Discrete Mathematics and Theoretical Computer Science **7**, 2005, 217–230

Karp-Miller Trees for a Branching Extension of VASS

Kumar Neeraj Verma [1] [†]Jean Goubault-Larrecq [2]

[1] *Institut für Informatik, TU München, Germany* `verma@in.tum.de`
[2] *LSV/ UMR 8643 CNRS & ENS Cachan; INRIA Futurs projet SECSI, France* `goubault@lsv.ens-cachan.fr`

received Jun 29, 2004, *accepted Oct 19, 2005*.

We study BVASS (Branching VASS) which extend VASS (Vector Addition Systems with States) by allowing addition transitions that merge two configurations. Runs in BVASS are tree-like structures instead of linear ones as for VASS. We show that the construction of Karp-Miller trees for VASS can be extended to BVASS. This entails that the coverability set for BVASS is computable. This allows us to obtain decidability results for certain classes of equational tree automata with an associative-commutative symbol. Recent independent work by de Groote et al. implies that decidability of reachability in BVASS is equivalent to decidability of provability in MELL (multiplicative exponential linear logic), which is still an open problem. Hence our results are also a step towards answering this question in the affirmative.

Keywords: branching vector addition systems, Karp-Miller trees, coverability, multiplicative exponential linear logic, equational tree automata.

1 Introduction

The purpose of this paper is to study Branching VASS (BVASS), a natural extension of both vector addition systems with states (VASS) and Parikh images of context-free grammars, and to show that emptiness, coverability and boundedness are decidable for this common extension, by extending the usual Karp-Miller tree construction. This allows us to obtain decidability results for certain classes of two-way equational tree automata modulo the theory AC of one associative-commutative symbol [Ver03a], which arise naturally from the study of certain cryptographic protocols, and which were the initial motivation behind our extension. However recent independent work by de Groote et al. [dGGS04] also implies that decidability of reachability of configurations in BVASS is equivalent to decidability of provability in MELL (multiplicative exponential linear logic), which is still an open problem. Hence our results are a step towards a positive answer to this question.

For the time being, let us introduce semi-linear sets, VASS, Petri nets, and Parikh images of context-free grammars, so as to explain what our extension is about. We apologize in advance for the length of the exposition, but we feel it is better to understand the concepts before we build on them.

[†]Work done while PhD student at LSV, and partially supported by the ACI VERNAM, the RNTL project EVA and the ACI jeunes chercheurs "Sécurité informatique, protocoles cryptographiques et détection d'intrusions".

Notation. We use a Prolog-like notation: although this is not standard, this will have several advantages, one being uniformity of notation. Fix a natural number p. We shall consider definite clauses of the form $P(t) \Leftarrow P_1(t_1), \ldots, P_n(t_n)$, where $n \in \mathbb{N}$, P, P_1, …, P_n are so-called *predicates* (a.k.a., *states*), and t, t_1, …, t_n are terms built from constants $\nu \in \mathbb{N}^p$, variables x, y, z, … (denoting p-tuples of natural numbers), and the symbol $+$ denoting componentwise addition of p-tuples of natural numbers. An *instance* of such a clause is obtained by replacing all variables by actual p-tuples of natural numbers and doing the obvious simplifications. An *integer program* $\mathcal{P}$ is any finite set of definite clauses of the above format. A *fact* is any atom $P(\nu)$ with $\nu \in \mathbb{N}^p$. *Derivations* (of $P(\nu)$) from $\mathcal{P}$ are inductively defined so that, given any instance $P(\nu) \Leftarrow P_1(\nu_1), \ldots, P_n(\nu_n)$ of a definite clause in $\mathcal{P}$, given any derivations Δ_i of $P_i(\nu_i)$ from $\mathcal{P}$, $1 \leq i \leq n$, the following is a derivation:

$$\frac{\begin{array}{ccc} \vdots\,\Delta_1 & & \vdots\,\Delta_n \\ P_1(\nu_1) & \ldots & P_n(\nu_n) \end{array}}{P(\nu)}$$

(The base cases are when $n = 0$, in which case the chosen instance is the fact $P(\nu)$.) A fact is *derivable* from $\mathcal{P}$ iff there is a derivation of it from $\mathcal{P}$. The *language* $L_{\mathcal{P}}(P)$ of the predicate (or state) P in $\mathcal{P}$ is the set of p-tuples $\nu \in \mathbb{N}^p$ such that $P(\nu)$ is derivable from $\mathcal{P}$. We say that $\nu \in \mathbb{N}^p$ is *recognized* at P in $\mathcal{P}$ iff $P(\nu) \in L_{\mathcal{P}}(P)$.

Semilinear sets. Recall that a *linear set* L of p-tuples of natural numbers is any set of the form $L_{\nu_0,B} = \{\nu_0 + \nu_1 + \ldots + \nu_k \mid k \in \mathbb{N} \text{ and } \nu_1, \ldots, \nu_k \in B\}$ for some finite set $B \subseteq \mathbb{N}^p$. A *semilinear set* is any finite union of linear sets. Semilinear sets are one of the fundamentals of automated verification systems, and are closed under union, intersection, complement, and projection; they are exactly the Presburger-definable subsets of $\mathbb{N}^p$ [GS66]. Now, given $\nu_0, \in \mathbb{N}^p$ and some finite $B \subseteq \mathbb{N}^p$, consider the following set of Horn clauses:

$$P(\nu_0) \tag{1}$$

$$P(x+\nu) \quad \Leftarrow \quad P(x) \qquad (\nu \in B) \tag{2}$$

It is easy to see that the set of p-tuples recognized at P in this program is exactly $L_{\nu_0,B}$. Given any semilinear set L, written as a union $\bigcup_{i=1}^n L_{\nu_0^i,B^i}$, we can as easily write the set of all clauses $P_i(\nu_0^i)$, $1 \leq i \leq n$ and $P_i(x+\nu) \Leftarrow P_i(x)$, $1 \leq i \leq n, \nu \in B^i$, where P_1, …, P_n are pairwise distinct predicates. The union of the sets of tuples recognized at each P_i is L. In particular, any semilinear set can be represented as $L_{\mathcal{P}}(R)$ for some R and some finite set $\mathcal{P}$ of definite clauses of the form (1) or

$$P(x+\nu) \Leftarrow Q(x), \quad (\text{where } \nu \in \mathbb{N}^p) \tag{3}$$

Conversely, for every finite set $\mathcal{P}$ of what we shall call *base/period clauses* (of the form (1) or (3)), the languages $L_{\mathcal{P}}(P)$ are semilinear, for every predicate P; this is a consequence of Parikh's Theorem, to be stated below.

Note that derivations from such integer programs are just *sequences* of applications of clauses (3) ending in one clause (1).

Parikh images. What about allowing for more complex clause formats? One possibility is to replace clauses (3) by the more general *addition clauses*:

$$P(x+y) \Leftarrow Q(x), R(y) \tag{4}$$

(x and y being distinct variables) and keep the above clauses (1). (Clauses (3) are easily seen to be implementable through these two.) Addition clauses state that given any p-tuple recognized at Q, and given any p-tuple recognized at R, their sum is recognized at P. It turns out that these clause formats encode naturally Parikh images of context-free languages; this has been used in one form or another by several authors, we take our presentation from [Ver03c]. Recall that the Parikh image of a set L of words over the finite alphabet $\mathcal{A} = \{a_1, \ldots, a_p\}$ is the set of all p-tuples $|w| = (|w|_1, \ldots, |w|_p)$, where w ranges over L, and $|w|_i$ is by convention the number of occurrences of a_i in the word w. The construction goes as follows. Take any context-free grammar in Chomsky normal form, i.e., with productions of the form $P \to a_i$, or $P \to \varepsilon$, or $P \to QR$ (where P, Q, R are non-terminals, and ε denotes the empty word). For each production $P \to a_i$, generate the clause $P((0, \ldots, 0, 1, 0, \ldots, 0))$, where the only '1' is at position i; for each production $P \to \varepsilon$, generate the clause $P((0, \ldots, 0))$; for each production $P \to QR$, generate $P(x+y) \Leftarrow Q(x), R(y)$. Then the language of P is the Parikh image of the language generated by the grammar with start symbol P [Ver03c].

Parikh's Theorem [Par66], once recast in our setting (see [Ver03c]), states that given any program $\mathcal{P}$ consisting of clauses of the form (1), (3) and (4), the languages $L_{\mathcal{P}}(P)$ are all semilinear sets, and effectively so. So there is a procedure that computes a set of base/period clauses (1), (2) from any such program $\mathcal{P}$, in such a way that the languages of P are preserved, for each P in $\mathcal{P}$. (A nice, generalized version of this appears in [AÉI02].)

Note that, while derivations in base/period programs are just sequences, derivations in the presence of addition clauses (4) exhibit a *branching behavior*. In a sense, Parikh's Theorem states that branching can be eliminated while preserving languages.

Petri nets and VASS. If, instead of allowing addition clauses (4), we allow *two-way* clauses of the form

$$P(x+\nu^\bullet) \Leftarrow Q(x+{}^\bullet\nu) \tag{5}$$

as another extension of base/period clauses, where $\nu^\bullet$ and ${}^\bullet\nu$ are elements of $\mathbb{N}^p$, then we get so-called *vector addition systems with states* (VASS) [HP79][‡]. The languages $L_{\mathcal{P}}(P)$ are called *reachability sets* (for state P) in this context. To simplify matters, we shall assume that $\min(\nu^\bullet, {}^\bullet\nu) = 0$, meaning that for every index i, either the ith component of $\nu^\bullet$ or the ith component of ${}^\bullet\nu$ is zero. This entails no loss of generality as far as reachability, or coverability, or boundedness, is concerned [Reu89]. E.g., $P(x+(2,3)) \Leftarrow Q(x+(4,1)))$ can be replaced by $P(x+(2,3)) \Leftarrow R(x)$ and $R(x) \Leftarrow Q(x+(4,1))$ for those purposes, with R a fresh state. In this case, there is no loss of generality either in abbreviating (5) as

$$P(x+\delta) \Leftarrow Q(x) \tag{6}$$

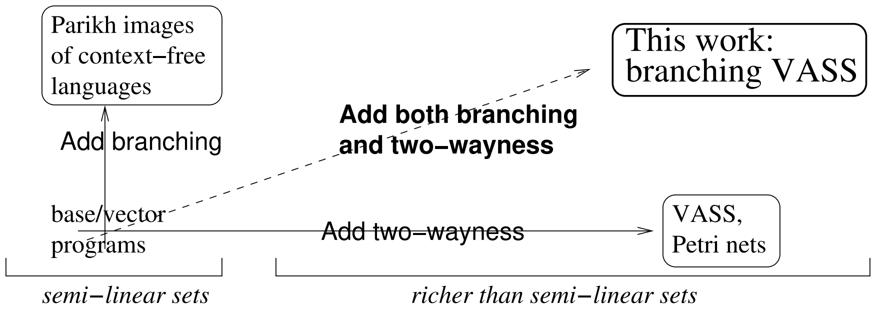
where $\delta \in \mathbb{Z}^p$ is a vector of *integers*, negative or positive, equal to $\nu^\bullet - {}^\bullet\nu$. Then, any derivation ending in an instance $P(\nu+\delta) \Leftarrow Q(\nu)$ of (6) simply infers $P(\nu+\delta)$ from $Q(\nu)$, provided $\nu+\delta$ is a non-negative tuple (in $\mathbb{N}^p$).

[‡] Up to the fact that VASS contain only one fact, the *initial marking*. This is inessential here.

Contrarily to Parikh images of context-free languages, two-way clauses strictly extend semilinear sets (provided $p \geq 3$), as there are VASS $\mathcal{P}$ and predicates P whose reachability set $L_{\mathcal{P}}(P)$ is not semilinear as soon as $p \geq 3$ [HP79]. VASS with just one state P are called *Petri nets*, and are as expressive as general VASS, as far as reachability sets are concerned [HP79]. Note that reachability of a fixed p-tuple in a given VASS is decidable [May84, Kos82, Mül84, Lam92, Reu89], but hard, both conceptually and complexity-wise (it is EXPSPACE-hard, and probably much higher).

A first step in deciding reachability of VASS is to construct the *Karp-Miller coverability tree* of the VASS $\mathcal{P}$ [KM69]. The construction is easier conceptually, although not primitive recursive. Karp-Miller trees compute all approximants of finite and infinite derivations, which we call *covering derivations* in this paper; the point is that there are only finitely many covering derivations, and this allows one to decide coverability and boundedness in VASS, in addition to being instrumental in deciding reachability.

This work: Branching VASS. The purpose of the present paper is to show that the Karp-Miller construction extends to the case of so-called *branching VASS*, or *BVASS*, which extend both Parikh images of context-free languages (sets of facts (1) and addition clauses (4)) and VASS (sets of facts (1) and two-way clauses (6)), by allowing all three kinds of clauses. I.e. *BVASS* are defined to consist of facts (1), addition clauses (4), and two-way clauses (6). BVASS exhibit both two-wayness and a branching behavior:



BVASS are clearly at least as expressive as Petri nets and VASS. At the moment, it is unknown whether we can effectively transform any BVASS into a VASS with the same reachability sets. (I.e., can we eliminate branching?) In fact, we do not know whether BVASS are strictly more expressive or just as expressive as VASS. An analogue of Parikh's Theorem would be needed here, but all known proof techniques for the latter that we know of fail on BVASS. Another extension to Petri nets that has been studied in the literature is ground rewrite systems modulo AC [MR98], which builds on the aforementioned decidability results for reachability in Petri nets. The latter do not seem to bear any relationship with BVASS.

Outline. Instead, we concentrate on generalizing the Karp-Miller construction to BVASS. While most of our arguments will look like the usual Karp-Miller construction, there is one difference. Remember that derivations in Petri nets are *sequences* of rule applications. The usual Karp-Miller construction organizes (approximants of) finite and infinite derivations, which we call the *covering derivations*, into a tree branching downwards, by sharing common prefixes; this Karp-Miller tree is finite by König's Lemma. With BVASS, derivations are trees *branching upwards*. There is little hope to organize such trees in a common structure (with both upward and downward branches, should it be called a jungle?). In particular, we lose the ability to use König's Lemma and conclude anything interesting this way. We show in this paper how König's Lemma can still be used, by building a special forest of covering derivations instead of burrowing our way through a jungle, to show that there are only finitely many covering derivations. The construction of covering derivations and their properties is set out in Section 2, the termination argument in Section 3.

We then apply this result to show that the emptiness, coverability and the boundedness problems for branching VASS are decidable, just like they are for VASS. Another consequence, which we briefly explain in Section 4, is that standard-two-way constant-only AC tree automata (extending the constant-only restriction of [Ver03c, Ver03b] with so-called *standard* + *push clauses*) have a decidable intersection-emptiness problem. Currently we know no alternative proof of this result, which is a justification of the usefulness of BVASS. We demonstrate a further extension of BVASS in Section 5, on which we basically know nothing, motivated by a more general notion of two-way constant-only AC tree automata. We conclude in Section 6.

Related work. While equational tree automata may have been the initial motivation behind this study, the interest of BVASS is however not limited to the narrow realm of equational tree automata. Recently de Groote et al. have independently defined *vector addition tree automata (VATA)*, which are essentially BVASS, and shown that decidability of provability in MELL (multiplicative exponential linear logic) is equivalent to decidability of reachability in VATA (see [dGGS04] for details)[§]. In other words, decidability of provability in MELL is equivalent to decidability of reachability in BVASS. No decidability questions are answered in [dGGS04]. Hence our Karp-Miller construction for BVASS can be seen as a step towards a positive answer to this open question. The fact that BVASS are a natural common generalization of two already well-established tools in computer science – Parikh images of context free languages, and VASS – and that they are useful in domains as diverse as equational tree automata and linear logic, confirms that BVASS are interesting objects to study. This connection between MELL and VATA (hence BVASS) generalizes the already known connection between ordinary VASS and the !-Horn fragment of MELL, which was used to obtain decidability result for the latter [Kan95]. See also [Kan94, Kan96] for connections between different fragments of linear logic and VASS.

For related work on equational tree automata, see [Ohs01, Lug03]. While these deal mainly with one-way variants, we have introduced two-way variants in order to deal with cryptographic protocols [GLRV05, GLV02, Ver03c, Ver03b, Ver03a]. Our study of BVASS was initially prompted by certain classes of these automata.

2 Covering Derivations

Covering derivations for branching VASS are defined in much the same way as for VASS. Definition 1 below should therefore not surprise the cognoscenti. The only new item of the definition, compared to VASS, would be item 3, which is due to the presence of addition clauses. If this were removed, we would get a definition of something very similar to the individual (finite prefixes of) *paths* in Karp-Miller trees of ordinary VASS. As we have said earlier, defining the Karp-Miller tree (jungle?) would be impossible, or at least obscure, in our extended setting.

We check all needed properties of covering derivations here. The challenge will be to show that there are only finitely many covering derivations, see Section 3.

[§] However 'BVASS' is not a new name that we have invented, and appears already in [VGL04]. This extension of VASS first appears in print in [Ver03a] where it is simply called extended VASS.

Definition 1 (Covering Derivation) *A* generalized fact *is any atom $P(\nu)$, where ν is in $(\mathbb{N} \cup \{\infty\})^p$. Addition, comparison are defined componentwise, with the convention that $\infty + n = n + \infty = \infty + \infty = \infty$ for every $n \in \mathbb{Z} \cup \{\infty\}$ and $n < \infty$ for every $n \in \mathbb{Z}$. For every i, $1 \leq i \leq p$, let $\nu[i]$ denote the ith component of ν. We write $\nu_1 <_i \nu_2$ to mean that $\nu_1 \leq \nu_2$ and $\nu_1[i] < \nu_2[i]$.*

Assume fixed a branching VASS $\mathcal{V}$. A covering derivation *Δ is a finite tree, each of whose nodes is labeled with a generalized fact and a clause, constructed using the following rules:*

1. *Whenever $P(\nu)$ is a clause in $\mathcal{V}$, the following is a covering derivation:*

$$\frac{}{P(\nu)}\ P(\nu)$$

2. *For every covering derivation*

$$\begin{array}{c}\vdots\ \Delta_1 \\ P_1(\nu'_1)\end{array}$$

such that $P_1(\nu'_1)$ only occurs once (namely, at the bottom) in Δ_1, for every transition $P(x+\delta) \Leftarrow P_1(x)$ in $\mathcal{V}$ such that $\nu'_1 + \delta \geq 0$,

$$\frac{\begin{array}{c}\vdots\ \Delta_1 \\ P_1(\nu'_1)\end{array}}{P(\nu')}\ P(x+\delta) \Leftarrow P_1(x)$$

is a covering derivation, where ν' is defined as the vector whose ith component is:

∞ *if there is a generalized fact $P(\nu'')$ in Δ_1 (that is, at or above $P_1(\nu'_1)$) such that $\nu'' <_i \nu'_1 + \delta$;*

$\nu'_1[i] + \delta[i]$ *otherwise.*

3. *For every covering derivations*

$$\begin{array}{c}\vdots\ \Delta_1 \\ P_1(\nu'_1)\end{array} \qquad\qquad \begin{array}{c}\vdots\ \Delta_2 \\ P_2(\nu'_2)\end{array}$$

such that $P_1(\nu'_1)$ only occurs once in Δ_1 and $P_2(\nu'_2)$ only occurs once in Δ_2, for every addition clause $P(x+y) \Leftarrow P_1(x), P_2(y)$ in $\mathcal{V}$,

$$\frac{\begin{array}{c}\vdots\ \Delta_1 \\ P_1(\nu'_1)\end{array} \quad \begin{array}{c}\vdots\ \Delta_2 \\ P_2(\nu'_2)\end{array}}{P(\nu')}\ P(x+y) \Leftarrow P_1(x), P_2(y)$$

is a covering derivation, where ν' is defined as the vector whose ith component is:

∞ *if there is a generalized fact $P(\nu'')$ in Δ_1 or Δ_2 such that $\nu'' <_i \nu'_1 + \nu'_2$;*

$\nu'_1[i] + \nu'_2[i]$ *otherwise.*

Intuitively, covering derivations compute "limits" of facts derivable in a branching VASS. This is made precise by Propositions 1 and 2 below.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{P_1(2,5)}\,C_1 \quad \dfrac{}{P_2(3,4)}\,C_2}{P_3(5,9)}\,C_3}{P_1(\infty,5)}\,C_4 \qquad \dfrac{\dfrac{\dfrac{}{P_1(2,5)}\,C_1 \quad \dfrac{}{P_2(3,4)}\,C_2}{P_3(5,9)}\,C_3}{P_2(\infty,4)}\,C_5}{P_3(\infty,9)}\,C_3}{P_1(\infty,5)}\,C_4$$

Fig. 1: A covering derivation for the BVASS in Example 1

Example 1 *Consider a branching VASS with the following set of clauses*

$$C_1 = P_1(2,5) \qquad C_2 = P_2(3,4) \qquad C_4 = P_1(x+(-2,-4)) \Leftarrow P_3(x)$$
$$C_3 = P_3(x+y) \Leftarrow P_1(x), P_2(y) \qquad C_5 = P_2(x+(2,-5)) \Leftarrow P_3(x)$$

Some facts derivable in this branching VASS are $P_1(2,5)$, $P_2(3,4)$, $P_3(5,9)$, $P_1(2+n,5)$, $P_3(5+n,9)$, $P_2(3+4n,4)$ *for all* $n \geq 0$. *Figure 1 shows an example of a covering derivation for this branching VASS. This covering derivation cannot be extended further, because the final fact* $P_1(\infty,5)$ *also occurs higher in the derivation, so items 2 and 3 of the definition do not apply. Intuitively, the meaning of* $P_1(\infty,5)$ *is that* $P_1(n,5)$ *is derivable from* C_1*–*C_5 *for arbitrarily high values of* $n \in \mathbb{N}$. *This will be made precise in Proposition 2.*

Proposition 1 *Let* $\mathcal{V}$ *be a branching VASS. If a fact* $P(\nu)$ *is derivable, then there is a covering derivation* Δ *of some generalized fact* $P(\nu')$ *such that for all* $1 \leq i \leq p$, *if* $\nu'[i] < \infty$ *then* $\nu'[i] = \nu[i]$.

Proof: We do induction on the size of the derivation of $P(\nu)$. We have the following cases:

(i) If $P(\nu)$ is derivable using the clause $P(\nu)$, then use Rule 1 of Definition 1; this satisfies the requirements.

(ii) Suppose $P(\nu_1+\delta)$ is derivable from the derivation Δ_1 of $P_1(\nu_1)$ using the clause $P(x+\delta) \Leftarrow P_1(x)$. By induction hypothesis we have a covering derivation Δ_1 of some generalized fact $P_1(\nu'_1)$, such that if $\nu'_1[i] < \infty$ then $\nu'_1[i] = \nu_1[i]$. We pick a minimal such Δ_1. Consequently $P_1(\nu'_1)$ does not occur except as conclusion in Δ_1. Clearly we have $\nu_1+\delta \geq 0$ and hence $\nu'_1+\delta \geq 0$. By using Rule 2 of Definition 1, we get a covering derivation Δ with root labeled by a generalized fact $P(\nu')$ with the property that if $\nu'[i] < \infty$ then $\nu'[i] = \nu'_1[i]+\delta[i]$. But then if $\nu'[i] < \infty$ then $\nu'_1[i] < \infty$ and hence $\nu'_1[i] = \nu_1[i]$, so $\nu'[i] = \nu_1[i]+\delta[i]$. Hence Δ is the required covering derivation.

(iii) Suppose $P(\nu_1+\nu_2)$ is derivable from the derivations of $P_1(\nu_1)$ and $P_2(\nu_2)$ using the clause $P(x+y) \Leftarrow P_1(x), P_2(y)$. By induction hypothesis, we have covering derivations Δ_1 and Δ_2 of $P_1(\nu'_1)$ and $P_2(\nu'_2)$ respectively such that for all i, if $\nu'_1[i] < \infty$ then $\nu'_1[i] = \nu_1[i]$, and if $\nu'_2[i] < \infty$ then $\nu'_2[i] = \nu_2[i]$. As in the previous case we may assume that $P_1(\nu'_1)$ only occurs in Δ_1 as the conclusion, and $P_2(\nu'_2)$ only occurs in Δ_2 as the conclusion. By using Rule 3 of Definition 1, we get a covering derivation Δ of some $P(\nu')$ with the property that if $\nu'[i] < \infty$, then $\nu'[i] = \nu'_1[i]+\nu'_2[i]$. But then if $\nu'[i] < \infty$, then $\nu'_1[i], \nu'_2[i] < \infty$, and hence $\nu'_1[i] = \nu_1[i]$ and $\nu'_2[i] = \nu_2[i]$, implying that $\nu'[i] = \nu_1[i]+\nu_2[i]$. Hence Δ is the required covering derivation. $\square$

Definition 2 (Linear Path) *Given a branching VASS $\mathcal{V}$, the* linear paths π *of $\mathcal{V}$ are the sequences* $P_1 \xrightarrow{e_1} P_2 \xrightarrow{e_2} \dots P_{n-1} \xrightarrow{e_{n-1}} P_n$ *, where $n \geq 1$, $P_1, \dots, P_n$ are states (predicates) of $\mathcal{V}$, and for each i, $1 \leq i < n$:*

- *either e_i is a two-way clause $P_{i+1}(x+\delta) \Leftarrow P_i(x)$;*
- *or e_i is a pair $\langle C; Q(\nu)\rangle$ of an addition clause $C = P_{i+1}(x+y) \Leftarrow P_i(x), Q(y)$, and of a fact $Q(\nu)$ that is derivable from $\mathcal{V}$. Here it is understood that the order of atoms in the body of an addition clause is irrelevant.*

We also say that π is a linear path from P_1 to P_n. *The elements e_i are called* edges. *In the first case, the* valuation $v(e_i)$ *of e_i is δ; in the second case, it is ν. The* valuation $v(\pi)$ *of the linear path π is $v(\pi) = \Sigma_{1 \leq i < n} v(e_i)$.*

If π_1 is the linear path $P_1 \xrightarrow{e_1} P_2 \xrightarrow{e_2} \dots P_{n-1} \xrightarrow{e_{n-1}} P_n$ and π_2 is the linear path $P_n \xrightarrow{e_n} P_{n+1} \xrightarrow{e_{n+1}} \dots P_{n+m-1} \xrightarrow{e_{n+m-1}} P_{n+m}$ then their concatenation $\pi_1\pi_2$ is defined as the linear path

$$P_1 \xrightarrow{e_1} P_2 \xrightarrow{e_2} \dots P_{n-1} \xrightarrow{e_{n-1}} P_n \xrightarrow{e_n} P_{n+1} \xrightarrow{e_{n+1}} \dots P_{n+m-1} \xrightarrow{e_{n+m-1}} P_{n+m}$$

Note that $\pi_1\pi_2$ is defined only when the last predicate of π_1 is equal to the first predicate of π_2. Clearly we have $v(\pi_1\pi_2) = v(\pi_1) + v(\pi_2)$.

Definition 3 (Admissible Linear Path) *Given $\nu \in (\mathbb{N} \cup \{\infty\})^p$, the linear path π is said to be* admissible *for ν if and only if, for each prefix π' of π, we have $\nu + v(\pi') \geq 0$. π is* admissible *for ν* with respect to $I \subseteq \{1, \dots, p\}$ *if and only if $\nu[I] + v(\pi')[I] \geq 0$ for all prefixes π' of π, where $\nu[I]$ denotes the tuple consisting of components $\nu[i]$ with $i \in I$.*

It is easy to see that if π is admissible for ν and $P_1(\nu)$ is derivable in $\mathcal{V}$ (in which case ν would have no infinite coordinate,) then $P_n(\nu + v(\pi))$ is derivable. Also if π_1 is admissible for ν and π_2 is admissible for $\nu + v(\pi_1)$, then $\pi_1\pi_2$ is admissible for ν.

Example 2 *The following is an example of a linear path.*

$$P_1 \xrightarrow{\langle C_3; P_2(3,4)\rangle} P_3 \xrightarrow{C_4} P_2 \xrightarrow{\langle C_3; P_1(50,5)\rangle} P_3$$

Note that the facts $P_2(3,4)$ and $P_1(50,5)$ are derivable in the branching VASS of Example 1. Its valuation is $(3,4) + (-2,-4) + (50,5) = (1,0) + (50,5) = (51,5)$. This linear path is admissible for every valuation ν. The linear path

$$P_3 \xrightarrow{C_4} P_2 \xrightarrow{\langle C_3; P_1(50,5)\rangle} P_3$$

which is a suffix of the previous one, has valuation $(-2,-4) + (50,5) = (48,1)$, but is admissible for ν only when $\nu \geq (2,4)$.

We require the following auxiliary lemma to prove Proposition 2, which is the most crucial result of our discussion on branching VASS.

Lemma 1 *Let $\mathcal{V}$ be a branching VASS. Let Δ be a covering derivation with the property that, given any generalized fact $P(\nu')$ occurring in this derivation, we can find a (non-generalized) fact ν such that*

- *$P(\nu)$ is derivable from $\mathcal{V}$,*
- *and for every i, if $\nu'[i] \neq \infty$, then $\nu[i] = \nu'[i]$.*

Suppose that Δ_2 is a subderivation of $P_2(\nu'_2)$ in Δ of the following form, containing a (not necessarily proper) subderivation Δ_1 (of $P_1(\nu'_1)$).

$$\begin{array}{c} \vdots\,\Delta_1 \\ P_1(\nu'_1) \\ \vdots \\ P_2(\nu'_2) \end{array}$$

Let J be a subset of $\{1,\ldots,p\}$ such that $\nu'_2[i] \neq \infty$ for all $i \in J$. Then we can find a linear path π from P_1 to P_2 such that π is admissible for ν'_1 with respect to J, and $\nu'_1[J] + v(\pi)[J] = \nu'_2[J]$.

Proof: We induct on the distance (number of steps) between $P_1(\nu'_1)$ and $P_2(\nu'_2)$ in Δ.

Suppose the distance is 0, i.e., $\Delta_2 = \Delta_1$, in particular $P_2(\nu'_2) = P_1(\nu'_1)$. Then the trivial linear path P_1 suffices. Otherwise, look at the last rule used in Δ_2.

- If Δ_2 is of the form

$$\begin{array}{c} \vdots\,\Delta_1 \\ P_1(\nu'_1) \\ \vdots \\ \dfrac{P_3(\nu'_3)}{P_2(\nu'_2)}\; P_2(x+\delta) \Leftarrow P_3(x) \end{array}$$

  Clearly $\nu'_3[J] + \delta[J] = \nu'_2[J]$. Also, for all $i \in J$, $\nu'_3[i] \neq \infty$. By induction hypothesis there is a linear path π' from P_1 to P_3 which is admissible for ν'_1 with respect to J, and such that $\nu'_1[J] + v(\pi')[J] = \nu'_3[J]$. Let the required linear path π be the concatenation of π' with $P_3 \xrightarrow{P_2(x+\delta)\Leftarrow P_3(x)} P_2$. We have $\nu'_1[J] + v(\pi)[J] = \nu'_1[J] + v(\pi')[J] + \delta[J] = \nu'_3[J] + \delta[J] = \nu'_2[J]$. In particular $\nu'_1[J] + v(\pi)[J] \geq 0$ and hence π is admissible for ν'_1 with respect to J.

- If Δ_2 is of the form

$$\begin{array}{c} \begin{array}{cc} \begin{array}{c} \vdots\,\Delta_1 \\ P_1(\nu'_1) \\ \vdots \\ P_3(\nu'_3) \end{array} & \begin{array}{c} \vdots\,\Delta_4 \\ P_4(\nu'_4) \end{array} \end{array} \\ \hline P_2(\nu'_2) \end{array}\; P_2(x+y) \Leftarrow P_3(x), P_4(y)$$

  where, without loss of generality, Δ_1 is contained in the subderivation leading to the left premise. Clearly $\nu'_2[J] = \nu'_3[J] + \nu'_4[J]$. Also for all $i \in J$, $\nu'_3[i] \neq \infty$ and $\nu'_4[i] \neq \infty$. By induction hypothesis we get a linear path π' from P_1 to P_3 which is admissible for ν'_1 with respect to J, and such that $\nu'_1[J] + v(\pi')[J] = \nu'_3[J]$. By assumption (this is where we need it!), $P(\nu_4)$ is derivable from $\mathcal{V}$ for some vector ν_4 such that, for all i such that $\nu'_4[i] \neq \infty$, $\nu_4[i] = \nu'_4[i]$. In particular $\nu_4[J] = \nu'_4[J]$. Let the required linear path π be the concatenation of π' with $P_3 \xrightarrow{\langle P_2(x+y) \Leftarrow P_3(x), P_4(x); P_4(\nu_4)\rangle} P_2$. This is a well-defined linear path. We have $\nu'_1[J] + v(\pi)[J] = \nu'_1[J] + v(\pi')[J] + \nu_4[J] = \nu'_3[J] + \nu_4[J] = \nu'_3[J] + \nu'_4[J] = \nu'_2[J]$. In particular $\nu'_1[J] + v(\pi)[J] \geq 0$ and hence π is admissible for ν'_1 with respect to J. $\square$

Example 3 *Let us look at Example 1 again. Looking at Figure 1, take Δ_2 to be the whole covering derivation (of $P_1(\infty,5)$), and Δ_1 to be the one on the left ending on $P_3(5,9)$. Take $J = \{2\}$.*

The linear path

$$P_3 \xrightarrow{C_4} P_1 \xrightarrow{\langle C_3; P_2(3,4)\rangle} P_3 \xrightarrow{C_4} P_1$$

fits the bill. The corresponding facts in a derivation in the BVASS are $P_3(5,9)$, $P_1(3,5)$, $P_3(6,9)$ and $P_1(4,5)$. This linear path is probably the one you expected; another one is

$$P_3 \xrightarrow{C_4} P_1 \xrightarrow{\langle C_3; P_2(3,4)\rangle} P_3 \xrightarrow{C_4} P_1 \xrightarrow{\langle C_3; P_2(7,4)\rangle} P_3 \xrightarrow{C_4} P_1$$

with the corresponding facts $P_3(5,9)$, $P_1(3,5)$, $P_3(6,9)$, $P_1(4,5)$, $P_3(11,9)$ and $P_1(9,5)$. The latter is meant to dispel the wrong intuition that linear paths (even admissible ones) should just correspond in some way to paths inside the covering derivation: this one jumps back from the P_3 node $P_3(\infty,9)$ to the P_1 node just above ($P_1(\infty,5)$). A similar phenomenon occurs in ordinary Karp-Miller trees [Reu89]. The new thing here is that linear paths can actually jump in a different *branch. For example, the following linear path starts from the upper-left P_3 node, goes down to the bottom P_3, jumps back to the P_2 node just above, then goes down again to the bottom P_3 and to P_1.*

$$P_3 \xrightarrow{C_4} P_1 \xrightarrow{\langle C_3; P_2(3,4)\rangle} P_3 \xrightarrow{C_5} P_2 \xrightarrow{\langle C_3; P_1(2,5)\rangle} P_3 \xrightarrow{C_4} P_1$$

The corresponding facts in this case are $P_3(5,9)$, $P_1(3,5)$, $P_3(6,9)$, $P_2(8,4)$, $P_3(10,9)$ and $P_1(8,5)$. We let the reader ponder about these examples.

Now we are ready to prove the required result:

Proposition 2 *Let $\mathcal{V}$ be a branching VASS. For every covering derivation Δ of some generalized fact $P(\nu')$, and for any $K \geq 0$ there is a tuple $\nu \in \mathbb{N}^p$ such that*

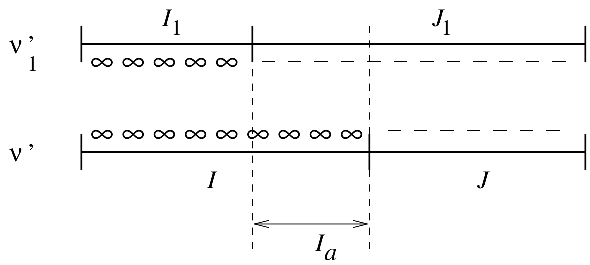
- *for every i such that $\nu'[i] = \infty$, $\nu[i] \geq K$;*
- *for every i such that $\nu'[i] \neq \infty$, $\nu[i] = \nu'[i]$;*
- *$P(\nu)$ is derivable from $\mathcal{V}$.*

Proof: By induction on Δ. If $P(\nu')$ is a fact in $\mathcal{V}$ (rule 1 of Definition 1), then $\nu = \nu'$ satisfies the requirements. Otherwise, look at the last rule in Δ:

- If Δ is constructed using rule 2 of Definition 1:

$$\dfrac{\begin{array}{c}\vdots\;\Delta_1\\ P_1(\nu'_1)\end{array}}{P(\nu')}\;P(x+\delta)\Leftarrow P_1(x)$$

then let I be the set of indices i such that $\nu'[i]=\infty$, J be the set of all other indices, let I_1 be the set of indices i such that $\nu'_1[i]=\infty$ and J_1 the set of all other indices. Clearly $I_1\subseteq I$, hence $J\subseteq J_1$; let I_a be the set $I\setminus I_1$ of additional indices that are infinite in I compared to I_1. The following picture may help:



In the sequel, for any integer N, write $\overline{N}$ the vector $(N,N,\ldots,N)$. The number of components in $\overline{N}$ will always be clear from context.

Buildings paths π_i for all $i\in I_a$. By definition of I_a, for each $i\in I_a$ there is a subderivation Δ^i of Δ_1 that derives $P(\nu'^i)$ for some generalized fact ν'^i such that: **(a)** $\nu'^i\leq\nu'_1+\delta$ and: **(b)** $\nu'^i[i]<\nu'_1[i]+\delta[i]$. By induction hypothesis the assumptions of Lemma 1 are satisfied of Δ_1 and the set of indices J_1. From this lemma we get a linear path π'_i from P to P_1 which is admissible for ν'^i with respect to J_1 and such that: **(c)** $\nu'^i[J_1]+v(\pi'_i)[J_1]=\nu'_1[J_1]$. Let π_i be the path from P to P obtained by concatenating π'_i with $P_1\xrightarrow{P(x+\delta)\Leftarrow P_1(x)}P$.

$$\dfrac{\begin{array}{c}\vdots\;\Delta^i\\ P(\nu'^i)\\ \vdots\\ P_1(\nu'_1)\end{array}}{P(\nu')}\;P(x+\delta)\Leftarrow P_1(x)\qquad\qquad \pi_i=P\longrightarrow\ldots\pi'_i\ldots\longrightarrow P_1\xrightarrow{\;P(x+\delta)\Leftarrow P_1(x)\;}P$$

Then: **(d)** $\nu'^i[J_1]+v(\pi_i)[J_1]=\nu'_1[J_1]+\delta[J_1]$. Indeed $\nu'^i[J_1]+v(\pi_i)[J_1]=\nu'^i[J_1]+v(\pi'_i)[J_1]+\delta[J_1]=\nu'_1[J_1]+\delta[J_1]$ by (c). By (d), and since $\nu'_1+\delta\geq\overline{0}$, it follows that $\nu'^i[J_1]+v(\pi_i)[J_1]\geq\overline{0}$. Hence π_i is admissible for ν'^i with respect to J_1. By (a) above, $\nu'^i\leq\nu'_1+\delta$, so π_i is admissible for $\nu'_1+\delta$ with respect to J_1.

We now observe that: **(e)** for every $j\in J_1$, $\nu'^i[j]$ is finite. Indeed, otherwise, by (a) $\nu'_1[j]$ would be infinite, contradicting the fact that j is in J_1.

We also observe that: **(f)** $v(\pi_i)[J] = \overline{0}$. Indeed, by (a) $\nu'^i \leq \nu'_1 + \delta$, and if we had $\nu'^i[j] < \nu'_1[j] + \delta[j]$ for some $j \in J$, then by definition of rule 2 of Definition 1 $\nu'[j]$ would be ∞, contradicting the fact that $j \in J$. So $\nu'^i[j] = \nu'_1[j] + \delta[j]$ for all $j \in J$, i.e., $\nu'^i[J] = \nu'_1[J] + \delta[J]$. By (d) and since $J \subseteq J_1$, $\nu'^i[J] + v(\pi_i)[J] = \nu'_1[J] + \delta[J]$, so $\nu'^i[J] + v(\pi_i)[J] = \nu'^i[J]$. But $\nu'^i[j]$ is finite for every $j \in J \subseteq J_1$ by (e), so $v(\pi_i)[J] = \overline{0}$, as claimed.

Next: **(g)** $v(\pi_i)[J_1] \geq \overline{0}$. The argument is similar. By (a) $\nu'^i \leq \nu'_1 + \delta$, so $\nu'^i[J_1] \leq \nu'_1[J_1] + \delta[J_1]$. Using (d), $\nu'^i[J_1] + v(\pi_i)[J_1] \geq \nu'^i[J_1]$. Claim (g) follows, since $\nu'^i[j]$ is finite for every $j \in J_1$ by (e).

Finally: **(h)** $v(\pi_i)[i] \geq 1$. Indeed, since i is in $I_a = I \setminus I_1 = I \cap J_1$, i is in J_1, so (d) applies, hence $\nu'^i[i] + v(\pi_i)[i] = \nu'_1[i] + \delta[i]$. By (b) $\nu'^i[i] < \nu'_1[i] + \delta[i]$, so $\nu'^i[i] + v(\pi_i)[i] > \nu'^i[i]$. Since $\nu'^i[i]$ is finite by (e) and the fact that $i \in J_1$, (h) obtains.

Building the path π. Let π be the concatenation of all linear paths π_i when i ranges over I_a, in any order. Since each π_i is a linear path from P to P, π is well-defined and is a linear path from P to P, too. Since $v(\pi_i)[J_1] \geq \overline{0}$ by (g), and each π_i is admissible for $\nu'_1 + \delta$ with respect to J_1, it is easy to see that π is admissible for $\nu'_1 + \delta$ with respect to J_1. Then, by (f), $v(\pi)[J] = \overline{0}$. Since $v(\pi_i)[i] \geq 1$ by (h) for all $i \in I_a$ and $v(\pi_i)[I_a] \geq \overline{0}$ by (g) and the fact that $I_a \subseteq J_1$, we obtain that $v(\pi)[I_a] \geq \overline{1}$.

Letting π^K be the concatenation of K copies of π, we can again see that the path π^K is admissible for $\nu'_1 + \delta$ with respect to J_1, $v(\pi^K)[J] = \overline{0}$ and $v(\pi^K)[I_a] \geq \overline{K}$. Choose some $K_1 \in \mathbb{N}$ such that for each prefix π' of π^K, $\overline{K_1} + v(\pi')[I_1] \geq \overline{0}$ and $\overline{K_1} + v(\pi^K)[I_1] \geq \overline{K}$. Choose $K_2 \in \mathbb{N}$ such that $\overline{K_2} + \delta[I_1] \geq \overline{K_1}$. By induction hypothesis there is a vector ν_1 such that $P_1(\nu_1)$ is derivable from $\mathcal{V}$, $\nu_1[I_1] \geq \overline{K_2}$, and $\nu_1[J_1] = \nu'_1[J_1]$.

Since $\nu'_1 + \delta \geq 0$ (by the simple fact that rule 2 of Definition 1 was applicable at all) and $\nu_1[J_1] = \nu'_1[J_1]$, we obtain $\nu_1[J_1] + \delta[J_1] \geq 0$. On the other hand, $\nu_1[I_1] + \delta[I_1] \geq \overline{K_2} + \delta[I_1] \geq \overline{K_1} \geq 0$. So $\nu_1 + \delta \geq 0$, therefore $P(\nu_1 + \delta)$ is derivable from $P_1(\nu_1)$ and the transition $P(x+\delta) \Leftarrow P_1(x)$.

Since $\nu_1[I_1] \geq \overline{K_2}$ and $\overline{K_2} + \delta[I_1] \geq \overline{K_1}$, $\nu_1[I_1] + \delta[I_1] \geq \overline{K_1}$. By the definition of K_1, this entails that $\nu_1[I_1] + \delta[I_1] + v(\pi')[I_1] \geq \overline{0}$ for every prefix π' of π^K, so π^K is admissible for $\nu_1 + \delta$ with respect to I_1.

Since $\nu_1[J_1] = \nu'_1[J_1]$ and π^K is admissible for $\nu'_1 + \delta$ with respect to J_1, π^K is also admissible for $\nu_1 + \delta$ with respect to J_1.

Since $I_1 \cup J_1$ is the set of all indices $\{1, \ldots, p\}$, and π^K is admissible for $\nu_1 + \delta$ with respect to both I_1 and J_1, π^K is admissible for $\nu_1 + \delta$.

This implies that $P(\nu_1 + \delta + v(\pi^K))$ is derivable from $\mathcal{V}$.

Building the fact ν from π. Let therefore ν be $\nu_1 + \delta + v(\pi^K)$. It remains to show that $\nu[I] \geq \overline{K}$ and $\nu[J] = \nu'[J]$.

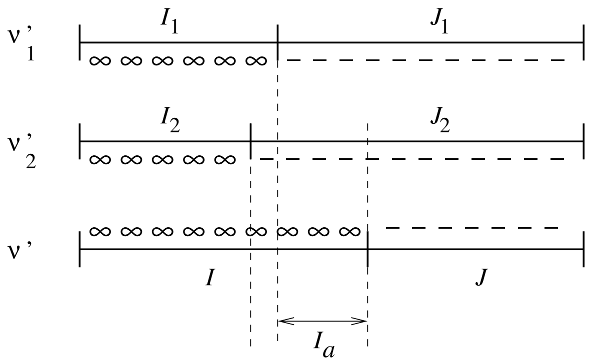
The second claim follows from the fact that $v(\pi)[J] = \overline{0}$, from which we infer $v(\pi^K)[J] = \overline{0}$, hence $\nu[J] = \nu_1[J] + \delta[J] + v(\pi^K)[J] = \nu_1[J] + \delta[J] = \nu'_1[J] + \delta[J]$ (since $\nu_1[J_1] = \nu'_1[J_1]$ and $J \subseteq J_1$) $= \nu'[J]$ (since $\nu'[J]$ has only finite components, and by definition finite components of ν' are sums of corresponding components of ν'_1 and δ).

The first claim will follow from the two sub-claims $\nu[I_1] \geq \overline{K}$ and $\nu[I_a] \geq \overline{K}$, since $I = I_1 \cup I_a$. For the first sub-claim, recall that $\nu_1[I_1] \geq \overline{K_2}$ and $\overline{K_2} + \delta[I_1] \geq \overline{K_1}$, so $\nu_1[I_1] + \delta[I_1] \geq \overline{K_1}$, and by the definition of K_1, this entails that $\nu_1[I_1] + \delta[I_1] + v(\pi^K)[I_1] \geq \overline{K}$, i.e., $\nu[I_1] \geq \overline{K}$. For the second sub-claim, recall that $v(\pi^K)[I_a] \geq \overline{K}$, and $\nu_1[J_1] = \nu'_1[J_1]$ (in particular, $\nu_1[I_a] = \nu'_1[I_a]$, since $I_a \subseteq J_1$); so $\nu[I_a] = \nu_1[I_a] + \delta[I_a] + v(\pi^K)[I_a] \geq \nu'_1[I_a] + \delta[I_a] + \overline{K} \geq \overline{K}$, since $\nu'_1 + \delta \geq \overline{0}$.

- Suppose Δ is constructed using rule 3 of Definition 1:

$$\dfrac{\begin{matrix} \vdots\,\Delta_1 & \vdots\,\Delta_2 \\ P_1(\nu'_1) & P_2(\nu'_2) \end{matrix}}{P(\nu')}\; P(x+y) \Leftarrow P_1(x), P_2(y)$$

then let I be the set of indices i such that $\nu'[i] = \infty$, J its complement, I_1 the set of indices i such that $\nu'_1[i] = \infty$, J_1 its complement, I_2 the set of indices i such that $\nu'_2[i] = \infty$, and J_2 its complement. Let I_a be $I \setminus (I_1 \cup I_2)$. The picture is now:



Buildings paths π_i for all $i \in I_a$. By induction hypothesis there are vectors ν_1 and ν_2 such that $P_1(\nu_1)$ and $P_2(\nu_2)$ are derivable, $\nu_1[J_1] = \nu'_1[J_1]$ and $\nu_2[J_2] = \nu'_2[J_2]$.

As in the previous case, define a linear path π_i for each $i \in I_a$, as follows. Contrarily to the previous case, π_i will depend on ν_1 and ν_2.

Since $i \in I_a$, there is a subderivation Δ^i of Δ_{b_i}, for some $b_i \in \{1,2\}$, that derives $P(\nu'^i)$ for some generalized fact ν'^i such that: **(a)** $\nu'^i \leq \nu'_1 + \nu'_2$, and: **(b)** $\nu'^i[i] < \nu'_1[i] + \nu'_2[i]$. By Lemma 1 there is a linear path π'_i from P to P_{b_i} which is admissible for ν'^i with respect to J_{b_i} and such that: **(c)** $\nu'^i[J_{b_i}] + v(\pi'_i)[J_{b_i}] = \nu'_{b_i}[J_{b_i}]$. Let $\overline{b}_i$ be 2 if $b_i = 1$, and 1 if $b_i = 2$. Then let π_i be the path from P to P obtained by concatenating π'_i with $P_{b_i} \xrightarrow{\langle C;\delta \rangle} P$, where C is the clause $P(x+y) \Leftarrow P_1(x), P_2(y)$, and $\delta = \nu_{\overline{b}_i}$.

$$\dfrac{\begin{matrix} \vdots\,\Delta^i & \\ P(\nu'^i) & \\ \vdots & \\ P_{b_i}(\nu'_{b_i}) & P_{\overline{b_i}}(\nu'_{\overline{b_i}}) \end{matrix}}{P(\nu')}\; P(x+y) \Leftarrow P_1(x), P_2(y) \qquad \pi_i = P \longrightarrow \ldots \pi'_i \ldots \longrightarrow P_{b_i} \xrightarrow{\langle P(x+y) \Leftarrow P_1(x), P_2(y); \nu_{\overline{b_i}} \rangle} P$$

The reasoning is then similar to the previous case; there are many slight differences, though.

We first show that: **(d)** $\nu'^{i}[J_{b_i}] + v(\pi_i)[J_{b_i}] = \nu'_{b_i}[J_{b_i}] + \delta[J_{b_i}]$. This is by (c).

By (d), and since $\nu'_{b_i} + \delta \geq \overline{0}$ as a sum of vectors in $\mathbb{N}^p$, we have $\nu'^{i}[J_{b_i}] + v(\pi_i)[J_{b_i}] \geq \overline{0}$. Hence π_i is admissible for ν'^{i} with respect to J_{b_i}. By (a), $\nu'^{i} \leq \nu'_1 + \nu'_2$, so π_i is admissible for $\nu'_1 + \nu'_2$ with respect to J_{b_i}.

We now have: **(e)** for every $j \in J_1 \cap J_2$, $\nu'^{i}[j]$ is finite. This is because by (a) $\nu'^{i} \leq \nu'_1 + \nu'_2$, and $\nu'_1[j] \neq \infty$ since $j \in J_1$, $\nu'_2[j] \neq \infty$ since $j \in J_2$.

Then: **(f)** $v(\pi_i)[J] = \overline{0}$. Indeed by (a) $\nu'^{i} \leq \nu'_1 + \nu'_2$. For every $j \in J$, by the definition of covering derivation $\nu'[j]$ would be infinite if $\nu'_i[j] < \nu'_1[j] + \nu'_2[j]$, and this would contradict the fact that j is in J. So $\nu'^{i}[J] = \nu'_1[J] + \nu'_2[J]$. Since $J \subseteq J_{\overline{b}_i}$, $\delta = \nu_{\overline{b}_i}$ and $\nu_{\overline{b}_i}[J_{\overline{b}_i}] = \nu'_{\overline{b}_i}[J_{\overline{b}_i}]$ by definition of $J_{\overline{b}_i}$, $\delta[J] = \nu'_{\overline{b}_i}[J]$. Using (d) and $J \subseteq J_{b_i}$, we obtain $\nu'^{i}[J] + v(\pi_i)[J] = \nu'_{b_i}[J] + \delta[J] = \nu'_{b_i}[J] + \nu'_{\overline{b}_i}[J] = \nu'_1[J] + \nu'_2[J] = \nu'^{i}[J]$. Since $J \subseteq J_1 \cap J_2$, by (e) $\nu'^{i}[j]$ is finite for every $j \in J$, so (f) obtains.

Next: **(g)** $v(\pi_i)[J_1 \cap J_2] \geq \overline{0}$. Indeed, since $\delta \in \mathbb{N}^p$, by (d) $\nu'^{i}[J_{b_i}] + v(\pi_i)[J_{b_i}] \geq \nu'^{i}[J_{b_i}]$. So $\nu'^{i}[J_1 \cap J_2] + v(\pi_i)[J_1 \cap J_2] \geq \nu'^{i}[J_1 \cap J_2]$. Claim (g) follows, since $\nu'^{i}[j]$ is finite for every $j \in J_1 \cap J_2$ by (e).

Finally: **(h)** $v(\pi_i)[i] \geq 1$. Indeed, by (b) $\nu'^{i}[i] < \nu'_{b_i}[i] + \nu'_{\overline{b}_i}[i]$. Since $i \in I_a \subseteq J_{\overline{b}_i}$, $\nu'_{\overline{b}_i}[i]$ is finite, hence equal to $\nu_{\overline{b}_i}[i] = \delta[i]$, so $\nu'^{i}[i] < \nu'_{b_i}[i] + \delta[i]$. Since $i \in I_a \subseteq J_{b_i}$, (d) applies, hence $\nu'^{i}[i] + v(\pi_i)[i] = \nu'_{b_i}[i] + \delta[i] > \nu'^{i}[i]$. Since $\nu'^{i}[i]$ is finite by (e) and the fact that $i \in J_1 \cap J_2$, (h) obtains.

Building the path π. Let π be the concatenation of all linear paths π_i when i ranges over I_a, in any order. This is a linear path from P to P, too. Since $v(\pi_i)[J_1 \cap J_2] \geq \overline{0}$ by (g), and each π_i is admissible for $\nu'_1 + \nu'_2$ with respect to J_{b_i}, it is easy to see that π is admissible for $\nu'_1 + \nu'_2$ with respect to $J_1 \cap J_2$. Then, by (f), $v(\pi)[J] = \overline{0}$. Since $v(\pi_i)[i] \geq 1$ by (h) for all $i \in I_a$ and $v(\pi_i)[I_a] \geq \overline{0}$ by (g) and the fact that $I_a \subseteq J_1 \cap J_2$, we obtain that $v(\pi)[I_a] \geq \overline{1}$.

Letting π^K be the concatenation of K copies of π, we can again see that the path π^K is admissible for $\nu'_1 + \nu'_2$ with respect to $J_1 \cap J_2$, $v(\pi^K)[J] = \overline{0}$ and $v(\pi^K)[I_a] \geq \overline{K}$. Choose $K_1 \in \mathbb{N}$ so that for each prefix π' of π^K, $\overline{K_1} + v(\pi')[I_1 \cup I_2] \geq \overline{0}$ and $\overline{K_1} + v(\pi^K)[I_1 \cup I_2] \geq \overline{K}$.

Recall we have already obtained vectors ν_1 and ν_2 such that $P_1(\nu_1)$ and $P_2(\nu_2)$ are derivable, $\nu_1[J_1] = \nu'_1[J_1]$ and $\nu_2[J_2] = \nu'_2[J_2]$, using the induction hypothesis. We would like to enforce $\nu_1[I_1] \geq \overline{K_1}$ and $\nu_2[I_2] \geq \overline{K_1}$ to proceed. However the constant K_1 depends on π, which depends on π_i, $i \in I_a$, which we built by concatenating π'_i with some edge labeled by $\nu_{\overline{b}_i}$: each new choice of ν_1, ν_2 may lead to a different constant K_1, so that we cannot assume that $\nu_1[I_1] \geq \overline{K_1}$ and $\nu_2[I_2] \geq \overline{K_1}$. Instead, we invoke the induction hypothesis again, and find two new vectors ν_1^1 and ν_2^1 such that $P_1(\nu_1^1)$ and $P_2(\nu_2^1)$ are derivable, $\nu_1^1[J_1] = \nu'_1[J_1]$ and $\nu_1^1[I_1] \geq \overline{K_1}$, and $\nu_2^1[J_2] = \nu'_2[J_2]$ and $\nu_2^1[I_2] \geq \overline{K_1}$.

Clearly $P(\nu_1^1 + \nu_2^1)$ is derivable from $P_1(\nu_1^1)$, $P_2(\nu_2^1)$, and the clause $P(x+y) \Leftarrow P_1(x), P_2(y)$.

Since $\nu_1^1[I_1] \geq \overline{K_1}$ and $\nu_2^1[I_2] \geq \overline{K_1}$, we infer $\nu_1^1[I_1 \cup I_2] + \nu_2^1[I_1 \cup I_2] \geq \overline{K_1}$. By the definition of K_1, for every prefix π' of π^K, $\nu_1^1[I_1 \cup I_2] + \nu_2^1[I_1 \cup I_2] + v(\pi')[I_1 \cup I_2] \geq 0$, so π^K is admissible for $\nu_1^1 + \nu_2^1$ with respect to $I_1 \cup I_2$.

Since $\nu_1^1[J_1] = \nu_1'[J_1]$, $\nu_2^1[J_2] = \nu_2'[J_2]$, and π^K is admissible for $\nu_1' + \nu_2'$ with respect to $J_1 \cap J_2$, π^K is also admissible for $\nu_1^1 + \nu_2^1$ with respect to $J_1 \cap J_2$.

Since the union of $I_1 \cup I_2$ and $J_1 \cap J_2$ is the set of all indices $\{1, \dots, p\}$, it follows that π^K is admissible for $\nu_1^1 + \nu_2^1$.

This implies that $P(\nu_1^1 + \nu_2^1 + v(\pi^K))$ is derivable from $\mathcal{V}$.

Building the fact ν from π. Let therefore ν be $\nu_1^1 + \nu_2^1 + v(\pi^K)$. It remains to show that $\nu[I] \geq \overline{K}$ and $\nu[J] = \nu'[J]$.

The second claim follows from the fact that $v(\pi)[J] = \overline{0}$, from which we infer $v(\pi^K)[J] = \overline{0}$, hence $\nu[J] = \nu_1^1[J] + \nu_2^1[J] + v(\pi^K)[J] = \nu_1^1[J] + \nu_2^1[J] = \nu_1'[J] + \nu_2'[J]$ (since $\nu_1^1[J_1] = \nu_1'[J_1]$ and $J \subseteq J_1$, and $\nu_2^1[J_2] = \nu_2'[J_2]$ and $J \subseteq J_2$) $= \nu'[J]$ (since $\nu'[J]$ has only finite components, and by definition finite components of ν' are sums of corresponding components of ν_1' and ν_2').

The first claim will follow from the two sub-claims $\nu[I_1 \cup I_2] \geq \overline{K}$ and $\nu[I_a] \geq \overline{K}$, since $I = I_1 \cup I_2 \cup I_a$. For the first sub-claim, recall that $\nu_1^1[I_1 \cup I_2] + \nu_2^1[I_1 \cup I_2] \geq \overline{K_1}$. By the definition of K_1, $\nu_1^1[I_1 \cup I_2] + \nu_2^1[I_1 \cup I_2] + v(\pi^K)[I_1 \cup I_2] \geq \overline{K}$, i.e., $\nu[I_1 \cup I_2] \geq \overline{K}$. For the second sub-claim, recall that $v(\pi^K)[I_a] \geq \overline{K}$, and $\nu_1^1[J_1] = \nu_1'[J_1]$ (in particular, $\nu_1^1[I_a] = \nu_1'[I_a]$, since $I_a \subseteq J_1$), and $\nu_2^1[J_2] = \nu_2'[J_2]$ (so $\nu_2^1[I_a] = \nu_2'[I_a]$, since $I_a \subseteq J_2$); so $\nu[I_a] = \nu_1^1[I_a] + \nu_2^1[I_a] + v(\pi^K)[I_a] \geq \nu_1'[I_a] + \nu_2'[I_a] + \overline{K} \geq \overline{K}$, since $\nu_1' + \nu_2' = \nu' \geq \overline{0}$. □

3 Termination

Now we are left to prove that there are only finitely many covering derivations. This is Theorem 1 below.

Remark 1 *Let $\mathcal{V}$ be a branching VASS, Δ a covering derivation. Suppose that Δ_2 is a subderivation of Δ of the form*

$$\begin{array}{c} \vdots\, \Delta_1 \\ P_1(\nu_1') \\ \vdots \\ P_2(\nu_2') \end{array}$$

containing a (not necessarily proper) subderivation Δ_1. For any index i, if $\nu_1'[i] = \infty$, then $\nu_2'[i] = \infty$. We say that ν_2' has at least as many infinite coordinates *as ν_1'. In case additionally $\nu_1'[i] \neq \infty$ and $\nu_2'[i] = \infty$ for some i, we say that ν_2' has* more infinite coordinates *than ν_1'. Otherwise, ν_1' and ν_2' have* the same infinite coordinates.

Lemma 2 *Let $\mathcal{V}$ be a branching VASS. Let Δ be a covering derivation of the form*

$$\begin{array}{c} \vdots\, \Delta_0 \\ P(\nu_0') \\ \vdots \\ P(\nu') \end{array}$$

with the same P, and such that $\nu' > \nu_0'$, by which we mean that $\nu'[i] \geq \nu_0'[i]$ for all i and $\nu'[i] > \nu_0'[i]$ for some i. Then ν' has more infinite coordinates than ν_0'.

Proof: Assume the contrary. From Remark 1, ν' and ν'_0 have the same infinite coordinates. Clearly Δ must have been constructed by using Rule 2 or Rule 3 of Definition 1. Accordingly we have the following two cases:

- Δ is of the form

$$\dfrac{\begin{array}{c}\vdots\,\Delta_1\\ P_1(\nu'_1)\end{array}}{P(\nu')}\; P(x+\delta) \Leftarrow P_1(x)$$

  Again from Remark 1, ν'_1 has the same infinite coordinates as ν'. Then from the construction in Rule 2 of Definition 1, we must have $\nu' = \nu'_1 + \delta$. Since $\nu' > \nu'_0$, it follows that $\nu'_1 + \delta \geq \nu'_0$ and for some i, $\nu'_1[i] + \delta[i] > \nu'_0[i]$. But Rule 2 of Definition 1 entails that $\nu'[i] = \infty$, while $\nu'_0[i] \neq \infty$ since $\nu'_0[i] < \nu'_1[i] + \delta[i]$, contradicting the fact that ν' and ν'_0 have the same infinite coordinates.

- Δ is of the form

$$\dfrac{\begin{array}{c}\vdots\,\Delta_1\\ P_1(\nu'_1)\end{array}\quad\begin{array}{c}\vdots\,\Delta_2\\ P_2(\nu'_2)\end{array}}{P(\nu')}\; P(x+y) \Leftarrow P_1(x), P_2(y)$$

  where we may assume without loss of generality that Δ_0 is a subderivation of Δ_1. From Remark 1, ν' and ν'_0 have the same infinite coordinates. Then from the construction in rule 3 of Definition 1, $\nu' = \nu'_1 + \nu'_2$. Since $\nu' > \nu'_0$, it follows that $\nu'_1 + \nu'_2 \geq \nu'_0$ and $\nu'_1[i] + \nu'_2[i] > \nu'_0[i]$ for some i, so by rule 3 of Definition 1 $\nu'[i] = \infty$. As above, $\nu'_0[i] \neq \infty$, leading to a contradiction. □

Define the height $H(\Delta)$ of the covering derivation Δ as the height of the corresponding tree. Formally:

Definition 4 (Height) *Taking the notations of Definition 1, the* height $H(\Delta)$ *of the covering derivation* Δ *is* 1 *if* Δ *was created by rule 1,* $1 + H(\Delta_1)$ *if by rule 2, and* $1 + \max(H(\Delta_1), H(\Delta_2))$ *if by rule 3.*

Given a branching VASS $\mathcal{V}$, we define a total ordering on covering derivations, based on height, as follows. For every $n \geq 1$, there are only finitely many, say k_n, covering derivations of height n. Let us enumerate them without repetition, arbitrarily as $\Delta_{n1}, \ldots, \Delta_{nk_n}$. Then define $\sqsubseteq$ by $\Delta_{mi} \sqsubseteq \Delta_{nj}$ if and only if $m < n$, or $m = n$ and $i \leq j$.

Lemma 3 *For a branching VASS $\mathcal{V}$, the total ordering $\sqsubseteq$ on covering derivations has the properties that:*

1. *if $H(\Delta_1) < H(\Delta_2)$ then $\Delta_1 \sqsubseteq \Delta_2$;*
2. *given any covering derivation Δ, there are only finitely many covering derivations Δ_1 such that $\Delta_1 \sqsubseteq \Delta$.*

Theorem 1 *Every branching VASS $\mathcal{V}$ has only finitely many covering derivations. Furthermore, the set of covering derivations of $\mathcal{V}$ can be effectively computed.*

Proof: We will construct a forest of all possible derivations; be aware that each node in this forest will be a whole derivation, not just a fact. Recall also that a forest is just a set of trees, which we shall call the *component trees* of the forest. This forest is constructed iteratively by adding one node at a time, using the following rules:

1. Each covering derivation constructed using rule 1 of Definition 1 is a root node. These are the only root nodes, so that there will be only finitely many component trees in the forest.

2. Suppose Δ is constructed using the derivation Δ_1 as defined in Rule 2 of Definition 1, and Δ_1 has been added in the forest, and Δ has not been added. Then we add Δ as a child of Δ_1.

3. Suppose Δ_1 and Δ_2 have been added in the forest, and Δ is constructed using them as defined in Rule 3 of Definition 1 and has not been added in the forest. Since $\sqsubseteq$ is total (see Lemma 3), either $\Delta_1 \sqsubseteq \Delta_2$ or $\Delta_2 \sqsubseteq \Delta_1$. Let Δ' be the greater of Δ_1 and Δ_2 in $\sqsubseteq$. Then we add Δ to the forest as a child of Δ'.

   This is the crux of the proof: by choosing exactly one of Δ_1, Δ_2 here, each node will have at most one parent, so we are indeed building a forest (not a jungle). The particular choice of Δ' ensures that the forest is finitely branching; this used to be trivial in the case of VASS.

It is clear that in this way all the derivations are added in the forest eventually. We claim that this process ends, i.e., the forest is finite. Assume the contrary. Since the number of component trees is finite, one of them would be infinite.

We see that the component trees are finitely branching. For any covering derivation Δ, the number of children created using Rule 2 above is limited by the number of clauses, and the number of children created using Rule 3 above is limited by the number of clauses times the finite number of covering derivations Δ_1 such that $\Delta_1 \sqsubseteq \Delta$. Then by König's lemma, there would be an infinite path in the component tree, consisting of covering derivations

$$\begin{array}{ccccc} \vdots\,\Delta_1 & \vdots\,\Delta_2 & & \vdots\,\Delta_k & \\ P_1(\nu'_1) & P_2(\nu'_2) & \dots & P_k(\nu'_k) & \dots \end{array}$$

By construction, for every $k \geq 1$, $P_k(\nu'_k)$ is one of the premises of the last rule used in deriving $P_{k+1}(\nu'_{k+1})$ in Δ_{k+1}. In particular, whenever $k < k'$, Δ_k occurs as a subderivation in $\Delta_{k'}$.

Since there are only finitely many predicate symbols, by the pigeonhole principle there must be some predicate symbol P such that there is an infinite subsequence $P(\nu'_{i_1}), P(\nu'_{i_2}), \dots, P(\nu'_{i_k}), \dots$, with $1 \leq i_1 < i_2 < \dots < i_k < \dots$ Since the ordering $\leq$ on $(\mathbb{N} \cup \{\infty\})^p$ is a well-quasi ordering (e.g., by a trivial extension of Dickson's Lemma), there is an infinite subsequence of the latter, say $P(\nu'_{j_1}), P(\nu'_{j_2}), \dots, P(\nu'_{j_k}), \dots$, with $1 \leq j_1 < j_2 < \dots < j_k < \dots$, such that $\nu'_{j_1} \leq \nu'_{j_2} \leq \dots \leq \nu'_{j_k} \leq \dots$ This sequence cannot stabilize, that is, there is no $k \geq 1$ such that $\nu'_{j_k} = \nu'_{j_{k+1}}$. Otherwise $\Delta_{j_{k+1}}$ would be of the form

$$\begin{array}{c} \vdots\,\Delta_{j_k} \\ P(\nu'_{j_k}) \\ \vdots \\ P(\nu'_{j_{k+1}}) \end{array}$$

where $\nu'_{j_k} = \nu'_{j_{k+1}}$. Since $P(\nu'_{j_{k+1}})$ occurs twice in it, no rule of Definition 1 applies, which entails that $\Delta_{j_{k+1}+1}$ cannot exist, a contradiction.

So $\nu'_{j_k} < \nu'_{j_{k+1}}$ for every $k \geq 1$. $\Delta_{j_{k+1}}$ is of the form

$$\begin{array}{c} \vdots\ \Delta_{j_k} \\ P(\nu'_{j_k}) \\ \vdots \\ P(\nu'_{j_{k+1}}) \end{array}$$

with $\nu'_{j_k} < \nu'_{j_{k+1}}$, so by Lemma 2, $\nu'_{j_{k+1}}$ has more infinite coordinates than ν'_{j_k}, for every $k \geq 1$. This clearly contradicts the fact that the sequence $(\Delta_{j_k})_{k\geq 1}$ is infinite. So the forest constructed at the beginning of this proof is finite, whence the claim. $\square$

Propositions 1, 2, and Theorem 1 entail that the emptiness, coverability and boundedness problems are decidable for branching VASS, just as they are for VASS, as we show next. Given a set S of predicates and a subset J of $\{1,\ldots,p\}$, the branching VASS $\mathcal{V}$ is *S,J-bounded* if and only if $\{\nu[J] | \nu \in \mathbb{N}^p$ and $\exists P \in S$ such that $P(\nu)$ is derivable from $\mathcal{V}\}$ is finite. It is now clear that $\mathcal{V}$ is S,J-bounded if and only all its covering derivations with some conclusion $P(\nu')$, $P \in S$, are such that $\nu'[i] \neq \infty$ for all $i \in J$, and this is decidable by Theorem 1. The *coverability problem* for $\mathcal{V}$, tuple ν_0, set of states P and subset J of $\{1,\ldots,p\}$, asks whether there is a fact $P(\nu)$ derivable from $\mathcal{V}$ such that $P \in S$ and $\nu[J] \geq \nu_0[J]$. This is equivalent to testing whether some covering derivation ends in some generalized fact $P(\nu')$ with $P \in S$ and $\nu'[J] \geq \nu_0[J]$, which is decidable by Theorem 1. The *emptiness problem* asks whether there is any tuple ν and any $P \in S$ such that $P(\nu)$ is derivable: this is decidable, as a special case of coverability.

4 Application to *AC* Automata

We apply our results to equational tree automata which were the initial motivation for this work. Given a signature Σ of function symbols with fixed arities and an equational theory $\mathcal{E}$ over terms built using symbols from Σ, an *$\mathcal{E}$-tree automaton $\mathcal{P}$* [Ver03c, Ver03a, Ver03b, GLV02] is a set of definite clauses of the form $P(t) \Leftarrow P_1(t_1),\ldots,P_n(t_n)$ where P, P_1, ..., P_n are (a.k.a. states), and t, t_1, ..., t_n are terms built from symbols in Σ and variables x, y, z, ... This is similar to BVASS where predicates represent states. The difference here is that we are working with terms instead of tuples of natural numbers. See also [CDG+97] (Chapter 7) which uses clausal notation to represent tree automata (in the absence of equational theories though). The additional advantage of this notation here is that it clarifies the relationship between BVASS and equational tree automata. Formally *derivable* ground atoms are defined using the rules:

$$\frac{P_1(t_1\sigma)\ldots P_n(t_n\sigma)}{P(t\sigma)}\ (P(t) \Leftarrow P_1(t_1),\ldots,P_n(t_n) \in \mathcal{P}) \qquad\qquad \frac{P(s)}{P(t)}\ (s =_{\mathcal{E}} t)$$

where σ is a ground substitution, and $=_{\mathcal{E}}$ is the congruence on terms induced by the theory $\mathcal{E}$. The language *accepted* by $\mathcal{P}$ is the set $L_{\mathcal{P}}(P)$ of terms t such that $P(t)$ is derivable from $\mathcal{P}$, where P is a designated final state. Note that the usual notion of tree automata, which accept regular languages, are conveniently considered [CDG+97] as $\mathcal{E}$-tree automata, by letting $\mathcal{E}$ be the empty theory and by suitably restricting the form of clauses.

Consider a signature Σ containing a binary symbol $+$ and constants $a_1,\ldots,a_p$. *AC* is the theory stating that $+$ is associative and commutative. We are also interested in the theory *ACU* which additionally says that 0 is unit of $+$, where 0 is a new constant added to the signature. Terms modulo *ACU* are exactly summations $\sum_{i=1}^{p} n_i a_i$, equivalently, tuples from $\mathbb{N}^p$. Terms modulo *AC* are exactly non-zero such tuples.

The *constant-only AC* and *ACU* automata are built from the following clauses exclusively [Ver03c, Ver03b]:

$$P(x+y) \Leftarrow P_1(x), P_2(y) \quad (7) \qquad P(a) \text{ where } a \text{ is a constant} \quad (9)$$

$$P(x) \Leftarrow P_1(x) \quad (8) \qquad P(0) \quad (10)$$

where clause (10) is present only in the *ACU* case. Considering terms as vectors, it is then natural to think of these automata as BVASS. $P(a_i)$ corresponds to the clause $P(0,\ldots,0,1,0,\ldots,0)$ where the only '1' is at position i. We don't require any two-way clause. The languages accepted in the *ACU* (resp. *AC*) case are then exactly $\mathcal{L}$ (resp. $\mathcal{L} \setminus \{(0,\ldots,0)\}$) where $\mathcal{L}$ is a semilinear set.

Now let's add *standard* $+$ *push clauses*:

$$P(x) \Leftarrow P_1(x+y) \quad (11)$$

Similar push clauses were considered in [Ver03c, Ver03b], except they were of the form

$$P(x_i) \Leftarrow Q(f(x_1,\ldots,x_n)), P_1(x_1),\ldots,P_{i-1}(x_{i-1}), P_{i+1}(x_{i+1}),\ldots,P_n(x_n)$$

where f is a free, i.e., non-equational symbol (in particular, not $+$). Standard $+$ push clauses are a timid attempt at allowing equational (AC, ACU) symbols in the body of clauses. In Section 5, we make an attempt at being a bit less shy. We shall remain in the constant-only case throughout here (where all free function symbols are constants a), for simplicity.

Clause (11) says that P should accept all the subterms (strict subterms in the *AC* case) of terms accepted at P_1. Such clauses can be added to constant only *ACU* automata without increasing expressiveness, and the following table gives a linear time procedure for eliminating clauses (11):

Clause of input automaton	Clauses of output automaton		
$P(0)$	$P(0)$	$P^\dagger(0)$	
$P(a)$	$P(a)$	$P^\dagger(a)$	$P^\dagger(0)$
$P(x) \Leftarrow P_1(x)$	$P(x) \Leftarrow P_1(x)$	$P^\dagger(x) \Leftarrow P_1^\dagger(x)$	
$P(x+y) \Leftarrow P_1(x), P_2(y)$	$P(x+y) \Leftarrow P_1(x), P_2(y)$	$P^\dagger(x+y) \Leftarrow P_1^\dagger(x), P_2^\dagger(x)$	
$P(x) \Leftarrow Q(x+y)$	$P(x) \Leftarrow Q^\dagger(x)$	$P^\dagger(x) \Leftarrow Q^\dagger(x)$	

where the new states $P^\dagger$ accept all terms s such that $s+u=t$ for some ground term t accepted at P and for some ground term u.

Unfortunately this procedure, or its simple variants, fail in the *AC* case, as the reader may verify. To solve this problem, we realize that this new clause (11) can also be translated to BVASS. The idea is that applying a standard $+$-push clause involves going through loops involving clauses of the form $R(x+(0,\ldots,0,-1,0,\ldots,0)) \Leftarrow R'(x)$ which remove constants from the term. This gives us a procedure for eliminating standard $+$-push clauses. For that we first make the following observation:

Remark 2 *Given any* $\nu \in (\mathbb{N} \cup \{\infty\})^p$*, the set* $L_<(\nu) = \{\nu' \in \mathbb{N}^p \mid (0,\ldots,0) < \nu' < \nu\}$ *is Presburger-definable, hence semilinear. We will use* $L'_<(\nu)$ *to denote the corresponding set* $\{\sum_{i=1}^{p} n_i a_i \mid (n_1,\ldots,n_p) \in L_<(\nu)\}$.

Call a *standard-two-way constant-only AC-automaton* any set of constant-only *AC* automata clauses and standard $+$-push clauses.

Lemma 4 *Given any standard-two-way constant-only AC-automaton $\mathcal{P}$, we can compute a constant-only AC automaton accepting the same language.*

Proof: We construct a BVASS $\mathcal{V}$ such that each state P in $\mathcal{P}$ accepts exactly the terms $\sum_{i=1}^{p} n_i a_i$ such that $P(n_1,\ldots,n_p)$ is derivable in $\mathcal{V}$. From Propositions 1 and 2, if $P(x) \Leftarrow Q(x+y)$ is any clause then the set of terms accepted at P using this clause is $L^Q_< = \bigcup L'_<(\nu)$ where the union is taken for all ν such that there is some covering derivation for $\mathcal{V}$, with root labeled by generalized fact $Q(\nu)$. From Theorem 1, since the number of such ν's is finite, $L^Q_<$ is semilinear, and can be accepted at a new state $Q^\dagger$ using other new states, and only clauses of constant-only *AC* automata. Then we can replace the clause $P(x) \Leftarrow Q(x+y)$ by the clause $P(x) \Leftarrow Q^\dagger(x)$. □

However note that while in the *ACU* case we are able to do this in linear time, in the *AC* case this algorithm is non-primitive recursive because the Karp-Miller construction (even for VASS) is not primitive recursive. The question whether the algorithm in the *AC* case can be improved is open.

Theorem 2 *The standard-two-way constant-only AC-automata accept exactly those semilinear sets which don't contain the term* 0*.*

5 Perspectives: A Further Extension of BVASS

We presented standard $+$ push clauses in Section 4. However we would really like to be able to deal with (not necessarily standard) $+$*-push clauses* of the form $P(x) \Leftarrow P_1(x+y), P_2(y)$ modulo *AC* or *ACU*. The same reduction as in Section 4 leads us to a further extension of BVASS by adding new clauses (interpreted over $\mathbb{N}^p$) of the form $P(x-y) \Leftarrow P_1(x), P_2(y)$, called *subtraction clauses*. The semantics of this clause is: "if facts $P_1(\nu_1)$ and $P_2(\nu_2)$ are derivable and $\nu_1 - \nu_2 \geq 0$ then the fact $P(\nu_1 - \nu_2)$ is derivable". However we don't know whether the construction of Karp-Miller trees can be further extended to deal with subtraction clauses.

Note that our extension of Karp-Miller trees gives us decidability of emptiness for BVASS. However the question of decidability of intersection-emptiness of BVASS (is there a tuple ν recognized at each of the states $P_1, \ldots, P_n$?) is still open. Clearly this problem subsumes the reachability problem for BVASS as well the reachability problem for VASS. As an aside, observe also that the reachability problem for BVASS subsumes the intersection-emptiness problem for BVASS. The idea is as follows. To decide whether some common tuple is accepted at states P_1 and P_2 in BVASS $\mathcal{V}_1$ and $\mathcal{V}_2$ respectively on p-tuples, we construct BVASS $\mathcal{V}''_1$ and $\mathcal{V}''_2$ on $2p$-tuples (with disjoint sets of states) such that in $\mathcal{V}''_1$, P_1 accepts exactly the tuples $(n_1,\ldots,n_p,0,\ldots,0)$ where P_1 accepts $(n_1,\ldots,n_p)$ in $\mathcal{V}_1$, and in $\mathcal{V}''_2$, P_2 accepts exactly the tuples $(0,\ldots,0,n_1,\ldots,n_p)$ where P_2 accepts $(n_1,\ldots,n_p)$ in $\mathcal{V}_2$. We define $\mathcal{V}$ to contain the clauses of $\mathcal{V}''_1$ and $\mathcal{V}''_2$, as well as the clauses $P(x+y) \Leftarrow P_1(x), P_2(y)$ and $P(x+\delta_i) \Leftarrow P(x)$ for some fresh P and for $1 \leq i \leq p$, where $\delta_i[i] = \delta_i[i+p] = -1$ and $\delta_i[j] = 0$ for $j \notin \{i, i+p\}$. Then P accepts the tuple $(0,\ldots,0)$ in $\mathcal{V}$ iff P_1 and P_2 accept some common tuple in $\mathcal{V}_1$ and $\mathcal{V}_2$ respectively. Similarly the reachability problem for VASS also subsumes the intersection-emptiness problem for VASS: we let the reader figure out the corresponding argument.

As we said in the introduction, decidability of the reachability problem for VASS is a difficult result and uses the Karp-Miller construction as an auxiliary result. On the other hand in the presence of subtraction clauses, the intersection-emptiness problem can actually be reduced to the emptiness problem, so emptiness is no longer easier than reachability! Here is the idea. To test whether two states P_1 and P_2 accept at least one common tuple, add the following clauses where P_0, P_3, P are fresh predicates:

$$\begin{array}{l} P_3(x-y) \Leftarrow P_1(x), P_2(y) \\ P_0((0,\ldots,0)) \\ P(x-y) \Leftarrow P_0(x), P_3(y) \end{array}$$

Then there is some common tuple accepted at both P_1 and P_2 iff there is some tuple accepted at P. This result shows the power of subtraction clauses since now doing the Karp-Miller construction already involves solving a problem at least as difficult as that of VASS reachability.

6 Conclusion

We have studied a natural generalization of both Parikh images of context-free languages and of vector addition systems with states (VASS, special case: Petri nets), where derivations are both two-way (as in the latter) and branching (as in the former). For these so-called branching VASS, we have constructed an analogue of the Karp-Miller coverability tree construction for Petri nets. This allows us to conclude, like in the ordinary coverability tree construction, that emptiness, coverability, and boundedness are decidable for the class of branching VASS. The construction for branching VASS differs from the simpler case of VASS in that we construct covering derivations (analogues of finite prefixes of paths in ordinary Karp-Miller trees) in isolation (Section 2). Doing this, we lose the possibility to appeal directly to König's Lemma; we nonetheless managed to show that there are only finitely many covering derivations from a given branching VASS (Section 3), by a more technical argument that builds a forest whose nodes are covering derivations, with a subtle selection of the (necessarily unique) parent of each non-root node.

We have shown (Section 4) how this produced a simple proof that a natural extension of the constant-only *AC* automata considered in [Ver03c, Ver03b] with so-called standard $+$-push clauses actually reduce to the case without standard $+$-push clauses. This seems to require non-primitive-recursive time, however, contrarily to the *ACU* case, which reduces also, but in linear time. In [Ver03a], this result on the constant-only case has also been used to deal with the general case (extending the automata of [Ver03c, Ver03b] with standard $+$-push clauses, not just the constant-only automata).

In turn, the results of Section 4 prod us to explore further extensions of branching VASS. We have explained what challenges this entailed (Section 5).

Besides arising as a natural common generalization of two already well-established tools in computer science – Parikh images of context-free languages, and VASS – and having application in equational tree automata, branching VASS are also useful in the completely different domain of linear logic, since decidability of reachability in branching BVASS is equivalent to decidability of provability in multiplicative exponential linear logic, which is still an open problem. This confirms that branching VASS are interesting objects to study.

The result that there are only finitely many covering derivations can also be seen as a first step towards a positive answer to the question whether the reachability problem for branching VASS, and hence whether provability in MELL, is decidable or not. This is deferred to a later paper; additional difficulties lurk ahead, though, and notably there are no such things as Kirchoff's laws [Lam92] in the extended context.

References

[AÉI02] Luca Aceto, Zoltán Ésik, and Anna Ingólfsdóttir. A fully equational proof of Parikh's theorem. *RAIRO, Theoretical Informatics and Applications*, 36:129–153, 2002.

[CDG+97] Hubert Comon, Max Dauchet, Rémi Gilleron, Florent Jacquemard, Denis Lugiez, Sophie Tison, and Marc Tommasi. Tree automata techniques and applications. `http://www.grappa.univ-lille3.fr/tata`, 1997.

[dGGS04] Philippe de Groote, Bruno Guillaume, and Sylvain Salvati. Vector addition tree automata. In *19th Annual IEEE Symposium on Logic in Computer Science (LICS 2004)*, pages 64–73, Turku, Finland, July 2004. IEEE Computer Society Press.

[GLRV05] Jean Goubault-Larrecq, Muriel Roger, and Kumar Neeraj Verma. Abstraction and resolution modulo AC: How to verify Diffie-Hellman-like protocols automatically. *Journal of Logic and Algebraic Programming*, 64(2):219–251, August 2005.

[GLV02] Jean Goubault-Larrecq and Kumar Neeraj Verma. Alternating two-way AC-tree automata. Research Report LSV-02-11, LSV, ENS Cachan, Cachan, France, September 2002.

[GS66] Seymour Ginsburg and Edwin H. Spanier. Semigroups, Presburger formulas and languages. *Pacific Journal of Mathematic*, 16(2):285–296, 1966.

[HP79] J. Hopcroft and J. J. Pansiot. On the reachability problem for 5-dimensional vector addition systems. *Theoretical Computer Science*, 8:135–159, 1979.

[Kan94] Max I. Kanovich. Linear logic as a logic of computations. *Annals of Pure and Applied Logic*, 67:183–212, 1994.

[Kan95] Max I. Kanovich. Petri nets, Horn programs, linear logic and vector games. *Annals of Pure and Applied Logic*, 75:107–135, 1995.

[Kan96] Max I. Kanovich. Linear logic automata. *Annals of Pure and Applied Logic*, 78:147–188, 1996.

[KM69] R. M. Karp and R. E. Miller. Parallel program schemata. *Journal of Computer and System Sciences*, 3(2):147–195, 1969.

[Kos82] S. R. Kosaraju. Decidability of reachability in vector addition systems. In *Proc. 14th Symp. Theory of Computing*, pages 267–281. ACM, 1982.

[Lam92] Jean-Luc Lambert. A structure to decide reachability in Petri nets. *Theoretical Computer Science*, 99:79–104, 1992.

[Lug03] Denis Lugiez. Counting and equality constraints for multitree automata. In Andrew D. Gordon, editor, *6th International Conference on Foundations of Software Science and Computational Structures (FoSSaCS'03)*, volume 2620 of *LNCS*, pages 328–342, Warsaw, Poland, 2003. Springer-Verlag.

[May84] E. W. Mayr. An algorithm for the general Petri net reachability problem. *SIAM Journal of Computing*, 13:441–460, 1984.

[MR98] Richard Mayr and Michaël Rusinowitch. Reachability is decidable for ground AC rewrite systems. In *Proceedings of the 3rd INFINITY Workshop*, Aalborg, Denmark, 1998.

[Mül84] H. Müller. The reachability problem for VAS. In *Advances in Petri nets*. Springer-Verlag LNCS 188, 1984.

[Ohs01] Hitoshi Ohsaki. Beyond regularity: Equational tree automata for associative and commutative theories. In Laurent Fribourg, editor, *10th Annual Conference of the European Association for Computer Science Logic (CSL'01)*, volume 2142 of *LNCS*, pages 539–553, Paris, France, September 2001. Springer-Verlag.

[Par66] Rohit J. Parikh. On context-free languages. *Journal of the ACM*, 13(4):570–581, October 1966.

[Reu89] Christophe Reutenauer. *Aspects Mathématiques des Réseaux de Pétri*. Masson, 1989.

[Ver03a] Kumar Neeraj Verma. *Automates d'arbres bidirectionnels modulo théories équationnelles*. PhD thesis, ENS Cachan, 2003.

[Ver03b] Kumar Neeraj Verma. On closure under complementation of equational tree automata for theories extending AC. In Moshe Vardi and Andrei Voronkov, editors, *10th International Conference on Logic for Programming Artificial Intelligence and Reasoning (LPAR'03)*, volume 2850 of *LNAI*, pages 183–197, Almaty, Kazakhstan, September 2003. Springer-Verlag.

[Ver03c] Kumar Neeraj Verma. Two-way equational tree automata for AC-like theories: Decidability and closure properties. In Robert Nieuwenhuis, editor, *14th International Conference on Rewriting Techniques and Applications (RTA'03)*, volume 2706 of *LNCS*, pages 180–196, Valencia, Spain, June 2003. Springer-Verlag.

[VGL04] Kumar Neeraj Verma and Jean Goubault-Larrecq. Karp-Miller trees for a branching extension of VASS. Research Report LSV-04-3, LSV, ENS Cachan, France, January 2004. Available at `http://www.lsv.ens-cachan.fr/Publis/RAPPORTS_LSV/PS/rr-lsv-2004-3.rr.ps`.