

On the Reachability Problem for Two-Dimensional Branching VASS

Clotilde Bizière 


LaBRI, Univ. Bordeaux, CNRS, Bordeaux INP, Talence, France

Thibault Hilaire 

LaBRI, Univ. Bordeaux, CNRS, Bordeaux INP, Talence, France

Jérôme Leroux 

LaBRI, Univ. Bordeaux, CNRS, Bordeaux INP, Talence, France

Grégoire Sutre 

LaBRI, Univ. Bordeaux, CNRS, Bordeaux INP, Talence, France

Abstract

Vectors addition systems with states (VASS), or equivalently Petri nets, are arguably one of the most studied formalisms for the modeling and analysis of concurrent systems. A central decision problem for VASS is reachability: whether there exists a run from an initial configuration to a final one. This problem has been known to be decidable for over forty years, and its complexity has recently been precisely characterized. Our work concerns the reachability problem for BVASS, a branching generalization of VASS. In dimension one, the exact complexity of this problem is known. In this paper, we prove that the reachability problem for 2-dimensional BVASS is decidable. In fact, we even show that the reachability set admits a computable semilinear presentation. The decidability status of the reachability problem for BVASS remains open in higher dimensions.

2012 ACM Subject Classification Theory of computation → Logic and verification

Keywords and phrases Vector addition systems, Reachability problem, Semilinear sets, Verification

Digital Object Identifier 10.4230/LIPIcs.MFCS.2025.22

Related Version *Full Version:* <https://arxiv.org/abs/2506.22561> [2]

1 Introduction

Vectors addition systems with states (VASS), or equivalently Petri nets, are arguably one of the most studied formalisms for the modeling and analysis of concurrent systems. A central decision problem for VASS is reachability: whether there exists a run from an initial configuration to a final one. This problem was shown decidable more than forty years ago [26] but its precise complexity was only established a few years ago [23, 9, 22]. Several VASS extensions have been introduced and studied, most notably unordered data nets [20], pushdown VASS [1, 19], and branching VASS [10, 31]. But so far, the reachability problem is still open for these models.

One of the first subclasses of VASS for which reachability was shown to be decidable is the class of 2-dimensional VASS. For this class, Hopcroft and Pansiot devised an algorithm that computes a finite description (more precisely, a semilinear presentation) of the reachability set [16]. As an immediate consequence, they obtained that reachability is decidable for this class. In fact, the algorithm of Hopcroft and Pansiot can be viewed as a refinement of the classical Karp-Miller algorithm [18] where the abstract pumping of cycles (putting ω in some components) is replaced by an exact acceleration of cycles (adding new vectors to the current set of periods).



© Clotilde Bizière, Thibault Hilaire, Jérôme Leroux, and Grégoire Sutre;
licensed under Creative Commons License CC-BY 4.0

50th International Symposium on Mathematical Foundations of Computer Science (MFCS 2025).

Editors: Paweł Gawrychowski, Filip Mazowiecki, and Michał Skrzypczak; Article No. 22; pp. 22:1–22:19

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

In this paper, we investigate the reachability problem for branching VASS (shortly called BVASS in the sequel), a branching generalization of VASS. More precisely, BVASS extend VASS with special branching transitions that merge configurations (by summing their vectors). This model has gained a lot of interest recently due to strong links with several fields in computer science such as cryptographic protocols [31], linear logic [10, 21], recursively parallel programs [5], timed pushdown systems [7], computational linguistic [29, 30], game semantics [8], equational tree automata [28, 25] and data logics [17, 4]. For instance, provability in the multiplicative exponential fragment of linear logic (MELL) is inter-reducible with the reachability problem in BVASS [10]. As mentioned before, the reachability problem is still open in arbitrary dimension for BVASS. In dimension one, the reachability problem is decidable and the exact complexity is known [15, 12]. The objective of our work was to investigate the decidability of the reachability problem for 2-dimensional BVASS.

Contributions. In this paper, we prove that the reachability problem for 2-dimensional BVASS is decidable. In fact, we even show that the reachability set admits a computable semilinear presentation. We propose an algorithm that essentially performs a forward symbolic exploration of a 2-dimensional BVASS given as input. Our algorithm is inspired from Hopcroft and Pansiot’s algorithm for classical 2-dimensional VASS [16]. The latter computes a symbolic reachability tree, but in our case we need an acyclic graph, that we call *exploration*, because of branching transition rules.

Compared to Hopcroft and Pansiot’s algorithm where pumped cycles are computed statically, in our case pumped cycles are computed dynamically since they exploit configurations that are discovered during the exploration. This feature complicates the proof of soundness of our algorithm. But the main challenge is the proof of termination. As usual, we proceed by contradiction and assume that the algorithm constructs an infinite exploration. A first source of difficulty in order to obtain a contradiction is that the set of pumped cycles is potentially infinite. A second source is the fact that we cannot consider any infinite path of the exploration. In fact, there are mutual dependencies between paths since the exploration is not necessarily a tree. We need to consider an infinite path that ultimately does not depend on the other paths, and we show that such a path always exist. We believe that our proof techniques could be applied to other algorithms that construct potentially infinite acyclic graphs.

Related Work. In general dimension, the coverability problem (a weak version of the reachability problem) and the boundedness problem are decidable for BVASS [31], and their precise complexity is known [11, 21]. The complexity of the reachability problem for bounded BVASS was established in [27]. The reachability problem for BVASS and pushdown VASS is still open. For pushdown VASS the problem is known to be decidable in the bidirected case [14]. In small dimensions, the above-mentioned idea of pumping cycles was successfully applied to the analysis of several VASS generalizations, and in particular to solve reachability for 2-dimensional VASS and extensions [3, 6, 13], coverability for 1-dimensional pushdown VASS [24], and reachability for 1-dimensional BVASS [15, 12].

Outline. Some preliminary background and notations are provided in Section 2. We define in Section 3 the model of BVASS and their semantics. Section 4 introduces a class of acyclic graphs, called *explorations*, where nodes are labeled by sets of configurations, and presents our reachability algorithm for 2-BVASS. We show in Section 5 that the explorations constructed by our algorithm, called *algorithmic* explorations, are sound and complete (for

the reachability set). This shows the partial correctness of our algorithm, and we then focus on its termination. We prove in Section 6 that any infinite graph that admits a finitely-branching spanning forest contains a “core” witness of infinity defined as a so-called directed and primary graph. Section 7 provides the proof of termination of our algorithm, and is decomposed into four subsections. First, we study the stabilization of cones, a form of acceleration for cones through the so-called notion of *modes*. Second, we show how to decompose the effect of paths in an exploration into a sum of so-called *elementary* vectors corresponding to previously mentioned *pumped cycles*, and *consecutive* vectors. Third, we prove that the periodic set associated to a primary infinite set of nodes in an algorithmic exploration is finitely-generated. Fourth, we assemble the results from the previous sections to deduce the termination of our algorithm. Section 8 concludes the paper. *Due to space limitations, detailed proofs are deferred to the full version [2] of the paper.*

2 Preliminaries

We denote by \mathbb{Z} the set of integers, \mathbb{N} the set of natural numbers, by \mathbb{Q} the set of rational numbers, and by $\mathbb{Q}_{\geq 0}$ the set of non-negative rational numbers. We also introduce $\mathbb{N}_{>0}$ and $\mathbb{Q}_{>0}$ defined as $\mathbb{N} \setminus \{0\}$ and $\mathbb{Q}_{\geq 0} \setminus \{0\}$, respectively. The powerset of a set S is written $\mathbb{P}(S)$.

Vectors. Vectors are typeset in bold face. Given $\mathbf{c} \in \mathbb{Q}^d$, we let $(\mathbf{c}(1), \dots, \mathbf{c}(d))$ denote the vector of rational numbers defining \mathbf{c} . We write $\mathbf{x} \leq \mathbf{y}$ for two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{Q}^d$ if $\mathbf{x}(i) \leq \mathbf{y}(i)$ for every $i \in \{1, \dots, d\}$. The sum of two vectors $\mathbf{x} + \mathbf{y}$ is defined component-wise. The sum operator over vectors is extended over sets $\mathbf{X}, \mathbf{Y} \subseteq \mathbb{Q}^d$ by $\mathbf{X} + \mathbf{Y} = \{\mathbf{x} + \mathbf{y} \mid \mathbf{x} \in \mathbf{X} \wedge \mathbf{y} \in \mathbf{Y}\}$. Given $\mathbf{X} \subseteq \mathbb{Q}^d$ and $\mathbf{x} \in \mathbb{Q}^d$, we define $\mathbf{x} + \mathbf{X}$ as $\{\mathbf{x}\} + \mathbf{X}$. The set $\mathbf{X} + \mathbf{x}$ is defined similarly. We also write $\mathbb{Q}_{\geq 0}\mathbf{X}$ the set $\{\lambda\mathbf{x} \mid \lambda \in \mathbb{Q}_{\geq 0} \wedge \mathbf{x} \in \mathbf{X}\}$.

Periodic sets and semilinear sets. A set $\mathbf{P} \subseteq \mathbb{Q}^d$ is said to be *periodic* if $\mathbf{0} \in \mathbf{P}$ and $\mathbf{P} + \mathbf{P} \subseteq \mathbf{P}$. Given a set $\mathbf{A} \subseteq \mathbb{Q}^d$, we denote by $\text{Per}(\mathbf{A})$ the set of finite sums $\mathbf{a}_1 + \dots + \mathbf{a}_k$ where $k \in \mathbb{N}$, and $\mathbf{a}_1, \dots, \mathbf{a}_k \in \mathbf{A}$. This periodic set is called the *periodic set spanned* by \mathbf{A} . A periodic set \mathbf{P} is said to be *finitely-generated* if $\mathbf{P} = \text{Per}(\mathbf{A})$ for some finite set $\mathbf{A} \subseteq \mathbb{Q}^d$. The sum of two periodic sets is a periodic set. Given a sequence $(\mathbf{P}_i)_{i \in I}$ of periodic sets indexed by a finite or infinite set I , we denote by $\sum_{i \in I} \mathbf{P}_i$ the periodic set $\text{Per}(\bigcup_{i \in I} \mathbf{P}_i)$. Notice that if I is finite, this definition of sum coincides with the previously introduced finite sum of subsets of \mathbb{Q}^d . A set $\mathbf{L} \subseteq \mathbb{N}^d$ is said to be *linear* if $\mathbf{L} = \mathbf{b} + \mathbf{P}$ where $\mathbf{b} \in \mathbb{N}^d$ and $\mathbf{P} \subseteq \mathbb{N}^d$ is a finitely-generated periodic set. A set $\mathbf{S} \subseteq \mathbb{N}^d$ is *semilinear* if \mathbf{S} is a finite union of linear subsets of \mathbb{N}^d .

Cones. A *cone* \mathbf{C} of \mathbb{Q}^d is a periodic subset of \mathbb{Q}^d such that $\mathbb{Q}_{>0}\mathbf{C} \subseteq \mathbf{C}$. The *cone spanned* by a set $\mathbf{A} \subseteq \mathbb{Q}^d$ is the cone denoted by $\text{Con}(\mathbf{A})$ and defined as $\mathbb{Q}_{>0}\text{Per}(\mathbf{A})$. A cone $\mathbf{C} \subseteq \mathbb{Q}^d$ is said to be *finitely-generated* if $\mathbf{C} = \text{Con}(\mathbf{A})$ for some finite set $\mathbf{A} \subseteq \mathbb{Q}^d$.

► **Lemma 2.1** ([16, Lemma 1.2]). *Let $\mathbf{P} \subseteq \mathbb{Z}^d$ be a periodic set. Then \mathbf{P} is a finitely-generated periodic set if, and only if, $\text{Con}(\mathbf{P})$ is a finitely-generated cone.*

Graphs. A *graph* is a pair $\mathcal{G} = (N, \rightarrow)$ where N is a set of *nodes*, and \rightarrow is a binary relation on N called the *edge relation*. The graph is said to be *empty* (resp. *finite*, *infinite*) when its set of nodes is empty (resp. finite, infinite). We denote by $\xrightarrow{+}$ the transitive closure of \rightarrow , and by $\xrightarrow{*}$ the reflexive closure of $\xrightarrow{+}$. The graph is called *acyclic* when $\xrightarrow{+}$ is irreflexive. We

associate with a node $n \in N$ the set of *ancestors* $\text{Anc}^{\mathcal{G}}(n) = \{m \in N \mid m \xrightarrow{*} n\}$, and the set of *descendants* $\text{Des}^{\mathcal{G}}(n) = \{m \in N \mid n \xrightarrow{*} m\}$. Ancestors and descendant are extended over sets of nodes $X \subseteq N$ as expected, by $\text{Anc}^{\mathcal{G}}(X) = \bigcup_{n \in X} \text{Anc}^{\mathcal{G}}(n)$ and $\text{Des}^{\mathcal{G}}(X) = \bigcup_{n \in X} \text{Des}^{\mathcal{G}}(n)$. A set of nodes $X \subseteq N$ verifying $X = \text{Anc}^{\mathcal{G}}(X)$ is said to be *ancestor-closed*. A node $n \in N$ is called a *leaf* if there does not exist a node $m \in M$ satisfying $n \rightarrow m$. A node n is called a *source* if there is no node m such that $m \rightarrow n$. The *restriction* of a graph $\mathcal{G} = (N, \rightarrow)$ to a set of nodes $X \subseteq N$ is the graph $(X, \rightarrow \cap (X \times X))$. A *node-labeled graph* is a triple $(N, \rightarrow, \lambda)$ where (N, \rightarrow) is a graph and λ is a function with domain N . The notions defined above for graphs naturally carry over to node-labeled graphs.

3 Branching VASS

A d -dimensional *branching vector addition system with states* (d -BVASS for short) is a pair $\mathcal{B} = (Q, \Delta)$ where Q is a finite non-empty set of *states* and $\Delta \subseteq (\mathbb{P}(Q) \times \mathbb{Z}^d \times Q)$ is a finite set of *transition rules*. A transition rule $\delta = (S, \mathbf{a}, q)$ in Δ consists in a set $S \subseteq Q$ of *input states*, a *displacement* $\mathbf{a} \in \mathbb{Z}^d$, and a single *output state* $q \in Q$. Intuitively, assuming that $S = \{q_1, \dots, q_k\}$, this transition rule can be seen as the rewriting rule $q_1(\mathbf{x}_1), \dots, q_k(\mathbf{x}_k) \rightarrow q(\mathbf{a} + \mathbf{x}_1 + \dots + \mathbf{x}_k)$ with formal parameters $\mathbf{x}_1, \dots, \mathbf{x}_k$. Note that our definition forbids a state from occurring twice on the left-hand side of a transition rule (as this left-hand side is given by a set of states). This restriction is only a matter of technical convenience. A transition rule $\delta = (S, \mathbf{a}, q)$ is called *initial* when $S = \emptyset$, *unary* when $|S| = 1$, and *branching* when $|S| \geq 2$. A d -dimensional *vector addition system with states* (d -VASS for short) is a d -BVASS $\mathcal{V} = (Q, \Delta)$ such that $|S| \leq 1$ for every transition rule (S, \mathbf{a}, q) in Δ .

We formulate the semantics of a d -BVASS $\mathcal{B} = (Q, \Delta)$ in terms of a configuration-set transformer Post . A *configuration* of \mathcal{B} is a pair (q, \mathbf{x}) in $Q \times \mathbb{N}^d$, also written as $q(\mathbf{x})$ in the sequel. By extension, given a set $\mathbf{X} \subseteq \mathbb{N}^d$, we let $q(\mathbf{X})$ denote the set of configurations $\{q\} \times \mathbf{X}$. The set of *initial* configurations of \mathcal{B} is $\{q(\mathbf{a}) \mid (\emptyset, \mathbf{a}, q) \in \Delta \text{ and } \mathbf{a} \geq \mathbf{0}\}$. For each transition rule $\delta = (S, \mathbf{a}, q)$ in Δ , we define the function $\text{Post}_\delta : \mathbb{P}(Q \times \mathbb{N}^d) \rightarrow \mathbb{P}(Q \times \mathbb{N}^d)$ by¹

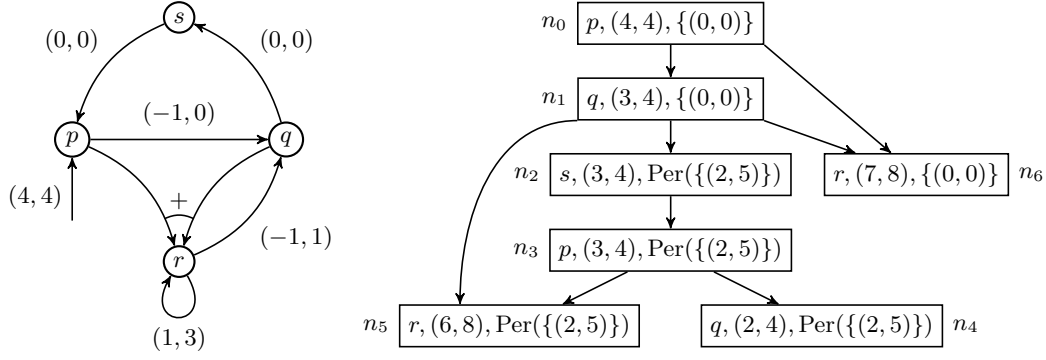
$$\text{Post}_\delta(C) = q \left(\left\{ \mathbf{y} \in \mathbb{N}^d \mid \exists D \subseteq C : S = \{r \mid r(\mathbf{z}) \in D\} \text{ and } \mathbf{y} = \mathbf{a} + \sum_{r(\mathbf{z}) \in D} \mathbf{z} \right\} \right)$$

for every set $C \subseteq Q \times \mathbb{N}^d$. We also introduce $\text{Post}_{\mathcal{B}} : \mathbb{P}(Q \times \mathbb{N}^d) \rightarrow \mathbb{P}(Q \times \mathbb{N}^d)$, defined by $\text{Post}_{\mathcal{B}}(C) = \bigcup_{\delta \in \Delta} \text{Post}_\delta(C)$. Note that $\text{Post}_{\mathcal{B}}$ is \subseteq -nondecreasing and that $\text{Post}_{\mathcal{B}}(\emptyset)$ coincides with the set of initial configurations of \mathcal{B} . The *reachability set* of \mathcal{B} , written $\llbracket \mathcal{B} \rrbracket$, is the \subseteq -least set $C \subseteq Q \times \mathbb{N}^d$ such that $\text{Post}_{\mathcal{B}}(C) \subseteq C$.

► **Example 3.1.** Consider the 2-BVASS \mathcal{E} depicted in Figure 1. It has four states p, q, r, s and seven transition rules, namely $(\emptyset, (4, 4), p)$, $(\{p, q\}, (0, 0), r)$ and five unary transition rules (see Figure 1). We have $p(4, 4) \in \llbracket \mathcal{E} \rrbracket$ since $(\emptyset, (4, 4), p) \in \Delta$. From $(\{p\}, (-1, 0), q) \in \Delta$ and $p(4, 4) \in \llbracket \mathcal{E} \rrbracket$ we get that $q(3, 4) \in \llbracket \mathcal{E} \rrbracket$. From $(\{p, q\}, (0, 0), r) \in \Delta$ and $p(4, 4), q(3, 4) \in \llbracket \mathcal{E} \rrbracket$ we get that $r(7, 8) \in \llbracket \mathcal{E} \rrbracket$. ◀

A set of configurations $C \subseteq Q \times \mathbb{N}^d$ is said to be *semilinear* if C is a finite union of sets of the form $q(\mathbf{L})$ where $q \in Q$ and $\mathbf{L} \subseteq \mathbb{N}^d$ is linear, i.e. a set of the form $\mathbf{b} + \text{Per}(\mathbf{A})$ for some $\mathbf{b} \in \mathbb{N}^d$ and some finite subset \mathbf{A} of \mathbb{N}^d . A *presentation* of a semilinear set of configurations

¹ We use double braces $\{\{\dots\}\}$ to denote multisets. Here, the condition $S = \{r \mid r(\mathbf{z}) \in D\}$ means that, firstly, the set S is equal to the set $\{r \mid r(\mathbf{z}) \in D\}$, and, secondly, $r_1 \neq r_2$ for every two distinct configurations $r_1(\mathbf{z}_1), r_2(\mathbf{z}_2)$ in D .



■ **Figure 1** The 2-BVASS \mathcal{E} from Example 3.1 (left) and an execution of $\text{Explore}(\mathcal{E})$ (right).

$C \subseteq Q \times \mathbb{N}^d$ is a finite set $\{(q_1, \mathbf{b}_1, \mathbf{A}_1), \dots, (q_k, \mathbf{b}_k, \mathbf{A}_k)\}$, where $q_j \in Q$, $\mathbf{b}_j \in \mathbb{N}^d$, and \mathbf{A}_j is a finite subset of \mathbb{N}^d , such that $C = \bigcup_{j=1}^k q_j(\mathbf{b}_j + \text{Per}(\mathbf{A}_j))$. The main contribution of this paper is the generalization to 2-BVASS of the following theorem.

► **Theorem 3.2** ([16]). *For every 2-VASS \mathcal{V} , the reachability set $\llbracket \mathcal{V} \rrbracket$ of \mathcal{V} is semilinear and a presentation of $\llbracket \mathcal{V} \rrbracket$ is computable from \mathcal{V} .*

In order to generalize Theorem 3.2 to 2-BVASS, we will extend to BVASS some techniques developed for classical VASS. A linear (i.e., non-branching) view of BVASS behaviors is required to do so. Intuitively, this view is obtained by instantiating transition rules $\delta = (S, \mathbf{a}, q)$ with $|S| \geq 2$ into unary transition rules. This instantiation can be performed at the semantic level or at the syntactic level. Let us make these ideas more concrete. We assume that $\mathcal{B} = (Q, \Delta)$ is a d -BVASS for the remainder of this section.

For each transition rule $\delta = (S, \mathbf{a}, q)$ in Δ , we introduce the binary relation $\stackrel{\delta}{\Rightarrow}$ on $Q \times \mathbb{N}^d$ defined as the set of pairs $(p(\mathbf{x}), q(\mathbf{y})) \in (Q \times \mathbb{N}^d)^2$ such that $p \in S$ and there exists a set $D \subseteq \llbracket \mathcal{B} \rrbracket$ verifying $(S \setminus \{p\}) = \{r \mid r(\mathbf{z}) \in D\}$ and $\mathbf{y} = \mathbf{a} + \mathbf{x} + \sum_{r(\mathbf{z}) \in D} \mathbf{z}$. The binary relation $\stackrel{\delta}{\Rightarrow}$ instantiates the transition rule δ at the semantic level as it relies on the reachability set $\llbracket \mathcal{B} \rrbracket$ of \mathcal{B} . We also introduce the binary *step* relation \Rightarrow on $Q \times \mathbb{N}^d$ defined as the union $\bigcup_{\delta \in \Delta} \stackrel{\delta}{\Rightarrow}$. The reflexive-transitive closure of \Rightarrow is denoted by $\stackrel{*}{\Rightarrow}$. It is readily seen that \Rightarrow and $\stackrel{*}{\Rightarrow}$ are diagonal.² Using square brackets to denote relational images³, we have $\text{Post}_{\mathcal{B}}(C) \subseteq \Rightarrow[C] \subseteq \stackrel{*}{\Rightarrow}[C] \subseteq \llbracket \mathcal{B} \rrbracket$ for every set of configurations $C \subseteq \llbracket \mathcal{B} \rrbracket$.

Let us now instantiate transition rules at the syntactic level. Given a finite set $F \subseteq Q \times \mathbb{N}^d$, the *instantiation of \mathcal{B} with F* , written $\mathcal{B}\langle F \rangle$, is the d -VASS $\mathcal{B}\langle F \rangle = (Q, \Delta')$ where Δ' is the set of triples $(\{p\}, \mathbf{a}', q)$ such that there exist a transition rule $(S, \mathbf{a}, q) \in \Delta$ with $p \in S$ and a set $D \subseteq F$ verifying $(S \setminus \{p\}) = \{r \mid r(\mathbf{z}) \in D\}$ and $\mathbf{a}' = \mathbf{a} + \sum_{r(\mathbf{z}) \in D} \mathbf{z}$. We observe that if $F \subseteq \llbracket \mathcal{B} \rrbracket$ then the step relation of $\mathcal{B}\langle F \rangle$ is contained in the step relation of \mathcal{B} .

² A binary relation \bowtie on $Q \times \mathbb{N}^d$ is called *diagonal* if $p(\mathbf{x}) \bowtie q(\mathbf{y})$ implies $p(\mathbf{x} + \mathbf{u}) \bowtie q(\mathbf{y} + \mathbf{u})$, for every configurations $p(\mathbf{x}), q(\mathbf{y}) \in Q \times \mathbb{N}^d$ and vector $\mathbf{u} \in \mathbb{N}^d$.

³ Given a binary relation \bowtie on a set S and a subset X of S , we let $\bowtie[X]$ denote the *relational image* of X under \bowtie , defined by $\bowtie[X] = \{y \in S \mid \exists x \in X : x \bowtie y\}$.

► **Remark 3.3.** In the definition of the instantiation $\mathcal{B}\langle F \rangle$, we require F to be finite solely to ensure that Δ' is finite. We could drop this requirement and obtain an “infinite d -VASS”, meaning that its set of transition rules is potentially infinite. The step relations of \mathcal{B} and of the resulting “infinite d -VASS” $\mathcal{B}\langle \llbracket \mathcal{B} \rrbracket \rangle$ coincide.

We conclude this section with notions that are specific to classical VASS. Consider a d -VASS $\mathcal{V} = (Q, \Delta)$. From now on, unary transition rules $(\{p\}, \mathbf{a}, q) \in \Delta$ will be written (p, \mathbf{a}, q) for short. A *path* of \mathcal{V} is a non-empty sequence $\theta = (p_1, \mathbf{a}_1, q_1) \cdots (p_k, \mathbf{a}_k, q_k)$ of unary transition rules $(p_i, \mathbf{a}_i, q_i) \in \Delta$ such that $q_i = p_{i+1}$ for all $i \in \{1, \dots, k-1\}$. Such a path θ is also shortly written $\theta = p_1 \xrightarrow{\mathbf{a}_1} q_1 \cdots \xrightarrow{\mathbf{a}_k} q_k$. We call p_1 and q_k the *start* and the *end* of θ , respectively. The *displacement* of θ is $\sum_{i=1}^k \mathbf{a}_i$. We say that θ is a *cycle* if $p_1 = q_k$. It is an *elementary cycle* if $p_1 = q_k$ and p_1, \dots, p_k are pairwise distinct.

► **Fact 3.4.** Consider a d -BVASS $\mathcal{B} = (Q, \Delta)$ and finite set $F \subseteq \llbracket \mathcal{B} \rrbracket$. Let $q \in Q$, $\mathbf{x} \in \mathbb{N}^d$ and let θ be an elementary cycle of $\mathcal{B}\langle F \rangle$ with displacement \mathbf{v} and with start (and end) q . If $\mathbf{x} \geq (c, \dots, c)$ where $c = |Q| \max_{i \in \{1, \dots, d\}} \max_{(S, \mathbf{a}, q) \in \Delta} -\mathbf{a}(i)$ then $q(\mathbf{x}) \xrightarrow{*} q(\mathbf{x} + \mathbf{v})$.

4 Reachability Set Computation for 2-BVASS

We present in this section an algorithm to compute the reachability set for 2-BVASS. More precisely, given a 2-BVASS \mathcal{B} , our algorithm returns a finite exploration of \mathcal{B} that is both sound and complete. We start by defining what we mean by sound and complete exploration.

► **Definition 4.1.** An exploration of a 2-BVASS $\mathcal{B} = (Q, \Delta)$ is a node-labeled acyclic graph $\mathcal{G} = (N, \rightarrow, \lambda)$ such that

1. the edge relation \rightarrow is well-founded, i.e., there is no infinite sequence n_0, n_1, \dots of nodes in N such that $n_{i+1} \rightarrow n_i$ for all $i \in \mathbb{N}$, and
2. each node $n \in N$ is labeled with $\lambda(n) = (\mathbf{a}_n, q_n, \mathbf{z}_n, \mathbf{P}_n)$ where $\mathbf{a}_n \in \mathbb{Z}^2$, $q_n \in Q$, $\mathbf{z}_n \in \mathbb{N}^2$, and \mathbf{P}_n is a periodic subset of \mathbb{N}^2 .

Intuitively, the label $\lambda(n) = (\mathbf{a}_n, q_n, \mathbf{z}_n, \mathbf{P}_n)$ of a node n provides, firstly, the displacement \mathbf{a}_n of the transition rule used to create n (this will be made clear later on and can be ignored for now), and, secondly, the set of configurations $q_n(\mathbf{z}_n + \mathbf{P}_n)$ associated with the node n . Recall that $\llbracket \mathcal{B} \rrbracket$ denotes the reachability set of a 2-BVASS \mathcal{B} . Similarly, we associate to an exploration $\mathcal{G} = (N, \rightarrow, \lambda)$ of \mathcal{B} the set of configurations $\llbracket \mathcal{G} \rrbracket = \bigcup_{n \in N} q_n(\mathbf{z}_n + \mathbf{P}_n)$. We say that \mathcal{G} is *sound* when $\llbracket \mathcal{G} \rrbracket \subseteq \llbracket \mathcal{B} \rrbracket$ and that it is *complete* when $\llbracket \mathcal{G} \rrbracket \supseteq \llbracket \mathcal{B} \rrbracket$. A node $n \in N$ is called *redundant* if there exists $s \in N$ verifying $s \xrightarrow{+} n$ and $q_n(\mathbf{z}_n + \mathbf{P}_n) \subseteq q_s(\mathbf{z}_s + \mathbf{P}_s)$. We say that \mathcal{G} is *non-redundant* when every redundant node is a leaf.

► **Example 4.2.** The node-labeled acyclic graph depicted on the right-hand side of Figure 1 is an exploration of the 2-BVASS \mathcal{E} depicted on the left-hand side (see also Example 3.1). For instance, the set of configurations associated with the node n_4 is $\{q(2 + 2k, 4 + 5k) \mid k \in \mathbb{N}\}$. The first component \mathbf{a}_n of $\lambda(n)$ is omitted in the figure to reduce clutter. As mentioned above, the vectors \mathbf{a}_n can be ignored for now, see Example 5.2 for actual values. ┘

As in Hopcroft and Pansiot’s algorithm for classical 2-VASS [16], a crucial ingredient of our algorithm is the *acceleration* of cycles. The purpose of cycle acceleration is to make the periodic sets \mathbf{P}_n grow. We will utilize three kinds of cycles. Consider an exploration $\mathcal{G} = (N, \rightarrow, \lambda)$ of a 2-BVASS $\mathcal{B} = (Q, \Delta)$. We associate to each node $n \in N$ the 2-VASS \mathcal{V}_n defined as the instantiation $\mathcal{B}\langle F \rangle$ of \mathcal{B} with the finite set of configurations $F = \{q_s(\mathbf{z}_s) \mid s \in N, s \xrightarrow{+} n\}$. In particular, if n is a source then $\mathcal{V}_n = \mathcal{B}\langle \emptyset \rangle = (Q, \Delta')$ where $\Delta' = \{(S, \mathbf{a}, q) \in \Delta \mid |S| = 1\}$ is

the set of unary transition rules in Δ . We introduce three finite subsets of \mathbb{Z}^2 , namely \mathbf{E}_n , \mathbf{C}_n and $\overline{\mathbf{C}}_n$, that correspond to the three kinds of cycles mentioned above. Let us define $c = |Q| \max_{i \in \{1,2\}} \max_{(S,a,q) \in \Delta} -a(i)$. We call c the *constant of iteration* of \mathcal{B} . Observe that c is the same constant as the one in Fact 3.4.

- \mathbf{E}_n is the set of vectors $\mathbf{v} \in \mathbb{Z}^2$ such that there exist an elementary cycle θ of \mathcal{V}_n with displacement \mathbf{v} and a node $s \in \text{Anc}(n)$ verifying q_s is the start of θ , $\mathbf{z}_s \geq (c, c)$ and $s \neq n$ implies $\mathbf{v} \geq (0, 0)$.
- \mathbf{C}_n is the set of vectors $\mathbf{v} \in \mathbb{Z}^2$ such that there exists a node $s \in \text{Anc}(n)$ verifying $q_s = q_n$, $\mathbf{v} = \mathbf{z}_n - \mathbf{z}_s$, $\mathbf{z}_n \not\geq (c, c)$ and $\mathbf{z}_s \not\geq (c, c)$.
- $\overline{\mathbf{C}}_n$ is the set of vectors $\mathbf{v} \in \mathbb{Z}^2$ such that there exists a node $s \in \text{Anc}(n)$ verifying $q_s = q_n$ and $\mathbf{v} = \mathbf{z}_n - \mathbf{z}_s$.

Note that $\mathbf{C}_n \subseteq \overline{\mathbf{C}}_n$. We also introduce the finite set \mathbf{I}_n defined by $\mathbf{I}_n = \mathbf{E}_n \cup \overline{\mathbf{C}}_n$ if $\sum_{m \rightarrow n} \mathbf{P}_m = \{(0, 0)\}$ and $\mathbf{I}_n = \mathbf{E}_n \cup \mathbf{C}_n$ otherwise. Vectors in \mathbf{E}_n , \mathbf{C}_n , and \mathbf{I}_n are respectively called *n-elementary*, *n-consecutive* and *n-iterable*.

► **Example 4.3.** Let us continue Example 4.2. The constant of iteration of \mathcal{E} is $c = 4$. By definition, $\mathbf{I}_{n_i} = \mathbf{E}_{n_i} \cup \overline{\mathbf{C}}_{n_i}$ for $i \in \{0, 1, 2, 6\}$, and $\mathbf{I}_{n_i} = \mathbf{E}_{n_i} \cup \mathbf{C}_{n_i}$ for $i \in \{3, 4, 5\}$. We first discuss *n-consecutive* vectors. We have $\mathbf{C}_{n_0} = \overline{\mathbf{C}}_{n_0} = \{(0, 0)\}$ since n_0 has no ancestor except itself. The set \mathbf{C}_{n_1} is empty because the first component of $\mathbf{z}_{n_1} = (3, 4)$ is strictly below c , hence, $\mathbf{z}_{n_1} \not\geq (c, c)$. Similarly, $\mathbf{C}_{n_2} = \mathbf{C}_{n_3} = \emptyset$. It is readily seen that $\overline{\mathbf{C}}_{n_1} = \overline{\mathbf{C}}_{n_2} = \{(0, 0)\}$ and that $\overline{\mathbf{C}}_{n_3} = \{(0, 0), (-1, 0)\}$. We now discuss *n-elementary* vectors. The 2-VASS $\mathcal{V}_{n_0} = \mathcal{E}(\emptyset)$ contains exactly one elementary cycle (up to rotation), namely $\theta_0 = p \xrightarrow{(-1,0)} q \xrightarrow{(0,0)} s \xrightarrow{(0,0)} p$. It follows that $\mathbf{E}_{n_0} = \{(-1, 0)\}$. In addition to the cycle θ_0 , the 2-VASS $\mathcal{V}_{n_1} = \mathcal{E}(\{p(4, 4)\})$ also contains the elementary cycle $\theta_1 = q \xrightarrow{(4,4)} r \xrightarrow{(-1,1)} q$. Still, the set \mathbf{E}_{n_1} is empty, because none of these two cycles contributes to \mathbf{E}_{n_1} . Indeed, even though the elementary cycle θ_0 contains the states q_{n_0} and q_{n_1} , its displacement $\mathbf{v}_0 = (-1, 0)$ is not in \mathbf{E}_{n_1} because $\mathbf{z}_{n_1} \not\geq (c, c)$ and $\mathbf{v}_0 \not\geq (0, 0)$. Analogously, the displacement of θ_1 is not in \mathbf{E}_{n_1} because $\mathbf{z}_{n_1} \not\geq (c, c)$. The elementary cycles of \mathcal{V}_{n_2} are θ_0 , θ_1 and θ_2 , where $\theta_2 = p \xrightarrow{(3,4)} r \xrightarrow{(-1,1)} q \xrightarrow{(0,0)} s \xrightarrow{(0,0)} p$. The displacement $\mathbf{v}_2 = (2, 5)$ of θ_2 is in \mathbf{E}_{n_2} since θ_2 contains the state q_{n_0} , $\mathbf{z}_{n_0} \geq (c, c)$ and $\mathbf{v}_2 \geq (0, 0)$. We get that $\mathbf{E}_{n_2} = \{(2, 5)\}$. Last, we observe that $\mathcal{V}_{n_2} = \mathcal{V}_{n_3}$ since the state q_{n_2} is not part of any branching transition. So \mathcal{V}_{n_3} has the same elementary cycles as \mathcal{V}_{n_2} , and we get that $\mathbf{E}_{n_3} = \mathbf{E}_{n_2} = \{(2, 5)\}$. Indeed, even though the elementary cycle θ_0 contains the state q_{n_3} , its displacement $\mathbf{v}_0 = (-1, 0)$ is not in \mathbf{E}_{n_3} because $\mathbf{z}_{n_3} \not\geq (c, c)$ and $\mathbf{v}_0 \not\geq (0, 0)$. ◀

Given a set $\mathbf{I} \subseteq \mathbb{Z}^2$ and a periodic set $\mathbf{P} \subseteq \mathbb{N}^2$, we introduce the periodic set $\text{Add}_{\mathbf{I}}^*(\mathbf{P})$ defined as the set of vectors $\mathbf{p} + \mathbf{v}_1 + \dots + \mathbf{v}_k$ where $\mathbf{p} \in \mathbf{P}$, $k \in \mathbb{N}$, and $\mathbf{v}_1, \dots, \mathbf{v}_k$ are vectors in \mathbf{I} such that $\mathbf{p} + \mathbf{v}_1 + \dots + \mathbf{v}_\ell \geq (0, 0)$ for every $\ell \in \{1, \dots, k\}$.

► **Lemma 4.4.** *For every finite sets $\mathbf{G} \subseteq \mathbb{N}^2$ and $\mathbf{I} \subseteq \mathbb{Z}^2$, we can effectively compute a finite set $\mathbf{H} \subseteq \mathbb{N}^2$ such that $\text{Add}_{\mathbf{I}}^*(\text{Per}(\mathbf{G})) = \text{Per}(\mathbf{H})$.*

Proof sketch. First of all, notice that Theorem 3.2 is not sufficient for proving that result since there exist semilinear periodic sets, like $\{(0, 0)\} \cup ((1, 1) + \mathbb{N}^2)$ that are not finitely-generated. From Lemma 2.1, we deduce that $\text{Add}_{\mathbf{I}}^*(\text{Per}(\mathbf{G}))$ is finitely-generated periodic set if $\text{Con}(\text{Add}_{\mathbf{I}}^*(\text{Per}(\mathbf{G})))$ is a finitely-generated cone. We observe that this cone is spanned by $\mathbf{G} \cup (\mathbf{I} \cap \mathbb{N}^2) \cup \mathbf{U}$ where \mathbf{U} is a set of axis, i.e. a subset of $\{(1, 0), (0, 1)\}$. It follows that there exists a finite set $\mathbf{H} \subseteq \mathbb{N}^2$ such that $\text{Add}_{\mathbf{I}}^*(\text{Per}(\mathbf{G})) = \text{Per}(\mathbf{H})$. Finally, with a step-by-step algorithm computing increasing finite subsets of $\text{Add}_{\mathbf{I}}^*(\text{Per}(\mathbf{G}))$ we eventually reach a set \mathbf{H} satisfying the lemma. ◀

Algorithm 1 $\text{Explore}(\mathcal{B})$.

Input: A 2-BVASS $\mathcal{B} = (Q, \Delta)$.
Output: A sound and complete finite exploration of \mathcal{B} .

```

1  $(N, \rightarrow, \lambda, R, W) := (\emptyset, \emptyset, \emptyset, \emptyset, \emptyset)$ 
2 foreach  $(S, \mathbf{a}, q) \in \Delta$  with  $S = \emptyset$  and  $\mathbf{a} \geq (0, 0)$  do
3   create a new node  $n$  (with  $n \notin N$ )
4    $N := N \cup \{n\}$ 
5    $\lambda(n) := (\mathbf{a}, q, \mathbf{a}, \{(0, 0)\})$ 
6    $W := W \cup \{n\}$ 
7 while  $W \neq \emptyset$  do
8   let  $n$  be a node in  $W$ 
9    $W := W \setminus \{n\}$ 
10   $\mathbf{P}_n := \text{Add}_{\mathbf{I}_n}^*(\mathbf{P}_n)$ 
11  if there exists  $s \in N$  verifying  $s \xrightarrow{+} n$  and  $q_n(\mathbf{z}_n + \mathbf{P}_n) \subseteq q_s(\mathbf{z}_s + \mathbf{P}_s)$  then
12     $R := R \cup \{n\}$ 
13  else
14    foreach  $M \subseteq N$  with  $n \in M$  and  $M \cap (R \cup W) = \emptyset$  do
15      foreach  $(S, \mathbf{a}, q) \in \Delta$  with  $S = \{\{q_m \mid m \in M\}\}$  do
16         $(\mathbf{z}', \mathbf{P}') := (\sum_{m \in M} \mathbf{z}_m, \sum_{m \in M} \mathbf{P}_m)$ 
17        let  $\mathbf{B}$  be a finite subset of  $\mathbb{N}^2$  such that  $(\mathbf{B} + \mathbf{P}') = (\mathbf{a} + \mathbf{z}' + \mathbf{P}') \cap \mathbb{N}^2$ 
18        foreach  $\mathbf{b} \in \mathbf{B}$  do
19          create a new node  $n'$  (with  $n' \notin N$ )
20           $(N, \rightarrow) := (N \cup \{n'\}, \rightarrow \cup \{(m, n') \mid m \in M\})$ 
21           $\lambda(n') := (\mathbf{a}, q, \mathbf{b}, \mathbf{P}')$ 
22           $W := W \cup \{n'\}$ 
23 return  $(N, \rightarrow, \lambda)$ 

```

Our algorithm, dubbed **Explore**, is defined in Algorithm 1. We use an abstract pseudocode to simplify the presentation. This raises implementability issues that are addressed in Remark 4.5. Given a 2-BVASS $\mathcal{B} = (Q, \Delta)$ as input, $\text{Explore}(\mathcal{B})$ iteratively computes an exploration of \mathcal{B} and then returns it. This exploration is maintained in the variables N , \rightarrow and λ , and is initially empty (see line 1). As in Definition 4.1, we let q_n , \mathbf{z}_n and \mathbf{P}_n denote the second, third and fourth components of $\lambda(n)$. The set of redundant nodes of the exploration is tracked in the variable R . The set of unprocessed nodes, called the *worklist*, is maintained in the variable W . Both variables R and W remain disjoint subsets of N during the execution of the algorithm. For each initial configuration $q(\mathbf{a})$ of \mathcal{B} , a new node is created and put in the worklist (see lines 2–6). After this initialization phase, $\text{Explore}(\mathcal{B})$ repeatedly selects a node from the worklist and processes it, as long as the worklist is non-empty (see lines 7–22). The processing of a node n consists in four steps. First, the node n is removed from the worklist (so n is considered processed afterwards). Second, the periodic set \mathbf{P}_n is enlarged using the set \mathbf{I}_n of n -iterable vectors (see line 10). This is the cycle *acceleration* step. Note that the set \mathbf{I}_n is finite and implicitly depends on the constant of iteration of \mathcal{B} and on the current exploration $(N, \rightarrow, \lambda)$ of \mathcal{B} . The assignment at line 10 actually means that the label $\lambda(n)$ of the node n is modified (in fact, only the fourth component of the label is modified). Third, the algorithm tests whether the node n is “covered” by one of its ancestors

(see line 11). If that is the case then n is added to the set R of redundant nodes and the processing of n stops. Otherwise, as a fourth step, the node n is expanded (see lines 14–22), meaning that for each set of processed and non-redundant nodes M containing n and for each transition rule (S, \mathbf{a}, q) that applies to M , finitely many children of n are created and put in the worklist. At first glance, one might want to create a single child labeled with $(\mathbf{a}, q, \mathbf{a} + \mathbf{z}', \mathbf{P}')$. This would be fine if $\mathbf{a} + \mathbf{z}' \geq (0, 0)$, however that is not the case in general. This is the reason why we need the finite set \mathbf{B} at line 17 and the **foreach**-loop at lines 18–22. The existence of such a finite set is explained in Remark 4.5. The observant reader will notice that the modification of the variables N and W at lines 20 and 22 has no impact on the **foreach**-loop iteration at line 14, since $N \setminus W$ remains constant. When the worklist becomes empty, the constructed exploration is returned at line 23. The reachability set of \mathcal{B} is then easily obtained from this exploration, provided that it is sound and complete.

► **Remark 4.5.** The abstract pseudocode used in Algorithm 1 raises some implementability issues. The first issue is that we liberally use periodic sets in the pseudocode, but these periodic sets need to admit a finite and computable representation. To address this issue, we observe that each periodic set defined in Algorithm 1 is finitely-generated and admits a computable finite spanning set. The only line where this property is non-trivial is line 10, but Lemma 4.4 provides the result. The second issue is the existence and computation of the finite set \mathbf{B} at line 17. To address this issue, we recall the following well-known fact (see for instance [16, Lemma 1.1]). Given a vector $\mathbf{v} \in \mathbb{Z}^d$ and a finite subset $\mathbf{A} = \{\mathbf{a}_1, \dots, \mathbf{a}_k\}$ of \mathbb{N}^d , the set $(\mathbf{v} + \text{Per}(\mathbf{A})) \cap \mathbb{N}^d$ is equal to $(\mathbf{B} + \text{Per}(\mathbf{A}))$ where $\mathbf{B} \subseteq \mathbb{N}^d$ is the finite set of vectors $\mathbf{v} + \alpha_1 \mathbf{a}_1 + \dots + \alpha_k \mathbf{a}_k$ such that $(\alpha_1, \dots, \alpha_k)$ is a minimal vector in \mathbb{N}^k satisfying $\mathbf{v} + \alpha_1 \mathbf{a}_1 + \dots + \alpha_k \mathbf{a}_k \geq \mathbf{0}$.

► **Example 4.6.** To illustrate Algorithm 1, we apply it on the 2-BVASS \mathcal{E} from Example 3.1. The resulting exploration is depicted in Figure 1. There is only one initial configuration, so the exploration has only one source n_0 . The nodes are added to (+) and removed from (−) the worklist in the order $+n_0, -n_0, +n_1, -n_1, +n_2, +n_6, -n_2, +n_3, -n_3, +n_4, +n_5$. So the nodes n_4, n_5 and n_6 are still in the worklist. The labels q_n and \mathbf{z}_n are straightforward. For the periodic sets \mathbf{P}_n , we first recall from Example 4.3 that $\mathbf{I}_{n_0} = \{(0, 0), (-1, 0)\}$, $\mathbf{I}_{n_1} = \{(0, 0)\}$, $\mathbf{I}_{n_2} = \{(0, 0), (2, 5)\}$ and $\mathbf{I}_{n_3} = \{(2, 5)\}$. The periodic set \mathbf{P}_{n_0} is $\text{Add}_{\mathbf{I}_{n_0}}^*(\{(0, 0)\}) = \{(0, 0)\}$. The node n_1 also has $\{(0, 0)\}$ as periodic set since $\mathbf{I}_{n_1} = \{(0, 0)\}$. The periodic set \mathbf{P}_{n_2} is $\text{Add}_{\mathbf{I}_{n_2}}^*(\{(0, 0)\}) = \text{Per}(\{(2, 5)\})$. Similarly, $\mathbf{P}_{n_3} = \text{Add}_{\mathbf{I}_{n_3}}^*(\text{Per}(\{(2, 5)\})) = \text{Per}(\{(2, 5)\})$. ◻

5 Soundness and Completeness of Algorithmic Explorations

This section is devoted to the partial correctness of our algorithm **Explore** (see Algorithm 1). Its termination is much more involved and will be the subject of the next sections. We first refine the notion of exploration to account for the behavior of our algorithm.

► **Definition 5.1.** An exploration $\mathcal{G} = (N, \rightarrow, \lambda)$ of a 2-BVASS $\mathcal{B} = (Q, \Delta)$ is algorithmic if it satisfies, for every node $n \in N$, the three following conditions:

1. the multiset $S = \{\{q_m \mid m \rightarrow n\}\}$ is a set and verifies $(S, \mathbf{a}_n, q_n) \in \Delta$,
2. the vector \mathbf{z}_n is in $\mathbf{a}_n + \sum_{m \rightarrow n} (\mathbf{z}_m + \mathbf{P}_m)$, and
3. the periodic set \mathbf{P}_n verifies $\mathbf{P}_n = \text{Add}_{\mathbf{I}_n}^*(\sum_{m \rightarrow n} \mathbf{P}_m)$.

Intuitively, Conditions 1 and 2 ensure that the exploration conforms to the semantics of 2-BVASS. As hinted before, the vector \mathbf{a}_n is the displacement of the transition rule leading to n (see Condition 1). Notice these two conditions entail in particular that $q_n(\mathbf{z}_n)$ is an initial configuration for every source n . Condition 3 corresponds to the previously-mentioned cycle acceleration step (see line 10 of Algorithm 1).

► **Example 5.2.** Let us get back to the exploration of Example 4.2 and give the actual values of the vectors \mathbf{a}_n . We take $\mathbf{a}_{n_0} = (4, 4)$, $\mathbf{a}_{n_1} = \mathbf{a}_{n_4} = (-1, 0)$ and $\mathbf{a}_{n_2} = \mathbf{a}_{n_3} = \mathbf{a}_{n_5} = \mathbf{a}_{n_6} = (0, 0)$. The restriction to $\{n_0, n_1, n_2, n_3\}$ of the resulting exploration is algorithmic. \lrcorner

► **Remark 5.3.** For every algorithmic exploration $\mathcal{G} = (N, \rightarrow, \lambda)$ and every node $n \in N$, the set $\text{Anc}(n)$ is finite. Indeed, by Condition 1, every node $n \in N$ has finite in-degree (i.e., the set of nodes $m \in N$ such that $m \rightarrow n$ is finite). As \rightarrow is well-founded by Condition 1 of Definition 4.1, it follows from König's Lemma that $\text{Anc}(n)$ is finite for every $n \in N$.

In the rest of this section, we show, firstly, that every algorithmic exploration is sound, and secondly, that our algorithm constructs explorations that are algorithmic and complete.

5.1 Soundness of algorithmic explorations

The main difficulty to establish the partial correctness of our algorithm **Explore** comes from the cycle acceleration step (see line 10 of Algorithm 1), which translates to Condition 3 of Definition 5.1. Contrary to usual cycle acceleration techniques, our cycle acceleration step is “retroactive” since the set \mathbf{I}_n used to accelerate a node n accounts for elementary cycles of \mathcal{V}_n that apply to an ancestor $s \stackrel{+}{\rightarrow} n$. Thus, to show that a given algorithmic exploration \mathcal{G} is sound, we transform \mathcal{G} into an alternative exploration \mathcal{H} whose periodic sets \mathbf{Q}_n are closed in the sense that each \mathbf{Q}_n already accounts for all potential cycles applicable to n , and we show that \mathcal{H} is sound. Let us make these ideas more precise.

Consider a 2-BVASS $\mathcal{B} = (Q, \Delta)$. For every configuration $q(\mathbf{x})$ of \mathcal{B} , we define the function $\text{Clo}_{q, \mathbf{x}} : \mathbb{P}(\mathbb{N}^2) \rightarrow \mathbb{P}(\mathbb{N}^2)$ by $\text{Clo}_{q, \mathbf{x}}(\mathbf{P}) = \{\mathbf{w} \in \mathbb{N}^2 \mid \exists \mathbf{u} \in \mathbf{P} : q(\mathbf{x} + \mathbf{u}) \stackrel{*}{\Rightarrow} q(\mathbf{x} + \mathbf{w})\}$. Observe that $\text{Clo}_{q, \mathbf{x}}$ is an upper closure operator⁴ on the partially-ordered set $(\mathbb{P}(\mathbb{N}^2), \subseteq)$. In fact, for every subset $\mathbf{P} \subseteq \mathbb{N}^2$, the set $\text{Clo}_{q, \mathbf{x}}(\mathbf{P})$ is the \subseteq -greatest subset $\mathbf{Q} \subseteq \mathbb{N}^2$ such that $q(\mathbf{x} + \mathbf{Q}) \subseteq \stackrel{*}{\Rightarrow} [q(\mathbf{x} + \mathbf{P})]$. Note also that $\text{Clo}_{q, \mathbf{x}}$ preserves periodicity, meaning that $\text{Clo}_{q, \mathbf{x}}(\mathbf{P})$ is periodic for every periodic subset $\mathbf{P} \subseteq \mathbb{N}^2$. The following technical lemma will allow us to relate $\text{Add}_{\mathbf{I}_n}^*$ and $\text{Clo}_{q_n, \mathbf{z}_n}$.

► **Lemma 5.4.** *For every configuration $q(\mathbf{x})$ of \mathcal{B} and periodic subset \mathbf{P} of \mathbb{N}^2 , it holds that $\text{Add}_{\mathbf{J}}^*(\mathbf{P}) = \text{Clo}_{q, \mathbf{x}}(\mathbf{P})$ where \mathbf{J} is the set of vectors $\mathbf{v} \in \mathbb{Z}^2$ such that there exists $\mathbf{y} \in \mathbb{N}^2$ and $\mathbf{u} \in \mathbf{P}$ verifying $(\mathbf{v} = \mathbf{y} - \mathbf{x} \text{ and } q(\mathbf{x} + \mathbf{u}) \stackrel{*}{\Rightarrow} q(\mathbf{y}))$ or $(\mathbf{v} = \mathbf{x} - \mathbf{y} \text{ and } q(\mathbf{y} + \mathbf{u}) \stackrel{*}{\Rightarrow} q(\mathbf{x}))$.*

► **Lemma 5.5.** *Every algorithmic exploration of a 2-BVASS is sound.*

Proof sketch. Let $\mathcal{G} = (N, \rightarrow, \lambda)$ be an algorithmic exploration of 2-BVASS \mathcal{B} . We introduce the family $(\mathbf{Q}_n)_{n \in N}$ of subsets of \mathbb{N}^2 defined, by well-founded recursion over \rightarrow , by $\mathbf{Q}_n = \text{Clo}_{q_n, \mathbf{z}_n}(\sum_{m \rightarrow n} \mathbf{Q}_m)$ for every node $n \in N$. Observe that each \mathbf{Q}_n is periodic and that $m \rightarrow n$ implies $\mathbf{Q}_m \subseteq \mathbf{Q}_n$. We show by well-founded induction over \rightarrow that, for all $n \in N$, we have $\mathbf{P}_n \subseteq \mathbf{Q}_n$ and $q_n(\mathbf{z}_n + \mathbf{Q}_n) \subseteq \llbracket \mathcal{B} \rrbracket$. Let $n \in N$ and assume that $\mathbf{P}_s \subseteq \mathbf{Q}_s$ and $q_s(\mathbf{z}_s + \mathbf{Q}_s) \subseteq \llbracket \mathcal{B} \rrbracket$ for all $s \in N$ with $s \stackrel{+}{\rightarrow} n$. We derive from Conditions 1 and 2 of Definition 5.1 that $q_n(\mathbf{z}_n + \sum_{m \rightarrow n} \mathbf{Q}_m) \subseteq \llbracket \mathcal{B} \rrbracket$. It follows that $q_n(\mathbf{z}_n + \mathbf{Q}_n) \subseteq \llbracket \mathcal{B} \rrbracket$. It remains to show that $\mathbf{P}_n \subseteq \mathbf{Q}_n$. Let \mathbf{J}_n denote the set \mathbf{J} defined in Lemma 5.4 with $q(\mathbf{x}) := q_n(\mathbf{z}_n)$ and $\mathbf{P} := \mathbf{Q}_n$. The crucial observation now is that \mathbf{E}_n and $\overline{\mathbf{C}}_n$ are both contained in \mathbf{J}_n , hence, $\mathbf{I}_n \subseteq \mathbf{J}_n$. We obtain from Lemma 5.4 that $\text{Add}_{\mathbf{I}_n}^*(\mathbf{Q}_n) \subseteq \text{Clo}_{q_n, \mathbf{z}_n}(\mathbf{Q}_n) = \mathbf{Q}_n$ since $\text{Clo}_{q_n, \mathbf{z}_n}$ is idempotent. Recall that $\mathbf{P}_m \subseteq \mathbf{Q}_m \subseteq \mathbf{Q}_n$ for every $m \in N$ with $m \rightarrow n$. We derive from Condition 3 of Definition 5.1 that $\mathbf{P}_n = \text{Add}_{\mathbf{I}_n}^*(\sum_{m \rightarrow n} \mathbf{P}_m) \subseteq \text{Add}_{\mathbf{I}_n}^*(\mathbf{Q}_n) \subseteq \mathbf{Q}_n$. \blacktriangleleft

⁴ An *upper closure operator* on a partially-ordered set (S, \leq) is any function $f : S \rightarrow S$ that is \leq -nondecreasing ($x \leq y$ implies $f(x) \leq f(y)$), extensive ($x \leq f(x)$), and idempotent ($f \circ f = f$).

5.2 Partial correctness of `Explore`

Consider an execution σ of `Explore`(\mathcal{B}), where \mathcal{B} is a 2-BVASS. We let $(N^j, \rightarrow^j, \lambda^j, R^j, W^j)_{j \in J}$ denote the successive values of the variables N , \rightarrow , λ , R and W at line 7 of Algorithm 1, just before the evaluation of the **while**-loop condition. The index set J is \mathbb{N} if the execution σ does not terminate (i.e., $W^j \neq \emptyset$ for all $j \in \mathbb{N}$). Otherwise, $J = \{0, \dots, \kappa\}$ where $\kappa \in \mathbb{N}$ is the number of iterations of the **while**-loop (i.e., $W^\kappa = \emptyset$ and $W^j \neq \emptyset$ for all $j \in \{0, \dots, \kappa - 1\}$). It is understood that, in both cases, the values of the variables after the **foreach**-loop at lines 2–6 are $N^0, \rightarrow^0, \lambda^0, R^0$ and W^0 . Moreover, if σ terminates then it returns $(N^\kappa, \rightarrow^\kappa, \lambda^\kappa)$ at line 23. For every $j \in J$, we let \mathcal{G}^j denote the labeled graph $(N^j, \rightarrow^j, \lambda^j)$, and we let $\hat{\mathcal{G}}^j$ denote the restriction of \mathcal{G}^j to the set of processed nodes $\hat{N}^j = N^j \setminus W^j$. The following lemma is easily derived from Algorithm 1 (the detailed proof is tedious but straightforward).

► **Lemma 5.6.** *Both \mathcal{G}^j and $\hat{\mathcal{G}}^j$ are non-redundant explorations of \mathcal{B} , for every $j \in J$. Moreover, $\hat{\mathcal{G}}^j$ is algorithmic and it holds that $\text{Post}_{\mathcal{B}}(\llbracket \hat{\mathcal{G}}^j \rrbracket) \subseteq \llbracket \mathcal{G}^j \rrbracket$.*

Notice that $N^j \subseteq N^{j+1}$, $\hat{N}^j \subseteq \hat{N}^{j+1}$ and $\rightarrow^j \subseteq \rightarrow^{j+1}$, for every $j \in J$ with $(j+1) \in J$. We introduce the “limit” values N^σ , \hat{N}^σ and \rightarrow^σ of the corresponding sequences, defined naturally by $N^\sigma = \bigcup_{j \in J} N^j$, $\hat{N}^\sigma = \bigcup_{j \in J} \hat{N}^j$ and $\rightarrow^\sigma = \bigcup_{j \in J} \rightarrow^j$. We also define λ^σ as the function with domain N^σ that maps each $n \in N^\sigma$ to the ultimate value of the sequence $(\lambda^j(n))_{j \in J, n \in N^j}$. The latter sequence is non-empty and ultimately constant. We let \mathcal{G}^σ denote the labeled graph $(N^\sigma, \rightarrow^\sigma, \lambda^\sigma)$, and we let $\hat{\mathcal{G}}^\sigma$ denote the restriction of \mathcal{G}^σ to \hat{N}^σ .

► **Lemma 5.7.** *For every 2-BVASS \mathcal{B} and every execution σ of `Explore`(\mathcal{B}), $\hat{\mathcal{G}}^\sigma$ is a non-redundant and algorithmic exploration of \mathcal{B} . If σ terminates then $\hat{\mathcal{G}}^\sigma$ is finite and complete, and σ returns $\hat{\mathcal{G}}^\sigma$. Otherwise, $\hat{\mathcal{G}}^\sigma$ is infinite.*

Proof sketch. Let σ be an execution of `Explore`(\mathcal{B}), where \mathcal{B} is a 2-BVASS. It is routinely checked that, for every $j \in J$,

1. for every $m, n \in N^\sigma$, if $n \in N^j$ and $m \rightarrow^\sigma n$ then $m \in N^j$ and $m \rightarrow^j n$, and
2. the restriction of $\hat{\mathcal{G}}^\sigma$ to \hat{N}^j coincides with $\hat{\mathcal{G}}^j$.

The first observation entails that the edge relation \rightarrow^σ of \mathcal{G}^σ is well-founded, hence, both \mathcal{G}^σ and $\hat{\mathcal{G}}^\sigma$ are explorations of \mathcal{B} . The second observation, combined with Lemma 5.6, entails that $\hat{\mathcal{G}}^\sigma$ is a non-redundant and algorithmic exploration of \mathcal{B} . If σ terminates then it returns \mathcal{G}^κ . Moreover, we have $\mathcal{G}^\kappa = \hat{\mathcal{G}}^\kappa = \hat{\mathcal{G}}^\sigma$ in that case, hence, $\hat{\mathcal{G}}^\sigma$ is finite and complete by Lemma 5.6. If σ does not terminate then \hat{N}^σ is infinite since $\hat{N}^j \subset \hat{N}^{j+1}$ for all $j \in \mathbb{N}$. ◀

Partial correctness of our algorithm `Explore` follows from Lemma 5.7. To prove termination, we will show in the next sections that every non-redundant and algorithmic exploration of a 2-BVASS is finite, under an additional technical condition called spannability and discussed in next section. Termination of our algorithm will then be ensured by Lemma 5.7.

6 Core Witnesses of Infinity for Spannable Graphs

A graph (N, \rightarrow) is called a *forest* if the edge relation \rightarrow is well-founded (hence acyclic) and the set $\{m \in N \mid m \rightarrow n\}$ contains at most one node for every node $n \in N$. A forest is said to be *finitely-branching* if the set of source nodes is finite and the set $\{n \in N \mid m \rightarrow n\}$ is finite for every node $m \in N$. We say that a graph $\mathcal{G} = (N, \rightarrow)$ is *spannable* if there exists a subrelation $\rightarrow_{\mathcal{F}} \rightarrow$ such that $\mathcal{F} = (N, \rightarrow_{\mathcal{F}})$ is a finitely-branching forest. In that case, \mathcal{F} is called a *spanning forest* of \mathcal{G} . Explorations $\hat{\mathcal{G}}^\sigma$ built by algorithm `Explore`(\mathcal{B}) are spannable.

This property is obtained by observing that along the execution of the algorithm, new created nodes are connected, thanks to an edge that we call special, to a node that is removed from the worklist at that step. The spanning forest is then obtained by restricting the exploration to the special edges.

A *branch* β of a forest $(N, \rightarrow_{\mathcal{F}})$ is an infinite sequence $\beta = (\beta_n)_{n \in \mathbb{N}}$ of nodes such that β_0 is a source node of \mathcal{F} and such that $\beta_i \rightarrow_{\mathcal{F}} \beta_{i+1}$ for every $i \geq 0$. Thanks to the Koenig's lemma, we know that any infinite finitely-branching forest admits a branch. Such a branch can be seen as a witness of infinity of \mathcal{F} . We extend this notion of witnesses to infinite spannable graphs in a non-trivial way. Naturally, denoting by \mathcal{F} a spanning forest of \mathcal{G} , any branch of \mathcal{F} is a kind of witness of infinity of \mathcal{G} . However such a witness depends somehow on the choice of \mathcal{F} and does not take into account the structure of \mathcal{G} .

Our “core” witness of infinity of an infinite spannable graph is defined thanks to the notion of *primary* and *directed* graphs. A graph $\mathcal{G} = (N, \rightarrow)$ is said to be *primary* if $N \setminus \text{Des}(n)$ is finite for every node $n \in N$. A graph $\mathcal{G} = (N, \rightarrow)$ is said to be *directed* if for every $n, m \in N$, there exists $s \in N$ such that $n \xrightarrow{*} s$ and $m \xrightarrow{*} s$. The following lemma will be useful to extract from an infinite spannable graph a “core” subset of nodes that explain its infinity.

► **Lemma 6.1.** *For every infinite spannable graph $\mathcal{G} = (N, \rightarrow)$, there exists an infinite ancestor-closed set $X \subseteq N$ such that the restriction of \mathcal{G} to X is primary and directed.*

Proof sketch. Let \mathcal{F} be a spanning forest of \mathcal{G} . We associate to each branch $\beta = (\beta_n)_{n \in \mathbb{N}}$ of \mathcal{F} , the set $\text{Anc}(\beta) = \bigcup_{n \in \mathbb{N}} \text{Anc}(\beta_n)$ of ancestors of β with respect to the graph \mathcal{G} . We introduce the preorder (reflexive and transitive) \sqsubseteq over the branches defined by $\alpha \sqsubseteq \beta$ if $\text{Anc}(\alpha) \subseteq \text{Anc}(\beta)$. A branch β is said to be *minimal* (for the relation \sqsubseteq) if for every branch α such that $\alpha \sqsubseteq \beta$, we have $\beta \sqsubseteq \alpha$. Notice that a branch β is minimal for \sqsubseteq if, and only if, $\text{Anc}(\beta)$ is minimal for the inclusion relation. We prove that a minimal branch exists by contradiction. Intuitively, if there does not exist a minimal branch then any branch admits a strictly smaller one (we can even select an eagerly smaller one). Since the number of sources of \mathcal{F} is finite, and the set $\{n \in N \mid m \rightarrow n\}$ is finite for every node $m \in N$, we can extract from this infinite sequence of branches, a subsequence that “converges” to another branch. We prove that this branch is necessarily minimal providing a contradiction. It follows that there exists a minimal branch β . Then, we show that $X = \text{Anc}(\beta)$ satisfies the lemma. ◀

► **Remark 6.2.** If (N, \rightarrow) is a primary graph then $\xrightarrow{*}$ is a well-quasi-order (wqo). In fact, let us consider an infinite sequence $(n_i)_{i \in \mathbb{N}}$ of nodes $n_i \in N$. Since \mathcal{G} is primary, the set $N_0 = N \setminus \text{Des}(n_0)$ is finite. If $n_i \in N_0$ for every $i \geq 1$ then there exists $i < j$ such that $n_i = n_j$ and in particular $n_i \xrightarrow{*} n_j$. Otherwise there exists $j \geq 1$ such that $n_j \notin N_0$ and in that case $n_0 \xrightarrow{*} n_j$. So, in any case, we have proved that there exists $i < j$ such that $n_i \xrightarrow{*} n_j$. Therefore $\xrightarrow{*}$ is a wqo.

7 Termination

The termination of Algorithm 1 is obtained by contradiction. We assume that the algorithm is not terminating and from an infinite execution we derive an infinite exploration. Such an exploration is spannable, algorithmic and infinite. Thanks to Lemma 6.1 we can extract a sub-exploration $(N, \rightarrow, \lambda)$ that is also directed and primary. We prove that the sequence of periodic sets \mathbf{P}_n for this exploration stabilizes. From Lemma 2.1 it is sufficient to prove that the sequence of cones $\text{Con}(\mathbf{P}_n)$ eventually stabilizes. Since the exploration is directed, it is sufficient to prove that the cone spanned by $\mathbf{P}_N = \bigcup_{n \in N} \mathbf{P}_n$ is finitely-generated.

This result is obtained by interpreting geometrically the acceleration step $\mathbf{P}_n := \text{Add}_{\mathbf{I}_n}^*(\mathbf{P}_n)$ performed at line 10, as the *so-called* \mathbf{v} -stabilization of $\text{Con}(\mathbf{P}_n)$ by vectors $\mathbf{v} \in \mathbf{I}_n$. More formally, the \mathbf{v} -stabilization of a cone $\mathbf{C} \subseteq \mathbb{Q}_{\geq 0}^2$ for some vector $\mathbf{v} \in \mathbb{Z}^2$ is the cone $(\mathbf{C} + \mathbb{Q}_{\geq 0}\mathbf{v}) \cap \mathbb{Q}_{\geq 0}^2$. A cone $\mathbf{C} \subseteq \mathbb{Q}_{\geq 0}^2$ is said to be \mathbf{v} -stable when it is equal to its \mathbf{v} -stabilization.

► **Example 7.1.** The cone $\mathbf{C} = \text{Con}(\{(1, 2), (2, 1)\})$ is $(-1, -1)$ -stable. It is not $(-3, -1)$ -stable since its $(-3, -1)$ -stabilization is $\mathbf{C} + \mathbb{Q}_{\geq 0}(0, 1)$. ◀

The \mathbf{v} -stabilization of a cone $\mathbf{C} \subseteq \mathbb{Q}_{\geq 0}^2$ spanned by a finite set of vectors $\mathbf{G} \subseteq \mathbb{N}^2$ is the cone spanned by $\mathbf{G} \cup \{\mathbf{v}\}$ when $\mathbf{v} \geq (0, 0)$, and the cone spanned by $\mathbf{G} \cup \mathbf{U}$ where \mathbf{U} is a subset of the set of *axis* $\{(1, 0), (0, 1)\}$ when $\mathbf{v} \not\geq (0, 0)$. Since at some step of the algorithm, no new axis are added, it follows that the cone $\text{Con}(\mathbf{P}_n)$ is \mathbf{v} -stable for every vector $\mathbf{v} \in \mathbf{I}_n \setminus \mathbb{Q}_{\geq 0}^2$. In fact a stronger stabilization property occurs: there exists a node n_0 such that for every $n, m \in \text{Des}(n_0)$, the cone $\text{Con}(\mathbf{P}_n)$ is \mathbf{v} -stable for every $\mathbf{v} \in \mathbf{I}_m \setminus \mathbb{Q}_{\geq 0}^2$. In order to capture this special node n_0 , we introduce in Section 7.1 the notion of *modes* of a cone and prove that modes of non-decreasing sequence of cones eventually stabilizes.

Thanks to the notion of modes, it will be clear that $\text{Con}(\mathbf{P}_n)$ is \mathbf{v} -stable for every vector $\mathbf{v} \in \mathbf{I}_n \setminus \mathbb{Q}_{\geq 0}^2$ but also for every vector $\mathbf{v} \in \mathbf{I}_m \setminus \mathbb{Q}_{\geq 0}^2$ where $m \in \text{Des}(n_0)$. This \mathbf{v} -stabilization with respect to vectors that can be discovered later by the algorithm will be useful with the decomposition of elementary vectors of \mathbf{E}_n introduced in Section 7.2 as finite sums of elementary vectors \mathbf{E}_m where m ranges over a finite set independent of n .

Thanks to the notion of modes and the decomposition of elementary cycles, we prove in Section 7.3 that the cones $\text{Con}(\mathbf{P}_N)$ is finitely-generated. We conclude our proof by contradiction in Section 7.4.

7.1 Modes

The \mathbf{v} -stabilization of a cone $\mathbf{C} \subseteq \mathbb{Q}_{\geq 0}^2$ does not change the cone when \mathbf{v} satisfies some conditions depending on the set of axis of \mathbf{C} and a natural number $h \in \mathbb{N}$ that provides a lower-bound on some negative components of \mathbf{v} . By *axis* of a cone $\mathbf{C} \subseteq \mathbb{Q}_{\geq 0}^2$ we mean a vector in $\mathbf{C} \cap \{(1, 0), (0, 1)\}$. The h -mode of \mathbf{C} is the set of vectors $\mathbf{v} \in \mathbb{Z}^2 \setminus \mathbb{N}^2$ such that \mathbf{C} is \mathbf{v} -stable, and such that the following two conditions hold:

- $\mathbf{v}(1) \geq -h$ if $(1, 0)$ is not an axis of \mathbf{C} .
- $\mathbf{v}(2) \geq -h$ if $(0, 1)$ is not an axis of \mathbf{C} .

► **Lemma 7.2.** *The h -mode of any non-decreasing sequence of cones $\mathbf{C}_0 \subseteq \mathbf{C}_1 \subseteq \dots \subseteq \mathbb{Q}_{\geq 0}^2$ eventually stabilizes.*

Proof sketch. By considering a suffix of the sequence, by discarding the trivial cases, and by symmetry (on the set of axis), we can assume that $(1, 0) \in \mathbf{C}_n \not\subseteq \mathbb{Q}_{\geq 0}(1, 0)$ for every $n \in \mathbb{N}$. We observe that the h -mode \mathbf{M}_n of \mathbf{C}_n can be decomposed into $\mathbf{Q}_n \cup (\mathbb{N} \times \{-h, \dots, -1\})$ where $\mathbf{Q}_n = \{\mathbf{v} \in \mathbf{M}_n \mid \mathbf{v}(1) < 0 \wedge \mathbf{v}(2) \in \{-h, \dots, 0\}\}$. By observing that $(\mathbf{Q}_n)_{n \in \mathbb{N}}$ is a non-increasing sequence of finite sets, we deduce that it eventually stabilizes. ◀

7.2 Cycle Decomposition

In this section, we prove that vectors in \mathbf{E}_n can be decomposed as finite sums of vectors in \mathbf{E}_m , vectors in \mathbf{P}_s and some consecutive vectors where m and s ranges over finite sets independent of n . The finiteness of those sets follows from the co-finiteness of $\text{Des}(n_0)$ and $\text{Des}(n_1)$ since the exploration is primary.

► **Lemma 7.3.** *For every primary and directed algorithmic exploration $(N, \rightarrow, \lambda)$ and for every node $n_0 \in N$ such that $\sum_{s \rightarrow n_0} \mathbf{P}_s \neq \{(0, 0)\}$, there exists a node $n_1 \in \text{Des}(n_0)$ such that for every node $n \in \text{Des}(n_0)$, we have:*

$$\mathbf{E}_n \subseteq \sum_{m \in \text{Des}(n_0) \setminus (\text{Des}(n_1) \setminus \{n_1\})} \text{Per}(\mathbf{E}_m) + \sum_{s \notin \text{Des}(n_0)} \mathbf{P}_s + \sum_{m | n_0 \xrightarrow{*} m \xrightarrow{+} n} \text{Per}(\mathbf{C}_m)$$

We define a node n_1 satisfying the previous lemma as follows. We fix a primary and directed exploration $(N, \rightarrow, \lambda)$ and a node $n_0 \in N$ such that $\sum_{s \rightarrow n_0} \mathbf{P}_s \neq \{(0, 0)\}$. We introduce an abstraction function $\alpha : \mathbb{N}^2 \rightarrow \{0, \dots, c\}^2$ defined by $\alpha(\mathbf{x})(i) = \mathbf{x}(i)$ if $\mathbf{x}(i) < c$ and $\alpha(\mathbf{x})(i) = c$ if $\mathbf{x}(i) \geq c$, for every $\mathbf{x} \in \mathbb{N}^2$ and every $i \in \{1, 2\}$, where c is the constant of iteration of \mathcal{B} . We introduce a partial order \sqsubseteq on the set of nodes $\text{Des}(n_0)$ defined by $s \sqsubseteq n$ if $s \xrightarrow{*} n$ and $q_s(\alpha(\mathbf{z}_s)) = q_n(\alpha(\mathbf{z}_n))$. Since $\xrightarrow{*}$ is a well-quasi-order on the set of nodes, and the equality is a well-quasi-order on the finite set $Q \times \{0, \dots, c\}^2$, the partial order \sqsubseteq is also a well-quasi-order as the intersection of two well-quasi-orders. In particular the set N_{\min} defined as the set of minimal nodes $n \in \text{Des}(n_0)$ for \sqsubseteq is finite and for every $n \in \text{Des}(n_0)$, there exists $s \in N_{\min}$ such that $s \sqsubseteq n$. Since the exploration is primary, the set $N \setminus \text{Des}(n_0)$ is finite. Moreover, as the exploration is directed and $N_{\min} \cup (N \setminus \text{Des}(n_0))$ is finite, there exists a node $n'_0 \in N$ such that $n \xrightarrow{*} n'_0$ for every $n \in N_{\min} \cup (N \setminus \text{Des}(n_0))$. Let Q_c be the set of states $q \in Q$ such that there exists a node $n \in \text{Des}(n'_0)$ satisfying $\mathbf{z}_n \geq (c, c)$ and $q_n = q$. For each $q \in Q_c$, we pick such a node n_q . Since the set $\{n_q \mid q \in Q_c\}$ is finite and the exploration is directed, there exists $n_1 \in \text{Des}(n'_0)$ such that $n_q \xrightarrow{*} n_1$ for every $q \in Q_c$. This node n_1 satisfies Lemma 7.3. Intuitively, this property is obtained by decomposing recursively either paths in the algorithmic exploration from a node $n \in \text{Des}(n_0)$ to a node m such that $q_n = q_m$ following intermediate nodes n satisfying $\mathbf{z}_n \geq (c, c)$ into elementary and consecutive cycles, and by replacing any elementary cycle using a transition coming from an instantiated node $n \in \text{Des}(n_0)$ by the elementary cycle obtained by using a node $s \in N_{\min}$ satisfying $s \sqsubseteq n$ rather than n for the instantiation. Since the effect of the original cycle is equal to the sum of the effect of the new one with the effect of a path from s to n , by recursively decomposing that path we prove that n_1 satisfies Lemma 7.3. Such a decomposition and the proof that n_1 satisfies Lemma 7.3 are fully detailed in [2].

7.3 Finitely-generated Cone

In this section, we prove the following lemma.

► **Lemma 7.4.** *For every primary and directed algorithmic exploration, the cone spanned by $\mathbf{P}_N = \sum_{n \in N} \mathbf{P}_n$ is finitely-generated.*

Let \mathcal{G} be a primary and directed algorithmic exploration and let \mathbf{P}_N be the periodic set $\sum_{n \in N} \mathbf{P}_n$. We introduce the set of axis $\mathbf{U} = \text{Con}(\mathbf{P}_N) \cap \{(1, 0), (0, 1)\}$, the constant of iteration c , and the set $\mathbf{Z} = \{\mathbf{z}_n \mid n \in N \wedge \mathbf{z}_n \not\geq (c, c)\}$. If $\mathbf{U} = \{(1, 0), (0, 1)\}$ we are done since in that case $\text{Con}(\mathbf{P}_N) = \mathbb{Q}_{\geq 0}^2$. So, we can assume that \mathbf{U} contains at most one vector.

► **Lemma 7.5.** *There exists $h \in \mathbb{N}$ such that $\mathbf{Z} \subseteq \{0, \dots, h\}^2 + \mathbf{U}^*$.*

Proof. Recall from Section 6 that $\xrightarrow{*}$ is a wqo. Given $q \in Q$, $d \in \{0, \dots, c-1\}$, and $i \in \{1, 2\}$, we introduce the set $N_{q,i,d}$ of nodes $n \in N$ such that $q_n = q$ and $\mathbf{z}_n(i) = d$. Since $\xrightarrow{*}$ is a wqo, the set $M_{q,i,d}$ of minimal elements of $N_{q,i,d}$ for this partial order is finite, and for every $n \in N_{q,i,d}$, there exists $m \in M_{q,i,d}$ such that $m \xrightarrow{*} n$. We pick $h \in \mathbb{N}$ satisfying $h \geq \mathbf{z}_m(i)$ for every $q \in Q$, $i \in \{1, 2\}$, $d \in \{0, \dots, c-1\}$ and $m \in M_{q,i,d}$.

Observe that for every $\mathbf{z} \in \mathbf{Z}$, there exists $n \in N$ such that $\mathbf{z} = \mathbf{z}_n \not\geq (c, c)$. It follows that there exists $i \in \{1, 2\}$ such that $d = \mathbf{z}_n(i)$ is in $\{0, \dots, c-1\}$. Let $q = q_n$ and notice that $n \in N_{q,i,d}$. It follows that there exists $m \in M_{q,i,d}$ such that $m \xrightarrow{*} n$. Observe that $\mathbf{z}_m(i) = d = \mathbf{z}_n(i)$ and $q_m = q = q_n$. Moreover $\mathbf{z}_m(\bar{i}) \leq h$. If $\mathbf{z}_n(\bar{i}) \leq h$ then $\mathbf{z} \in \{0, \dots, h\}^2$. If $\mathbf{z}_n(\bar{i}) > h$, then $\mathbf{z}_n - \mathbf{z}_m \in \mathbb{Q}_{\geq 0} \mathbf{u}$ for some $\mathbf{u} \in \{(1, 0), (0, 1)\}$. It follows that \mathbf{u} is in $\text{Con}(\mathbf{P}_n)$ since $\mathbf{z}_n - \mathbf{z}_m$ is a consecutive vector in \mathbf{C}_n . Hence it is also in \mathbf{U} . As $\mathbf{z}_m \in \{0, \dots, h\}^2$ we deduce that $\mathbf{z} \in \{0, \dots, h\}^2 + \mathbf{U}^*$. We have proved the lemma. \blacktriangleleft

In the sequel, h denotes a fix natural number satisfying the previous lemma and $h \geq c$. We put $\mathbf{P}'_n = \sum_{s \rightarrow n} \mathbf{P}_s$ for every node $n \in N$. If $\mathbf{P}'_n = \{(0, 0)\}$ for every $n \in N$ then $\text{Con}(\mathbf{P}_N) = \{(0, 0)\}$ and we are done also in that case. So, we can assume that there exists a node $n \in N$ such that $\mathbf{P}'_n \neq \{(0, 0)\}$.

Let us prove that there exists a node n_0 such that \mathbf{U} is the set of axis of $\text{Con}(\mathbf{P}_{n_0})$ and $\mathbf{P}'_{n_0} \neq \{(0, 0)\}$. If $\mathbf{U} = \emptyset$, it is sufficient to consider a node $n_0 \in N$ such that $\mathbf{P}'_{n_0} \neq \{(0, 0)\}$. If $\mathbf{U} \neq \emptyset$, it is sufficient to consider a node n_0 such that the unique vector of \mathbf{U} is in $\text{Con}(\mathbf{P}'_{n_0})$. By replacing n_0 by a descendant of n_0 , thanks to Lemma 7.2, we can assume that the h -mode of $\text{Con}(\mathbf{P}_n)$ is equal to the h -mode of $\text{Con}(\mathbf{P}_{n_0})$ for every $n \in \text{Des}(n_0)$.

In the sequel, given a set $\mathbf{X} \subseteq \mathbb{Q}^2$, we introduce $\mathbf{X}^+ = \mathbf{X} \cap \mathbb{Q}_{\geq 0}^2$ and $\mathbf{X}^- = \mathbf{X} \setminus \mathbb{Q}_{\geq 0}^2$.

► **Lemma 7.6.** *The set $\text{Con}(\mathbf{U}) + \sum_{m \in \text{Des}(n_0)} \text{Con}(\mathbf{C}_m^+)$ is a finitely-generated cone.*

Proof sketch. We just observe that $\mathbf{C}_m \subseteq \mathbf{Z} - \mathbf{Z}$ and $\mathbf{Z} \subseteq \{0, \dots, h\}^2 + \mathbf{U}^*$. \blacktriangleleft

Now, let $n_1 \in \text{Des}(n_0)$ satisfying Lemma 7.3.

► **Lemma 7.7.** *The set $\text{Con}(\mathbf{P}_N)$ can be decomposed as follows:*

$$\sum_{m \in \text{Des}(n_0) \setminus (\text{Des}(n_1) \setminus \{n_1\})} \text{Con}(\mathbf{E}_m^+) + \sum_{s \notin \text{Des}(n_0)} \text{Con}(\mathbf{P}_s) + \text{Con}(\mathbf{U}) + \sum_{m \in \text{Des}(n_0)} \text{Con}(\mathbf{C}_m^+)$$

Proof sketch. Let \mathbf{K} be the cone given above. Clearly \mathbf{E}_m^+ where $m \in \text{Des}(n_0) \setminus (\text{Des}(n_1) \setminus \{n_1\})$, \mathbf{P}_s where $s \notin \text{Des}(n_0)$, \mathbf{U} and \mathbf{C}_m^+ where $m \in \text{Des}(n_0)$ are included in $\text{Con}(\mathbf{P}_N)$. It follows that $\mathbf{K} \subseteq \text{Con}(\mathbf{P}_N)$. We prove the converse inclusion by induction on the well-foundedness of the relation \rightarrow , showing that $\mathbf{P}_n \subseteq \mathbf{K}$ for every $n \in N$. The crucial observation is the fact that even if in the right-hand side we discard vectors in \mathbf{E}_m^- and \mathbf{C}_m^- , since the h -mode of $\text{Con}(\mathbf{P}_n)$ does not depend on the node $n \in \text{Des}(n_0)$, we can apply the decomposition of \mathbf{E}_n given by Lemma 7.3 to prove the inclusion $\mathbf{P}_n \subseteq \mathbf{K}$. \blacktriangleleft

Lemma 7.6 and Lemma 7.7 show that $\text{Con}(\mathbf{P}_N)$ is a finitely-generated cone.

7.4 Wrap-Up

In this section, we prove the termination of our algorithm by assembling the results from the previous sections. We start with the following observation.

► **Lemma 7.8.** *Let $\mathcal{G} = (N, \rightarrow, \lambda)$ be an algorithmic exploration of a 2-BVASS \mathcal{B} and let $s, n \in N$. If $s \xrightarrow{*} n$, $q_s = q_n$ and $\mathbf{P}_s = \mathbf{P}_n$ then $\text{Con}(\mathbf{P}_n)$ is $(\mathbf{z}_n - \mathbf{z}_s)$ -stable.*

Proof sketch. The decomposition mentioned in Section 7.2 provides a way to decompose a path from s to n in the algorithmic exploration into elementary cycles and consecutive cycles, with cutting points from intermediate nodes m satisfying $\mathbf{z}_m \not\geq (c, c)$. With such a decomposition, we deduce that $\mathbf{z}_n - \mathbf{z}_s$ is a sum of vectors in \mathbf{P}_n and vectors $\mathbf{v} \in \mathbf{I}_m$ where

m ranges over the intermediate nodes of the considered path. Since $\text{Con}(\mathbf{P}_m)$ is \mathbf{v} -stable for each $\mathbf{v} \in \mathbf{I}_m$ and $\text{Con}(\mathbf{P}_m) = \text{Con}(\mathbf{P}_n)$, we deduce that $\text{Con}(\mathbf{P}_n)$ is \mathbf{v} -stable for all those vectors $\mathbf{v} \in \mathbf{I}_m$. We deduce that $\text{Con}(\mathbf{P}_n)$ is $(\mathbf{z}_n - \mathbf{z}_s)$ -stable. \blacktriangleleft

► **Lemma 7.9.** *Every non-redundant, algorithmic, primary and directed exploration of a 2-BVASS is finite.*

Proof. Let $\mathcal{G} = (N, \rightarrow, \lambda)$ be a non-redundant, algorithmic, primary and directed exploration of a 2-BVASS $\mathcal{B} = (Q, \Delta)$. By Lemma 7.4, the set $\mathbf{P} = \sum_{n \in N} \mathbf{P}_n$ is a finitely-generated periodic set. This entails that $\mathbf{P} = \sum_{n \in N'} \mathbf{P}_n$ for some finite set $N' \subseteq N$. As \mathcal{G} is directed, there exists a node $n_0 \in N$ such that $n' \xrightarrow{*} n_0$ for every $n' \in N'$. This entails that $\mathbf{P}_n = \mathbf{P}$ for all $n \in \text{Des}(n_0)$. Assume, by contradiction, that N is infinite. Since \mathcal{G} is primary, the set $\text{Des}(n_0)$ is infinite and the binary relation $\xrightarrow{*}$ is a wqo on N . By Dickson's Lemma, the binary relation \preceq on N defined by $m \preceq n$ if $q_m = q_n$ and $\mathbf{z}_m \leq \mathbf{z}_n$, is also a wqo on N . We derive that there exists an infinite sequence n_1, n_2, n_3, \dots of nodes in $\text{Des}(n_0)$ such that $n_1 \xrightarrow{+} n_2 \xrightarrow{+} n_3 \xrightarrow{+} \dots$, $q_{n_1} = q_{n_2} = q_{n_3} = \dots$, and $\mathbf{z}_{n_1} \leq \mathbf{z}_{n_2} \leq \mathbf{z}_{n_3} \leq \dots$. Note that $\mathbf{P}_{n_i} = \mathbf{P}$ for all $i \geq 1$. We deduce from Lemma 7.8 that $\text{Con}(\mathbf{P})$ is $(\mathbf{z}_{n_i} - \mathbf{z}_{n_1})$ -stable for all $i \geq 1$. Since the vector $\mathbf{z}_{n_i} - \mathbf{z}_{n_1}$ is in \mathbb{N}^2 , we get that it is in $\text{Con}(\mathbf{P})$. It follows that $\mathbf{z}_{n_i} \in \mathbf{z}_{n_1} + \text{Con}(\mathbf{P})$ for all $i \geq 1$. So the set $\mathbf{B} = \{\mathbf{z}_{n_i} \mid i \geq 1\}$ is contained in $\mathbf{z}_{n_1} + \text{Con}(\mathbf{P})$. We obtain from [16, Lemma 1.2] that $\mathbf{B} + \mathbf{P} = \mathbf{B}' + \mathbf{P}$ for some finite subset $\mathbf{B}' \subseteq \mathbf{B}$. This entails that $\mathbf{z}_{n_j} \in (\mathbf{z}_{n_i} + \mathbf{P})$ for some $i < j$. Observe that $q_{n_j}(\mathbf{z}_{n_j} + \mathbf{P}_{n_j}) \subseteq q_{n_i}(\mathbf{z}_{n_i} + \mathbf{P}_{n_i})$. Moreover, we have $n_i \xrightarrow{+} n_j$ since $i < j$. So the node n_j is redundant, but n_j is not a leaf since $n_j \xrightarrow{+} n_{j+1}$. This contradicts our assumption that \mathcal{G} is non-redundant. \blacktriangleleft

► **Corollary 7.10.** *Every non-redundant, algorithmic and spannable exploration of a 2-BVASS is finite.*

Proof. This corollary follows from Lemma 6.1, Lemma 7.9 and the following observation. Given an exploration $\mathcal{G} = (N, \rightarrow, \lambda)$ of a 2-BVASS \mathcal{B} and an ancestor-closed subset X of N , the restriction of \mathcal{G} to X is also an exploration of \mathcal{B} . Moreover, if \mathcal{G} is non-redundant (resp., algorithmic) then the restriction of \mathcal{G} to X is also non-redundant (resp., algorithmic). \blacktriangleleft

As indicated in Section 6, for every 2-BVASS \mathcal{B} and every execution σ of $\text{Explore}(\mathcal{B})$, the constructed exploration $\hat{\mathcal{G}}^\sigma$ is spannable. We derive from Lemmas 5.5 and 5.7 and Corollary 7.10 that every execution of $\text{Explore}(\mathcal{B})$ terminates and returns a sound and complete finite exploration of \mathcal{B} . The reachability set of \mathcal{B} is then easily obtained from this exploration. We obtain the following theorem.

► **Theorem 7.11.** *For every 2-BVASS \mathcal{B} , the reachability set $\llbracket \mathcal{B} \rrbracket$ of \mathcal{B} is semilinear and a presentation of $\llbracket \mathcal{B} \rrbracket$ is computable from \mathcal{B} .*

8 Conclusion

In this paper, we have shown that the reachability set of a 2-BVASS admits a computable semilinear presentation. This entails that the reachability problem for 2-BVASS is decidable. Our approach, which is inspired from Hopcroft and Pansiot's algorithm for classical 2-VASS [16], does not provide any upper bound on the complexity of this problem. The decidability status of the reachability problem for d -BVASS remains open in arbitrary dimension.

References

- 1 Mohamed Faouzi Atig and Pierre Ganty. Approximating petri net reachability along context-free traces. In Supratik Chakraborty and Amit Kumar, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2011, December 12-14, 2011, Mumbai, India*, volume 13 of *LIPICs*, pages 152–163. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2011. doi:10.4230/LIPICs.FSTTCS.2011.152.
- 2 Clotilde Bizière, Thibault Hilaire, Jérôme Leroux, and Grégoire Sutre. On the reachability problem for two-dimensional branching VASS, 2025. arXiv:2506.22561.
- 3 Michael Blondin, Matthias Englert, Alain Finkel, Stefan Göller, Christoph Haase, Ranko Lazic, Pierre McKenzie, and Patrick Totzke. The reachability problem for two-dimensional vector addition systems with states. *J. ACM*, 68(5):34:1–34:43, 2021. doi:10.1145/3464794.
- 4 Mikolaj Bojanczyk, Claire David, Anca Muscholl, Thomas Schwentick, and Luc Segoufin. Two-variable logic on data trees and XML reasoning. In Stijn Vansummeren, editor, *Proceedings of the Twenty-Fifth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 26-28, 2006, Chicago, Illinois, USA*, pages 10–19. ACM, 2006. doi:10.1145/1142351.1142354.
- 5 Ahmed Bouajjani and Michael Emmi. Analysis of recursively parallel programs. *ACM Trans. Program. Lang. Syst.*, 35(3):10:1–10:49, 2013. doi:10.1145/2518188.
- 6 Dmitry Chistikov, Wojciech Czerwinski, Filip Mazowiecki, Lukasz Orlikowski, Henry Sinclair-Banks, and Karol Węgrzycki. The tractability border of reachability in simple vector addition systems with states. In *65th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2024, Chicago, IL, USA, October 27-30, 2024*, pages 1332–1354. IEEE, 2024. doi:10.1109/FOCS61266.2024.00086.
- 7 Lorenzo Clemente, Slawomir Lasota, Ranko Lazic, and Filip Mazowiecki. Timed pushdown automata and branching vector addition systems. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017*, pages 1–12. IEEE Computer Society, 2017. doi:10.1109/LICS.2017.8005083.
- 8 Conrad Cotton-Barratt, Andrzej S. Murawski, and C.-H. Luke Ong. ML and extended branching VASS. In Hongseok Yang, editor, *Programming Languages and Systems - 26th European Symposium on Programming, ESOP 2017, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2017, Uppsala, Sweden, April 22-29, 2017, Proceedings*, volume 10201 of *Lecture Notes in Computer Science*, pages 314–340. Springer, 2017. doi:10.1007/978-3-662-54434-1_12.
- 9 Wojciech Czerwinski, Slawomir Lasota, Ranko Lazic, Jérôme Leroux, and Filip Mazowiecki. The reachability problem for Petri nets is not elementary. *J. ACM*, 68(1):7:1–7:28, 2021. doi:10.1145/3422822.
- 10 Philippe de Groote, Bruno Guillaume, and Sylvain Salvati. Vector addition tree automata. In *19th IEEE Symposium on Logic in Computer Science (LICS 2004), 14-17 July 2004, Turku, Finland, Proceedings*, pages 64–73. IEEE Computer Society, 2004. doi:10.1109/LICS.2004.1319601.
- 11 Stéphane Demri, Marcin Jurdzinski, Oded Lachish, and Ranko Lazic. 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.
- 12 Diego Figueira, Ranko Lazic, Jérôme Leroux, Filip Mazowiecki, and Grégoire Sutre. Polynomial-space completeness of reachability for succinct branching VASS in dimension one. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, volume 80 of *LIPICs*, pages 119:1–119:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPICs.ICALP.2017.119.
- 13 Alain Finkel, Jérôme Leroux, and Grégoire Sutre. Reachability for two-counter machines with one test and one reset. In Sumit Ganguly and Paritosh K. Pandya, editors, *38th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS*

- 2018, December 11-13, 2018, Ahmedabad, India, volume 122 of *LIPICs*, pages 31:1–31:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPICs.FSTTCS.2018.31.
- 14 Moses Ganardi, Rupak Majumdar, Andreas Pavlogiannis, Lia Schütze, and Georg Zetsche. Reachability in bidirected pushdown VASS. In Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*, volume 229 of *LIPICs*, pages 124:1–124:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICs.ICALP.2022.124.
 - 15 Stefan Göller, Christoph Haase, Ranko Lazić, and Patrick Totzke. A polynomial-time algorithm for reachability in branching VASS in dimension one. In *ICALP*, volume 55 of *LIPICs*, pages 105:1–105:13. Schloss Dagstuhl, 2016. doi:10.4230/LIPICs.ICALP.2016.105.
 - 16 John E. Hopcroft and Jean-Jacques Pansiot. On the reachability problem for 5-dimensional vector addition systems. *Theor. Comput. Sci.*, 8:135–159, 1979. doi:10.1016/0304-3975(79)90041-0.
 - 17 Florent Jacquemard, Luc Segoufin, and Jérémie Dimino. $\text{Fo2}(<, +1, \sim)$ on data trees, data tree automata and branching vector addition systems. *Log. Methods Comput. Sci.*, 12(2), 2016. doi:10.2168/LMCS-12(2:3)2016.
 - 18 Richard M. Karp and Raymond E. Miller. Parallel Program Schemata. *J. Comput. Syst. Sci.*, 3(2):147–195, 1969. doi:10.1016/S0022-0000(69)80011-5.
 - 19 Ranko Lazic. The reachability problem for vector addition systems with a stack is not elementary. *CoRR*, abs/1310.1767, 2013. doi:10.48550/arXiv.1310.1767.
 - 20 Ranko Lazic, Thomas Christopher Newcomb, Joël Ouaknine, A. W. Roscoe, and James Worrell. Nets with tokens which carry data. *Fundam. Informaticae*, 88(3):251–274, 2008. URL: <http://content.iospress.com/articles/fundamenta-informaticae/fi88-3-03>.
 - 21 Ranko Lazić and Sylvain Schmitz. Nonelementary complexities for branching VASS, MELL, and extensions. *ACM Trans. Comput. Log.*, 16(3):20:1–20:30, 2015. doi:10.1145/2733375.
 - 22 Jérôme Leroux. The reachability problem for Petri nets is not primitive recursive. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 1241–1252. IEEE, 2021. doi:10.1109/FOCS52979.2021.00121.
 - 23 Jérôme Leroux and Sylvain Schmitz. Reachability in vector addition systems is primitive-recursive in fixed dimension. In *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24-27, 2019*, pages 1–13. IEEE, 2019. doi:10.1109/LICS.2019.8785796.
 - 24 Jérôme Leroux, Grégoire Sutre, and Patrick Totzke. On the coverability problem for pushdown vector addition systems in one dimension. In Magnús M. Halldórsson, Kazuo Iwama, Naoki Kobayashi, and Bettina Speckmann, editors, *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part II*, volume 9135 of *Lecture Notes in Computer Science*, pages 324–336. Springer, 2015. doi:10.1007/978-3-662-47666-6_26.
 - 25 Denis Lugiez. Counting and equality constraints for multitree automata. In Andrew D. Gordon, editor, *Foundations of Software Science and Computational Structures, 6th International Conference, FOSSACS 2003 Held as Part of the Joint European Conference on Theory and Practice of Software, ETAPS 2003, Warsaw, Poland, April 7-11, 2003, Proceedings*, volume 2620 of *Lecture Notes in Computer Science*, pages 328–342. Springer, 2003. doi:10.1007/3-540-36576-1_21.
 - 26 Ernst W. Mayr. An algorithm for the general petri net reachability problem. *SIAM J. Comput.*, 13(3):441–460, 1984. doi:10.1137/0213029.
 - 27 Filip Mazowiecki and Michal Pilipczuk. Reachability for bounded branching VASS. In Wan J. Fokkink and Rob van Glabbeek, editors, *30th International Conference on Concurrency Theory, CONCUR 2019, August 27-30, 2019, Amsterdam, the Netherlands*, volume 140 of *LIPICs*, pages 28:1–28:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPICs.CONCUR.2019.28.

- 28 Hitoshi Ohsaki. Beyond regularity: Equational tree automata for associative and commutative theories. In Laurent Fribourg, editor, *Computer Science Logic, 15th International Workshop, CSL 2001. 10th Annual Conference of the EACSL, Paris, France, September 10-13, 2001, Proceedings*, volume 2142 of *Lecture Notes in Computer Science*, pages 539–553. Springer, 2001. doi:10.1007/3-540-44802-0_38.
- 29 Owen Rambow. Multiset-valued linear index grammars: Imposing dominance constraints on derivations. In James Pustejovsky, editor, *32nd Annual Meeting of the Association for Computational Linguistics, 27-30 June 1994, New Mexico State University, Las Cruces, New Mexico, USA, Proceedings*, pages 263–270. Morgan Kaufmann Publishers / ACL, 1994. doi:10.3115/981732.981768.
- 30 Sylvain Schmitz. On the computational complexity of dominance links in grammatical formalisms. In Jan Hajic, Sandra Carberry, and Stephen Clark, editors, *ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden*, pages 514–524. The Association for Computer Linguistics, 2010. URL: <https://aclanthology.org/P10-1053/>.
- 31 Kumar Neeraj Verma and Jean Goubault-Larrecq. Karp-miller trees for a branching extension of VASS. *Discret. Math. Theor. Comput. Sci.*, 7(1):217–230, 2005. doi:10.46298/DMTCS.350.