

A Polynomial-Time Algorithm for Reachability in Branching VASS in Dimension One

Stefan Göller^{*1}, Christoph Haase^{*1}, Ranko Lazić^{†2}, and Patrick Totzke^{†2}

- 1 LSV, CNRS & ENS Cachan
Université Paris-Saclay, France
{goeller,haase}@lsv.ens-cachan.fr
- 2 DIMAP, Department of Computer Science
University of Warwick, United Kingdom
{r.s.lazic,p.totzke}@warwick.ac.uk

Abstract

Branching VASS (BVASS) generalise vector addition systems with states by allowing for special branching transitions that can non-deterministically distribute a counter value between two control states. A run of a BVASS consequently becomes a tree, and reachability is to decide whether a given configuration is the root of a reachability tree. This paper shows P-completeness of reachability in BVASS in dimension one, the first decidability result for reachability in a subclass of BVASS known so far. Moreover, we show that coverability and boundedness in BVASS in dimension one are P-complete as well.

1998 ACM Subject Classification F.1.1 Models of Computation

Keywords and phrases branching vector addition systems, reachability, coverability, boundedness

1 Introduction

Vector addition systems with states (VASS), equivalently known as Petri nets, are a fundamental model of computation which comprise a finite-state controller with a finite number of counters ranging over the naturals. The number of counters is usually referred to as the dimension of the VASS. A configuration $q(\mathbf{n})$ of a VASS in dimension d consists of a control state q and a valuation $\mathbf{n} \in \mathbb{N}^d$ of the counters. A transition of a VASS can increment and decrement counters and is enabled in a configuration whenever the resulting counter values are all non-negative, otherwise the transition is disabled. Consequently, VASS induce an infinite transition system. Three of the most fundamental decision problems for VASS are reachability, coverability and boundedness. Given a target configuration $q(\mathbf{n})$ and some initial configuration, reachability is to decide whether starting in the initial configuration there exists a path ending in $q(\mathbf{n})$ in the induced infinite transition system. Coverability asks whether some configuration $q(\mathbf{n}')$ can be reached for some $\mathbf{n}' \geq \mathbf{n}$, where \geq is defined component-wise. Boundedness is the problem to decide whether there are infinitely many different configurations reachable from a given starting configuration. Those decision problems find a plethora of applications, for instance in the verification of concurrent programs. Coverability can, for example, be used in order to validate mutual exclusion properties of

* Supported by Labex Digicosme, Univ. Paris-Saclay, project VERICONISS.

† Supported by the EPSRC, grants EP/M011801/1 and EP/M027651/1.



shared-memory concurrent programs [6]; reachability is a key underlying decision problem in the verification of liveness properties of finite-data asynchronous programs [5]. Even though the complexity of coverability and boundedness are well-understood and known to be EXPSPACE-complete [12, 14], the precise complexity of reachability remains a major unsolved problem; a non-primitive recursive upper bound (\mathbf{F}_{ω^3}) has only recently been established [11] and the best known lower bound is EXPSPACE [12].

The situation is even more dissatisfying when considering *branching extensions* of VASS. Such *branching VASS (BVASS)* are additionally equipped with special branching transitions of the form (q, p, p') . When in a configuration $q(\mathbf{n})$, a BVASS can simultaneously non-deterministically branch into configurations $p(\mathbf{m})$ and $p'(\mathbf{m}')$ such that $\mathbf{n} = \mathbf{m} + \mathbf{m}'$. Reachability of a configuration $q(\mathbf{n})$ then is to decide whether there exists a proof tree whose root is labelled with $q(\mathbf{n})$ and whose leaves are all labelled with designated target control states in which all counters have value zero; coverability and boundedness are defined analogously as above. While coverability and boundedness are known to be 2-EXPTIME-complete [3], reachability in BVASS is not known to be decidable, not even in any fixed dimension. Recently, non-elementary lower bounds for reachability in BVASS have been obtained [10]. Reachability in BVASS is closely related and in fact equivalent to decidability of the multiplicative-exponential fragment of linear logic [2], and also an underlying decision problem in various other applications for instance in computational linguistics, cryptographic protocol verification, data logics and concurrent program verification; see [10] for more details.

The primary contribution of this paper is to provide a polynomial-time algorithm for reachability in BVASS in dimension one (BVASS₁) and to show that reachability is in fact P-complete. To the best of our knowledge, we give the first decidability result for reachability in a fragment of BVASS. Let us remark that a decidability result, in particular with such low complexity is actually quite surprising. On the one hand, due to the infinite state space of BVASS₁ it is not immediate that reachability is decidable. In particular, the emptiness problem for conjunctive grammars over a unary alphabet, which can be seen as a slight generalisation of BVASS₁ with special alternating transitions that can simultaneously branch into two control states while retaining the same counter value (known as ABVASS₁), is undecidable [9]. On the other hand, if we disallow branching rules in ABVASS₁ and thus obtain AVASS₁ then reachability is PSPACE-complete [15, 8].

Due to the presence of only one single counter, it is possible to establish a small-model property and to show that if a configuration is reachable in a BVASS₁ then there exists a so-called reachability tree of exponential size. What causes a main challenge when establishing a polynomial-time algorithm is that this bound is optimal in the sense that, as we show in Section 3, there exist families of BVASS₁ whose reachability trees are inherently of exponential size, and which also contain an exponential number of different counter values. Consequently, reachability cannot be witnessed in polynomial time by explicitly constructing a witnessing reachability tree. Instead, in Section 4 we show that polynomial-time computable certificates for the reachability of a configuration suffice. These certificates have two parts: the first is a table that, for certain $d > 0$ contains those pairs of control states q and residue classes r modulo d such that $q(n)$ is reachable for some sufficiently large n with $n \equiv r \pmod{d}$. This is called residue reachability and described in Section 4.1. The second part, described in Section 4.2, is a compressed collection of incomplete small reachability trees, so-called expandable partial reachability trees, whose leaves are either accepting configurations or have some ancestor node with the same control state and a strictly smaller counter. In the latter case, the corresponding subtree can be repeated arbitrarily often,

which leaves some configuration with an arbitrarily large counter value in a certain residue class. This eventually enables us to witness the existence of a reachability tree via residue reachability.

In Section 5, we show that coverability and boundedness are P-complete for BVASS₁. For coverability, the upper bound follows easily via a reduction to reachability. For boundedness, this is not the case and we require a specifically tailored argument.

Due to space constraints, the proofs of some statements can be found in an appendix.

2 Preliminaries

We write \mathbb{Z} and \mathbb{N} for the sets of integers and non-negative integers, respectively, and define $[i, j] \stackrel{\text{def}}{=} \{i, i+1, \dots, j-1, j\}$, for given integers $i < j$. For $d \geq 1$ we define $\mathbb{Z}_d \stackrel{\text{def}}{=} [0, d-1]$.

The set of finite words over alphabet A is denoted by A^* and the length of a word $w \in A^*$ is written as $|w|$. For two words $u, v \in A^*$, we say u is a *prefix* of v (written as $u \preceq v$) if $v = uw$ for some $w \in A^*$. It is a *strict prefix* ($u \prec v$) if $u \preceq v$ and $u \neq v$. We say u and v are *incomparable* if neither $u \preceq v$ nor $v \preceq u$. A set $U \subseteq A^*$ is *prefix-closed* if for all $u \in U$ and all $v \in A^*$ we have that $v \preceq u$ implies $v \in U$.

Let Σ be a set. A Σ -labelled (finite) tree is a mapping $T: U \rightarrow \Sigma$ where $U \subseteq A^*$ is a non-empty finite prefix-closed set of nodes for some finite set A . For $V \subseteq U$, we define $T(V) \stackrel{\text{def}}{=} \{T(v) \mid v \in V\}$. A *leaf* of T is a node $u \in U$ such that there is no $v \in U$ with $u \prec v$; every node of T that is not a leaf is called *inner node*. A node u is an *ancestor* (resp. *descendant*) of a node v if $u \preceq v$ (resp. $v \preceq u$) and a *strict ancestor* (resp. *strict descendant*) if $u \prec v$ (resp. $v \prec u$). For any node u we define the *subtree of T rooted at u* as $T^{\downarrow u}: u^{-1}U \rightarrow \Sigma$, where $u^{-1}U \stackrel{\text{def}}{=} \{x \in A^* \mid ux \in U\}$ and $T^{\downarrow u}(x) \stackrel{\text{def}}{=} T(ux)$. Note that $u^{-1}U$ is a prefix-closed subset of A . We define $h(u) \stackrel{\text{def}}{=} \max\{|x| \mid x \in u^{-1}U\}$ to be the *height* of the subtree rooted at u and define $h(T) \stackrel{\text{def}}{=} h(\varepsilon)$. Note that $h(u) = 0$ if, and only if, u is a leaf. We say T is *binary* if $U \subseteq \{0, 1\}^*$; in this case if for some node $u \in U$ we have that $u0 \in U$, then $u0$ the *left child* of u and if $u1 \in U$ we say that $u1$ is the *right child* of u .

2.1 Branching Vector Addition Systems

In the following, \mathbf{n} and \mathbf{z} will denote elements from \mathbb{N}^k and \mathbb{Z}^k , respectively; addition on \mathbb{Z}^k is defined component-wise.

► **Definition 1.** Let $k \geq 1$. A *k-dimensional branching vector addition system with states* (BVASS_k) is a tuple $\mathcal{B} = (Q, \Delta, F)$ where Q is a finite set of *control states*, $\Delta \subseteq Q^3 \cup (Q \times \{-1, 0, 1\}^k \times Q)$ is a finite set of *transitions*, and $F \subseteq Q$ is a set of *final states*. The *size* $|\mathcal{B}|$ of a BVASS is defined as $|\mathcal{B}| \stackrel{\text{def}}{=} |Q| + k \cdot |\Delta|$.

The semantics of BVASS is given in terms of reachability trees. A *partial reachability tree* of a BVASS_k \mathcal{B} is a $Q \times \mathbb{N}^k$ -labelled binary tree $T: U \rightarrow Q \times \mathbb{N}^k$, where each inner node $u \in U$ with $T(u) = (q, \mathbf{n})$ satisfies exactly one of the following conditions:

- $u0, u1 \in U$, and if $T(u0) = (p, \mathbf{n}_0)$ and $T(u1) = (p', \mathbf{n}_1)$, then $\mathbf{n} = \mathbf{n}_0 + \mathbf{n}_1$ and $(q, p, p') \in \Delta$; or
- $u0 \in U, u1 \notin U$, and if $T(u0) = (p, \mathbf{n}_0)$, then $\mathbf{n}_0 = \mathbf{n} + \mathbf{z}$ and $(q, \mathbf{z}, p) \in \Delta$.

Note that in the second condition, counter values can be seen as being propagated top down. A *reachability tree* is a partial reachability tree T where $T(u) \in F \times \{0\}^k$ for all leaves u of T . We call these nodes *accepting* nodes. For each $j \in \mathbb{N}$ we say that a partial reachability tree T is *j-bounded* if $T(u) \in Q \times [0, j]^k$ for all $u \in U$. We call $Q \times \mathbb{N}^k$ the set of *configurations*



■ **Figure 1** Illustration of the BVASS₁ \mathcal{B}_n . The reachability set of the control state q_n is the singleton set $\{2^n\}$, and a reachability tree for $q(0)$ contains all counter values between 0 and 2^n .

of \mathcal{B} and for the sake of readability often write its elements (q, \mathbf{n}) as $q(\mathbf{n})$. We say that a configuration $q(\mathbf{n})$ is *reachable* if there exists a reachability tree T with $T(\varepsilon) = q(\mathbf{n})$. Note that in particular every configuration in $F \times \{0\}^k$ is reachable. The reachability set $\text{reach}(q)$ of a control state q is defined as $\text{reach}(q) \stackrel{\text{def}}{=} \{\mathbf{n} \in \mathbb{N} \mid q(\mathbf{n}) \text{ is reachable}\}$. The decision problem that we mainly focus on in this paper is *reachability*, defined as follows:

REACHABILITY IN BVASS_k

INPUT: A BVASS_k $\mathcal{B} = (Q, \Delta, F)$, a control state q and $\mathbf{n} \in \mathbb{N}^k$ encoded in unary.

QUESTION: Is $q(\mathbf{n})$ reachable?

Our main result is that reachability is P-complete in dimension one.

► **Theorem 2.** *Reachability in BVASS₁ is P-complete.*

3 Lower Bounds

As a warm-up exercise and in order to familiarise ourselves with BVASS₁, we begin with proving a couple of lower bounds for the reachability problem. First, it is not difficult to see that the reachability problem is P-hard via a reduction from the monotone circuit value problem (MCVP) [13]. By simulating \vee -gates of a Boolean by non-deterministic branching and \wedge -gates by splitting transitions, the following statement can easily be obtained.

► **Proposition 3.** *Let \mathcal{C} be a Boolean circuit. There exists a logspace computable BVASS₁ \mathcal{B} with a control state q such that $q(0)$ is reachable if, and only if, \mathcal{C} evaluates to true.*

A challenging aspect when providing a polynomial-time upper bound for reachability in BVASS₁ is that reachability trees may be of exponential size and may contain an exponential number of nodes labelled with distinct counter values. To see this, consider the family $(\mathcal{B}_n)_{n \geq 0}$ of BVASS₁, where $\mathcal{B}_n \stackrel{\text{def}}{=} (Q_n, \Delta_n, F)$ and where $Q_n \stackrel{\text{def}}{=} \{q, q_f\} \cup \{q_0, \dots, q_n\}$, $\Delta_n \stackrel{\text{def}}{=} \{(q, +1, q), (q, 0, q_n)\} \cup \{(q_i, q_{i-1}, q_{i-1}) \mid 0 < i \leq n\} \cup \{(q_0, -1, q_f)\}$ and $F \stackrel{\text{def}}{=} \{q_f\}$. The construction is illustrated in Figure 1. It is easily seen that $q_i(N)$ is reachable if, and only if, $N = 2^i$. Observe that $\text{reach}(q) = \{0, \dots, 2^n\}$ is finite and that the reachability tree of $q(0)$ contains all counter values between 0 and 2^n . In particular, this allows us to obtain the following hardness result in which the updates of the BVASS₁ are from $\{-1, 0, +1\}$ (i.e. encoded in unary), but the initial configuration is given in binary, via a straight-forward reduction from the NP-complete SUBSET SUM problem [13].

► **Proposition 4.** *Reachability in BVASS₁ is NP-hard if the initial configuration $q(\mathbf{n})$ is given in binary.*

It is worth mentioning that the previous lemma enables us to derive as a corollary an NP-lower bound for reachability in BVASS₂. This is in contrast to VASS where there is no difference between the NL-completeness of reachability in dimensions one and two [16, 4].

► **Corollary 5.** *Reachability in BVASS₂ is NP-hard.*

4 Reachability in BVASS₁

Here, we show that reachability in BVASS₁ is decidable in polynomial time, thereby establishing the P upper bound claimed in Theorem 2. In the first part, we consider a variation of the reachability problem in which we are only interested in reaching configurations that are sufficiently large and lie in a certain residue class. Subsequently, we will apply this intermediate result for showing that reachability can be witnessed by small partial reachability trees. Finally, we put everything together in order to obtain a polynomial-time algorithm.

4.1 The Residue Reachability Problem

A cornerstone of our algorithm for reachability in BVASS₁ is the polynomial-time decidability of the following variant of the reachability problem for BVASS₁:

RESIDUE REACHABILITY FOR BVASS₁

INPUT: A BVASS₁ $\mathcal{B} = (Q, \Delta, F)$, a configuration $q_0(n_0)$ and $d \geq 1$, where n_0 and d are given in unary.

QUESTION: Does there exist some $n \geq n_0$ such that $q_0(n)$ is reachable and $n \equiv n_0 \pmod{d}$?

The main result of this section is that residue reachability for BVASS₁ is decidable in polynomial time. Notice that setting $d = 1$ allows for checking whether there exists some $n \geq n_0$ such that $q(n)$ is reachable. We first introduce some auxiliary definitions that allow us to abstract away concrete counter values of reachability trees. A *partial d -residue tree* is a binary tree $T: U \rightarrow Q \times \mathbb{Z}_d$, where each inner node $u \in U$ with $T(u) = (q, n)$ satisfies precisely one of the following conditions:

- (i) $u0, u1 \in U$, and if $T(u0) = (p, m_0)$ and $T(u1) = (p', m_1)$ then $n \equiv m_0 + m_1 \pmod{d}$ and $(q, p, p') \in \Delta$;
- (ii) $u0 \in U, u1 \notin U$, and if $T(u0) = (p, m)$ then $m = n + z \pmod{d}$ and $(q, z, p) \in \Delta$.

We call a configuration from $Q \times \mathbb{Z}_d$ a *residue configuration*. Given a set of configurations S , its *residue* is $S/\mathbb{Z}_d \stackrel{\text{def}}{=} \{(q, n \pmod{d}) \in Q \times \mathbb{Z}_d \mid q(n) \in S\}$. Likewise, given a partial reachability tree $T: U \rightarrow Q \times \mathbb{N}$, the *residue T/\mathbb{Z}_d of T* is $T/\mathbb{Z}_d: U \rightarrow Q \times \mathbb{Z}_d$, where $T/\mathbb{Z}_d(u) \stackrel{\text{def}}{=} T(u)/\mathbb{Z}_d$ for all $u \in U$. Clearly, T/\mathbb{Z}_d is a partial residue tree.

For the remainder of this section, fix some BVASS₁ $\mathcal{B} = (Q, \Delta, F)$, some configuration $q_0(n_0)$ and some $d \geq 1$, where n_0 and d are given in unary. In order to decide residue reachability, one might be tempted to start with an initial configuration and then to repeatedly apply transitions of \mathcal{B} modulo d until the desired residue configuration is discovered. Such an approach would, however, not be sound as it may lead to residue configurations that, informally speaking, can only be obtained by forcing the counter to drop below zero. Also, the simple alternative of constructing a sufficiently large reachability tree is futile as it may be of exponential size, cf. Section 3. In order to balance between those two extremes, we introduce reachability trees in which all nodes except of the root are required to be bounded by some value $j \in \mathbb{N}$: a partial reachability tree $T: U \rightarrow Q \times \mathbb{N}$ is *almost j -bounded* if $T(u) \in Q \times [0, j]$ for all $u \in U \setminus \{\varepsilon\}$. Note that every j -bounded partial reachability tree is almost j -bounded. The following constant will be particularly useful:

$$N \stackrel{\text{def}}{=} |Q| \cdot d.$$

Moreover, by S we denote the set of configurations for which there exists an $(n_0 + N)$ -bounded reachability tree and define for $i < j$:

$$S \stackrel{\text{def}}{=} \{(q, m) \in Q \times \mathbb{N} \mid q(m) \text{ has an } (n_0 + N)\text{-bounded reachability tree}\}$$

$$S[i, j] \stackrel{\text{def}}{=} S \cap Q \times [i, j].$$

► **Lemma 6.** *The set S is computable in polynomial time.*

For any set of residue configurations (modulo d) $V, W \subseteq Q \times \mathbb{Z}_d$, we define the following sets that contain the result of an application of a transition of \mathcal{B} modulo d :

$$\Delta(V) \stackrel{\text{def}}{=} \{(q, r - z \bmod d) \mid (q, z, p) \in \Delta, (p, r) \in V\}$$

$$\Delta(V, W) \stackrel{\text{def}}{=} \{(q, r_0 + r_1 \bmod d) \mid (q, p_0, p_1) \in \Delta, (p_0, r_0) \in V, (p_1, r_1) \in W\}.$$

Next, we inductively define a sequence of sets $R_i \subseteq Q \times \mathbb{Z}_d$ for $i \geq 0$ whose fixed point will allow for deciding residue reachability. The set R_0 consists of those pairs of control states and residue classes that can be witnessed by a reachability tree that is almost $(n_0 + N)$ -bounded and whose root has a counter value at least $n_0 + N$, and the R_i for $i > 0$ are obtained by application of Δ :

$$R_0 \stackrel{\text{def}}{=} \{(q, n \bmod d) \in Q \times \mathbb{Z}_d \mid$$

$$n \geq n_0 + N, q(n) \text{ has an almost } (n_0 + N)\text{-bounded reachability tree}\}$$

$$R_{i+1} \stackrel{\text{def}}{=} R_i \cup \Delta(R_i) \cup \Delta(R_i, S/\mathbb{Z}_d) \cup \Delta(S/\mathbb{Z}_d, R_i) \cup \Delta(R_i, R_i).$$

Since the cardinality of each R_i is at most N , it is easily seen that the sequence $(R_i)_{i \geq 0}$ reaches a fixed point which can be computed in polynomial time.

► **Lemma 7.** *The fixed point $R \stackrel{\text{def}}{=} \bigcup_{i \geq 0} R_i$ equals R_N and is computable in polynomial time.*

In particular, R together with S yields the whole residue reachability set.

► **Lemma 8.** *The set $X \stackrel{\text{def}}{=} R \cup S[n_0, n_0 + N]/\mathbb{Z}_d$ is computable in polynomial time. Moreover,*

$$X = \{(q, n \bmod d) \mid q \in Q, n \in \text{reach}(q), n \geq n_0\}.$$

Proof (sketch). Polynomial-time computability of X follows immediately from Lemmas 6 and 7. The proof of the stated equality is quite technical though not too difficult and deferred to the appendix. The crucial part for the inclusion “ \subseteq ” is to show that for every $i \in [0, N]$ and each $(q, r) \in R_i$ there exists some $n \in \text{reach}(q)$ with $n \geq n_0 + N - i$ and $n \equiv r \bmod d$ by induction on i . For the converse inclusion the only interesting case is when a potential reachability tree T is not $(n_0 + N)$ -bounded. One first shows that all \prec -maximal nodes u in T with $T(u) \notin S$ satisfy $T(u)/\mathbb{Z}_d \in R_0$ and uses the fact that $\Delta(R, R) \subseteq R$ and $\Delta(R) \subseteq R$ to conclude $T(\varepsilon)/\mathbb{Z}_d \in R$. ◀

The main result of this section now follows directly from Lemma 8.

► **Theorem 9.** *Residue reachability for $BVASS_1$ is decidable in polynomial time.*

4.2 Expandable Partial Reachability Trees

We now employ our result on residue reachability to show that small partial reachability trees suffice in order to witness reachability. The key idea is to identify branches of partial reachability trees that end in a leaf and which could, informally speaking, be copied or pumped an arbitrary number of times, thus achieving a counter value in the leaf that is large enough and lies in a certain residue class of some modulus. Residue reachability then witnesses that such a leaf could be completed in order to yield a reachability tree. For the remainder of this section, fix some BVASS₁ $\mathcal{B} = (Q, \Delta, F)$.

Let us first introduce a couple of auxiliary definitions. Given a partial reachability tree $T: U \rightarrow Q \times \mathbb{N}$ and $v, w \in U$, the *lowest common ancestor* of $v, w \in U$ is defined as

$$lca(v, w) \stackrel{\text{def}}{=} \max\{u \in U \mid u \preceq v \text{ and } u \preceq w\},$$

where the maximum is taken with respect to \preceq . Let $T(u) = q(n)$, we define functions $state(u) \stackrel{\text{def}}{=} q$ and $counter(u) \stackrel{\text{def}}{=} n$ that allow us to access the control state and the counter value at u , respectively.

► **Definition 10.** A node $v \in U$ is *increasing* if there is a proper ancestor $u \prec v$ such that $state(u) = state(v)$ and $counter(u) < counter(v)$; the maximal such u is called the *anchor* of v . We say that T is *exclusive* if the least common ancestor of any two distinct increasing leaves is a proper ancestor of at least one of their anchors. Finally, we call T *expandable* if

- T is exclusive,
- every leaf v of T is either accepting or an increasing leaf,
- every increasing leaf v with anchor u such that $T(v) = q(n)$ and $T(u) = q(m)$ induces a valid instance of the residue reachability problem, i.e., $q(l)$ is reachable for some $l \geq n$ and $l \equiv n \pmod{n - m}$.

A node u is said to be *exclusive* resp. *expandable* if $T^{\downarrow u}$ is.

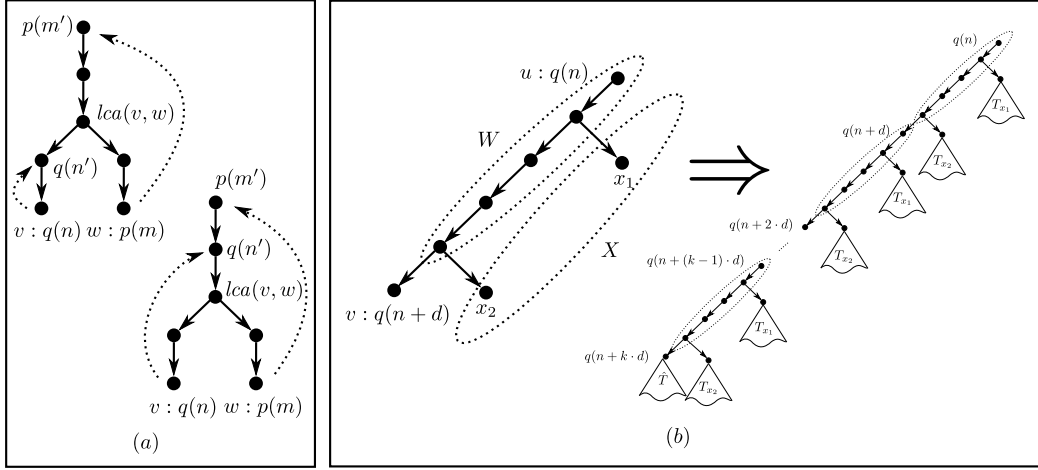
Observe that nodes cannot be both accepting and increasing because increasing nodes have strictly positive counter values and accepting nodes must have counter value zero. Exclusive and non-exclusive partial reachability trees are illustrated in Figure 2(a). The next lemma states a useful fact that directly follows from the pigeon-hole principle: whenever the counter increases on a branch by a certain amount then the branch contains an increasing node and its anchor.

► **Lemma 11.** *Let u and v be nodes of a partial reachability tree such that $u \prec v$ and $counter(u) + |Q| \leq counter(v)$. Then there exists an increasing node v' with anchor u' such that $u \preceq u' \prec v' \preceq v$.*

The following lemma shows that every reachability tree gives rise to an expandable reachability tree whose nodes have counter values bounded polynomially in $|\mathcal{B}|$.

► **Lemma 12.** *Suppose $q(n)$ is reachable and let $B \stackrel{\text{def}}{=} 2 \cdot |Q| + n$. Then there exists an expandable B -bounded partial reachability tree with root $q(n)$.*

Proof. Let T be a reachability tree with $T(\varepsilon) = q(n)$. We call a node w of T *large* if $counter(w) = B$. We obtain a partial reachability tree T' from T as follows. By Lemma 11, every large node w gives rise to at least one pair of nodes (u, v) such that $u \prec v \preceq w$ and v is an increasing node with anchor u . For every large node w that is minimal with respect to \preceq , we assign the maximal such pair $pair(w) \stackrel{\text{def}}{=} (u, v)$ with respect to the lexicographical ordering on nodes (more precisely, $(u, v) \preceq (u', v')$ if either, $u \prec u'$, or $u = u'$ and $v \preceq v'$).



■ **Figure 2** (a) Illustration an exclusive (top) and a non-exclusive (bottom) partial reachability tree. Here, v and w are pumping nodes and anchor relationships are depicted as dashed arrows. (b) Illustration of the pumping argument in Lemma 14.

Let $T' : U' \rightarrow Q \times \mathbb{N}$ denote the tree that one obtains from T by replacing all subtrees of T that are rooted at some node v such that $\text{pair}(w) = (u, v)$ for some minimal (with respect to \preceq) large node w in T by $\{v\}$ itself, i.e. such nodes v become leaves. We now prove that T' is B -bounded and exclusive:

- T' is B -bounded since the w above are chosen minimal with respect to \preceq and hence $\text{counter}(u) \leq B$ for all nodes $u \in U'$.
- T' is exclusive, which can be seen as follows. Striving for a contradiction, suppose that T' is not exclusive. Then there are distinct increasing nodes v, v' with anchors u, u' such that $u, u' \preceq w \stackrel{\text{def}}{=} \text{lca}(v, v')$. Since $\text{counter}(w) = \text{counter}(w_0) + \text{counter}(w_1) \leq B$, we have $\text{counter}(w_0) \leq B/2$ or $\text{counter}(w_1) \leq B/2$, and assume without loss of generality that $\text{counter}(w_0) \leq B/2$. Since $B - B/2 \geq |Q|$, by Lemma 11 there is another increasing node v'' with anchor u'' such that $w_0 \preceq u'' \prec v''$, contradicting the assumed maximality of (u, v) .
- Every leaf is accepting or increasing, by definition of T' .
- Finally, every increasing leaf u in T' induces a positive residue-reachability instance. Since T is a reachability tree, we have that $T(u)$ is reachable and thus $T'(u)$ is reachable. So in particular, it is reachable modulo $d = 1$, i.e. if $T'(u) = q(n)$, then we can choose $(q(n), 1)$ as the required valid instance of residue reachability. ◀

We now turn towards the converse direction and show that every expandable tree witnesses reachability. We first state an auxiliary lemma about structural properties of nodes in exclusive trees whose proof can be found in the appendix.

► **Lemma 13.** *For every node u of an expandable partial reachability tree the following hold:*

- (i) *If u is the anchor of an increasing leaf v then u is expandable and all nodes w such that $u \prec w \preceq v$ are not expandable.*
- (ii) *u has at most one child that is not expandable.*

The previous lemma enables us to show that an expandable partial reachability tree implies the existence of a reachability tree.

► **Lemma 14.** *Let $T: U \rightarrow Q \times \mathbb{N}$ be an expandable partial reachability tree. Then for all $u \in U$, $T(u)$ is reachable or u is not expandable.*

Proof. We prove the lemma by induction on $h(u)$. For the induction base, assume $h(u) = 0$, hence u is a leaf. Then u is either accepting and thus $T(u)$ is reachable, or u is not accepting and therefore an increasing leaf and so $T^{\downarrow u}$ is not expandable by Lemma 13(i).

For the induction step, suppose u is expandable. We distinguish two cases:

- *All children of u are expandable.* We only treat the case when u has two children, the case when u has one child follows as a special case. Since the children $u0$ and $u1$ of u are expandable, by the induction hypothesis there are reachability trees $T_0: U_0 \rightarrow Q \times \mathbb{N}$ and $T_1: U_1 \rightarrow Q \times \mathbb{N}$ with $T_0(\varepsilon) = T(u0)$ and $T_1(\varepsilon) = T(u1)$. We define the following tree $T_u: V \rightarrow Q \times \mathbb{N}$, where $V \stackrel{\text{def}}{=} \{0\}U_0 \cup \{1\}U_1 \cup \{\varepsilon\}$, $T_u(\varepsilon) \stackrel{\text{def}}{=} T(u)$ and $T_u(i\varepsilon) \stackrel{\text{def}}{=} T_i(\varepsilon)$ for all $i \in \{0, 1\}$. Now T_u is a reachability tree, hence $T_u(\varepsilon) = T(u)$ is reachable.
- *Some child of u is not expandable.* For simplicity of presentation, let $u = \varepsilon$, the cases when $u \neq \varepsilon$ can be proven analogously. Moreover, let us assume that $T(u) = q(n)$. By Lemma 13(ii) there is at most one such child, without loss of generality let $u0 = 0$ be this child. Moreover, since u is expandable and $u0$ is not expandable it must hold that u is the anchor of some unique increasing leaf v , we may assume without loss of generality $v = u0^\ell$ for some $\ell \geq 1$. We must have $T(v) = q(n + d)$ for some $d \geq 1$. Let $W = \{0^i \mid i \in [0, \ell - 1]\}$ be the set all nodes in T “on the path from u to v ” without v . Let $X \stackrel{\text{def}}{=} \{0^i 1 \in U \mid i \in [0, \ell - 1]\}$ be the set of all right children of nodes in W .

By Lemma 13(i), all nodes in $\{0^i \mid i \in [1, \ell]\}$ are not expandable and consequently, Lemma 13(ii) implies that all nodes in X are expandable. Hence by induction hypothesis, for every $x \in X$ there is a reachability tree $T_x: U_x \rightarrow Q \times \mathbb{N}$ such that $T_x(\varepsilon) = T(x)$.

It remains to show that $T(u) = q(n)$ is reachable. Since T is expandable there exists some $m \geq n + d$ such that $q(m)$ is reachable and $m \equiv n \pmod{d}$. Let us assume $m = n + d + k \cdot d$ for some $k \geq 0$ and let $\widehat{T}: Z \rightarrow Q \times \mathbb{N}$ be a reachability tree for $q(m)$.

We construct the following reachability tree T' (formal definition below) for $q(n)$ as the tree one obtains from T by replacing the leaf v by the tree T repeatedly exactly k times and by adding to the counter values of the resulting nodes from 0^* in the i -th copy the counter value $i \cdot d$. This procedure is illustrated in Figure 2(b). Note that this process yields a partial reachability tree in which every leaf is accepting except for the leaf $0^{(k+1) \cdot \ell}$; therefore we replace this leaf by the tree $\widehat{T}: Z \rightarrow Q \times \mathbb{N}$. Recall that $T_x: U_x \rightarrow Q \times \mathbb{N}$ is a reachability tree for $T(x) = T_x(\varepsilon)$. Formally, we define $T': \left(0^{(k+1) \cdot \ell} Z \cup \bigcup_{i=0}^k 0^{i \cdot \ell} (W \cup \bigcup \{xU_x \mid x \in X\})\right) \rightarrow Q \times \mathbb{N}$, where

- $T'(0^{(k+1) \cdot \ell} z) \stackrel{\text{def}}{=} \widehat{T}(z)$ for all $z \in Z$,

and for all $i \in [0, k]$ we put

- $T'(0^{i \cdot \ell} w) \stackrel{\text{def}}{=} i \cdot d + T(w)$ for all $w \in W$, and

- $T'(0^{i \cdot \ell} xy) \stackrel{\text{def}}{=} T_x(y)$ for all $x \in X$ and all $y \in U_x$.

It easily checked that the result is a reachability tree for $T'(\varepsilon) = q(n)$. ◀

A consequence of the previous lemma is that in particular $T(\varepsilon)$ is reachable for every expandable partial reachability tree T . By combining Lemmas 12 and 14, we obtain the following characterisation of reachability in BVASS₁.

► **Proposition 15.** *A node $q(n)$ is reachable if, and only if, there exists an expandable B -bounded partial reachability tree T with $T(\varepsilon) = q(n)$, where $B \stackrel{\text{def}}{=}} 2 \cdot |Q| + n$.*

Algorithm 1 An alternating logspace procedure for reachability in BVASS₁.

```

1: procedure REACH( $q(n)$ )
2:   if  $n \notin [0, B]$  then return false
3:   if  $q(n) \in F \times \{0\}$  then return true
4:   else non-deterministically guess  $t \in \Delta \cap (\{q\} \times Q \times Q \cup \{q\} \times \{-1, 0, 1\} \times Q)$ 
5:     if  $t = (q, p_1, p_2) \in Q^3$  then
6:       non-deterministically guess  $m_1, m_2 \in [0, B]$  s.t.  $n = m_1 + m_2$ 
7:       return (REACH( $p_1(m_1)$ ) and REACH( $p_2(m_2)$ ))
8:       or (ANCHORREACH( $q(n), p_1(m_1)$ ) and REACH( $p_2(m_2)$ ))
9:       or (ANCHORREACH( $q(n), p_2(m_2)$ ) and REACH( $p_1(m_1)$ ))
10:    else let  $t = (q, z, p) \in Q \times \{-1, 0, 1\} \times Q$ 
11:      return REACH( $p(n+z)$ ) or ANCHORREACH( $q(n), p(n+z)$ )

12: procedure ANCHORREACH( $q(n), p(m)$ )
13:   if  $\{n, m\} \not\subseteq [0, B]$  then return false
14:   if  $p = q$  and  $m > n$  and RESIDUEREACH( $q(n), m-n$ ) then return true
15:   else non-deterministically guess  $t \in \Delta \cap (\{p\} \times Q \times Q \cup \{p\} \times \{-1, 0, 1\} \times Q)$ 
16:     if  $t = (p, p_1, p_2) \in Q^3$  then
17:       non-deterministically guess  $m_1, m_2 \in [0, B]$  s.t.  $m = m_1 + m_2$ 
18:       return ANCHORREACH( $q(n), p_1(m_1)$ ) and REACH( $p_2(m_2)$ )
19:       or ANCHORREACH( $q(n), p_2(m_2)$ ) and REACH( $p_1(m_1)$ )
20:     else let  $t = (p, z, p') \in Q \times \{-1, 0, 1\} \times Q$ 
21:       return ANCHORREACH( $q(n), p'(m+z)$ )
  
```

4.3 The Algorithm

In this section, we provide an alternating logspace procedure for reachability in BVASS₁. This shows that reachability in BVASS₁ is decidable in deterministic polynomial time since alternating logspace equals deterministic polynomial time [1]. We employ the characterisation of reachability in BVASS₁ in terms of expandable B -bounded partial reachability of Proposition 15. First, by Theorem 9 we may assume the existence of an alternating logspace procedure for residue reachability in BVASS₁, i.e., an alternating logspace procedure RESIDUEREACH($q(n_0), d$) that has an accepting computation if, and only if, $q(n)$ is reachable for some $n \geq n_0$ and $n \equiv n_0 \pmod{d}$. By application of this procedure, we show that one can construct an alternating logspace procedure REACH($q(n)$) that takes a configuration $q(n)$ as input and that has an accepting computation if, and only if, there exists an expandable B -bounded partial reachability tree T with $T(\varepsilon) = q(n)$.

The idea is to simply to guess an expandable B -bounded partial reachability tree T in a top-down manner. The procedure REACH is defined above in Algorithm 1. First in Line 2, REACH rejects whenever the counter value n is not in $[0, B]$ and accepts if $q(n)$ is an accepting configuration (Line 3). Thus, subsequently we may assume that $n \in [0, B]$. In Line 4, we non-deterministically choose a transition $t \in \Delta$. If $t = (q, p_1, p_2) \in Q^3$ is a branching rule, we non-deterministically guess how n can be decomposed as $n = m_1 + m_2$. Moreover, we non-deterministically guess whether the currently processed inner node of T labelled by $q(n)$ will be an anchor of some pumping leaf “below.” If not then we simply recursively call REACH($p_1(m_1)$) and REACH($p_2(m_2)$) (Line 7). Otherwise, $q(n)$ will be the anchor of some pumping leaf that is either in the subtree “rooted at” $p_1(m_1)$ (Line 8) or in the subtree “rooted at” $p_2(m_2)$ (Line 9). Speaking in terms of Lemma 13, either the inner node corresponding to configuration $p_1(m_1)$ is not exclusive or the one for $p_2(m_2)$ is not exclusive. Suppose $p_1(m_1)$ is not exclusive, we then call a procedure ANCHORREACH($q(n), p(m_1)$) that takes two configurations as arguments and tacitly assumes the first argument $q(n)$ is the anchor and the second argument $p_1(m_1)$ corresponds to some inner node that lies between the anchor and the pumping leaf it will eventually correspond to.

In more detail, analogously to REACH the procedure ANCHORREACH first checks whether the counter values of its inputs both lie in $[0, B]$ (Line 13). If so it checks whether $p(m)$ corresponds to a valid pumping leaf of $q(n)$, i.e., it induces a positive instance of the residue reachability problem by invoking RESIDUEREACH($q(n), m - n$) (Line 14). If not then a rule $t \in \Delta$ is non-deterministically chosen (Line 15), and in case t is a branching rule, it is non-deterministically chosen which “child” of $p(m)$ is not exclusive, the other child is simply checked for reachability by invoking procedure REACH (Lines 18 and 19).

Obviously, REACH and ANCHORREACH can be implemented in alternating logspace since the involved counter values lie in the interval $[-1, B + 1]$ and can hence be stored using a logarithmic number of bits.

5 Coverability and Boundedness

In this section, we show that the coverability and boundedness problem for BVASS₁ are also P-complete. The two problems are defined as follows:

COVERABILITY AND BOUNDEDNESS IN BVASS₁

INPUT: A BVASS₁ $\mathcal{B} = (Q, \Delta, F)$, a control state q and $n \in \mathbb{N}$ encoded in unary.

QUESTION: *Coverability:* Is there $m \geq n$ such that $q(m)$ is reachable?

Boundedness: Is $\text{reach}(q)$ finite?

If $q(n)$ is a positive instance of coverability then we call the configuration $q(n)$ *coverable*. A state q is *unbounded* whenever $\text{reach}(q)$ is unbounded (i.e. infinite).

Hardness for P is in both cases easily seen and similar to the P-hardness reduction from MCVP in Proposition 3.

Moreover, the P upper bound for coverability follows easily from the P upper bound for residue reachability since $q(n)$ is coverable if, and only if, the pair $(q(n), 1)$ is a positive instance of the residue reachability problem.

► **Theorem 16.** *Coverability in BVASS₁ is P-complete.*

The P upper bound for boundedness, however, cannot be derived immediately. In particular, as discussed in Section 3, there exists a family of BVASS₁ $(\mathcal{B}_n)_{n \geq 0}$ with some control state q such that $\text{reach}(q)$ is finite but of cardinality 2^n .

For the remainder of this section, fix some BVASS₁ $\mathcal{B} = (Q, \Delta, F)$. We first provide sufficient and necessary criteria that witness that a control state is unbounded. Call a node v in a reachability tree *decreasing* if there is an ancestor $u \prec v$ with $\text{state}(u) = \text{state}(v)$ and $\text{counter}(u) > \text{counter}(v)$. The following lemma, whose proof is deferred to the appendix, shows that a reachability tree that contains some decreasing node witnesses that the control state at its root is unbounded.

► **Lemma 17.** *If a reachability tree T with $T(\varepsilon) = q(n)$ contains a decreasing node then q is unbounded.*

Conversely, the next lemma shows that a reachability tree whose root is labelled with a configuration with a sufficiently large counter value gives rise to a reachability tree which contains a decreasing node, informally speaking, shortly below its root.

► **Lemma 18.** *Suppose $n > 2^{|Q|}$ with $n \in \text{reach}(q)$. There exists a reachability tree $T: U \rightarrow Q \times \mathbb{N}$ for $q(n')$ where $n' \geq n$, and which contains a decreasing node v with $|v| \leq |Q|$.*

A consequence of the two previous lemmas is that q is unbounded if, and only if, $\text{reach}(q)$ contains some $n > 2^{|Q|}$. Even though the reachability trees in Lemma 18 are sufficient witnesses for unboundedness, they still contain much more information than necessary and are potentially of exponential size. In order to verify the existence of such a tree, exact counter values and in fact the subtrees rooted in v as well as all incomparable nodes can be abstracted away, as shown in the lemma below.

Let us write $\text{src}(t) \stackrel{\text{def}}{=} q$, $\text{trg}(t) \stackrel{\text{def}}{=} \{p, p'\}$ and $\text{eff}(t) \stackrel{\text{def}}{=} 0$, for the *source* and *target states* and the *effect* of a branching transition $t = (q, p, p') \in Q^3$, respectively. Similarly, for $t = (q, z, p)$ define $\text{src}(t) \stackrel{\text{def}}{=} q$, $\text{trg}(t) \stackrel{\text{def}}{=} \{p\}$ and $\text{eff}(t) \stackrel{\text{def}}{=} z$.

► **Lemma 19.** *A control state p_0 is unbounded if, and only if, there is a sequence of control states and transitions $p_0 t_1 p_1 t_2 \cdots t_k p_k$ with $k \leq |Q|$ and some index $j < k$ such that*

- (i) $p_{i-1} = \text{src}(t_i)$ and $p_i \in \text{trg}(t_i)$ for all $1 \leq i \leq k$;
- (ii) $p_k = p_j$ and $p_i \neq p_j$ for all $0 \leq i < j$;
- (iii) $p(0)$ is coverable for every $p \in \bigcup_{i=1}^k \text{trg}(t_i)$; and
- (iv) for every $j < i \leq k$, there exists $n_i \leq |Q| + 1$ such that

- a. if $t_i = (p_{i-1}, p_i, p'_i) \in Q^3$ or $t_i = (p_{i-1}, p'_i, p_i) \in Q^3$ then $p'_i(n_i)$ is coverable, else $n_i = 0$,
- b. $\sum_{i=j+1}^k n_i > \sum_{i=j+1}^k \text{eff}(t_i)$.

The last condition (iv) expresses that the cyclic suffix is consistent with the transition relation and guarantees a decreasing node.

Proof. If p_0 is unbounded, then by Lemma 18 we can take a reachability tree T containing a short decreasing node v , i.e., with $|v| \leq |Q|$. This decreasing node provides the claimed sequence: Conditions (i) and (ii) are immediate; for condition (iii) notice that for each mentioned state p some configuration $p(n)$ is reachable, as guaranteed by the respective subtree of T . This means in particular that $p(0)$ is coverable.

For (iv), first notice that the combined effect $\sum_{i=j+1}^k \text{eff}(t_i)$ of those transitions used between v (where $\text{state}(v) = p_k$) and its anchor (with state $p_j = p_k$) is bounded by $|v| = k \leq |Q|$. Secondly, as for condition (iii), we can assume that for all p'_i such that either $t_i = (p_{i-1}, p'_i, p_i) \in Q^3$ or $t_i = (p_{i-1}, p_i, p'_i) \in Q^3$, some configuration $p'_i(m_i)$ is reachable. For those $i \leq k$ where $t_i \notin Q^3$, let $m_i \stackrel{\text{def}}{=} 0$. Now, for all $j < i \leq k$, define $n_i \stackrel{\text{def}}{=} \min\{|Q| + 1, m_i\}$.

Case (iv)(a) holds immediately by definition of the n_i . To show Case (iv)(b) we distinguish two cases. In case $m_i \geq |Q| + 1$ for some such i it follows that $n_i = |Q| + 1$ and hence $\sum_{i=j+1}^k n_i \geq |Q| + 1 > \sum_{i=j+1}^k \text{eff}(t_i)$. Otherwise, if all $m_i < |Q| + 1$ then for all i it holds that $n_i = m_i$ and so $\sum_{i=j+1}^k n_i \leq \sum_{i=j+1}^k \text{eff}(t_i)$ contradicts that v is a decreasing node.

For the converse direction, assume a sequence as claimed above. Conditions (i)-(iii) imply the existence of a reachability tree for some $p_0(n)$. Condition (iv) ensures that there is such a tree with a decreasing node. We conclude by Lemma 17. ◀

Lemma 19 provides a characterisation of unbounded states that directly translates into an alternating logspace algorithm for the boundedness problem, similar to Algorithm 1, which yields the P upper bound. In particular, observe that a witnessing sequence satisfying Conditions (i) and (ii), as well as the numbers $n_i \leq |Q| + 1$ can be guessed non-deterministically in logarithmic space. Moreover, Conditions (iii) and (iv) are decidable in polynomial time by Theorem 16.

► **Theorem 20.** *Boundedness in BVASS₁ is P-complete.*

6 Conclusion

We showed that reachability, coverability and boundedness in $BVASS_1$ are all P-complete and thereby established the first decidability result for reachability in a subclass of BVASS. This low complexity is quite surprising since the general reachability problem for BVASS is at least non-elementary [10] and there exist families of instances of $BVASS_1$ -reachability problems whose reachability trees contain an exponential number of distinct counter values, cf. Section 3. The approach developed in this paper shows that it is not necessary to explicitly construct a full reachability tree in order to witness reachability. In fact, we showed in Section 4 that the existence of so-called residue and expandable reachability trees suffices in order to decide reachability and can be witnessed in polynomial time.

Our approach is quite specific to having only one counter available in $BVASS_1$ and does not seem to immediately generalise to higher dimensions. Nevertheless, we believe that this paper spreads some optimism and provides sufficient evidence that obtaining results for reachability in general BVASS is not impossible.

References

- 1 A.K. Chandra, D. Kozen, and L.J. Stockmeyer. Alternation. *J. ACM*, 28(1):114–133, 1981. doi:10.1145/322234.322243.
- 2 Ph. de Groote, B. Guillaume, and S. Salvati. Vector addition tree automata. In *Logic in Computer Science, LICS*, pages 64–73. IEEE Computer Society, 2004. doi:10.1109/LICS.2004.1319601.
- 3 S. Demri, M. Jurdziński, O. Lachish, and R. Lazić. The covering and boundedness problems for branching vector addition systems. *J. Comput. Syst. Sci.*, 79(1):23–38, 2013. doi:10.1016/j.jcss.2012.04.002.
- 4 M. Englert, R. Lazić, and P. Totzke. Reachability in two-dimensional unary vector addition systems with states is NL-complete. In *Logic in Computer Science, LICS*, 2016. To appear.
- 5 P. Ganty and R. Majumdar. Algorithmic verification of asynchronous programs. *ACM Trans. Program. Lang. Syst.*, 34(1):6, 2012. doi:10.1145/2160910.2160915.
- 6 S.M. German and A.P. Sistla. Reasoning about systems with many processes. *J. ACM*, 39(3):675–735, 1992. doi:10.1145/146637.146681.
- 7 R. Greenlaw, H.J. Hoover, and W.L. Ruzzo. *Limits to parallel computation: P-completeness theory*. Oxford University Press, 1995.
- 8 P. Jančar and Z. Sawa. A note on emptiness for alternating finite automata with a one-letter alphabet. *Inf. Process. Lett.*, 104(5):164–167, 2007. doi:10.1016/j.ipl.2007.06.006.
- 9 A. Jez and A. Okhotin. Conjunctive grammars over a unary alphabet: Undecidability and unbounded growth. *Theory Comput. Syst.*, 46(1):27–58, 2010. doi:10.1007/s00224-008-9139-5.
- 10 R. Lazić and S. Schmitz. Nonelementary complexities for branching VASS, MELL, and Extensions. *ACM Trans. Comput. Log.*, 16(3):20, 2015. doi:10.1145/2733375.
- 11 J. Leroux and S. Schmitz. Demystifying reachability in vector addition systems. In *Logic in Computer Science, LICS*, pages 56–67. IEEE, 2015. doi:10.1109/LICS.2015.16.
- 12 R.J. Lipton. The reachability problem requires exponential space. *Yale University*, Technical Report 62, 1976.
- 13 C.H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.
- 14 C. Rackoff. The covering and boundedness problems for vector addition systems. *Theor. Comput. Sci.*, 6:223–231, 1978. doi:10.1016/0304-3975(78)90036-1.

- 15 O. Serre. Parity games played on transition graphs of one-counter processes. In *Foundations of Software Science and Computation Structures, FoSSaCS*, pages 337–351, 2006. doi:10.1007/11690634_23.
- 16 L.G. Valiant and M. Paterson. Deterministic one-counter automata. *J. Comput. Syst. Sci.*, 10(3):340–350, 1975. doi:10.1016/S0022-0000(75)80005-5.

A

 Missing Proofs

A.1 Missing Proofs from Section 3

An instance of MCVP is a Boolean circuit \mathcal{C} consisting of n gates g_1, \dots, g_n such that for all $k \in [1, n]$ either $g_k = \top$, $g_k = \perp$ or there are $1 \leq i, j < k$ such that $g_k = g_i \vee g_j$ or $g_k = g_i \wedge g_j$. MCVP is to decide whether \mathcal{C} evaluates to true, i.e. if g_n evaluates to true. We note that MCVP is the canonical P-complete problem [7]. The following proposition gives the lower bound for Theorem 2.

► **Proposition 3.** *Let \mathcal{C} be a Boolean circuit. There exists a logspace computable BVASS₁ \mathcal{B} with a control state q such that $q(0)$ is reachable if, and only if, \mathcal{C} evaluates to true.*

Proof. From \mathcal{C} we derive a BVASS₁ $\mathcal{B} \stackrel{\text{def}}{=} (Q, \Delta, F)$, where $Q \stackrel{\text{def}}{=} \{q_1, \dots, q_n\}$, $F \stackrel{\text{def}}{=} \{q_i \mid g_i = \top\}$ and $\Delta \stackrel{\text{def}}{=} \{(q_k, q_i, q_j) \mid g_k = g_i \wedge g_j\} \cup \{(g_k, 0, g_i), (g_k, 0, g_j) \mid g_k = g_i \vee g_j\}$. Hence, \wedge -gates are simulated by splits and \vee -gates by non-deterministic branching. It is easily seen that g_n evaluates to true if, and only if, $q_k(0)$ is reachable in \mathcal{B} . ◀

► **Proposition 4.** *Reachability in BVASS₁ is NP-hard if the initial configuration $q(n)$ is given in binary.*

Proof. We first show that for any $m \in \mathbb{N}$ given in binary, we can in logarithmic space extend \mathcal{B}_n constructed above with a control state q_m such that $\text{reach}(q_m) = \{m\}$. Let $m = \sum_{0 \leq i \leq n} b_i \cdot 2^i$ with $b_i \in \{0, 1\}$ be the binary representation of m . We introduce additional fresh control states q_m^i , $0 \leq i \leq n$, transitions $(q_m, 0, q_m^n)$ and (q_m^0, q_0, q_f) , and for every $1 \leq i \leq n$ transitions $(q_m^i, 0, q_m^{i-1})$ if $b_i = 0$ and (q_m^i, q_i, q_m^{i-1}) if $b_i = 1$. It is easily checked that $\text{reach}(q_m) = \{m\}$.

In order to show hardness for NP, we reduce from the problem SUBSET SUM. Given a finite set $S = \{m_1, \dots, m_k\} \subseteq \mathbb{N}$ and $t \in \mathbb{N}$ with all numbers encoded in binary, SUBSET SUM is the problem to decide whether there are $c_1, \dots, c_k \in \{0, 1\}$ such that $t = \sum_{1 \leq i \leq k} c_i \cdot m_i$ and is known to be NP-complete [13]. As shown above, we can construct a BVASS \mathcal{B} with control states q_{m_1}, \dots, q_{m_k} such that $\text{reach}(q_{m_i}) = \{m_i\}$. We introduce additional fresh control states q_{c_1}, \dots, q_{c_k} that allow us to non-deterministically make a choice for every c_i by introducing for every $1 \leq i < k$ transitions $(q_{c_i}, 0, q_{c_{i+1}})$ and $(q_{c_i}, q_{m_i}, q_{c_{i+1}})$. It is now easily seen that the instance (S, t) of SUBSET SUM is valid if, and only if, $q_{c_1}(t)$ is reachable. ◀

► **Corollary 21.** *Reachability in BVASS₂ is NP-hard.*

Proof (sketch). The statement follows from an easy adaption of the proof of Proposition 4. It suffices to show how to construct a BVASS₂ that reaches the control state q_{c_1} from Proposition 4 with counter values $(t, 0)$. But this can easily be achieved by first adding a non-deterministic number of times $(1, 1)$ to the counter and then by branching into the control states q_{c_1} and q_t , where q_t is suitably adjusted such that $\text{reach}(q_t) = \{(0, t)\}$. ◀

A.2 Missing Proofs from Section 4.1

► **Lemma 6.** *The set S is computable in polynomial time.*

Proof. We note that N is polynomially bounded in $|\mathcal{B}| + |d|$. Moreover, $S \subseteq Q \times [0, n + N]$ and S can be computed in polynomial time by using a dynamic programming approach. ◀

► **Lemma 7.** *The fixed point $R \stackrel{\text{def}}{=} \bigcup_{i \geq 0} R_i$ equals R_N and is computable in polynomial time.*

Proof. Analogously to the computation of S in Lemma 6, one shows that R_0 is computable in polynomial time. To see that $R = R_N$, note that by definition we have $R_i \subseteq R_{i+1}$ for all $i \in \mathbb{N}$. If $R_i \subset R_{i+1}$, there is at least one pair from $Q \times \mathbb{Z}_d$ that is in R_{i+1} and not in R_i . Since there are at most N such pairs, the sequence stabilises after at most N steps at R_N . Since N is polynomial in $|\mathcal{B}| + d$, consequently R_N can also be computed in polynomial time. ◀

► **Lemma 8.** *The set $X \stackrel{\text{def}}{=} R \cup S[n_0, n_0 + N] / \mathbb{Z}_d$ is computable in polynomial time. Moreover,*

$$X = \{(q, n \bmod d) \mid q \in Q, n \in \text{reach}(q), n \geq n_0\}.$$

Proof. Polynomial-time computability of X follows immediately from the polynomial time computability of S (Lemma 6) and of R (Lemma 7). It thus remains to prove that $X = \{(q, n \bmod d) \mid q \in Q, n \in \text{reach}(q), n \geq n_0\}$.

(“ \subseteq ”) Trivially, $S[n_0, n_0 + N] / \mathbb{Z}_d \subseteq \{(q, n \bmod d) \mid q \in Q, n \in \text{reach}(q), n \geq n_0\}$ since $S \subseteq \{(q, n) \mid q \in Q, n \in \text{reach}(q)\}$. Hence it remains to show that R is contained in $\{(q, n \bmod d) \mid q \in Q, n \in \text{reach}(q), n \geq n_0\}$.

To prove this, we show that for every $i \in [0, N]$ and each $(q, r) \in R_i$ there exists some $n \in \text{reach}(q)$ with $n \geq n_0 + N - i$ and $n \equiv r \pmod{d}$ by induction on i . We note that this is sufficient to prove since $R = R_N$ and thus for each $(q, r) \in R$ there exists some $n \in \text{reach}(q)$ with $n \geq n_0$ and $n \equiv r \pmod{d}$.

For the induction base, i.e. $i = 0$, we recall that for each $(q, r) \in R_0$ there exists some $n \geq n_0 + N = n_0 + N - i$ such that $n \equiv r \pmod{d}$ and there is some almost $(n_0 + N)$ -bounded reachability tree whose root is labelled with $q(n)$ by definition of R_0 ; in particular $n \in \text{reach}(q)$.

For the induction step, let $i+1 \leq N$ and let us assume $(q, r) \in R_{i+1}$. If already $(q, r) \in R_i$ then (q, r) satisfies the desired property immediately by the induction hypothesis. Otherwise, if $(q, r) \in \Delta(R_i)$ then $r \equiv r' - z \pmod{d}$ for some $(q', r') \in R_i$ and some $(q, z, q') \in \Delta$. By the induction hypothesis, there exists some $n' \in \text{reach}(q')$ with $n' \geq n_0 + N - i$ and $n' \equiv r' \pmod{d}$. For $n = n' - z$, we have $n \equiv r' - z \equiv r \pmod{d}$ and since $n \geq n' - |z| \geq n_0 + N - i - |z| \geq n_0 + N - (i + 1) \geq n_0 \geq 0$ it follows $n \in \text{reach}(q)$.

It remains to consider the case when $(q, r) \in \Delta(R_i, S/\mathbb{Z}_d) \cup \Delta(S/\mathbb{Z}_d, R_i) \cup \Delta(R_i, R_i)$. We only treat the case $(q, r) \in \Delta(R_i, S/\mathbb{Z}_d)$, the other cases can be proven analogously. In this case we have $r \equiv r' - n'' \pmod{d}$ for some $(p', r') \in R_i$ and some $(p'', n'') \in S$, where $(q, p', p'') \in \Delta$. Clearly, $n'' \in \text{reach}(p'')$ by definition of S . By the induction hypothesis, there exists some $n' \geq n_0 + N - i$ such that $n' \in \text{reach}(p')$ and $n' \equiv r' \pmod{d}$. Hence $n \in \text{reach}(q)$, since $n' \in \text{reach}(p')$ and $n'' \in \text{reach}(p'')$. Obviously $n \equiv r \pmod{d}$ and, finally, $n \geq n' \geq n_0 + N - i \geq n_0 + N - (i + 1)$.

(“ \supseteq ”) Assume some $q(n)$ is reachable for some $n \geq n_0$. We prove that $(q, n \bmod d) \in X$. To this end, let us fix some reachability tree $T: U \rightarrow Q \times \mathbb{N}$ for $q(n)$. If T is $(n_0 + N)$ -bounded it follows that $(q, n) \in S$ and we are done since $n_0 \leq n \leq n_0 + N$.

Consequently, let us assume that T is not $(n_0 + N)$ -bounded. First, observe that $T(u) \in F \times \{0\} \subseteq S$ for all leaves $u \in U$. In addition, the set of nodes $V \stackrel{\text{def}}{=} \{u \in U \mid T(u) \notin S\}$ is non-empty for otherwise T would be $(n_0 + N)$ -bounded. Moreover, V is prefix-closed and note that every \preceq -maximal node v in V satisfies $T(v)/\mathbb{Z}_d \in R_0$ by the choice of V . For every node $v \in V$ that is not \preceq -maximal, we either have $T(v)/\mathbb{Z}_d \in \Delta(T(v_0)/\mathbb{Z}_d)$ (if v_0 is the only child of v) or $T(v)/\mathbb{Z}_d \in \Delta(T(v_0)/\mathbb{Z}_d, T(v_1)/\mathbb{Z}_d)$ (if v has two children v_0 and v_1). Moreover, note that $\Delta(R) \subseteq R$ and $\Delta(R, R) \subseteq R$. This shows that $T/\mathbb{Z}_d(V)$ is contained in R , in particular $T/\mathbb{Z}_d(\varepsilon) \in R$ which proves $(q, n \bmod d) \in R \subseteq X$. ◀

A.3 Missing Proofs from Section 4.2

► **Lemma 11.** *Let u and v be nodes of a partial reachability tree such that $u \prec v$ and $\text{counter}(u) + |Q| \leq \text{counter}(v)$. Then there exists an increasing node v' with anchor u' such that $u \preceq u' \prec v' \preceq v$.*

Proof. The counter value of a node exceeds that of its parent by at most one. Consequently, for every $\text{counter}(u) \leq i \leq \text{counter}(v)$ there is a node $u \preceq u_i \preceq v$ with $\text{counter}(u_i) = i$ and further, for all $j < k$ in between $\text{counter}(u)$ and $\text{counter}(v)$ it holds that $u_j \prec u_k$. Since $\text{counter}(v) - \text{counter}(u) \geq |Q|$, there must be some $\text{counter}(u) \leq j < k \leq \text{counter}(v)$ such that $\text{state}(u_j) = \text{state}(u_k)$. Then $u' \stackrel{\text{def}}{=} u_j$ and $v' \stackrel{\text{def}}{=} u_k$ satisfy the claim. ◀

► **Lemma 13.** *For every node u of an expandable partial reachability tree the following hold:*

- (i) *If u is the anchor of an increasing leaf v then u is expandable and all nodes w such that $u \prec w \preceq v$ are not expandable.*
- (ii) *u has at most one child that is not expandable.*

Proof. Regarding (i), every anchor u' of every other increasing leaf v' of $T^{\downarrow u}$ is a strict descendant of u , since otherwise $u, u' \preceq \text{lca}(v, v')$, contradicting T being exclusive. Consequently, T being exclusive implies u being exclusive. Moreover, for w such that $u \prec w \preceq v$, due to an anchor being maximal, v has no anchor on the subtree rooted at any such w . Hence, w violates the condition of being expandable.

Regarding (ii), suppose u_0 and u_1 are both not expandable. This can only be if both are not exclusive, as T is expandable. If all increasing leaves of $T^{\downarrow u_0}$ had their anchors in $T^{\downarrow u_0}$, then u_0 would be exclusive. Hence there is some increasing leaf v_0 in $T^{\downarrow u_0}$ with anchor u_0 such that $u_0 \preceq u$. Likewise, we find an increasing leaf v_1 with anchor u_1 such that $u_1 \preceq u$ in $T^{\downarrow u_1}$. But then $u_0, u_1 \preceq \text{lca}(v_0, v_1)$ and hence T is not exclusive, a contradiction. ◀

A.4 Missing Proofs from Section 5

► **Lemma 17.** *If a reachability tree T with $T(\varepsilon) = q(n)$ contains a decreasing node then q is unbounded.*

Proof. It suffices to observe that one can unfold the cyclic suffix of a decreasing node v by replacing the subtree rooted in v by that one rooted in $u \prec v$. This construction is analogous to the construction in the proof of Lemma 14, with the only difference that the effect of the cycle is negative here. The result of such an operation is a reachability tree whose root is labelled with a configuration that has the same control state and whose counter value is strictly increased. Moreover, this reachability tree still contains a decreasing node. Such an unfolding can therefore be repeated arbitrarily often, from which the claim follows. ◀

► **Lemma 18.** *Suppose $n > 2^{|Q|}$ with $n \in \text{reach}(q)$. There exists a reachability tree $T: U \rightarrow Q \times \mathbb{N}$ for $q(n')$ where $n' \geq n$, and which contains a decreasing node v with $|v| \leq |Q|$.*

Proof. In any reachability tree it is possible to collapse the part between two nodes $u \prec v$ if $\text{state}(u) = \text{state}(v)$ and $\text{counter}(u) \leq \text{counter}(v)$, that is, to replace the subtree rooted in u by the one rooted in v . The result of this is a reachability tree with fewer nodes and where the root has the same state and a counter value at least as large as in the original tree.

Thus, we may assume with no loss of generality a reachability tree T with root $T(\varepsilon) = q(n)$ for $n \geq 2^{|Q|}$ and such that for any two nodes $u \prec v$ with $\text{state}(u) = \text{state}(v)$, it holds that $\text{counter}(u) > \text{counter}(v)$.

In order to find a decreasing node, we move from the root downwards, always choosing the successor with the largest counter value. This way, the counter value of a chosen node is at least half as large as the counter of its parent. Since the value in the root is greater or equal to $2^{|Q|}$, this means that the produced sequence is longer than $|Q|$. In particular, the prefix of length $|Q|$ must contain a decreasing node. ◀