Karp-Miller Trees for a Branching Extension of VASS
Kumar Neeraj Verma, Jean Goubault-Larrecq

To cite this version:

HAL Id: hal-00959038
https://inria.hal.science/hal-00959038
Submitted on 13 Mar 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Karp-Miller Trees for a Branching Extension of VASS

Kumar Neeraj Verma and Jean Goubault-Larrecq

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


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”.

1365–8050 c 2005 Discrete Mathematics and Theoretical Computer Science (DMTCS), Nancy, France
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 \( n \). We shall consider definite clauses of the form
\[
P(t) \leftarrow P_1(t_1), \ldots, P_n(t_n),
\]
where \( n \in \mathbb{N} \), \( P, P_1, \ldots, P_n \) are so-called predicates (a.k.a., states), and \( t, t_1, \ldots, t_n \) are terms built from constants \( v \in \mathbb{N}^n \), variables \( x, y, z, \ldots \) (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 \( P \) is any finite set of definite clauses of the above format. A fact is any atom \( P(v) \) with \( v \in \mathbb{N}^n \). Derivations (of \( P(v) \)) from \( P \) are inductively defined so that, given any instance \( P(v) \leftarrow P_1(v_1), \ldots, P_n(v_n) \) of a definite clause in \( P \), given any derivations \( \Delta_i \) of \( P_i(v_i) \) from \( P \), \( 1 \leq i \leq n \), the following is a derivation:
\[
\begin{align*}
\vdots & \quad \Delta_1 \quad \vdots \\
P_1(v_1) & \quad \ldots \\
\vdots & \quad \Delta_n \quad \vdots \\
P(v)
\end{align*}
\]
(The base cases are when \( n = 0 \), in which case the chosen instance is the fact \( P(v) \).) A fact is derivable from \( P \) iff there is a derivation of it from \( P \). The language \( L_P(P) \) of the predicate (or state) \( P \) in \( P \) is the set of \( p \)-tuples \( v \in \mathbb{N}^n \) such that \( P(v) \) is derivable from \( P \). We say that \( v \in \mathbb{N}^n \) is recognized at \( P \) in \( P \) iff \( P(v) \in L_P(P) \).

Semilinear sets. Recall that a linear set \( L \) of \( p \)-tuples of natural numbers is any set of the form \( L_{v_0,B} = \{v_0 + v_1 + \ldots + v_k \mid k \in \mathbb{N} \text{ and } v_1, v_2, v_3 \in B\} \) 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 \( v_0, v \in \mathbb{N}^n \) and some finite \( B \subseteq \mathbb{N}^p \), consider the following set of Horn clauses:
\[
P(v_0)
\]
\[
P(x + v) \iff P(x) \quad (v \in B)
\]
It is easy to see that the set of \( p \)-tuples recognized at \( P \) in this program is exactly \( L_{v_0,B} \). Given any semilinear set \( L \), written as a union \( \bigcup_{i=1}^n L_{v_i,B_i} \), we can as easily write the set of all clauses \( P(v_i) \), \( 1 \leq i \leq n \) and \( P_i(x + v) \leftarrow P_i(x) \), \( 1 \leq i \leq n, v \in B_i \), where \( P_1, \ldots, P_n \) 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 \( L_{P}(R) \) for some \( R \) and some finite set \( P \) of definite clauses of the form (1) or
\[
P(x + v) \iff Q(x), \quad (v \in \mathbb{N}^p)
\]
Conversely, for every finite set \( P \) of what we shall call base/period clauses (of the form (1) or (3)), the languages \( L_P(P) \) 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:

\[ P(x + y) \iff Q(x), R(y) \]  

(\(x\) 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 \(A = \{a_1, \ldots, a_p\}\) 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 \(P \rightarrow a_i\), or \(P \rightarrow \varepsilon\), or \(P \rightarrow QR\) (where \(P, Q, R\) are non-terminals, and \(\varepsilon\) denotes the empty word). For each production \(P \rightarrow a_i\), generate the clause \(P((0, \ldots, 0, 1, 0, \ldots, 0))\), where the only ‘1’ is at position \(i\); for each production \(P \rightarrow \varepsilon\), generate the clause \(P((0, \ldots, 0))\); for each production \(P \rightarrow QR\), generate \(P(x + y) \iff Q(x), R(y)\). 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 \(P\) consisting of clauses of the form (1), (2) and (3), the languages \(L_P(P)\) 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 \(P\), in such a way that the languages of \(P\) are preserved, for each \(P\) in \(P\). (A nice, generalized version of this appears in [AE02].)

Note that, while derivations in base/period programs are just sequences, derivations in the presence of addition clauses (3) 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

\[ P(x + \nu^*) \iff Q(x + \nu) \]  

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

\[ P(x + \delta) \iff Q(x) \]  

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

\footnote{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 \( p \geq 3 \) \cite{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 \cite{HP79}. Note that reachability of a fixed \( p \)-tuple in a given VASS is decidable \cite{May84,Kos82,Mul84,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 \( \mathcal{P} \) \cite{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:

![Diagram showing Parikh images of context-free languages, addition clauses, VASS, and BVASS]

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 \cite{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
Karp-Miller Trees for a Branching Extension of VASS

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 Groot 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.

---

8 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 $\nu$ 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 $i$th component of $\nu$. 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 $\Delta$ 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:

   \[
   \begin{array}{c}
   \vdash \Delta \\
   P(\nu)
   \end{array}
   \]

2. For every covering derivation

   \[
   \begin{array}{c}
   \vdash \Delta_1 \\
   P_1(\nu')
   \end{array}
   \]

   such that $P_1(\nu')$ only occurs once (namely, at the bottom) in $\Delta_1$, for every transition $P(x + \delta) \Leftarrow P_1(x)$ in $\mathcal{V}$ such that $\nu_1'[i] + \delta[i] \geq 0$,

   \[
   \begin{array}{c}
   \vdash \Delta_1 \\
   P_1(\nu')
   \end{array}
   \]

   is a covering derivation, where $\nu'$ is defined as the vector whose $i$th component is:

   \[
   \infty \text{ if there is a generalized fact } P(\nu'') \text{ in } \Delta_1 \text{ (that is, at or above } P_1(\nu')) \text{ such that } \nu'' <_i \nu'[i] + \delta; \\
   \nu'[i] + \delta[i] \text{ otherwise.}
   \]

3. For every covering derivations

   \[
   \begin{array}{c}
   \vdash \Delta_1 \\
   P_1(\nu')
   \end{array}
   \quad \quad \quad
   \begin{array}{c}
   \vdash \Delta_2 \\
   P_2(\nu'_2)
   \end{array}
   \]

   such that $P_1(\nu')$ only occurs once in $\Delta_1$ and $P_2(\nu'_2)$ only occurs once in $\Delta_2$, for every addition clause $P(x + y) \Leftarrow P_1(x), P_2(y)$ in $\mathcal{V}$,

   \[
   \begin{array}{c}
   \vdash \Delta_1 \\
   P_1(\nu')
   \end{array}
   \quad \quad \quad
   \begin{array}{c}
   \vdash \Delta_2 \\
   P_2(\nu'_2)
   \end{array}
   \]

   is a covering derivation, where $\nu'$ is defined as the vector whose $i$th component is:

   \[
   \infty \text{ if there is a generalized fact } P(\nu'') \text{ in } \Delta_1 \text{ or } \Delta_2 \text{ such that } \nu'' <_i \nu'[i] + \nu'_2; \\
   \nu'[i] + \nu'_2[i] \text{ otherwise.}
   \]

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

Proposition 1

Example 1

Consider a branching VASS with the following set of clauses

\[
\begin{align*}
C_1 &= P_1(2, 5) \\
C_2 &= P_2(3, 4) \\
C_3 &= P_1(x + y) \iff P_1(x), P_2(y) \\
C_4 &= P_1(x + \{2, -4\}) \iff P_3(x) \\
C_5 &= P_2(x + \{2, -5\}) \iff P_3(x)
\end{align*}
\]

Some facts derivable in this branching VASS are \(P_1(2, 5), P_2(3, 4), P_3(5, 9), P_1(2 + n, 5), P_3(5 + n, 9), P_5(3 + 4n, 4)\) 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 \(\nu\) be a branching VASS. If a fact \(P(\nu)\) is derivable, then there is a covering derivation \(\Delta\) 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 \(\Delta_1\) of \(P(\nu_1)\) using the clause \(P(x + \delta) \iff P_1(x)\). By induction hypothesis we have a covering derivation \(\Delta_1\) of some generalized fact \(P(\nu_1')\), such that if \(\nu_1'[i] < \infty\) then \(\nu_1'[i] = \nu_1[i]\). We pick a minimal such \(\Delta_1\). Consequently \(P_1(\nu_1')\) does not occur except as conclusion in \(\Delta_1\). Clearly we have \(\nu_1 + \delta \geq 0\) and hence \(\nu_1'[i] + \delta \geq 0\). By using Rule 2 of Definition 1 we get a covering derivation \(\Delta\) 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 \(\Delta\) 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 \(P(x + y) \iff P_1(x), P_2(y)\). By induction hypothesis, we have covering derivations \(\Delta_1\) and \(\Delta_2\) 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 \(\Delta_1\) as the conclusion, and \(P_2(\nu_2')\) only occurs in \(\Delta_2\) as the conclusion. By using Rule 3 of Definition 1 we get a covering derivation \(\Delta\) 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 \(\Delta\) is the required covering derivation. \(\square\)
**Definition 2 (Linear Path)** Given a branching VASS $\mathcal{V}$, the linear paths $\pi$ of $\mathcal{V}$ are the sequences $P_1 \xrightarrow{e_1} P_2 \xrightarrow{e_2} \cdots P_{n-1} \xrightarrow{e_{n-1}} P_n$, where $n \geq 1$, $P_1, \ldots, P_n$ 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 $(C; Q(\nu))$ 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 $\pi$ 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 $\delta$; in the second case, it is $\nu$. The valuation $v(\pi)$ of the linear path $\pi$ is $v(\pi) = \Sigma_{1 \leq i < n} v(e_i)$.

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

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

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

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

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

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

$$P_1 \xrightarrow{(C_3; P_3(3,4))} P_2 \xrightarrow{C_4} P_3 \xrightarrow{(C_3; P_3(50,5))} P_4$$

Note that the facts $P_2(3,4)$ and $P_3(50,5)$ 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 $\mathbf{v}$. The linear path

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

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

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 $\Delta$ be a covering derivation with the property that, given any generalized fact $P(\mathcal{V})$ occurring in this derivation, we can find a (non-generalized) fact $\mathcal{V}$ such that

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

Suppose that $\Delta_2$ is a subderivation of $P_2(\mathcal{V}^'_2)$ in $\Delta$ of the following form, containing a (not necessarily proper) subderivation $\Delta_1$ (of $P_1(\mathcal{V}^'_1)$).

$$
\vdots \Delta_1 \\
P_1(\mathcal{V}^'_1) \\
\vdots \\
P_2(\mathcal{V}^'_2)
$$

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

**Proof:** We induct on the distance (number of steps) between $P_1(\mathcal{V}^'_1)$ and $P_2(\mathcal{V}^'_2)$ in $\Delta$.

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

- If $\Delta_2$ is of the form

  $$
  \vdots \Delta_1 \\
P_1(\mathcal{V}^'_1) \\
\vdots \\
P_2(\mathcal{V}^'_2)
  $$

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

- If $\Delta_2$ is of the form

  $$
  \vdots \Delta_1 \\
P_1(\mathcal{V}^'_1) \\
\vdots \\
P_2(\mathcal{V}^'_2)
  $$

  where, without loss of generality, $\Delta_1$ is contained in the subderivation leading to the left premise. Clearly $\mathcal{V}^'_2[J] = \mathcal{V}^'_3[J] + \mathcal{V}^'_4[J]$. Also for all $i \in J$, $\mathcal{V}^'_3[i] \neq \infty$ and $\mathcal{V}^'_4[i] \neq \infty$. By induction hypothesis
we get a linear path \( \pi' \) from \( P_1 \) to \( P_3 \) which is admissible for \( \mathcal{V}'_1 \) with respect to \( \Delta \), and such that \( \mathcal{V}'_1[J] + \nu(\mathcal{V}')[J] = \mathcal{V}'_2[J] \). By assumption (this is where we need it!), \( P(\mathcal{V}_4) \) is derivable from \( \mathcal{V}' \) for some vector \( \mathcal{V}_4 \) such that, for all \( i \) such that \( \mathcal{V}'_4[i] \neq \infty, \mathcal{V}_4[i] = \mathcal{V}'_4[i] \). In particular \( \mathcal{V}_4[J] = \mathcal{V}'_4[J] \). Let the required linear path \( \pi \) be the concatenation of \( \pi' \) with \( P_3 \rightarrow \left( P_5(x+y) = P_4(x)P_2(P_4) \right) \rightarrow P_2 \). This is a well-defined linear path. We have \( \mathcal{V}'_1[J] + \nu(\mathcal{V})[J] = \mathcal{V}'_1[J] + \nu(\mathcal{V}')[J] + \mathcal{V}_4[J] = \mathcal{V}'_3[J] + \mathcal{V}_4[J] = \mathcal{V}'_3[J] + \mathcal{V}_4[J] = \mathcal{V}'_2[J] \). In particular \( \mathcal{V}'_1[J] + \nu(\pi)[J] \geq 0 \) and hence \( \pi \) is admissible for \( \mathcal{V}'_1 \) with respect to \( \Delta \).

**Example 3** Let us look at Example 2 again. Looking at Figure 2, take \( \Delta_2 \) to be the whole covering derivation (of \( P_1(\infty, 5) \)), and \( \Delta_1 \) to be the one on the left ending on \( P_3(5, 9) \). Take \( J = \{2\} \).

The linear path \( P_3 \rightarrow C_4 \rightarrow P_1(C_2, P_3(3, 4)) \rightarrow P_3 \rightarrow C_4 \rightarrow P_1(C_3, P_3(7, 4)) \rightarrow P_3 \rightarrow C_4 \rightarrow P_1 \)

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 \( P_1(5, 5) \). This linear path is probably the one you expected; another one is

\[
\begin{align*}
P_3 \rightarrow & C_4 \rightarrow P_1(C_2, P_3(3, 4)) \rightarrow P_3 \rightarrow C_4 \rightarrow P_1(C_3, P_3(7, 4)) \rightarrow P_3 \rightarrow C_4 \rightarrow P_1 \\
& \text{with the corresponding facts } P_3(5, 9), P_1(3, 5), P_3(6, 9), P_1(4, 5), P_3(11, 9) \text{ and } P_1(9, 5).
\end{align*}
\]

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 \( P_1 \) 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 \( P_1 \):

\[
\begin{align*}
P_3 \rightarrow & C_4 \rightarrow P_1(C_2, P_3(3, 4)) \rightarrow P_3 \rightarrow C_5 \rightarrow P_2(C_3, P_3(2, 5)) \rightarrow P_3 \rightarrow C_4 \rightarrow P_1 \\
& \text{The corresponding facts in this case are } P_3(5, 9), P_1(3, 5), P_3(6, 9), P_2(8, 4), P_3(10, 9) \text{ and } P_1(8, 5).
\end{align*}
\]

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 \( \Delta \) of some generalized fact \( P(\mathcal{V}') \), and for any \( K \geq 0 \) there is a tuple \( \nu \in \mathbb{N}^P \) such that

- for every \( i \) such that \( \mathcal{V}'[i] = \infty, \mathcal{V}[i] \geq K; \)

- for every \( i \) such that \( \mathcal{V}'[i] \neq \infty, \mathcal{V}[i] = \mathcal{V}'[i]; \)

- \( P(\mathcal{V}) \) is derivable from \( \mathcal{V}' \).

**Proof:** By induction on \( \Delta \). If \( P(\mathcal{V}') \) is a fact in \( \mathcal{V}' \) (rule 1 of Definition 1), then \( \nu = \mathcal{V}' \) satisfies the requirements. Otherwise, look at the last rule in \( \Delta \):
Karp-Miller Trees for a Branching Extension of VASS

- If $\Delta$ is constructed using rule [2] of Definition [1]

\[
\frac{\vdots}{\Delta_1} \quad \frac{P_i(v'_1)}{P(v')} \quad P(x + \delta) \iff P_1(x)
\]

then let $I$ be the set of indices $i$ such that $v'[i] = \infty$, $J$ be the set of all other indices, let $I_1$ be the set of indices $i$ such that $v'_1[i] = \infty$ 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:

\[
\begin{array}{c}
\begin{array}{c}
\vdots
\end{array}

\begin{array}{c}
I_1
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
\vdots
\end{array}

\begin{array}{c}
I_a
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
\vdots
\end{array}

\begin{array}{c}
J
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
\vdots
\end{array}

\begin{array}{c}
J_1
\end{array}
\end{array}
\]

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 $\pi_i$ for all $i \in I_a$.** By definition of $I_a$, for each $i \in I_a$ there is a subderivation $\Delta'$ of $\Delta_1$ that derives $P(v''_i)$ for some generalized fact $v''_i$ such that: (a) $v''_i \leq v'_1 + \delta$ and: (b) $v''_i[i] < v'_1[i] + \delta[i]$. By induction hypothesis the assumptions of Lemma [1] are satisfied of $\Delta_1$ and the set of indices $J_1$. From this lemma we get a linear path $\pi'_i$ from $P$ to $P_1$ which is admissible for $v''_i$ with respect to $J_1$ and such that: (c) $v''_i[J_1] + v(\pi'_i)[J_1] = v'_1[J_1]$. Let $\pi_i$ be the path from $P$ to $P$ obtained by concatenating $\pi'_i$ with $P_1$ by $P(\pi_i) = P \xrightarrow{P(v''_i)} P_1 \xrightarrow{P(\pi'_i)} P$. Then: (d) $v''_i[J_1] + v(\pi_i)[J_1] = v'_1[J_1] + \delta[J_1]$. Indeed $v''_i[J_1] + v(\pi_i)[J_1] = v''_i[J_1] + v(\pi'_i)[J_1] + \delta[J_1] = v'_1[J_1] + \delta[J_1]$ by (c). By (d), and since $v'_1 + \delta \geq 0$, it follows that $v''_i[J_1] + v(\pi_i)[J_1] \geq 0$. Hence $\pi_i$ is admissible for $v''_i$ with respect to $J_1$. By (a) above, $v''_i \leq v'_1 + \delta$, so $\pi_i$ is admissible for $v'_1 + \delta$ with respect to $J_1$.

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

---

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

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

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

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

Letting \( \pi^K \) be the concatenation of \( K \) copies of \( \pi \), we can again see that the path \( \pi^K \) is admissible for \( v'_1 + \delta \) with respect to \( J_1 \), \( v(\pi^K)[J] = \emptyset \) and \( v(\pi^K)[I_a] \geq K. \) Choose some \( K_1 \in \mathbb{N} \) such that for each prefix \( \pi' \) of \( \pi^K, K_1 + v(\pi'')[I_a] \geq \emptyset \) and \( K_1 + v(\pi^K)[I_a] \geq K. \) Choose \( K_2 \in \mathbb{N} \) such that \( K_2 + \delta[I_a] \geq K_1. \) By induction hypothesis there is a vector \( v_1 \) such that \( P(v_1) \) is derivable from \( v', v_1[I_a] \geq K_2, \) and \( v_1[I_a] = v'_1[I_a]. \)

Since \( v'_1 + \delta \geq 0 \) (by the simple fact that rule [2] of Definition [1] was applicable at all) and \( v_1[I_a] = v'_1[I_a], \) we obtain \( v_1[I_a] + \delta[I_a] \geq 0. \) On the other hand, \( v_1[I_a] + \delta[I_a] \geq K_2 + \delta[I_a] \geq K_1 \geq 0. \) So \( v_1[I_a] + \delta \geq 0, \) therefore \( P(v_1[I_a] + \delta) \) is derivable from \( P(v_1[I_a]) \) and the transition \( P(x + \delta) \leftarrow P_1(x). \)

Since \( v_1[I_a] + \delta[I_a] \geq K_2 + \delta[I_a] \geq K_1, v_1[I_a] + \delta[I_a] \geq K_1. \) By the definition of \( K_1, \) this entails that \( v_1[I_a] + \delta[I_a] + v(\pi')[I_a] \geq \emptyset \) for every prefix \( \pi' \) of \( \pi^K, \) so \( \pi^K \) is admissible for \( v_1[I_a] + \delta[I_a] \) with respect to \( I_a. \)

Since \( v_1[I_a] = v'_1[I_a] \) and \( \pi^K \) is admissible for \( v'_1[I_a] + \delta[I_a] \) with respect to \( J_1, \) \( \pi^K \) is also admissible for \( v_1[I_a] + \delta[I_a] \). This implies that \( P(v_1[I_a] + \delta[I_a] + v(\pi^K)[I_a]) \) is derivable from \( v'. \)

**Building the fact \( v \) from \( \pi. \)** Let therefore \( v \) be \( v_1 + \delta + v(\pi^K). \) It remains to show that \( v[J] \geq K \) and \( v[J] = v'[J]. \)

The second claim follows from the fact that \( v(\pi)[J] = \emptyset, \) from which we infer \( v(\pi^K)[J] = \emptyset, \) hence \( v[J] = v_1[J] + \delta[J] + v(\pi^K)[J] = v'_1[J] + \delta[J] = v'[J] + \delta[J] \) (since \( v_1[I_a] = v'_1[I_a] \) and \( J \subseteq J_1 \) for the second sub-claim, recall that \( v_1[I_a] \geq K_2 \) and \( K_2 + \delta[I_a] \geq K_1, \) so \( v_1[I_a] + \delta[I_a] \geq K_1, \) and by the
definition of $K_1$, this entails that $\nu_1[I_1] + \delta[I_1] + \nu(\pi^k)[I_1] \geq \overline{K}$, i.e., $\nu[I_1] \geq \overline{K}$. For the second subclaim, recall that $\nu(\pi^k)[I_0] \geq \overline{K}$, and $\nu_1[I_1] = \nu_1'[I_1]$ (in particular, $\nu_1[I_0] = \nu_1'[I_0]$, since $I_0 \subseteq J_1$); so $\nu[I_0] = \nu_1[I_0] + \delta[I_0] + \nu(\pi^k)[I_0] \geq \nu_1'[I_0] + \delta[I_0] + \overline{K} \geq \overline{K}$, since $\nu_1' + \delta \geq 0$.

- Suppose $\Delta$ is constructed using rule 3 of Definition 1:

\[
\begin{array}{c}
\vdots \Delta_1 \\ \vdots \Delta_2 \\ P_1(\nu_1') \\ P_2(\nu_2') \\ \overline{P(\nu')} \\
\end{array}
\]

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 $\pi_i$ for all $i \in I_a$.** By induction hypothesis there are vectors $\nu_1$ and $\nu_2$ such that $P_1(\nu_1)$ and $P_2(\nu_2)$ are derivable, $\nu_1[I_1] = \nu_1'[I_1]$ and $\nu_2[J_2] = \nu_2'[J_2]$.

As in the previous case, define a linear path $\pi_i$ for each $i \in I_a$, as follows. Contrarily to the previous case, $\pi_i$ will depend on $\nu_1$ and $\nu_2$.

Since $i \in I_a$, there is a subderivation $\Delta_i'$ of $\Delta_{b_i}$, for some $b_i \in \{1, 2\}$, that derives $P(\nu')$ for some generalized fact $\nu'$ such that: (a) $\nu'[i] \leq \nu_1' + \nu_2'$, and (b) $\nu'_{[i]} < \nu_1'[i] + \nu_2'[i]$. By Lemma 1 there is a linear path $\pi_i'$ from $P$ to $P_{b_i}$ which is admissible for $\nu'$ with respect to $J_{b_i}$ and such that: (c) $\nu'[J_{b_i}] + \nu(\pi'_i)[J_{b_i}] = \nu'_{b_i}[J_{b_i}]$. Let $b_i$ be 2 if $b_i = 1$, and 1 if $b_i = 2$. Then let $\pi_i$ be the path from $P$ to $P$ obtained by concatenating $\pi_i'$ with $P_{b_i} \xrightarrow{(C, \delta)} P$, where $C$ is the clause $P(x+y) \iff P_1(x), P_2(y)$, and $\delta = \nu_{b_i}$.

\[
\begin{array}{c}
\vdots \Delta_i' \\ P(\nu') \\
\vdots \\
P_{b_i}(\nu_{b_i'}) \\ P(\nu') \\
\end{array}
\]

\[
\begin{array}{c}
\pi_i = P \longrightarrow \ldots \pi_i' \ldots \longrightarrow P_{b_i} \xrightarrow{(P(x+y) \iff P_1(x), P_2(y), \nu_{b_i})} P \\
\end{array}
\]
The reasoning is then similar to the previous case; there are many slight differences, though.
We first show that: (d) $v^i[J_0] + v(\pi_i)[J_0] = v'_i[J_0] + 0$. This is by (c).

By (d), and since $v'_i + 0 \geq 0$ as a sum of vectors in $\mathbb{N}^p$, we have $v^i[J_0] + v(\pi_i)[J_0] \geq 0$. Hence $\pi_i$ is admissible for $v^i$ with respect to $J_{b_t}$. By (a), $v^i \leq v'_i + v'_2$, so $\pi_i$ is admissible for $v'_i + v'_2$ with respect to $J_{b_t}$.

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

Then: (f) $v(\pi_i)[J] = 0$. Indeed by (a) $v^i \leq v'_1 + v'_2$. For every $j \in J$, by the definition of covering derivation $v^i[j]$ would be infinite if $v'_1[j] < v'_2[j]$, and this would contradict the fact that $j$ is in $I$. So $v^i[j] = v'_1[j] + v'_2[j]$. Since $J \subseteq J_{b,t}$, $\delta = v(\pi_i)$ and $v_{b,t}[J_{b,t}] = v'_i[J_{b,t}]$ by definition of $J_{b,t}$, so $\delta[j] = v'_i[j]$. Using (d) and $J \subseteq J_{b,t}$, we obtain $v^i[j] + v(\pi_i)[J] = v'_i[j] + \delta[j] = v'_i[j] + v'_i[j] = v'_i[j] + v'_2[j] = v^i[j]$. Since $J \subseteq J_1 \cap J_2$ by (e) $v^i[j]$ is finite for every $j \in J$, so (f) obtains.

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

Finally: (h) $v(\pi_i)[i] \geq 1$. Indeed, by (b) $v^i[i] \leq v'_i[i] + v'_2[i]$. Since $i \in I_a \subseteq J_{b,t}$, $v'_2[i]$ is finite, hence equal to $v(\pi_i)[i] = \delta[i]$, so $v^i[i] \leq v'_i[i] + \delta[i]$. Since $i \in I_a \subseteq J_{b,t}$, (d) applies, hence $v^i[i] + v(\pi_i)[i] = v'_i[i] + v'_2[i] = v^i[i]$. Since $v^i[i]$ is finite by (e) and the fact that $i \in J_1 \cap J_2$, (h) obtains.

**Building the path $\pi$.** Let $\pi$ be the concatenation of all linear paths $\pi_i$ when $i$ ranges over $I_a$, in any order. This is a linear path from $P$ to $P$, too. Since $v(\pi_i)[J_1 \cap J_2] \geq 0$ by (g), and each $\pi_i$ is admissible for $v'_1 + v'_2$ with respect to $J_{b,t}$, it is easy to see that $\pi$ is admissible for $v'_1 + v'_2$ with respect to $J_1 \cap J_2$. Then, by (f), $v(\pi)[J] = 0$. Since $v(\pi_i)[i] \geq 1$ by (h) for all $i \in I_a$ and $v(\pi_i)[I_a] \geq 0$ by (g) and the fact that $I_a \subseteq J_1 \cap J_2$, we obtain that $v(\pi)[I_a] = 0$.

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

Recall we have already obtained vectors $v_1$ and $v_2$ such that $P_1(v_1)$ and $P_2(v_2)$ are derivable, $v_1[J_1] = v'_1[J_1]$ and $v_2[J_2] = v'_2[J_2]$, using the induction hypothesis. We would like to enforce $v'_1[I_1] \geq K_1$ and $v'_2[I_2] \geq K_1$ to proceed. However the constant $K_1$ depends on $\pi$, which depends on $\pi_i$, $i \in I_a$, which we built by concatenating $\pi'_i$ with some edge labeled by $v_{b,t}$: each new choice of $\pi_i$, $v_2$ may lead to a different constant $K_1$, so that we cannot assume that $v'_1[I_1] \geq K_1$ and $v'_2[I_2] \geq K_1$.

Instead, we invoke the induction hypothesis again, and find two new vectors $v'_1$ and $v'_2$ such that $P_1(v'_1)$ and $P_2(v'_2)$ are derivable, $v'_1[I_1] = v'_1[I_1]$ and $v'_1[I_1] \geq K_1$, and $v'_2[I_2] = v'_2[I_2]$ and $v'_2[I_2] \geq K_1$. Clearly $P(v'_1 + v'_2)$ is derivable from $P_1(v'_1), P_2(v'_2)$, and the clause $P(x + y) = P_1(x), P_2(y)$.

Since $v'_1[I_1] \geq K_1$ and $v'_2[I_2] \geq K_1$, we infer $v^i[I_1 \cup I_2] + v'_i[I_1 \cup I_2] \geq K_1$. By the definition of $K_1$, for every prefix $\pi'$ of $\pi^K$, $v^i[I_1 \cup I_2] + v_i[I_1 \cup I_2] + v(\pi')[I_1 \cup I_2] \geq 0$, so $\pi^K$ is admissible for $v'_1 + v'_2$ with respect to $I_1 \cup I_2$. 

Since $v_1'[J_1] = v_2'[J_1]$, $v_2'[J_2] = v_2'[J_2]$, and $\pi^K$ is admissible for $v'_1 + v'_2$ with respect to $J_1 \cap J_2$, $\pi^K$ is also admissible for $v'_1 + v'_2$ 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 $\{1, \ldots, p\}$, it follows that $\pi^K$ is admissible for $v'_1 + v'_2$.

This implies that $P(v'_1 + v'_2 + v(\pi^K))$ is derivable from $\pi'$.

**Building the fact $v$ from $\pi$.** Let therefore $v = v'_1 + v'_2 + v(\pi^K)$. It remains to show that $v[I] \geq K$ and $v[J] = v'[J]$.

The second claim follows from the fact that $v(\pi)[J] = 0$, from which we infer $v(\pi^K)[J] = 0$, hence $v[J] = v'_1[J] + v'_2[J] + v(\pi^K)[J] = v'_1[J] + v'_2[J] = v'_3[J]$ (since $v'_1[J_1] = v'_1[J_1]$ and $J \subseteq J_1$, and $v'_2[J_2] = v'_2[J_2]$ and $J \subseteq J_2 = v'[J]$ (since $v'[J]$ has only finite components, and by definition finite components of $v'$ are sums of corresponding components of $v'_1$ and $v'_2$).

The first claim will follow from the two sub-claims $v[I_1 \cup I_2] \geq K$ and $v[I_2] \geq K$, since $I = I_1 \cup I_2 \cup I_0$. For the first sub-claim, recall that $v'_1[I_1 \cup I_2] + v'_2[I_1 \cup I_2] \geq K$. By the definition of $K_1$, $v'_1[I_1 \cup I_2] + v'_2[I_1 \cup I_2] + v(\pi^K)[I_1 \cup I_2] \geq K$, i.e., $v[I_1 \cup I_2] \geq K$. For the second sub-claim, recall that $v(\pi^K)[I_0] \geq K$, and $v'_1[J_1] = v'_1[J_1]$ (in particular, $v'_1[I_0] = v'_1[I_0]$, since $I_0 \subseteq J_1$), and $v'_2[J_2] = v'_2[J_2]$ (so $v'_2[I_0] = v'_2[I_0]$, since $I_0 \subseteq J_2$). So $v[I_0] = v'_1[I_0] + v'_2[I_0] + v(\pi^K)[I_0] \geq v'_1[I_0] + v'_2[I_0] + K \geq K$, since $v'_1 + v'_2 = v' \geq 0$.

**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, $\Delta$ a covering derivation. Suppose that $\Delta_2$ is a subderivation of $\Delta$ of the form

\[ \vdots \Delta_1 \]
\[ P_1(v'_1) \]
\[ \vdots \]
\[ P_2(v'_2) \]

containing a (not necessarily proper) subderivation $\Delta_1$. For any index $i$, if $v'_1[i] = \infty$, then $v'_2[i] = \infty$. We say that $v'_2$ has at least as many infinite coordinates as $v'_1$. In case additionally $v'_1[i] \neq \infty$ and $v'_2[i] = \infty$ for some $i$, we say that $v'_2$ has more infinite coordinates than $v'_1$. Otherwise, $v'_1$ and $v'_2$ have the same infinite coordinates.

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

\[ \vdots \Delta_0 \]
\[ P(v'_0) \]
\[ \vdots \]
\[ P(v') \]

with the same $P$, and such that $v' > v'_0$, by which we mean that $v'_i[i] \geq v'_0[i]$ for all $i$ and $v'[i] > v'_0[i]$ for some $i$. Then $v'$ has more infinite coordinates than $v'_0$. 
Proof: Assume the contrary. From Remark \[1\], \(v'\) and \(v'_0\) have the same infinite coordinates. Clearly \(\Delta\) must have been constructed by using Rule 2 or Rule 3 of Definition 1. Accordingly we have the following two cases:

- \(\Delta\) is of the form
  \[
  \frac{\begin{array}{c}
  \vdots \\
  \Delta_1 \\
  \frac{P_1(v'_1)}{P(v')} \\
  P(x + \delta) \leq P_1(x)
  \end{array}}{\vdots \\
  \Delta_2}
  \]
  Again from Remark \[1\], \(v'_1\) has the same infinite coordinates as \(v'\). Then from the construction in Rule 2 of Definition \[1\] we must have \(v' = v'_1 + \delta\). Since \(v' > v'_0\), it follows that \(v'_1 + \delta \geq v'_0\) and for some \(i\), \(v'_1[i] + \delta[i] > v'_0[i]\). But Rule 2 of Definition \[1\] entails that \(v'[i] = \infty\), while \(v'_0[i] \neq \infty\) since \(v'_0[i] < v'_1[i] + \delta[i]\), contradicting the fact that \(v'\) and \(v'_0\) have the same infinite coordinates.

- \(\Delta\) is of the form
  \[
  \frac{\begin{array}{c}
  \vdots \\
  \Delta_1 \\
  \frac{P_1(v'_1)}{P(v')} \\
  P(x + \delta) \leq P_1(x), P_2(y)
  \end{array}}{\vdots \\
  \Delta_2}
  \]
  where we may assume without loss of generality that \(\Delta_0\) is a subderivation of \(\Delta_1\). From Remark \[1\], \(v'\) and \(v'_0\) have the same infinite coordinates. Then from the construction in rule 3 of Definition \[1\] \(v' = v'_1 + v'_2\). Since \(v' > v'_0\), it follows that \(v'_1 + v'_2 \geq v'_0\) and \(v'_1[i] + v'_2[i] > v'_0[i]\) for some \(i\), so by rule 3 of Definition \[1\] \(v'[i] = \infty\). As above, \(v'_0[i] \neq \infty\), leading to a contradiction. \(\square\)

Define the height \(H(\Delta)\) of the covering derivation \(\Delta\) 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 \(\Delta\) is 1 if \(\Delta\) was created by rule 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 \(\subseteq\) by \(\Delta_{ni} \subseteq \Delta_{nj}\) if and only if \(m < n\), or \(m = n\) and \(i \leq j\).

**Lemma 3** For a branching VASS \(\mathcal{V}'\), the total ordering \(\subseteq\) 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 \(\Delta\), there are only finitely many covering derivations \(\Delta_1\) 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 $\Delta$ is constructed using the derivation $\Delta_1$ as defined in Rule 2 of Definition 1 and $\Delta_1$ has been added in the forest, and $\Delta$ has not been added. Then we add $\Delta$ as a child of $\Delta_1$.

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

   This is the crux of the proof: by choosing exactly one of $\Delta_1, \Delta_2$ here, each node will have at most one parent, so we are indeed building a forest (not a jungle). The particular choice of $\Delta'$ 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 $\Delta$, 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 $\Delta_1$ such that $\Delta_1 \subseteq \Delta$. Then by König's lemma, there would be an infinite path in the component tree, consisting of covering derivations

$$\ldots : \Delta_1 : \Delta_2 : \ldots : \Delta_k : \ldots$$

By construction, for every $k \geq 1$, $P_k (v'_k)$ is one of the premises of the last rule used in deriving $P_{k+1} (v'_{k+1})$ in $\Delta_{k+1}$. In particular, whenever $k < k'$, $\Delta_k$ 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 $P(v'_{i_1}), P(v'_{i_2}), \ldots, P(v'_{i_k}), \ldots$, with $1 \leq i_1 < i_2 < \ldots < i_k < \ldots$. 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 $P(v'_{j_1}), P(v'_{j_2}), \ldots, P(v'_{j_k}), \ldots$, with $1 \leq j_1 < j_2 < \ldots < j_k < \ldots$, such that $v'_{j_1} \leq v'_{j_2} \leq \ldots \leq v'_{j_k} \leq \ldots$ This sequence cannot stabilize, that is, there is no $k \geq 1$ such that $v'_{j_k} = v'_{j_{k+1}}$. Otherwise $\Delta_{j_{k+1}}$ would be of the form

$$\ldots : \Delta_k : P(v'_{j_k}) : \ldots : P(v'_{j_{k+1}})$$

where $v'_{j_k} = v'_{j_{k+1}}$. Since $P(v'_{j_{k+1}})$ occurs twice in it, no rule of Definition 1 applies, which entails that $\Delta_{j_{k+1} + 1}$ cannot exist, a contradiction.
So $v'_k < v'_{k+1}$ for every $k \geq 1$. $\Delta_{k+1}$ is of the form

\[
\begin{align*}
\vdots & \Delta_k \\
P(v'_k) & \\
\vdots & \\
P(v'_{k+1})
\end{align*}
\]

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

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 $\{v[J]|v \in N^P$ and $\exists P \in S$ such that $P(v)$ is derivable from $\mathcal{V}\}$ is finite. It is now clear that $\mathcal{V}$ is $S,J$-bounded if and only if all its covering derivations with some conclusion $P(v')$, $P \in S$, are such that $v'[i] \neq \infty$ for all $i \in J$, and this is decidable by Theorem [1]. The \textbf{coverability problem} for $\mathcal{V}$, tuple $v_0$, set of states $P$ and subset $J$ of $\{1, \ldots, p\}$, asks whether there is a fact $P(v)$ derivable from $\mathcal{V}$ such that $P \in S$ and $v[J] \geq v_0[J]$. This is equivalent to testing whether some covering derivation ends in some generalized fact $P(v')$ with $P \in S$ and $v'[J] \geq v_0[J]$, which is decidable by Theorem [1]. The \textbf{emptiness problem} asks whether there is any tuple $v$ and any $P \in S$ such that $P(v)$ 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 $\Sigma$ of function symbols with fixed arities and an equational theory $E$ over terms built using symbols from $\Sigma$, an $E$-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 $P$, $P_1, \ldots, P_n$ are (a.k.a. states), and $t, t_1, \ldots, t_n$ are terms built from symbols in $\Sigma$ and variables $x, y, z, \ldots$. 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 \textbf{derivable} ground atoms are defined using the rules:

\[
\frac{P_1(t_1) \ldots P_n(t_n)}{P(t)} \quad (P(t) \leftarrow P_1(t_1), \ldots, P_n(t_n) \in \mathcal{P})
\]

where $\sigma$ is a ground substitution, and $=_E$ is the congruence on terms induced by the theory $E$. The language \textbf{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 $E$-tree automata, by letting $E$ be the empty theory and by suitably restricting the form of clauses.

Consider a signature $\Sigma$ 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]:

\[
P(x + y) \iff P_1(x), P_2(y) \quad (7) \]
\[
P(a) \quad (9)
\]
\[
P(x) \iff P_1(x) \quad (8)
\]
\[
P(0) \quad (10)
\]

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:

\[
P(x) \iff P_1(x + y) \quad (11)
\]

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

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

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):

<table>
<thead>
<tr>
<th>Clause of input automaton</th>
<th>Clauses of output automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P(0) )</td>
<td>( P(0) )</td>
</tr>
<tr>
<td>( P(a) )</td>
<td>( P(a) )</td>
</tr>
<tr>
<td>( P(x) \iff P_1(x) )</td>
<td>( P(x) \iff P_1(x) )</td>
</tr>
<tr>
<td>( P(x + y) \iff P_1(x), P_2(y) )</td>
<td>( P(x + y) \iff P_1(x), P_2(y) )</td>
</tr>
<tr>
<td>( P(x) \iff Q(x + y) )</td>
<td>( P(x) \iff Q^*(x) )</td>
</tr>
</tbody>
</table>

where the new states \( P^* \) 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 \( R(x + (0, \ldots, 0, -1, 0, \ldots, 0)) \iff R'(x) \) 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 \( L_\prec(\nu) = \{\nu' \in \mathbb{N}^p \mid (0, \ldots, 0) < \nu' < \nu\} \) is Presburger-definable, hence semilinear. We will use \( L_\prec(\nu) \) to denote the corresponding set \( \{\sum_{i=1}^{p} n_i a_i \mid (n_1, \ldots, n_p) \in L_\prec(\nu)\} \).
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 $P(x) \iff Q(x+y)$ is any clause then the set of terms accepted at $P$ using this clause is $L_{0}^{Q} = \bigcup L_{n}(\nu)$ where the union is taken for all $\nu$ 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 $\nu$’s is finite, $L_{0}^{Q}$ is semilinear, and can be accepted at a new state $Q'$ using other new states, and only clauses of constant-only AC automata. Then we can replace the clause $P(x) \iff Q(x+y)$ by the clause $P(x) \iff Q'(y)$.

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 $P(x) \iff P_{1}(x+y), P_{2}(y)$ 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 $P(x-y) \iff P_{1}(x), P_{2}(y)$, 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 $\nu$ 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 $P(x+y) \iff P_{1}(x), P_{2}(y)$ and $P(x+\delta_{i}) \iff P(x)$ 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:

$$
P_3(x - y) \leftarrow P_1(x), P_2(y)
$$

$$
P_0((0, \ldots, 0))
$$

$$
P(x - y) \leftarrow P_0(x), P_3(y)
$$

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 3). 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 $\text{AC}$ automata considered in [Ver03c, Ver03b] with so-called standard $+\text{-}push$ clauses actually reduce to the case without standard $+\text{-}push$ clauses. This seems to require non-primitive-recursive time, however, contrarily to the $\text{ACU}$ 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 $+\text{-}push$ 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


Karp-Miller Trees for a Branching Extension of VASS


