k+1}}$, where $L = \max\{\text{size}(R), \text{size}(\mathbf{t})\}$.

(iii) Let $\mathcal{D}'$ be a derivation of $\mathcal{B}$ obtained from $\mathcal{D}^\ddagger$ by undoing the linearisation done in (i), i.e. by unfolding each rule in $\mathcal{D}^\ddagger$ which is not a unary rule of $\mathcal{B}$ into a binary rule of $\mathcal{B}$ and a subderivation of $\mathcal{D}$. It is straightforward to check that $\mathcal{D}'$ is also an initialised admissible covering of $\mathbf{t}$. We repeat from (i) with $\mathcal{D}'$ instead of $\mathcal{D}$, until there are no excessively high leaves.

There are two obstacles to developing the scheme into a valid proof:

- Since the definition of R in (i) involves adding derived vectors (the ones at the nodes one edge away from the path π), we have no bound on size(R) in terms of size($\mathcal{B}$) and size($\mathbf{t}$), and therefore neither on L in (ii).
- Even if we obtain a bound on L, Lemma 3 gives us no guarantees about the shape of $\mathcal{D}^\ddagger$ in (ii) in relation to the shape of $\mathcal{D}^\dagger$. Hence, although the length of $\mathcal{D}^\ddagger$ is bounded, we are not able to deduce that, after the unfolding in (iii), $\mathcal{D}'$ has fewer excessively high leaves than $\mathcal{D}$.

The key to overcoming both obstacles is observing that, essentially, Rackoff's proof of Lemma 3 shows more than is stated in that result. Firstly, any initialised admissible covering has a contraction which is a short initialised admissible covering, and secondly, the length of the latter is bounded by the sizes of the target vector and only the negative entries in the rules of the VAS. More precisely, we have:

Lemma 4. *If a VAS $\langle k, \mathbf{a}, R\rangle$ has an initialised admissible covering $\mathcal{D}$ of $\mathbf{t} \in \mathbb{N}^k$, then it has one which is a contraction of $\mathcal{D}$ and whose length is at most* $(\max(R^-) + \max(\mathbf{t}) + 2)^{(3k)!}$.

Proof. Although this is a reworking of the proof of Lemma 3, we present it in detail to show exactly how the new conclusions are obtained.

For an initialised admissible covering $\mathcal{D}$ of $\mathbf{t} \in \mathbb{N}^k$ in a VAS $\langle k, \mathbf{a}, R\rangle$, let $m(\mathcal{D}, \mathbf{t}, \langle k, \mathbf{a}, R\rangle)$ be the smallest length of a contraction of $\mathcal{D}$ that is also an initialised admissible covering of $\mathbf{t}$ in $\langle k, \mathbf{a}, R\rangle$. Trivially, $m(\mathcal{D}, \mathbf{t}, \langle k, \mathbf{a}, R\rangle)$ is at most the length of $\mathcal{D}$. For $L, k \in \mathbb{N}$, we then let:

$$M_L(k) = \sup\big\{m(\mathcal{D}, \mathbf{t}, \langle k, \mathbf{a}, R\rangle) \;:\; \mathcal{D} \text{ is an initialised admissible covering of } \mathbf{t} \in \mathbb{N}^k \text{ in VAS } \langle k, \mathbf{a}, R\rangle, \text{ and } \max(R^-) + \max(\mathbf{t}) \leq L\big\}.$$

The set of tuples $(\mathcal{D}, \mathbf{t}, \langle k, \mathbf{a}, R\rangle)$, over which the supremum of the $m(\mathcal{D}, \mathbf{t}, \langle k, \mathbf{a}, R\rangle)$ values is taken in the definition of $M_L(k)$, is always infinite, and hence it is not *a priori* clear that the number $M_L(k)$ is well defined. The following lemma implies that, and it paves the way to an easy inductive proof of Lemma 4.

Lemma 5. *For all $L \in \mathbb{N}$, the following inequalities hold:*

$$M_L(k) \leq \begin{cases} 1 & \text{if } k = 0, \\ \big(L \cdot M_L(k-1)\big)^k + M_L(k-1) & \text{if } k \geq 1. \end{cases}$$

Proof. The case when $k = 0$ is trivial. For every $k \geq 1$, it is sufficient to prove that for every initialised admissible covering $\mathcal{D}$ of $\mathbf{t} \in \mathbb{N}^k$ in a VAS $\langle k, \mathbf{a}, R\rangle$, where $\max(R^-) + \max(\mathbf{t}) \leq L$, the following inequality holds:

$$m(\mathcal{D}, \mathbf{t}, \langle k, \mathbf{a}, R\rangle) \leq \big(L \cdot M_L(k-1)\big)^k + M_L(k-1). \tag{1}$$

Let $B = M_L(k-1) \cdot \max(R^-) + \max(\mathbf{t})$. We consider the following two cases: (a) $\mathcal{D}$ is B-bounded, and (b) $\mathcal{D}$ is not B-bounded.

Assume that $\mathcal{D}$ is B-bounded. Note that if $\widehat{\mathcal{D}}(n) = \widehat{\mathcal{D}}(n')$ and n' precedes n, then the derivation $\mathcal{D}[n \leftarrow n']$ obtained by the contracting substitution is also an initialised B-bounded covering of $\mathbf{t}$. By performing such substitutions repeatedly, then we will eventually obtain a contraction of $\mathcal{D}$ that is an initialised B-bounded covering of $\mathbf{t}$, and such that the vectors derived at its nodes are mutually distinct; the length of such a derivation is clearly at most B^k. We have now proved (1) in case (a) because

$$B^k = \big(M_L(k-1) \cdot \max(R^-) + \max(\mathbf{t})\big)^k \leq \big(L \cdot M_L(k-1)\big)^k,$$

where the inequality follows from the assumption that $\max(R^-) + \max(\mathbf{t}) \leq L$.

We now handle case (b), i.e., when $\mathcal{D}$ is not B-bounded. In this case there are derivations: $\mathcal{D}_1$ in the VAS $\langle k, \mathbf{a}, R\rangle$, and $\mathcal{D}_2$ in the VAS $\langle k, \widehat{\mathcal{D}_1}(\varepsilon), R\rangle$, such that:

- $\mathcal{D} = \mathcal{D}_1; \mathcal{D}_2$,
- $\mathcal{D}_1$ is B-bounded except for the vector $\widehat{\mathcal{D}_1}(\varepsilon)$ derived at its last node,
- $\widehat{\mathcal{D}_1}(\varepsilon)$ is not B-bounded because there is a place $i \in [1, k]$ such that $\widehat{\mathcal{D}_1}(\varepsilon)(i) \geq B$.

Observe that, as in case (a), we can choose a contraction $\mathcal{D}'_1$ of $\mathcal{D}_1$ that is an initialised derivation of $\widehat{\mathcal{D}_1}(\varepsilon)$, B-bounded except for the vector derived at its last node, and of length at most $B^k + 1$. Moreover, letting $I = [1, k] \setminus \{i\}$, note that $\mathcal{D}_2(I)$ is an initialised admissible covering of $\mathbf{t}(I)$ in the VAS $\langle k, \widehat{\mathcal{D}_1}(\varepsilon), R\rangle(I)$, and hence there is a contraction $\mathcal{D}'_2$ of $\mathcal{D}_2$ such that $\mathcal{D}'_2(I)$ is also an initialised admissible covering of $\mathbf{t}(I)$, and of length at most $M_L(|I|) = M_L(k-1)$.

Observe that $\mathcal{D}'_1; \mathcal{D}'_2$ is a contraction of $\mathcal{D}$, and that it is of length at most

$$B^k + M_L(k-1) \leq \big(L \cdot M_L(k-1)\big)^k + M_L(k-1).$$

In order to establish (1) in case (b) we argue that $\mathcal{D}'_1; \mathcal{D}'_2$ is a initialised admissible covering of $\mathbf{t}$ in $\langle k, \mathbf{a}, R\rangle$. It suffices to prove that for every node n in $\mathcal{D}'_2$, we have $\widehat{\mathcal{D}'_2}(n)(i) \geq \mathbf{t}(i) \geq 0$. This follows from

$$\widehat{\mathcal{D}'_1}(\varepsilon)(i) = \widehat{\mathcal{D}_1}(\varepsilon)(i) \geq B \geq \big(M_L(k-1) - 1\big) \cdot \max(R^-) + \max(\mathbf{t}),$$

and the number of applications of rules in $\mathcal{D}'_2$ being at most $M_L(k-1) - 1$. □

Now, to prove Lemma 4, we show by induction on $k \in \mathbb{N}$ that $M_\ell(k) \leq \ell^{(3k)!}$, where $\ell = \max(R^-) + \max(\mathbf{t}) + 2$. The base case, when $k = 0$, is trivial.[4] If we assume $M_\ell(k-1) \leq \ell^{(3(k-1))!}$ then we have:

$$M_\ell(k) \leq \big(\ell \cdot M_\ell(k-1)\big)^k + M_\ell(k-1) \leq \big(\ell \cdot M_\ell(k-1)\big)^{k+1} \leq \big(\ell^{1+(3(k-1))!}\big)^{k+1} \leq \ell^{(3k)!},$$

where the first inequality holds by Lemma 5, the second is true because $\ell \geq 2$, and the third follows from the inductive hypothesis. □

We are now in a position to show that, indeed, if a given BVAS has an initialised admissible covering of a given vector of non-negative integers, then it has one of at most doubly-exponential height. Although that is all that is required in this article, we can easily infer a little more:

Lemma 6. *If a BVAS $\langle k, A_0, R_1, R_2\rangle$ has an initialised admissible covering $\mathcal{D}$ of $\mathbf{t} \in \mathbb{N}^k$, then it has one which is a contraction of $\mathcal{D}$ and whose height is at most* $(\max((R_1 \cup R_2)^-) + \max(\mathbf{t}) + 2)^{(3k)!}$.

Proof. We follow the scheme in (i)–(iii) for $\mathcal{B} = \langle k, A_0, R_1, R_2\rangle$, with "excessively high" replaced by "of height more than $(\max((R_1 \cup R_2)^-) + \max(\mathbf{t}) + 2)^{(3k)!}$", and with the application of Lemma 3 in (ii) replaced by an application of Lemma 4.

Let $\mathcal{D}$, n, $\mathcal{V} = \langle k, \mathcal{D}(n), R\rangle$ and $\mathcal{D}^\dagger$ be as in (i). Since $\mathcal{D}$ is admissible, we have that, in particular for all nodes n'' that are one edge away from the path π from n to the root, $\widehat{\mathcal{D}}(n'') \geq \mathbf{0}$. Hence, $\max(R^-) \leq \max((R_1 \cup R_2)^-)$, and so by Lemma 4, $\mathcal{V}$ has an initialised admissible covering $\mathcal{D}^\ddagger$ of $\mathbf{t}$, which is a contraction of $\mathcal{D}^\dagger$ and whose length is at most

$$\big(\max(R^-) + \max(\mathbf{t}) + 2\big)^{(3k)!} \leq \big(\max((R_1 \cup R_2)^-) + \max(\mathbf{t}) + 2\big)^{(3k)!}.$$

[4] Recall that $0! = 1$.

As outlined in (iii), $\mathcal{D}^\ddagger$ can be unfolded into an initialised admissible covering $\mathcal{D}'$ of $\mathbf{t}$ in $\mathcal{B}$. By taking care that, for each node in $\mathcal{D}^\ddagger$ that corresponds to a node n' in $\mathcal{D}$ with a child n'' not on the path π, the unfolding is performed so that the subderivation of $\mathcal{D}$ rooted at n'' is attached on the same side as n'' is in relation to π, we obtain $\mathcal{D}'$ which is in addition a contraction of $\mathcal{D}$.

Let n' be the leaf of $\mathcal{D}'$ that was obtained from the unique leaf of $\mathcal{D}^\ddagger$. The height of n' equals the length of $\mathcal{D}^\ddagger$, so it is not excessively high. By the properties of $\mathcal{D}'$, there is an injection ι from the leaves of $\mathcal{D}'$ to the leaves of $\mathcal{D}$ which does not decrease heights and such that $\iota(n') = n$. Since the height of n is excessively high, we conclude that $\mathcal{D}'$ has fewer excessively high leaves than $\mathcal{D}$, as required. □

Therefore, to decide the covering problem, it suffices to search for an initialised admissible covering of at most doubly-exponential height. Note, however, that the size of a binary tree of doubly-exponential height can be triply exponential, and hence vectors derived in a derivation of doubly-exponential height may contain triply-exponential entries. In order to prove the main result of this section, i.e., that the covering problem for BVAS is in 2ExpTime, we need to avoid having to manipulate such large numbers. That is achieved by our next result, Proposition 7, which shows that whether a derivation is admissible and a covering can be verified accurately with arithmetic where values that are not less than a sufficiently large bound B are replaced by ∞.

Let us first, for integral a and B, denote by $\overline{a}^B$ the *B-truncation* of a:

$$\overline{a}^B = \begin{cases} a & \text{if } a < B, \\ \infty & \text{otherwise.} \end{cases}$$

For an integral vector $\mathbf{v}$, we define its B-truncation $\overline{\mathbf{v}}^B$ pointwise. We then let, for a derivation $\mathcal{D} : \mathcal{T} \to \mathbb{Z}^k$, its *$B$-derived* vectors $\widehat{\mathcal{D}}^B(n)$ be obtained by B-truncating its derived vectors and propagating ∞ entries along all simple paths to the root, as follows. Here, the B-truncation of ∞ is ∞, and sums that contain ∞ evaluate to ∞:

- if n is a leaf in $\mathcal{T}$, then $\widehat{\mathcal{D}}^B(n) = \overline{\mathcal{D}(n)}^B$;
- if n has one child n' in $\mathcal{T}$, then $\widehat{\mathcal{D}}^B(n) = \overline{\mathcal{D}(n) + \widehat{\mathcal{D}}^B(n')}^B$;
- if n has two children n' and n'' in $\mathcal{T}$, then $\widehat{\mathcal{D}}^B(n) = \overline{\mathcal{D}(n) + \widehat{\mathcal{D}}^B(n') + \widehat{\mathcal{D}}^B(n'')}^B$.

Proposition 7. *Suppose $\mathcal{B} = \langle k, A_0, R_1, R_2 \rangle$ is a BVAS, $\mathbf{t} \in \mathbb{N}^k$, $\mathcal{D}$ is a derivation in $\mathcal{B}$ of height at most H, and $B \geq H \cdot \max((R_1 \cup R_2)^-) + \max(\mathbf{t})$. Then $\mathcal{D}$ is an admissible covering of $\mathbf{t}$ iff, for each node n in $\mathcal{D}$, $\widehat{\mathcal{D}}^B(n) \geq \mathbf{0}$, and $\widehat{\mathcal{D}}^B(\varepsilon) \geq \mathbf{t}$.*

Proof. That $\mathcal{D}$ being admissible and a covering of $\mathbf{t}$ implies $\widehat{\mathcal{D}}^B(n) \geq \mathbf{0}$ for all n and $\widehat{\mathcal{D}}^B(\varepsilon) \geq \mathbf{t}$ is trivial, since for each n and i, $\widehat{\mathcal{D}}^B(n)(i)$ either equals $\widehat{\mathcal{D}}(n)(i)$ or is ∞.

For the other direction, it suffices to consider n and i such that $\widehat{\mathcal{D}}^B(n)(i) = \infty$. We can also assume $(*)$ that $\widehat{\mathcal{D}}(n')(i) \geq 0$ has been shown for all descendents n' of n. Since ∞ entries propagate pointwise towards the root, there must exist $n^\dagger$ which is either n or a descendent of n, such that $\widehat{\mathcal{D}}^B(n^\dagger)(i) = \infty$, and which does not have a descendent with the same property. From the definition of B-derived vectors, we have that $\widehat{\mathcal{D}}(n^\dagger)(i) \geq B$. Recalling the assumption $(*)$ and that the number of additions of a rule of $\mathcal{B}$ along the simple path from $n^\dagger$ to n is less than H, we have

$$\widehat{\mathcal{D}}(n)(i) \geq \widehat{\mathcal{D}}(n^\dagger)(i) - H \cdot \max((R_1 \cup R_2)^-) \geq B - H \cdot \max((R_1 \cup R_2)^-) \geq \max(\mathbf{t}) \geq \mathbf{t}(i) \geq 0,$$

as required. □

Theorem 8. *Covering for BVAS is in* 2ExpTime.

Proof. Let $\mathcal{B} = \langle k, A_0, R_1, R_2 \rangle$ be a BVAS and $\mathbf{t} \in \mathbb{N}^k$. Let $N = \text{size}(\mathcal{B}) + \text{size}(\mathbf{t})$. If

$$\ell = \max((R_1 \cup R_2)^-) + \max(\mathbf{t}) + 2$$

then $\ell \leq 2^N$, and without any loss of generality we can assume that $3k \leq N$.

Lemma 6 implies that if there is an initialised admissible covering of $\mathbf{t}$ in $\mathcal{B}$ then there is one of height at most $\ell^{(3k)!} \leq (2^N)^{N!} \leq 2^{2^{\mathsf{C}_1 N \log N}}$, for some constant $\mathsf{C}_1 > 1$. If we set $H = 2^{2^{\mathsf{C}_1 N \log N}}$ and $B = H^2$, then from Proposition 7 it follows that in order to establish existence of an initialised admissible covering of $\mathbf{t}$ in $\mathcal{B}$, it suffices to:

- guess an initialised derivation $\mathcal{D}$ in $\mathcal{B}$ of height at most H;
- guess the B-derived vectors at all nodes in $\mathcal{D}$, and for every node and its children, verify that they satisfy the equations defining B-derived vectors, and that they are non-negative;
- verify that the B-derived vector at the root covers $\mathbf{t}$.

We argue that the guessing and verification of such a structure can be carried out by an alternating Turing machine with exponential space, and hence the covering problem is in 2ExpTime [19]. The alternating Turing machine starts at the root of the derivation, it uses non-deterministic states to guess the rules labelling the current node and its children, and their B-derived vectors, and it uses universal states to proceed with the guessing and verification process to both children (for nodes labelled by binary rules) in parallel. All those tasks can indeed be carried out by a Turing machine with only exponential space because it can represent—in binary—and manipulate numbers of doubly-exponential magnitude. □

We remark that, for BVAS whose dimension is fixed, covering is in ExpTime. That follows from the proof of Theorem 8, where if k is fixed then H and B are only singly exponential in N, so polynomial space suffices for the alternating Turing machine.

4. Upper bound for the boundedness problem

Let us say that a derivation $\mathcal{D}$ is *self-covering* iff, for some node n, the vector derived at n is less than or equal to the one at the root, and less in at least one place, i.e. $\widehat{\mathcal{D}}(n) < \widehat{\mathcal{D}}(\varepsilon)$.

The following fact tells us that boundedness is equivalent to non-existence of an initialised admissible self-covering derivation. The "if" part is easy. The "only if" part was inferred by Verma and Goubault-Larrecq, using the properties of their extension of Karp and Miller's procedure.

Theorem 9. *[16] A BVAS produces infinitely many vectors iff it has an initialised admissible self-covering derivation.*

In the simpler setting of VAS, to conclude that boundedness is in ExpSpace, Rackoff showed that if an initialised admissible self-covering derivation exists, then there exists one of at most doubly-exponential length:

Lemma 10. *[4, Section 4] If a VAS $\mathcal{V} = \langle k, \mathbf{a}, R\rangle$ has an initialised admissible self-covering derivation, then it has one whose length is at most $2^{2^{\mathsf{C}_2 L \log L}}$, where $L = \mathrm{size}(R)$ and C_2 is some constant.*

Starting from the same simple idea as in Section 3, consider the following scheme for proving that, if a BVAS $\mathcal{B} = \langle k, A_0, R_1, R_2\rangle$ has an initialised admissible self-covering derivation $\mathcal{D}$, then it has one of at most doubly-exponential height:

(I) Let node n be such that $\widehat{\mathcal{D}}(n) < \widehat{\mathcal{D}}(\varepsilon)$, and pick a simple path π in $\mathcal{D}$ which is from a leaf to the root and passes through n. Let $\mathcal{V}$ be the VAS defined as in (i) in Section 3, i.e. its axiom is the label of the leaf of π and its rules R are obtained by linearising the binary rules on π. Thus, $\mathcal{V}$ has a derivation $\mathcal{D}^\dagger$ whose sequence of derived vectors is the same as the sequence of derived vectors along π in $\mathcal{D}$. In particular, $\mathcal{D}^\dagger$ is initialised, admissible and self-covering.

(II) By Lemma 10, $\mathcal{V}$ has an initialised admissible self-covering derivation $\mathcal{D}^\ddagger$ whose length is at most $2^{2^{\mathsf{C}_2 L \log L}}$, where $L = \mathrm{size}(R)$.

(III) Let $\mathcal{D}'$ be a derivation of $\mathcal{B}$ obtained from $\mathcal{D}^\ddagger$ by undoing the linearisation done in (I), as in (iii) in Section 3, and let π' be the path in $\mathcal{D}'$ that is from a leaf to the root and corresponds to $\mathcal{D}^\ddagger$. It is straightforward to check that $\mathcal{D}'$ is also initialised, admissible and self-covering.

(IV) Let H be the length of π', which equals the length of $\mathcal{D}^\ddagger$. For each node n' that is one edge away from π' in $\mathcal{D}'$ (i.e., that was attached in (III)), the subderivation of $\mathcal{D}'$ rooted at n' is an initialised admissible covering of $\min((H-1)\cdot\max(R^-)+1, \widehat{\mathcal{D}'}(n'))$. By Lemma 6, $\mathcal{B}$ has an initialised admissible

covering $\mathcal{D}^*_{n'}$ of the same vector, whose height is at most

$$\begin{aligned}\left(\max((R_1 \cup R_2)^-) + \max\left(\min\left((H-1)\cdot\max(R^-)+1, \widehat{\mathcal{D}'}(n')\right)\right)+2\right)^{(3k)!} \\ \leq \left(\max((R_1 \cup R_2)^-) + (H-1)\cdot\max(R^-)+3\right)^{(3k)!} \\ \leq \left(H\cdot\max((R_1 \cup R_2)^-)+3\right)^{(3k)!}.\end{aligned}$$

Let $\mathcal{D}''$ be obtained from $\mathcal{D}'$ by performing each substitution $[n' \leftarrow \mathcal{D}^*_{n'}]$. The threshold $(H-1)\cdot\max(R^-)+1$ is such that $\mathcal{D}''$ is still admissible and self-covering, certainly it is still initialised, and $H + (H\cdot\max((R_1 \cup R_2)^-)+3)^{(3k)!}$ bounds its height.

Of course, we have the same problem as the first one in Section 3: we have no bound on size(R) in terms of size($\mathcal{B}$), and therefore neither on H in (IV). Seeking therefore a refinement of Lemma 10, we find that the key ingredient in its proof is:

Lemma 11. *[4, Lemma 4.5] Suppose $\mathcal{V} = \langle k, \mathbf{a}, R\rangle$ is a VAS, $I \subseteq [1,k]$ and $B > 1$. If $\mathcal{V}$ has an initialised I-B-bounded self-covering derivation, then it has one whose length is at most $B^{(\mathrm{size}(R))^{\mathsf{C}_3}}$, where C_3 is some constant.*

In turn, at the centre of the proof of Lemma 11, Rackoff invokes the following theorem of Borosh and Treybig on small solutions of integer linear programming problems. Recall that the interval notations denote sets of integers.

Theorem 12. *[20] Let $\mathbf{A} \in (-m,m)^{k\times n}$ and $\mathbf{b} \in (-m,m)^k$, where $k,n,m \in \mathbb{N}$. If there exists $\mathbf{x} \in \mathbb{N}^n$ such that $\mathbf{Ax} \geq \mathbf{b}$, then there exists $\mathbf{y} \in [0, (\max\{n,m\})^{\mathsf{C}_4 k}]^n$ such that $\mathbf{Ay} \geq \mathbf{b}$, where C_4 is some constant.*

When we examine feeding a VAS $\langle k, \mathbf{a}, R\rangle$ for which we have a bound on $\max(R^-)$ but not on $\max(R^+)$ into Rackoff's proof of Lemma 11, we discover that Theorem 12 is invoked for bounded k, unbounded n, $\mathbf{A}$ whose entries are bounded below but not above, and $\mathbf{b}$ whose entries are bounded above but not below. Surprisingly, this is where we can make progress. We now show that, if we can afford roughly one exponential more, small solutions exist for $\mathbf{A}$ and $\mathbf{b}$ which are only one-sidedly bounded by m. Moreover, the number of non-zero entries in the small solutions and their values are bounded only in terms of k and m.

Theorem 13. *Let $\mathbf{A} \in (-m,\infty)^{k\times n}$ and $\mathbf{b} \in (-\infty,m)^k$, where $k,n,m \in \mathbb{N}$. If there exists $\mathbf{x} \in \mathbb{N}^n$ such that $\mathbf{Ax} \geq \mathbf{b}$, then there exists $\mathbf{y} \in [0,L]^n$ such that $|\mathrm{supp}(\mathbf{y})| \leq L$ and $\mathbf{Ay} \geq \mathbf{b}$, where $L = m^{2^{\mathsf{C}_5 k^2}}$ and C_5 is some constant.*

In order to reformulate Theorem 13 so that it becomes amenable to a proof by induction on k (cf. Lemma 15), we define $F_k(m)$, for all integers $k \geq 1$ and $m \geq 2$, by:

$$F_k(m) = \begin{cases} m & \text{if } k = 1, \\ \left(F_{k-1}(2m)\right)^{4\mathsf{C}_4 k^2} & \text{if } k > 1, \end{cases}$$

where C_4 is the constant from Theorem 12, which we can assume is at least 1.

Proposition 14. *For all integers $k \geq 1$ and $m \geq 2$, we have $F_k(m) \leq m^{(4\mathsf{C}_4)^k \cdot (2k)!}$.*

Proof. The proof is by induction on k. The base case, i.e., when $k = 1$, is trivial. If we assume that $F_{k-1}(m) \leq m^{(4\mathsf{C}_4)^{k-1} \cdot (2(k-1))!}$ for all integers $m \geq 2$, then we have:

$$F_k(m) = \big(F_{k-1}(2m)\big)^{4\mathsf{C}_4 k^2} \leq \big((2m)^{(4\mathsf{C}_4)^{k-1} \cdot (2(k-1))!}\big)^{4\mathsf{C}_4 k^2} \leq m^{(4\mathsf{C}_4)^k \cdot (2k)!},$$

where the equality holds by the definition of $F_k(m)$, and the first inequality by the inductive hypothesis. □

Observe that there is a constant C_5 such that, for all integers $k \geq 1$ and $m \geq 2$, we have $F_k(m) \leq m^{(4\mathsf{C}_4)^k \cdot (2k)!} \leq m^{2^{\mathsf{C}_5 k^2}}$. Hence, and since Theorem 13 is true trivially when $k = 0$ or $m \leq 1$, Theorem 13 follows from the following lemma.

Lemma 15. *Let $\mathbf{A} \in (-m, \infty)^{k \times n}$ and $\mathbf{b} \in (-\infty, m)^k$, where $k \geq 1$ and $m \geq 2$. If there exists $\mathbf{x} \in \mathbb{N}^n$ such that $\mathbf{A}\mathbf{x} \geq \mathbf{b}$, then there exists $\mathbf{y} \in [0, F_k(m)]^n$ such that $|\mathrm{supp}(\mathbf{y})| \leq F_k(m)$ and $\mathbf{A}\mathbf{y} \geq \mathbf{b}$.*

Proof. We can assume without any loss of generality that, for each $j \in [1, n]$, there exists $\mathbf{x} \in \mathbb{N}^n$ such that $\mathbf{A}\mathbf{x} \geq \mathbf{b}$ and $\mathbf{x}(j) \geq 1$. Otherwise, consider $\mathbf{A}' = \mathbf{A}_{\bullet(-j)}$, where there exists no $\mathbf{x} \in \mathbb{N}^n$ such that $\mathbf{A}\mathbf{x} \geq \mathbf{b}$ and $\mathbf{x}(j) \geq 1$.

The proof is by induction on k. First we consider the base case when $k = 1$. If $\mathbf{b} \leq 0$ then $\mathbf{A}\mathbf{y} \geq \mathbf{b}$ for $\mathbf{y} = \mathbf{0}$. If, however, $\mathbf{b} > 0$ then the existence of $\mathbf{x} \in \mathbb{N}^n$ such that $\mathbf{A}\mathbf{x} \geq \mathbf{b}$ implies that there must be $i \in [1, n]$ such that $\mathbf{A}(1, i) > 0$. Then, we have $\mathbf{A}\mathbf{y} \geq \mathbf{b}$ for $\mathbf{y} = m \cdot \mathbf{e}_i$.

For the inductive step we consider the following three cases. Essentially, if either $\mathbf{b}$ contains a large negative entry or $\mathbf{A}$ contains a large positive entry, then we remove that row of $\mathbf{A}$ and argue by the inductive hypothesis and the largeness of the entry. Otherwise, we have a lower bound for all entries of $\mathbf{b}$ and an upper bound for all entries of $\mathbf{A}$, and we invoke Theorem 12.

Case 1: There exists $i \in [1, k]$ such that $\mathbf{b}(i) \leq -m \cdot (F_{k-1}(m))^2$. Let $\mathbf{A}' = \mathbf{A}_{(-i)\bullet}$ and let $\mathbf{b}' = \mathbf{b}_{-i}$. By the inductive hypothesis, there exists $\mathbf{y} \in [0, F_{k-1}(m)]^n$—and hence $\mathbf{y} \in [0, F_k(m)]^n$—such that $|\mathrm{supp}(\mathbf{y})| \leq F_{k-1}(m) < F_k(m)$ and $\mathbf{A}'\mathbf{y} \geq \mathbf{b}'$. The assumption that $\mathbf{A}(i, j) > -m$ for all $j \in [1, n]$ then implies that $\mathbf{A}_{i\bullet}\mathbf{y} > -m \cdot (F_{k-1}(m))^2 \geq \mathbf{b}(i)$, and hence we have $\mathbf{A}\mathbf{y} \geq \mathbf{b}$.

Case 2: There exist $i \in [1, k]$ and $j \in [1, n]$ such that $\mathbf{A}(i, j) \geq 2m \cdot (F_{k-1}(2m))^2$, and there exists $\mathbf{x} \in \mathbb{N}^n$ such that $\mathbf{A}\mathbf{x} \geq \mathbf{b}$ and $\mathbf{x}(j) \geq 1$. Let $\mathbf{A}' = \mathbf{A}_{(-i)\bullet}$, let $\mathbf{b}' = \mathbf{b}_{-i}$, and let $\mathbf{b}'' = \mathbf{b}' - \mathbf{A}_{(-i)j}$. Note that $\mathbf{A}'(\mathbf{x} - \mathbf{e}_j) \geq \mathbf{b}''$ and that, since $\mathbf{x}(j) \geq 1$, we have $\mathbf{x} - \mathbf{e}_j \in \mathbb{N}^n$. Observe also that $\mathbf{b}'' \in (-\infty, 2m)^{k-1}$ and hence, by the inductive hypothesis, there exists $\mathbf{y} \in [0, F_{k-1}(2m)]^n$ such that $|\mathrm{supp}(\mathbf{y})| \leq F_{k-1}(2m)$ and $\mathbf{A}'\mathbf{y} \geq \mathbf{b}''$.

Let $\mathbf{z} = \mathbf{y} + \mathbf{e}_j$. Note that then $\mathbf{z} \in [0, F_{k-1}(2m) + 1]^n \subseteq [0, F_k(m)]^n$ and $|\mathrm{supp}(\mathbf{y})| \leq F_{k-1}(2m) + 1 \leq F_k(m)$, and hence we only need to establish that $\mathbf{Az} \geq \mathbf{b}$. We have:

$$(\mathbf{Az})(i) = \mathbf{A}_{i\bullet}(\mathbf{y} + \mathbf{e}_j) \geq \mathbf{A}(i,j) - m \cdot (F_{k-1}(2m))^2 \geq m \cdot (F_{k-1}(2m))^2 \geq m \geq \mathbf{b}(i),$$

where the first inequality follows from $\mathbf{A} \in (-m, \infty)^{k \times n}$, from $\mathbf{y} \in [0, F_{k-1}(2m)]$, and from $|\mathrm{supp}(\mathbf{y})| \leq F_{k-1}(2m)$; and the second inequality follows from the assumption that $\mathbf{A}(i,j) \geq 2m \cdot (F_{k-1}(2m))^2$. Moreover, we have:

$$(\mathbf{Az})_{-i} = \mathbf{A}'(\mathbf{y} + \mathbf{e}_j) = \mathbf{A}'\mathbf{y} + \mathbf{A}_{(-i)j} \geq \mathbf{b}'' + \mathbf{A}_{(-i)j} = \mathbf{b}' = \mathbf{b}_{-i}.$$

Case 3: Neither Case 1 nor Case 2 applies. Observe that, in this case, every column of $\mathbf{A}$ is in $[-m, 2m \cdot (F_{k-1}(2m))^2]^k$, and $\mathbf{b} \in [-m \cdot (F_{k-1}(m))^2, m]^k$. The number of distinct columns of $\mathbf{A}$ is therefore at most $(3m \cdot (F_{k-1}(2m))^2)^k \leq (F_{k-1}(2m))^{4k}$, and so without loss of generality we may assume $n \leq (F_{k-1}(2m))^{4k}$. By Theorem 12, there exists $\mathbf{y} \in [0, F_{k-1}(2m)^{4\mathsf{C}_4 k^2}]^n = [0, F_k(m)]^n$ such that $|\mathrm{supp}(\mathbf{y})| \leq (F_{k-1}(2m))^{4k} \leq F_k(m)$ and $\mathbf{Ay} \geq \mathbf{b}$. □

Having proved Theorem 13, we can use it to obtain a revision of Lemma 11, where the dependence of the bound on the size of the set of rules is replaced by dependences on the minimum negative entry in a rule and the dimension. The price to pay is that the revised bound is doubly exponential in the dimension.

Lemma 16. *Suppose $\mathcal{V} = \langle k, \mathbf{a}, R \rangle$ is a VAS, $I \subseteq [1,k]$ and $B > 1$. If $\mathcal{V}$ has an initialised I-B-bounded self-covering derivation, then it has one of length at most $((\max(R^-)+1) \cdot B)^{2^{\mathsf{C}_6 k^2}}$, where C_6 is some constant.*

Proof. Let $d = |I|$, and let $\mathcal{D}$ be a minimal (i.e., shortest) initialised I-B-bounded self-covering derivation. By the self-covering property of $\mathcal{D}$, we can decompose it as $\mathcal{D}_1; \mathcal{D}_2$, where $\mathcal{D}_1$ and $\mathcal{D}_2$ are I-B-bounded derivations, the vector $\widehat{\mathcal{D}_1}(\varepsilon)$ derived at the last node of $\mathcal{D}_1$ is the label of the first node of $\mathcal{D}_2$, and is less than the vector $\widehat{\mathcal{D}_2}(\varepsilon)$ derived at the last node of $\mathcal{D}_2$. By the minimality of $\mathcal{D}$, the length of $\mathcal{D}_1$ is at most $B^d \leq B^k$ (cf. the proof of Lemma 5).

We claim that the length of $\mathcal{D}_2$ is at most

$$(B^k + 1)^2 + B^k \cdot \left(\left((\max(R^-) + 1) \cdot (B^k + 1)^2 \right)^2 \right)^{2^{\mathsf{C}_5 k^2}},$$

where C_5 is the constant from Theorem 13, which implies the lemma since

$$B^k + (B^k + 1)^2 + B^k \cdot \left(\left((\max(R^-) + 1) \cdot (B^k + 1)^2 \right)^2 \right)^{2^{\mathsf{C}_5 k^2}} \leq ((\max(R^-) + 1) \cdot B)^{2^{\mathsf{C}_6 k^2}},$$

where C_6 is some constant.

The argument for the claim follows Rackoff's proof of Lemma 11, except that it uses Theorem 13 instead of Theorem 12. We therefore skip some details that are the same as in the original.

The following notions will be useful, where $\mathcal{D}'$ is a derivation of $\mathcal{V}$:

- a *segment* of $\mathcal{D}'$, from a node n to a node n', is a derivation $\mathcal{D}^\dagger$ whose first node (i.e., leaf) is labelled by $\widehat{\mathcal{D}'}(n)$ and whose remaining nodes are labelled by the sequence of rules in $\mathcal{D}'$ from node n'' to node n', where n'' is next after n;
- the *effect* of $\mathcal{D}'$, written $\Delta(\mathcal{D}')$, is the sum of its rules, which equals the difference between its last and first derived vectors;
- $\mathcal{D}'$ is an *I-loop* iff $\Delta(\mathcal{D}')(I) = \mathbf{0}$;
- $\mathcal{D}'$ is a *simple* I-loop iff it is an I-loop and no proper segment of it is an I-loop;
- a segment $\mathcal{D}^\dagger$ of $\mathcal{D}'$ from n to n' is a *light* I-loop iff it is an I-loop and, for every node n'' in the interior of the segment, there exists n''' outside of the interior such that $\widehat{\mathcal{D}'}(n'')(I)$ equals $\widehat{\mathcal{D}'}(n''')(I)$;
- *deleting* a segment of $\mathcal{D}'$, which is from n to n', results in the contraction $\mathcal{D}'[n' \leftarrow n]$;
- *inserting* a derivation $\mathcal{D}^\dagger$ of $\mathcal{V}$ into $\mathcal{D}'$ at a node n inserts the sequence of rules in $\mathcal{D}^\dagger$ immediately after n in $\mathcal{D}'$ (the first node of $\mathcal{D}^\dagger$ is irrelevant).

The rest of the proof consists of three stages: analysing $\mathcal{D}_2$ to obtain a solution to a certain integer programming problem, applying Theorem 13 to get a small solution to the same problem, and synthesising from the small solution a short I-B-bounded derivation $\mathcal{D}_2''$ of $\mathcal{V}$ whose initial vector is the same as that of $\mathcal{D}_2$ and is strictly covered by the last derived vector in $\mathcal{D}_2''$.

For the first stage, observe that:

- if $\mathcal{D}_2$ does not have a segment of length at least 2 which is a light simple I-loop, then its length is at most $(B^d + 1)^2 \leq (B^k + 1)^2$;
- deleting any light simple I-loop does not alter the set of all I-restrictions of the derived vectors in $\mathcal{D}_2$.

Hence, starting with $\mathcal{D}_2$, there exists a sequence of deletions of light simple I-loops of length at least 2, which finishes with a derivation $\mathcal{D}_2'$ whose length is at most $(B^k + 1)^2$ and for which the set of all I-restrictions of its derived vectors is the same as for $\mathcal{D}_2$. Let $E \subseteq \mathbb{Z}^k$ be the set of all effects of the deleted I-loops (of course, their I-restrictions equal $\mathbf{0}$), and for each $e \in E$:

- let $\mathbf{x}_e$ be the number of deleted I-loops whose effect is e;
- let $\mathcal{D}_e^\dagger$ be some deleted I-loop whose effect is e.

By the definition of $\mathcal{D}_2'$ and the self-covering property of $\mathcal{D}_2$, we have $\Delta(\mathcal{D}_2) = \Delta(\mathcal{D}_2') + \sum_{e \in E} \mathbf{x}_e \cdot e > 0$,

i.e., there is $i \in [1,k]$ such that:

$$\left(\sum_{e\in E} \mathbf{x}_e \cdot e\right)([1,k] \setminus \{i\}) \quad \geq \quad -\Delta(\mathcal{D}'_2)([1,k] \setminus \{i\}), \tag{2}$$

$$\left(\sum_{e\in E} \mathbf{x}_e \cdot e\right)(i) \quad \geq \quad 1 - \Delta(\mathcal{D}'_2)(i). \tag{3}$$

The system of inequalities (2)–(3) states that $\mathbf{x} \in \mathbb{N}^{|E|}$ satisfies $\mathbf{A}\mathbf{x} \geq \mathbf{b}$, where $\mathbf{A} \in [-\max(R^-) \cdot B^k, \infty)^{k\times|E|}$ since the length of any simple I-loop is at most B^k+1, and $\mathbf{b} \in (-\infty, \max(R^-)\cdot(B^k+1)^2+1]^k$ since the length of $\mathcal{D}'_2$ is at most $(B^k+1)^2$. Letting $m = (\max(R^-)+1)\cdot(B^k+1)^2$ and applying Theorem 13, we get $\mathbf{y} \in [0,L]^{|E|}$ such that $|\mathrm{supp}(\mathbf{y})| \leq L$ and $\mathbf{A}\mathbf{y} \geq \mathbf{b}$, where $L = m^{2^{\mathsf{c}_5 k^2}}$.

For the last stage, recall that the set of all I-restrictions of the derived vectors in $\mathcal{D}'_2$ is the same as for $\mathcal{D}_2$. Let $\mathcal{D}''_2$ be obtained from $\mathcal{D}'_2$ by, for each $e \in E$, inserting $\mathbf{y}_e$ times the I-loop $\mathcal{D}^\dagger_e$ at some node for which the I-restriction of its derived vector equals the I-restriction of the first and last derived vectors in $\mathcal{D}^\dagger_e$ (those two I-restrictions are the same). It is straightforward to check that $\mathcal{D}''_2$ is I-B-bounded and that its initial vector is the same as that of $\mathcal{D}_2$. The latter is also strictly covered by the last derived vector in $\mathcal{D}''_2$ because $\mathbf{y}$ satisfies the inequalities (2)–(3). It remains to observe that the length of $\mathcal{D}''_2$ is at most $(B^k+1)^2 + B^k \cdot L^2$, which establishes the claim by the assumed minimality of $\mathcal{D}$. □

The final step in obtaining a revision of Lemma 10 that we can apply to VAS whose rules are bounded below but not above is to substitute in its proof uses of Lemma 11 by uses of Lemma 16. That yields the next result, which shows that we could indeed afford the extra exponential in Theorem 13. Although it has filtered through to Lemma 16, it disappears in the following proof.

Lemma 17. *If a VAS $\mathcal{V} = \langle k, \mathbf{a}, R\rangle$ has an initialised admissible self-covering derivation, then it has one of length at most $(2(\max(R^-)+1))^{2^{\mathsf{c}_7 k^3}}$, where c_7 is some constant.*

Proof. Given a VAS $\mathcal{V} = \langle k, \mathbf{a}, R\rangle$ and $I \subseteq [1,k]$, let $m(\langle k, \mathbf{a}, R\rangle, I)$ be the smallest length of an initialised self-covering derivation $\mathcal{D}$ such that $\mathcal{D}(I)$ is admissible in $\mathcal{V}(I)$. If there is none, by convention $m(\langle k, \mathbf{a}, R\rangle, I) = 0$.

For $L \geq 2$ and $i \in \mathbb{N}$, we then let:

$$m_L(i) = \sup\big\{m(\langle k, \mathbf{a}, R\rangle, I) \ : \ |I| = i,\ \langle k, \mathbf{a}, R\rangle \text{ is a VAS and } \max(R^-)+1 \leq L\big\}.$$

The set over which the supremum of the $m(\langle k, \mathbf{a}, R\rangle, I)$ values is taken in the definition of $m_L(i)$ is always infinite, and hence it is not *a priori* clear that the number $m_L(i)$ is well defined.

By Lemma 16, $m_L(0) \leq (2L)^{2^{\mathsf{c}_6 k^2}}$.

Suppose $\mathcal{D}$ is an initialised self-covering derivation in a VAS $\mathcal{V} = \langle k, \mathbf{a}, R\rangle$, such that $\mathcal{D}(I)$ is admissible in $\mathcal{V}(I)$, $\max(R^-)+1 \leq L$ and $|I| = i+1$. Let $B = \max(R^-) \cdot m_L(i)$.

If $\mathcal{D}$ is I-B-bounded, then by Lemma 16, $\mathcal{V}$ has an initialised self-covering derivation $\mathcal{D}'$ such that $\mathcal{D}'(I)$ is admissible in $\mathcal{V}(I)$ and its length is at most $((\max(R^-)+1)\cdot\max(R^-)\cdot m_L(i))^{2^{\mathsf{C}_6 k^2}} \leq (L^2\cdot m_L(i))^{2^{\mathsf{C}_6 k^2}}$.

Otherwise, $\mathcal{D}$ is not I-B-bounded, so we can write it as $\mathcal{D}_1;\mathcal{D}_2$, where $\mathcal{D}_1$ is I-B-bounded except for its last derived vector $\widehat{\mathcal{D}_1}(\varepsilon)$, which is also the first vector in $\mathcal{D}_2$. Let $j\in I$ be such that $\widehat{\mathcal{D}_1}(\varepsilon)(j)\geq B$. Without any loss of generality, we can assume that $\mathcal{D}_2$ is self-covering, and that the length of $\mathcal{D}_1$ is at most $B^{i+1}+1\leq (L\cdot m_L(i))^k+1$. Now, letting $I'=I\setminus\{j\}$, the VAS $\mathcal{V}'=\langle k,\widehat{\mathcal{D}_1}(\varepsilon),R\rangle$ must have an initialised self-covering derivation $\mathcal{D}'_2$ such that $\mathcal{D}'_2(I')$ is admissible in $\mathcal{V}'(I')$ and its length is at most $m_L(i)$. Since $\widehat{\mathcal{D}_1}(\varepsilon)(j)\geq B=\max(R^-)\cdot m_L(i)$, we have that $(\mathcal{D}_1;\mathcal{D}'_2)(I)$ is admissible in $\mathcal{V}(I)$. Moreover, the length of $\mathcal{D}_1;\mathcal{D}'_2$ is at most $(L\cdot m_L(i))^k+m_L(i)\leq (L^2\cdot m_L(i))^{2^{\mathsf{C}_6 k^2}}$.

We conclude that $m_L(0)\leq (2L)^{2^{\mathsf{C}_6 k^2}}$ and $m_L(i+1)\leq (L^2\cdot m_L(i))^{2^{\mathsf{C}_6 k^2}}$. It follows that, for all $i\in[0,k]$,

$$m_L(i)\leq \left(L^{2^{1+i\cdot \mathsf{C}_6 k^2}}\right)^i\cdot (2L)^{2^{(i+1)\cdot \mathsf{C}_6 k^2}}.$$

Thus, $m_L(k)\leq L^{2^{\log k + k\cdot \mathsf{C}_6 k^2}}\cdot (2L)^{2^{(k+1)\cdot \mathsf{C}_6 k^2}}\leq (2L)^{2^{\mathsf{C}_7 k^3}}$ for some constant C_7. □

Theorem 18. *Boundedness for BVAS is in* 2ExpTime.

Proof. We fix the proof scheme in (I)–(IV) by using Lemma 17 instead of Lemma 10 in (II), and thus deduce that every unbounded BVAS $\langle k,A_0,R_1,R_2\rangle$ has an initialised admissible self-covering derivation whose height is at most

$$H+\big(H\cdot\max((R_1\cup R_2)^-)+3\big)^{(3k)!}\leq \big(2(\max((R_1\cup R_2)^-)+1)\big)^{2^{\mathsf{C}_8 k^3}}$$

for a constant C_8, since $H\leq (2(\max(R^-)+1))^{2^{\mathsf{C}_7 k^3}}$ and $\max(R^-)\leq\max((R_1\cup R_2)^-)$. Moreover, the argument in (IV) shows that, to establish existence of such an initialised admissible self-covering derivation, it suffices to guess and verify an admissible self-covering derivation which is a path with single edges attached to it, all of whose derived vectors are doubly-exponentially bounded, and such that the vectors that label the nodes off the path are coverable. By Lemma 6 and Proposition 7, as in the proof of Theorem 8, each of the instances of covering is decidable in alternating exponential space. We conclude that the boundedness problem is in co-AExpSpace, which equals AExpSpace, which equals 2ExpTime [19]. □

As with Theorem 8, for BVAS whose dimension is fixed, the bounds in the proof of Theorem 18 are only singly exponential in its size, so we have membership of APSpace, which is included in ExpTime [19].

5. Lower bounds

We shall obtain lower bounds for covering and boundedness for BVAS by reducing from the following problem, which is for a simple class of programs with natural-valued variables (called counters) and with non-deterministic and universal branchings. We first introduce the programs, and then state the problem, whose

AExpSpace-hardness (and thus 2ExpTime-hardness [19]) is a straightforward consequence of standard translations from Turing machines to counter machines (e.g., by simulating the tape by two stacks and encoding the latter by counters).

An *alternating counter program* is a finite sequence of lines, which are numbered by 1, 2, ... Each line contains a command, which is one of: an increment of a counter ($x := x + 1$), a decrement of a counter ($x := x - 1$), a zero test (`if` $x = 0$ `then` L `else` L'), a non-deterministic jump (`goto` L `or` L'), a universal jump (`goto` L `and` L'), or termination (`halt`). In every program, `halt` occurs only as the last command.

A computation of such a program is a tree labelled by configurations, each of which is a line number together with a counter valuation. The root is labelled by the initial configuration: line number 1 with all counters having value 0. Decrements of counters with value 0 cannot be performed. Nodes with more than one child are labelled by configurations from which there is a universal jump: they have two children, to whom they pass their counter valuation unchanged. We say that a configuration is terminated iff its line number is the last in the program.

Doubly-exponential halting Given an alternating counter program with N lines, does it have a finite computation in which every counter value is at most 2^{2^N} and every leaf configuration is terminated?[5]

Our main technical goal in the rest of this section is to show how to compute, in polynomial time, BVAS which simulate alternating counter programs as long as their counters are doubly-exponentially bounded. Instead of programming the BVAS directly, we use a slightly higher-level formalism. We now define the latter, and establish a lemma which relates it with BVAS, where the emphasis is on the covering and boundedness properties.

Branching net programs are defined like alternating counter programs, except that they have no zero tests and no universal jumps, but they can contain calls of subroutines (`gosub` L) and returns from subroutines (`return`), as well as joinings of computations (`gojoin` L). The call-return stack involved is bounded, since we require that each subroutine can be assigned a level so that subroutines of level i can only call subroutines of level $i + 1$. That includes the main program, whose level is 0. Moreover, only jumps to commands in the same subroutine are permitted, and in every subroutine which is not the main program, `return` occurs only as the last command. The `gojoin` L and `halt` commands may occur only in the main program (i.e., at level 0), where such lines L must contain `halt`. The effect of a `gojoin` L command is to launch a new copy of the program and, provided it terminates at line L, add its final counter valuation pointwise to the current counter valuation.

A computation of a branching net program is therefore also a tree, but of opposite orientation compared with alternating counter programs. Each leaf is labelled by the initial configuration: empty call-return stack,

[5]The bound of 2^{2^N} suffices for AExpSpace-hardness since the program can be padded to N^k lines.

line number 1, and all counters having value 0. Nodes with more than one child are results of the launches and joins: the command at the left-hand child's line number L is `gojoin` L' where L' is the right-hand child's line number, the line number at the parent node is $L+1$, and the value of each counter at the parent node is the sum of its values at the child nodes (and the three call-return stacks are empty). We say that a configuration is reachable iff it is at the root of some computation.

Lemma 19. *Given a branching net program $\mathcal{M}$ with N lines, a BVAS $\mathcal{B}(\mathcal{M})$ of size $O(N^2)$ is computable in space logarithmic in N such that:*

(a) *given a line number L, a vector $\mathbf{t}_L$ is computable in space logarithmic in N such that $\mathcal{M}$ can reach a configuration whose line number is L iff $\mathcal{B}(\mathcal{M})$ can produce some $\mathbf{v} \geq \mathbf{t}_L$;*

(b) *$\mathcal{M}$ can reach only finitely many configurations iff $\mathcal{B}(\mathcal{M})$ can produce only finitely many vectors.*

Proof. A simple translation from net programs with N lines to VAS of size $O(N^2)$ was described by Esparza [21, §7], where the former are the subclass of branching net programs obtained by disallowing the `gojoin` L commands. It operates in space logarithmic in N, and outputs VAS which have a separate place (i.e. vector component) for each counter in the program and for each line in the program. The translation is straightforward to extend to branching net programs and BVAS: each `gojoin` L' command at a line L results in a binary rule whose -1 entries ensure that the two derived vectors being summed correspond to lines L and L'. For (a), it suffices to let $\mathbf{t}_L$ have entry 1 in the place for line L and all other entries 0. For (b), we recall that the call-return stack of $\mathcal{M}$ is bounded and note that vectors produced by $\mathcal{B}(\mathcal{M})$ have only entries 0 or 1 in the places for the lines of $\mathcal{M}$, so $\mathcal{M}$ can reach only finitely many configurations iff all its counters are bounded, which is the case iff $\mathcal{B}(\mathcal{M})$ can produce only finitely many vectors. □

We now present the main technical result in this section. Although it is phrased in terms of the properties of branching net programs that correspond to covering and boundedness for BVAS (cf. Lemma 19), the bulk of the proof shows how to construct, in polynomial time, branching net programs that simulate alternating counter programs as long as their counters are doubly-exponentially bounded.

Lemma 20. *Given an alternating counter program $\mathcal{C}$ with N lines, we have that a branching net program $\mathcal{M}(\mathcal{C})$ with $O(N)$ lines and a line number L_{fin} are computable in time polynomial in N such that the following are equivalent:*

- *$\mathcal{C}$ has a finite computation in which every counter value is at most 2^{2^N} and every leaf configuration is terminated;*
- *$\mathcal{M}(\mathcal{C})$ can reach a configuration whose line number is L_{fin};*
- *$\mathcal{M}(\mathcal{C})$ can reach infinitely many configurations.*

PROOF. We show how to construct $\mathcal{M}(\mathcal{C})$ based on the following plan:

- since computations of $\mathcal{C}$ are trees that start from the root, whereas those of $\mathcal{M}(\mathcal{C})$ are trees that start from the leaves, $\mathcal{M}(\mathcal{C})$ will simulate $\mathcal{C}$ in reverse;
- it is trivial for $\mathcal{M}(\mathcal{C})$ to simulate increments and decrements of counters in $\mathcal{C}$, but to simulate zero tests (which are not available in branching net programs), we can use Lipton's polynomial-time construction [5] (cf. the presentation by Esparza [21, Section 7]), which employs $O(N)$ counters and works as long as values of counters in $\mathcal{C}$ are at most 2^{2^N};
- since universal jumps in $\mathcal{C}$ copy counter valuations, whereas joinings of computations in $\mathcal{M}(\mathcal{C})$ sum them, $\mathcal{M}(\mathcal{C})$ will use auxiliary counters to store one of the counter valuations before each joining and to verify its equality with the other counter valuation afterwards;
- before each reverse step of $\mathcal{C}$, $\mathcal{M}(\mathcal{C})$ can attempt to verify that the current configuration of $\mathcal{C}$ is initial, in which case it will pass through line number L_{fin} and enter a loop that makes a counter unbounded.

Let $x_1, \ldots, x_K$ be the counters of $\mathcal{C}$. Then $\mathcal{M}(\mathcal{C})$ has counters x_j and $\bar{x}_j$ for each $1 \leq j \leq K$, x'_j and $\bar{x}'_j$ for each $1 \leq j \leq K+1$, s_i and $\bar{s}_i$ for each $0 \leq i \leq N$, and y_i, $\bar{y}_i$, z_i and $\bar{z}_i$ for each $0 \leq i < N$ (N is the number of lines in $\mathcal{C}$). At the beginning, $\mathcal{M}(\mathcal{C})$ performs a subroutine $Init_N(x_1, \ldots, x_K)$ which for each x_j non-deterministically chooses a value from $[0, 2^{2^N}]$, and ensures that:

(1) for all $1 \leq j \leq K$, $x_j + \bar{x}_j = 2^{2^N}$;
(2) for all $1 \leq j \leq K+1$, $x'_j = 0$ and $\bar{x}'_j = 0$;
(3) for all $0 \leq i \leq N$, $s_i = 0$ and $\bar{s}_i = 2^{2^i}$;
(4) for all $0 \leq i < N$, $y_i = 2^{2^i} = z_i$ and $\bar{y}_i = 0 = \bar{z}_i$.

The rest of the main program in $\mathcal{M}(\mathcal{C})$ consists of a segment that begins at a line $Step_L$, for each line L in $\mathcal{C}$, starting with the last (i.e. N, whose command is `halt`). For each L, the segment attempts to simulate in reverse a step of $\mathcal{C}$ that leads to line L and to the current values of $x_1, \ldots, x_K$, which may be an increment, a decrement, a successful zero test, an unsuccessful zero test, a non-deterministic jump, or a universal jump. In addition, if L is the first line in $\mathcal{C}$ (i.e. 1), the segment may verify that $x_1, \ldots, x_K$ are all zero, and then pass through L_{fin} and make a counter unbounded; and if L occurs as the second destination of a universal jump in $\mathcal{C}$, the segment may terminate $\mathcal{M}(\mathcal{C})$, making it ready for joining in a reverse simulation of such a jump. The segments are programmed so that, at the beginning and at the end (if any) of each, properties (1)–(4) hold.

More specifically, from each line $Step_L$, $\mathcal{M}(\mathcal{C})$ non-deterministically chooses to perform one of the following, where line numbers are inserted as appropriate, and `goto` L' abbreviates `goto` L' `or` L'. Recalling that subroutines in branching net programs do not have parameters, we have that $\mathcal{M}(\mathcal{C})$ contains, for example, a separate copy of the subroutine $Test_i(c, \bar{c})$ for each i, c and $\bar{c}$ that occur in an actual call.

- If $L > 1$ and the command at line $L-1$ in $\mathcal{C}$ is $x_j := x_j + 1$, then do:

$$x_j := x_j - 1;\ \bar{x}_j := \bar{x}_j + 1;$$
$$\texttt{goto}\ \mathit{Step}_{L-1}.$$

- If $L > 1$ and the command at line $L-1$ in $\mathcal{C}$ is $x_j := x_j - 1$, then do:

$$x_j := x_j + 1;\ \bar{x}_j := \bar{x}_j - 1;$$
$$\texttt{goto}\ \mathit{Step}_{L-1}.$$

- For any line $L^\dagger$ of $\mathcal{C}$ whose command is of the form $\texttt{if}\ x_j = 0\ \texttt{then}\ L\ \texttt{else}\ L'$, do:

$$\texttt{gosub}\ \mathit{Test}_N(x_j, \bar{x}_j);\ \texttt{gosub}\ \mathit{Test}_N(\bar{x}_j, x_j);$$
$$\texttt{goto}\ \mathit{Step}_{L^\dagger},$$

  where $\mathit{Test}_i(c, \bar{c})$ is a subroutine that transfers a non-deterministic amount from $\bar{c}$ to s_i and then attempts to decrement s_i exactly 2^{2^i} times by a subroutine Dec_i, while keeping constant $c + \bar{c}$ and $s_i + \bar{s}_i$:

$$\begin{aligned}
\mathit{Test}_i(c, \bar{c}):\quad & c := c + 1;\ \bar{c} := \bar{c} - 1;\\
& s_i := s_i + 1;\ \bar{s}_i := \bar{s}_i - 1;\\
& \texttt{goto}\ \mathit{Test}_i(c, \bar{c})\ \texttt{or}\ \mathit{exit};\\
\mathit{exit}:\quad & \texttt{gosub}\ \mathit{Dec}_i;\ \texttt{return}.
\end{aligned}$$

  Thus, assuming $s_N = 0$, we have that the calls to $\mathit{Test}_N(x_j, \bar{x}_j)$ and $\mathit{Test}_N(\bar{x}_j, x_j)$ can succeed iff $x_j = 0$, in which case again $x_j = 0$ and $s_N = 0$.

- For any line $L^\dagger$ of $\mathcal{C}$ whose command is of the form $\texttt{if}\ x_j = 0\ \texttt{then}\ L'\ \texttt{else}\ L$, do:

$$x_j := x_j - 1;\ x_j := x_j + 1;$$
$$\texttt{goto}\ \mathit{Step}_{L^\dagger}.$$

- For any line $L^\dagger$ of $\mathcal{C}$ whose command is of the form $\texttt{goto}\ L\ \texttt{or}\ L'$, or of the form $\texttt{goto}\ L'\ \texttt{or}\ L$, do $\texttt{goto}\ \mathit{Step}_{L^\dagger}$.

- For any line $L^\dagger$ of $\mathcal{C}$ whose command is of the form $\texttt{goto}\ L\ \texttt{and}\ L'$, do:

$$\texttt{gosub}\ \mathit{Move}_N(x_1, \ldots, x_K);\ \texttt{gosub}\ \mathit{Fin}_N;$$
$$\texttt{gojoin}\ \mathit{Step}_{L'};\ \texttt{gosub}\ \mathit{Ver}_N(x_1, \ldots, x_K);$$
$$\texttt{goto}\ \mathit{Step}_{L^\dagger},$$

  where $\mathit{Move}_N(x_1, \ldots, x_K)$ transfers x_j and $\bar{x}_j$ to x'_j and $\bar{x}'_j$ for each $1 \le j \le K$:

$$\begin{aligned}
Move_N(x_1,\dots,x_K):\quad & \texttt{gosub } Trans_N(x_1,x'_1);\ \texttt{gosub } Trans_N(\bar{x}_1,\bar{x}'_1);\ \texttt{gosub } Dec_N;\\
& \dots\\
& \texttt{gosub } Trans_N(x_K,x'_K);\ \texttt{gosub } Trans_N(\bar{x}_K,\bar{x}'_K);\ \texttt{gosub } Dec_N;\\
& \texttt{return};
\end{aligned}$$

$$\begin{aligned}
Trans_N(c,c'):\quad & \texttt{goto } loop \texttt{ or } exit;\\
loop:\quad & c := c-1;\ c' := c'+1;\\
& s_N := s_N+1;\ \bar{s}_N := \bar{s}_N-1;\\
& \texttt{goto } Trans_N(c,c');\\
exit:\quad & \texttt{return},
\end{aligned}$$

and where Fin_N empties $\bar{s}_i$ for all $0 \le i \le N$, and y_i and z_i for all $0 \le i < N$ (their duals s_i, $\bar{y}_i$ and $\bar{z}_i$ are already zero, and the emptying is undone by `gojoin` $Step_{L'}$), and $Ver_N(x_1,\dots,x_K)$ uses the auxiliary counters x'_{K+1} and $\bar{x}'_{K+1}$ to check that x_j (i.e. its value after the universal jump to L') equals x'_j (i.e. x_j's value after the universal jump to L) and then to empty x'_j and $\bar{x}'_j$, for all $1 \le j \le K$:

$$\begin{aligned}
Ver_N(x_1,\dots,x_K):\quad & \texttt{gosub } Trans^2_N(x_1,x'_1,x'_{K+1});\ \texttt{gosub } Trans^2_N(\bar{x}_1,\bar{x}'_1,\bar{x}'_{K+1});\ \texttt{gosub } Dec_N;\\
& \texttt{gosub } Trans_N(x'_{K+1},x_1);\ \texttt{gosub } Trans_N(\bar{x}'_{K+1},\bar{x}_1);\ \texttt{gosub } Dec_N;\\
& \dots\\
& \texttt{gosub } Trans^2_N(x_K,x'_K,x'_{K+1});\ \texttt{gosub } Trans^2_N(\bar{x}_K,\bar{x}'_K,\bar{x}'_{K+1});\ \texttt{gosub } Dec_N;\\
& \texttt{gosub } Trans_N(x'_{K+1},x_K);\ \texttt{gosub } Trans_N(\bar{x}'_{K+1},\bar{x}_K);\ \texttt{gosub } Dec_N;\\
& \texttt{return};
\end{aligned}$$

$$\begin{aligned}
Trans^2_N(c,c',c''):\quad & \texttt{goto } loop \texttt{ or } exit;\\
loop:\quad & c := c-1;\ c' := c'-1;\ c'' := c''+1;\\
& s_N := s_N+1;\ \bar{s}_N := \bar{s}_N-1;\\
& \texttt{goto } Trans^2_N(c,c',c'');\\
exit:\quad & \texttt{return}.
\end{aligned}$$

- For any line $L^\dagger$ of $\mathcal{C}$ whose command is of the form `goto` L' `and` L, do `halt`.
- If $L = 1$, then do:

$$\begin{aligned}
& \texttt{gosub } Test_N(x_1,\bar{x}_1);\ \texttt{gosub } Test_N(\bar{x}_1,x_1);\\
& \dots\\
& \texttt{gosub } Test_N(x_K,\bar{x}_K);\ \texttt{gosub } Test_N(\bar{x}_K,x_K);\\
L_{fin}:\quad & x_{K+1} := x_{K+1}+1;\ \texttt{goto } L_{fin}.
\end{aligned}$$

It remains to define the subroutines $Init_N(x_1, \ldots, x_K)$, Dec_i for each $0 \leq i \leq N$, which attempts to perform $s_i := s_i - 1$; $\bar{s}_i := \bar{s}_i + 1$ exactly 2^{2^i} times, and Fin_N. The code for Dec_0 is trivial. For Dec_{i+1}, the auxiliary counters y_i, $\bar{y}_i$, z_i and $\bar{z}_i$ are used to provide two nested loops that count from 2^{2^i} to 0 each, and so iterate $2^{2^i} \cdot 2^{2^i} = 2^{2^{i+1}}$ times together. Whether a loop counter has reached zero is checked by the already defined subroutine $Test_i(c, \bar{c})$, which may call Dec_i. Its side effect, that c is complemented if it is zero, ensures that Dec_{i+1} finishes with $y_i = 2^{2^i} = z_i$ and $\bar{y}_i = 0 = \bar{z}_i$, which are also assumed at the start (cf. property (4)).

$$
\begin{array}{ll}
Dec_{i+1}: & y_i := y_i - 1;\ \bar{y}_i := \bar{y}_i + 1; \\
Dec'_{i+1}: & z_i := z_i - 1;\ \bar{z}_i := \bar{z}_i + 1; \\
 & s_{i+1} := s_{i+1} - 1;\ \bar{s}_{i+1} := \bar{s}_{i+1} + 1; \\
 & \texttt{goto } Dec'_{i+1} \texttt{ or } exit'_{i+1}; \\
exit'_{i+1}: & \texttt{gosub } Test_i(z_i, \bar{z}_i);\ \texttt{goto } Dec_{i+1} \texttt{ or } exit_{i+1}; \\
exit_{i+1}: & \texttt{gosub } Test_i(y_i, \bar{y}_i);\ \texttt{return}
\end{array}
$$

Definitions of $Init_N(x_1, \ldots, x_K)$ and Fin_N are similar, where the former ensures $\bar{s}_i = 2^{2^i}$ and $y_i = 2^{2^i} = z_i$ in the increasing order of i, the latter empties those counters in the decreasing order of i, and some further details can found in Esparza's presentation of Lipton's construction [21, Section 7]).

To conclude that $\mathcal{M}(\mathcal{C})$ has $O(N)$ lines and is computable in time polynomial in N, observe that:

- the number of lines of each $Step_L$ segment is $O(1)$, except that there are $O(N)$ lines from $Step_1$;
- the number of subroutines $Test_i(c, \bar{c})$ required is $O(N)$ and each has $O(1)$ lines;
- each subroutine Dec_i has $O(1)$ lines;
- the number of subroutines $Trans_N(c, c')$ and $Trans^2_N(c, c', c'')$ required is $O(N)$ and each has $O(1)$ lines. □

From the 2ExpTime-hardness of the doubly-exponential halting problem for alternating counter programs, and Lemmas 20 and 19, we infer the same for the covering and boundedness problems for BVAS. We remark that, since Esparza's translation from net programs outputs VAS whose rules contain only entries -1, 0 or 1 [21, §7], the BVAS in Lemma 19 have the same property, and consequently covering and boundedness are 2ExpTime-hard already for that subclass.

Theorem 21. *Covering and boundedness for BVAS are* 2ExpTime*-hard.*

6. Concluding remarks

The extra work in this article in relation to the proofs of Lipton and Rackoff [5, 4], and the recent result that reachability for BVAS is 2ExpSpace-hard [22] (the highest known lower bound for VAS is Lipton's),

indicate that BVAS are not a trivial extension of VAS.

We would like to thank Serge Haddad (LSV, Cachan) for numerous discussions about VAS and their extensions, Sylvain Schmitz (LSV, Cachan) for pointing us to Rambow's work [12], and Alexander Schrijver (CWI, Amsterdam) for correspondence about integer linear programming.

References

[1] S. Demri, M. Jurdziński, O. Lachish, R. Lazić, The covering and boundedness problems for branching vector addition systems, in: FSTTCS, volume 4 of *LIPIcs*, Schloss Dagstuhl, 2009, pp. 181–192.

[2] W. Reisig, Petri Nets: An Introduction, volume 4 of *Monographs in Theor. Comput. Sci. An EATCS Series*, Springer, 1985.

[3] J. Esparza, M. Nielsen, Decidability issues for Petri nets — a survey, Bull. EATCS 52 (1994) 244–262.

[4] C. Rackoff, The covering and boundedness problems for vector addition systems, Theor. Comput. Sci. 6 (1978) 223–231.

[5] R. J. Lipton, The Reachability Problem Requires Exponential Space, Technical Report 62, Dep. Comput. Sci., Yale Univ., 1976.

[6] L. Rosier, H.-C. Yen, A multiparameter analysis of the boundedness problem for vector addition systems, J. Comput. Syst. Sci. 32 (1986) 105–135.

[7] E. W. Mayr, A. R. Meyer, The complexity of the word problems for commutative semigroups and polynomial ideals, Adv. Math. 46 (1982) 305–329.

[8] P. Habermehl, On the complexity of the linear-time mu-calculus for Petri nets, in: ICATPN, volume 1248 of *Lect. Notes Comput. Sci.*, Springer, 1997, pp. 102–116.

[9] M. F. Atig, P. Habermehl, On Yen's path logic for Petri nets, in: RP, volume 5797 of *Lect. Notes Comput. Sci.*, Springer, 2009, pp. 51–63.

[10] S. Demri, On selective unboundedness of VASS, in: INFINITY, volume 39 of *El. Proc. Theor. Comput. Sci.*, pp. 1–15. 2010.

[11] A. Urquhart, The complexity of decision procedures in relevance logic II, J. Symb. Log. 64 (1999) 1774–1802.

[12] O. Rambow, Multiset-valued linear index grammars: imposing dominance constraints on derivations, in: ACL, Morgan Kaufmann, 1994, pp. 263–270.

[13] S. Schmitz, On the computational complexity of dominance links in grammatical formalisms, in: ACL, The Association for Computer Linguistics, 2010, pp. 514–524.

[14] P. de Groote, B. Guillaume, S. Salvati, Vector addition tree automata, in: LICS, IEEE, 2004, pp. 64–73.

[15] R. M. Karp, R. E. Miller, Parallel program schemata, J. Comput. Syst. Sci. 3 (1969) 147–195.

[16] K. N. Verma, J. Goubault-Larrecq, Karp-Miller trees for a branching extension of VASS, Discr. Math. and Theor. Comput. Sci. 7 (2005) 217–230.

[17] M. Bojańczyk, A. Muscholl, T. Schwentick, L. Segoufin, Two-variable logic on data trees and XML reasoning, J. ACM 56 (2009).

[18] R. Valk, G. Vidal-Naquet, Petri nets and regular languages, J. Comput. Syst. Sci. 23 (1981) 299–325.

[19] A. Chandra, D. Kozen, L. Stockmeyer, Alternation, J. ACM 28 (1981) 114–133.

[20] I. Borosh, L. B. Treybig, Bounds on positive integral solutions of linear Diophantine equations, Proc. AMS 55 (1976) 299–304.

[21] J. Esparza, Decidability and complexity of Petri net problems — an introduction, in: Lectures on Petri Nets I: Basic Models, volume 1491 of *Lect. Notes Comput. Sci.*, Springer, 1998, pp. 374–428.

[22] R. Lazić, The reachability problem for branching vector addition systems requires doubly-exponential space, Inf. Process. Lett. 110 (2010) 740–745.