HAL
open science

Hyper-Ackermannian Bounds for Pushdown Vector Addition Systems

Jérôme Leroux, M. Praveen, Grégoire Sutre

To cite this version:

Jérôme Leroux, M. Praveen, Grégoire Sutre. Hyper-Ackermannian Bounds for Pushdown Vector Addition Systems. CSL-LICS 2014 - Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science, Jul 2014, Vienna, Austria. pp.Article 63, 10.1145/2603088.2603146 . hal-00989109

HAL Id: hal-00989109

https://hal.science/hal-00989109v1

Submitted on 9 May 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.

Hyper-Ackermannian Bounds for Pushdown Vector Addition Systems

Jérôme Leroux
LaBRI, UMR CNRS 5800
University of Bordeaux
Talence, France

M. Praveen
LaBRI, UMR CNRS 5800
University of Bordeaux
Talence, France

Grégoire Sutre
LaBRI, UMR CNRS 5800
University of Bordeaux
Talence, France

***Abstract*—This paper studies the boundedness and termination problems for vector addition systems equipped with one stack. We introduce an algorithm, inspired by the Karp & Miller algorithm, that solves both problems for the larger class of well-structured pushdown systems. We show that the worst-case running time of this algorithm is hyper-Ackermannian for pushdown vector addition systems. For the upper bound, we introduce the notion of bad nested words over a well-quasi-ordered set, and we provide a general scheme of induction for bounding their lengths. We derive from this scheme a hyper-Ackermannian upper bound for the length of bad nested words over vectors of natural numbers. For the lower bound, we exhibit a family of pushdown vector addition systems with finite but large reachability sets (hyper-Ackermannian).**

I. Introduction

Vector addition systems (VAS) are an important class of well-structured transition systems. They are powerful enough to model concurrent infinite-state systems but still many important properties are decidable, like boundedness, termination and reachability. Adding more power to VAS while retaining decidability is an active area of research (see, e.g., [12], [15], [24], [10], [17], [4]). Techniques developed for VAS are often extended to the more general class of well-structured systems. In this work, we extend the power of VAS by adding one stack. We give decidability and complexity results, and show how these fit in the general scheme of well-structured systems.

It is well known that a finite-state machine equipped with two stacks is Turing powerful. Replacing one of the stacks by multiple stacks over a singleton stack alphabet (and without bottom test) will result in VAS with one stack, which we study here. This model is close to the border of undecidability and very little is known about the decidability of many problems. Here we prove that the boundedness and termination problems are decidable for this model. The basis of these results is the reduced reachability tree, which is also the basis for many results about VAS and well-structured systems. Our main contributions are twofold.

First, we introduce well-structured pushdown systems, of which pushdown VAS are an important class. We extend the well-known *reduced reachability tree* technique for well-structured transition systems [14] to the pushdown case. The reduced reachability tree allows us to decide the boundedness and termination problems for well-structured pushdown systems.

Second, we give hyper-Ackermannian upper and lower bounds for the size of the reduced reachability trees of pushdown VAS. For the upper bound, we introduce the notion of bad nested words over a well-quasi-ordered set, and we provide a general scheme of induction for bounding their lengths. We derive from this scheme a hyper-Ackermannian upper bound for the length of bad nested words over vectors of natural numbers. For the lower bound, we exhibit a family of pushdown vector addition systems with finite but large reachability sets (hyper-Ackermannian).

Many decidability results about well-structured systems depend on the behavior of well-quasi-orders. In the presence of a stack, making these work require new insights, as done in our results mentioned above. While results like Dickson's lemma and Higman's lemma can prove decidability of some algorithms, deriving complexity bounds for such algorithms has recently received attention [9], [13], [18]. Such bounds help to compare the power of different models and understand the structure of the models under study. Again, making these work in the presence of a stack require substantial extensions, as we have done for deriving bounds for the size of the reduced reachability tree. In particular, we extend the framework for deriving bounds on bad sequences [13] to bad nested sequences [3].

The upper and lower bounds that we derive in this paper for the size of the reduced reachability tree do not apply for the complexity of the problems themselves, since there may be better algorithms. The best known lower bound is the non-elementary lower bound given by [19]. Very few upper bounds are known and none of them match this lower bound. Closing these gaps are challenging open problems.

Related work. In [5], a model called RVASS is introduced, which allows vector additions during stack operations. However, the semantics of this model resembles that of branching VAS [10] rather than that of a VAS with a stack. In [6], a different version of well-structured pushdown systems are introduced, which are systems where both the set of control states and the stack alphabet can be infinite and are well-quasi-ordered. They consider the coverability problem for subclasses. In contrast, the model we consider here has a finite stack alphabet and we consider termination and boundedness for the whole class. In [1], the authors introduce a model called

pushdown automata with data and prove that the control state reachability problem is decidable. In this model again, the set of messages that can be pushed onto the stack can be infinite. The authors introduce a well-quasi-order on this set to decide control state reachability. It is not clear whether the techniques developed for models with infinite stack alphabet can be used in our model and vice-versa. A general framework for well-structured transition systems with auxiliary storage has been introduced in [7]. Decidability is shown for the coverability problem in some subclasses that have a ranking function compatible with the transition relation. Our focus on the other hand is specifically on reduced reachability trees and deriving upper bounds for algorithms based on them. The authors of [7] extend their work in [8] to decide some branching properties of asynchronous programs. They prove that termination and other problems are decidable for our model, provided counters are decreased only when the stack is empty. A similar restriction is imposed by the authors of [26] on their model called multi-set pushdown systems, closely related to our model. We do not impose such restrictions. The fundamental techniques used for deriving Ackermannian and hyper-Ackermannian bounds are developed mostly in [13], [25], on top of which we develop techniques for working with the stack.

Outline. We present in Section III our reduced reachability tree technique[1] for well-structured pushdown systems. Section IV provides a general scheme of induction for bounding the length of bad nested words. Basic notions on fast growing functions are recalled in Section V. We derive in Section VI a multiply-recursive upper bound for the length of bad nested sequences over $\mathbb{N}^d$. Section VII shows that the worst-case size of the reduced reachability tree is hyper-Ackermannian for pushdown VAS.

II. Preliminaries

A *quasi-order* on a set S is a binary relation $\preceq$ on S that is reflexive and transitive. As usual, we write $s \prec t$ when $s \preceq t$ and $t \not\preceq s$. A *partial-order* is an antisymmetric quasi-order.

A *well-quasi-order* (wqo) on S is a quasi-order $\preceq$ such that every infinite sequence $s_0, s_1, s_2, \dots$ in S contains an infinite nondecreasing subsequence $s_{i_0} \preceq s_{i_1} \preceq s_{i_2} \cdots$ (with $i_0 < i_1 < i_2 \cdots$).

The concatenation of a finite sequence (resp. word) σ with a finite or infinite one w is denoted by σ, w (resp. $\sigma \cdot w$). The length of a finite sequence or word σ is denoted $|\sigma|$.

III. Well-Structured Pushdown Systems

A *pushdown system* is a 4-tuple $\mathcal{P} = \langle S, s_{\text{init}}, \Gamma, \Delta \rangle$ where S is a set of *states*, $s_{\text{init}} \in S$ is the *initial state*, Γ is a finite *stack alphabet*, and $\Delta \subseteq S \times \mathtt{Op} \times S$ is a set of *rules*. The set $\mathtt{Op}$ of *operations* is given by $\mathtt{Op} = \{\varepsilon\} \cup \{\mathsf{push}(\gamma), \mathsf{pop}(\gamma) \mid \gamma \in \Gamma\}$. Observe that we do *not* require S to be finite. Informally, a rule $(s, \mathtt{op}, t)$ means that the pushdown system can move from state s to state t by performing the operation $\mathtt{op}$ on the stack. We say that $\mathcal{P}$ is *finitely-branching* if the set $\{(\mathtt{op}, t) \mid (s, \mathtt{op}, t) \in \Delta\}$ is finite for every state $s \in S$. A *simulation relation* for $\mathcal{P}$ is any binary relation R on S satisfying the following condition: for every $(s, \mathtt{op}, t) \in \Delta$ and $s' \in S$ such that $(s, s') \in R$, there exists $t' \in S$ such that $(s', \mathtt{op}, t') \in \Delta$ and $(t, t') \in R$.

We define the operational semantics of a pushdown system $\mathcal{P} = \langle S, s_{\text{init}}, \Gamma, \Delta \rangle$ as follows. A *configuration* is a pair $\chi = (s, \sigma)$ where $s \in S$ is a state and $\sigma \in \Gamma^*$ is a word denoting the contents of the stack. The *initial configuration* is $\chi_{\text{init}} = (s_{\text{init}}, \varepsilon)$. The *transition relation* is the least binary relation $\rightarrow \subseteq (S \times \Gamma^*) \times (S \times \Gamma^*)$ satisfying the following rules:

$$\frac{(s, \varepsilon, t) \in \Delta}{(s, \sigma) \rightarrow (t, \sigma)}$$

$$\frac{(s, \mathsf{push}(\gamma), t) \in \Delta}{(s, \sigma) \rightarrow (t, \sigma \cdot \gamma)} \qquad \frac{(s, \mathsf{pop}(\gamma), t) \in \Delta}{(s, \sigma \cdot \gamma) \rightarrow (t, \sigma)}$$

A *run* is a (finite or infinite) sequence $\chi_0, \chi_1, \chi_2, \dots$ of configurations such that $\chi_0 = \chi_{\text{init}}$ and $\chi_{i-1} \rightarrow \chi_i$ for every index $i > 0$ of the sequence. The *reachability set* is the set of all configurations that occur on some run.

We investigate two verification questions on pushdown systems, namely *termination* and *boundedness*. The former asks whether all runs of a given pushdown system are finite, and the latter asks whether its reachability set is finite. Recall that these questions are decidable for pushdown systems with finitely many states. In our setting, the set of states may be infinite (e.g., $Q \times \mathbb{N}^d$ for pushdown vector addition systems with states, see Section VII). By extending well-known techniques from the theory of well-structured transition systems [14], [2], [16], we obtain a class of pushdown systems for which termination and boundedness are decidable.

Definition III.1. *A* well-structured pushdown system *is a finitely-branching pushdown system* $\mathcal{P} = \langle S, s_{init}, \Gamma, \Delta \rangle$ *equipped with a well-quasi-order* $\preceq \subseteq S \times S$ *on states such that* $\preceq$ *is a simulation relation for* $\mathcal{P}$.

Remark III.2. The monotonicity condition of Definition III.1 matches the one given in [2] for infinite-state *labeled* transition systems. In our setting, the labels of [2] are operations $\mathtt{op} \in \mathtt{Op}$. More general monotonicity conditions have been proposed in [16] for *unlabeled* infinite-state transition systems. The condition of Definition III.1 corresponds to the *strong compatibility* requirement of [16].

The *reachability tree* of a well-structured pushdown system $\mathcal{P} = \langle S, s_{\text{init}}, \Gamma, \Delta, \preceq \rangle$ is the directed unordered tree defined as follows. Nodes of the tree are labeled by configurations of $\mathcal{P}$. The root r is labeled by the initial configuration χ_{init}, written $r : \chi_{\text{init}}$. Each node $n : \chi$ has one child $m : \psi$ for each configuration ψ such that $\chi \rightarrow \psi$. Note that the reachability tree of $\mathcal{P}$ is finitely branching.

The reachability tree is usually infinite for well-structured pushdown systems. We show how to truncate it while preserving enough information to decide termination and boundedness. A node $n : (s, \sigma)$ *subsumes* a node $m : (t, \tau)$ if n is

[1] This section motivates the developments of the following sections, but its results are only used in Section VII.

a proper ancestor of m, $s \preceq t$ and σ is a prefix of σ' for every node $n' : (s', \sigma')$ on the path from n to m. The *reduced reachability tree* of $\mathcal{P}$ is the largest prefix of the reachability tree such that every subsumed node has no child.

Proposition III.3. *The reduced reachability tree of any well-structured pushdown system is finite.*

Proof. Consider a well-structured pushdown system $\mathcal{P} = \langle S, s_{\text{init}}, \Gamma, \Delta, \preceq \rangle$. By contradiction, assume that its reduced reachability tree $\mathcal{T}$ is infinite. Recall that this tree is finitely branching. By König's lemma, $\mathcal{T}$ contains an infinite branch $n_0 : (s_0, \sigma_0) \to n_1 : (s_1, \sigma_1) \to n_2 : (s_2, \sigma_2) \to \cdots$. Observe that $\sigma_0 = \varepsilon$ since n_0 is the root of the tree. We extract from $n_0, n_1, n_2, \ldots$ an infinite subsequence $n_{i_0}, n_{i_1}, n_{i_2}, , \ldots$ defined as follows:

$$\begin{array}{rcl} i_0 & = & 0 \\ i_{k+1} & = & \min \{ i > i_k \mid \forall j > i_k, |\sigma_i| \leq |\sigma_j| \} \end{array}$$

By definition, $|\sigma_{i_k}| \leq |\sigma_j|$ for all $j, k \in \mathbb{N}$ with $i_k \leq j$. It follows from the operational semantics of pushdown systems that σ_{i_k} is a prefix of σ_j for all $j, k \in \mathbb{N}$ with $i_k \leq j$. Since $\preceq$ is a well-quasi-order, there exists $k < l$ such that $s_{i_k} \preceq s_{i_l}$. We obtain that the node n_{i_k} subsumes the node n_{i_l}. This entails that n_{i_l} has no child in $\mathcal{T}$, which is impossible. □

Proposition III.4. *A well-structured pushdown system $\mathcal{P}$ has an infinite run if, and only if, its reduced reachability tree contains a subsumed node.*

Proof. If $\mathcal{P}$ has an infinite run, then its reachability tree contains an infinite branch. Since the reduced reachability tree of $\mathcal{P}$ is finite, this infinite branch necessarily contains a subsumed node. Hence, the reachability tree of $\mathcal{P}$ contains a subsumed node, and so does its reduced reachability tree. For the converse, assume that the reachability tree of $\mathcal{P}$ contains a subsumed node. The path from the root to this subsumed node yields a run

$$(s_0, \sigma_0) \xrightarrow{\mathtt{op}_1} \cdots \xrightarrow{\mathtt{op}_k} (s_k, \sigma_k) \xrightarrow{\mathtt{op}_{k+1}} \cdots \xrightarrow{\mathtt{op}_l} (s_l, \sigma_l)$$

such that $k < l$, $s_k \preceq s_l$ and σ_k is a prefix of σ_j for all $k \leq j \leq l$. Here, we have decorated each transition of the run with its associated operation. Let $z_k, \ldots, z_l$ be the words satisfying $\sigma_j = \sigma_k \cdot z_j$ for all $k \leq j \leq l$. Define $(t_k, \tau_k) = (s_l, \sigma_l)$. Since $s_k \preceq t_k$ and $\preceq$ is a simulation relation for $\mathcal{P}$, we derive that there exists $t_{k+1}, \ldots, t_l$ such that, for every $k < j \leq l$, $s_j \preceq t_j$ and $(t_{j-1}, \mathtt{op}_j, t_j) \in \Delta$. Observe that the sequence of operations $\mathtt{op}_{k+1}, \ldots, \mathtt{op}_l$ never pops from σ_k. Hence, we may also perform $\mathtt{op}_{k+1}, \ldots, \mathtt{op}_l$ starting from $\sigma_l = \tau_k \cdot z_k$. So the above run may be extended by appending

$$(s_l, \sigma_l) = (t_k, \tau_k \cdot z_k) \xrightarrow{\mathtt{op}_{k+1}} \cdots \xrightarrow{\mathtt{op}_l} (t_l, \tau_k \cdot z_l)$$

We obtain an infinite run by repeating this construction. □

The boundedness problem requires stronger monotonicity requirements. A well-structured pushdown system $\mathcal{P} = \langle S, s_{\text{init}}, \Gamma, \Delta, \preceq \rangle$ is called *strict* when $\preceq$ is a partial order and $\prec$ is a simulation relation. Similarly, in the reachability tree of $\mathcal{P}$, a node $n : (s, \sigma)$ *strictly subsumes* a node $m : (t, \tau)$ if n subsumes m, and $s \prec t$ or $|\sigma| < |\tau|$.

Proposition III.5. *A strict well-structured pushdown system $\mathcal{P}$ has an infinite reachability set if, and only if, its reduced reachability tree contains a strictly subsumed node.*

Proof. Assume that $\mathcal{P}$ has an infinite reachability set. Let $\mathcal{T}$ be the largest prefix of its reachability tree such that, on each branch, all nodes have distinct labels. The tree $\mathcal{T}$ is infinite since every configuration in the reachability set is the label of some node in $\mathcal{T}$. By König's lemma, it follows that the reachability tree of $\mathcal{P}$ contains an infinite branch where all nodes have distinct labels. Since the reduced reachability tree of $\mathcal{P}$ is finite, this infinite branch necessarily contains two nodes $n : (s, \sigma)$ and $m : (t, \tau)$, both contained in the reduced reachability tree, such that n subsumes m. Recall that $(s, \sigma) \neq (t, \tau)$. If $|\sigma| < |\tau|$ then n strictly subsumes m. Otherwise, $\sigma = \tau$ since σ is a prefix of τ. It follows that $s \neq t$, which entails that $s \prec t$ as $\preceq$ is a partial order. We get, again, that n strictly subsumes m.

For the converse, assume that the reduced reachability tree of $\mathcal{P}$ contains a strictly subsumed node. The path from the root to this subsumed node yields a run

$$(s_0, \sigma_0) \xrightarrow{\mathtt{op}_1} \cdots \xrightarrow{\mathtt{op}_k} (s_k, \sigma_k) \xrightarrow{\mathtt{op}_{k+1}} \cdots \xrightarrow{\mathtt{op}_l} (s_l, \sigma_l)$$

such that $k < l$, $s_k \preceq s_l$, σ_k is a prefix of σ_j for all $k \leq j \leq l$, and $s_k \prec s_l$ or $|\sigma_k| < |\sigma_l|$. With the same arguments as in the proof of Proposition III.4, we extend this run into an infinite one by iterating the sequence of operations $\mathtt{op}_{k+1}, \ldots, \mathtt{op}_l$. If $s_k \prec s_l$ then the resulting infinite run visits infinitely many states (as $\preceq$ is a partial order). If $|\sigma_k| < |\sigma_l|$ then it visits infinitely many stack contents. In both cases, we obtain that $\mathcal{P}$ has an infinite reachability set. □

IV. Bad Nested Sequences Over a WQO Set

Nested words [3] offer a convenient formalism to study some properties of pushdown systems. Since we are dealing with sets of reachable configurations, our notation is slightly different from that of [3], where the main concern is the theory of languages. For us, an alphabet S is usually the set of states of a pushdown system, and the content of the stack is abstracted by its length.

Definition IV.1. *A* nested sequence *over a set S is a (finite or infinite) sequence $(s_0, h_0), (s_1, h_1), \ldots$ of elements in $S \times \mathbb{N}$ satisfying $h_0 = 0$ and $h_j \in h_{j-1} + \{-1, 0, 1\}$ for every index $j > 0$ of the sequence.*

Example IV.2. The empty sequence is a nested sequence. The nested sequence $(a, 0), (b, 1), (c, 2), (d, 1)$ over $\{a, b, c, d\}$ can be depicted as follows:

$$\begin{array}{lllll} 2- & & & c & \\ 1- & & b & & d \\ 0- & a & & & \end{array}$$

Remark IV.3. In a well-structured pushdown system, every run can be seen as a nested sequence, by mapping each

configuration (s, σ) to the pair $(s, |\sigma|)$. Similarly, every branch of the reachability tree can be seen as a nested sequence.

Definition IV.4. *A nested sequence* $(s_0, h_0), (s_1, h_1), \ldots$ *over a quasi-ordered set* $(S, \preceq)$ *is said to be* good *if there exists* $i < j$ *such that* $s_i \preceq s_j$ *and* $h_{i+1}, \ldots, h_j \geq h_i$. *A* bad *nested sequence is one that is not good.*

Example IV.5. The following nested sequence over $(\mathbb{N}, \leq)$ is bad:

$$\begin{array}{lcccccccccccc}
3- & & & & 0 & & & & & & & & \\
2- & & & 1 & & 0 & & & & 0 & & & \\
1- & & 2 & & & & 1 & & 1 & & 0 & & 0 \\
0- & 3 & & & & & & 2 & & & & 1 & & 0
\end{array}$$

Remark IV.6. A nested sequence $(s_0, 0), (s_1, 0), \ldots$ is good iff $s_i \preceq s_j$ for some $i < j$. Hence good/bad nested sequences are generalizations of good/bad (non-nested) sequences [13].

Remark IV.7. Consider a branch in the reachability tree of some well-structured pushdown system. The nested sequence associated with the branch is good if, and only if, the branch contains a subsumed node.

Lemma IV.8. *Bad nested sequences over a wqo are finite.*

Proof. Similar to the proof of Proposition III.3. □

The finiteness of the reduced reachability tree presented in the previous section relies on the finiteness of bad nested sequences. In order to derive complexity arguments, bounds on these sequences must be obtained. *Normed* wqo sets provide a framework for deriving such bounds. We first present some simple examples of (non-nested) bad sequences that motivate normed wqo sets.

The following nested sequence shows that the length of bad nested sequences over a wqo cannot be bounded since there exist bad nested sequences over $(\mathbb{N}, \leq)$ of length n for every n.

$$0- \quad 100 \quad 99 \quad 98 \quad \ldots \quad 1 \quad 0$$

Hence, the initial element must be bounded somehow. In the algorithms presented in the previous section, nested sequences starts from the same element, the initial state of the well-structured pushdown system. However, even if the initial element is fixed, the problem is not yet solved as shown by the following bad nested sequence over $(\mathbb{N}^2, \leq)$.

$$0- \quad (0,1) \quad (100,0) \quad (99,0) \quad (98,0) \quad \ldots \quad (1,0) \quad (0,0)$$

This problem appears because the second element of the sequence is arbitrarily large compared to the first one. In order to control the growth of values in a nested sequence, we use the notion of *normed* wqo sets.

Definition IV.9. *A* norm *for a wqo set* $(S, \preceq)$ *is a function* $||.|| : S \mapsto \mathbb{N}$ *such that* $\{s \in S \mid ||s|| \leq n\}$ *is finite for every* n. *The structure* $(S, \preceq, ||.||)$ *is called a* normed wqo.

Example IV.10. The wqo set $(\mathbb{N}^d, \leq)$ is normed by the function that maps any vector to its largest component.

Definition IV.11. *A nested sequence* $(s_0, h_0), (s_1, h_1), \ldots$ *over a* normed *wqo set* $(S, \preceq, ||.||)$ *is said to be* n-controlled *for some* $n \in \mathbb{N}$ *if* $||s_j|| \leq n + j$ *for every index* j *of the sequence.*

While there exist bad nested sequences of arbitrarily long lengths, König's lemma shows that there exists a maximal length for n-controlled bad nested sequences. Indeed, these sequences can be organized in a forest where nodes are labeled by elements in $S \times \mathbb{N}$. By definition of the norm function, this forest is finite branching and the number of roots is finite. Hence, if the forest is infinite, the König's lemma shows that it contains an infinite branch. Since infinite nested sequences are good we get a contradiction.

Let $\mathrm{BAD}_S(n)$ be the set of n-controlled bad nested sequences over S. This set is non empty since it contains the empty sequence. The *maximal length* function $L_S : \mathbb{N} \mapsto \mathbb{N}$ is defined as follows:

$$L_S(n) = \max\{|w| \ \mid \ w \in \mathrm{BAD}_S(n)\}$$

The maximal length function L_S satisfies a recurrence relation that we call the *descent equation*. For this, we need the set $S|_{\leq n}$ defined as follows:

$$S|_{\leq n} = \{s \in S \ \mid \ ||s|| \leq n\}$$

Given $s \in S|_{\leq n}$, let us denote by $\mathrm{BAD}_{S,s}(n)$ the set of n-controlled bad nested sequences starting from $(s, 0)$. This set is non-empty since it contains $(s, 0)$. Let $L_{S,s}(n)$ be the number defined as follows:

$$L_{S,s}(n) = \max\{|w| \ \mid \ w \in \mathrm{BAD}_{S,s}(n)\}$$

The following equality is immediate for every $n \in \mathbb{N}$ such that $S|_{\leq n}$ is non empty:

$$L_S(n) = \max_{s \in S|_{\leq n}} L_{S,s}(n)$$

Let $(S/s, \preceq, ||.||)$ be the normed wqo set where S/s (called the *quotient* of S by $s \in S$) is defined as $S/s = \{t \in S \mid s \not\preceq t\}$. In the following, we prove the following equality, called the *descent equation* for every $n \in \mathbb{N}$ and for every $s \in S|_{\leq n}$:

$$L_{S,s}(n) = 1 + L_{S/s}(n+1) + L_{S/s}(n + 1 + L_{S/s}(n+1))$$

Let us first prove that $L_{S,s}(n)$ is bounded by the above term. We pick a n-controlled bad nested sequence w over S starting from $(s, 0)$ with $s \in S|_{\leq n}$. The nested sequence w can be decomposed into $w = (s, 0)w'$ where w' is a sequence of elements in $S \times \mathbb{N}$. Since w is bad, w' is in fact a sequence of elements in $(S/s) \times \mathbb{N}$. Let us introduce the maximal prefix w'' of w' such that w'' is a sequence of elements in $S \times (\mathbb{N} \backslash \{0\})$. Let us show that $|w'| \leq L_{S/s}(n+1)$. The sequence w'' can be decomposed into $w'' = (s_1, h_1), \ldots, (s_k, h_k)$ where $k = |w''|$. Note that $(s_1, h_1 - 1), \ldots, (s_k, h_k - 1)$ is a $(n+1)$-controlled bad nested sequence over the wqo set S/s. Thus, $|w''| \leq L_{S/s}(n+1)$. Now, let us consider the sequence w''' such that $w' = w'', w'''$. Let us prove that $|w'''| \leq L_{S/s}(n +$

$1+k)$. Observe that if w''' is empty the relation is immediate. So, let us assume that w''' is non empty. By maximality of w'', we deduce that w''' starts with an element in $S \times \{0\}$. Thus w''' is a $(n+1+k)$-controlled bad nested sequence over S/s. We get $|w''| \leq L_{S/s}(n+1+k)$. Since $L_{S/s}$ is monotonic and $k \leq L_{S/s}(n+1)$, we deduce that $|w'''|$ is bounded by $L_{S/s}(n+1+L_{S/s}(n+1))$. Hence $|w|$ is bounded by $1+L_{S/s}(n+1)+L_{S/s}(n+1+L_{S/s}(n+1))$. We have proved one inequality of the descent equation, namely that $L_{S,s}(n)$ is bounded by $1+L_{S/s}(n+1)+L_{S/s}(n+1+L_{S/s}(n+1))$ for every $n \in \mathbb{N}$ and for every $s \in S|_{\leq n}$.

To prove the converse inequality, we need to concatenate bad nested sequences. The concatenation will be possible thanks to the following lemma.

Lemma IV.12. *Every $w \in \mathrm{BAD}_S(n)$ with $|w| = L_S(n)$ is either empty or the last element is in $S \times \{0\}$.*

Proof. If $L_S(n) = 0$, then the proof is immediate. Assume that $L_S(n) > 0$ and let us consider a n-controlled bad nested sequence w such that $|w| = L_S(n)$. Let (s,h) be the last element of w. By contradiction, suppose that $h > 0$. It is readily seen that the sequence obtained by appending $(s, h-1)$ to w is also a n-controlled bad nested sequence. This contradicts the maximality of w. We derive that $h = 0$, which concludes the proof of the lemma. □

Let us now complete the proof of the descent equation, by showing that $1+L_{S/s}(n+1)+L_{S/s}(n+1+L_{S/s}(n+1))$ is bounded by $L_{S,s}(n)$ for every $n \in \mathbb{N}$ and for every $s \in S|_{\leq n}$. Pick $n \in \mathbb{N}$ and $s \in S|_{\leq n}$. Let $n_1 = L_{S/s}(n+1)$. There exists $w_1 \in \mathrm{BAD}_{S/s}(n+1)$ with $|w_1| = n_1$. According to Lemma IV.12, if $n_1 > 0$ then the last element of w_1 is in $S \times \{0\}$. Let $n_2 = L_{S/s}(n+1+n_1)$. By definition, there exists w_2 in $\mathrm{BAD}_{S/s}(n+1+n_1)$ with $|w_2| = n_2$. We introduce the nested sequence $w = (s,0), w_1', w_2$ where w_1' is obtained from w_1 by replacing every occurrence of $(t,h) \in S \times \mathbb{N}$ by $(t, h+1)$. Observe that w is a bad nested sequence that is n-controlled. Since $w \in \mathrm{BAD}_{S,s}(n)$, we get $L_{S,s}(n) \geq |w|$. It follows that $1+L_{S/s}(n+1)+L_{S/s}(n+1+L_{S/s}(n+1))$ is bounded by $L_{S,s}(n)$, which concludes the proof of the descent equation.

In order to simplify the presentation of the descent equation, we introduce the function $K_S : \mathbb{N} \mapsto \mathbb{N}$ defined for every $n \in \mathbb{N}$ as follows:

$$K_S(n) = n + L_S(n)$$

Intuitively, $K_S(n)$ explicitly accounts for the starting value n by adding it to $L_S(n)$. It is understood that, like L_S, the function K_S is parametrized by a normed wqo set $(S, \preceq, ||.||)$.

Now, just observe that for every $n \in \mathbb{N}$ and every $s \in S|_{\leq n}$, the descent equation entails that:

$$\begin{aligned} & n + L_{S,s}(n) \\ & \quad = n + 1 + L_{S/s}(n+1) + L_{S/s}(n+1+L_{S/s}(n+1)) \\ & \quad = K_{S/s}(K_{S/s}(n+1)) \end{aligned}$$

In particular, if $S|_{\leq n}$ is non empty, we obtain that:

$$\begin{aligned} K_S(n) &= n + L_S(n) \\ &= n + \max_{s \in S|_{\leq n}} L_{S,s}(n) \\ &= \max_{s \in S|_{\leq n}} n + L_{S,s}(n) \\ &= \max_{s \in S|_{\leq n}} K_{S/s}(K_{S/s}(n+1)) \end{aligned}$$

We have proved the following theorem.

Theorem IV.13. *For every normed wqo set $(S, \preceq, ||.||)$ and for every $n \in \mathbb{N}$, we have:*

$$K_S(n) = \begin{cases} \max_{s \in S|_{\leq n}} K_{S/s}(K_{S/s}(n+1)) & \text{if } S|_{\leq n} \neq \emptyset \\ n & \text{otherwise} \end{cases}$$

V. Fast Growing Functions

The next sections classify the maximal length function in the *fast-growing hierarchy* [20]. We shall work with set-theoretic ordinals less than ω^ω, written in Cantor's Normal Form. Each limit ordinal λ has a *canonical fundamental sequence* $(\lambda_n)_{n \in \mathbb{N}}$, satisfying $\lambda_0 < \lambda_1 < \cdots < \lambda_n < \lambda_{n+1} < \cdots$, with λ being the supremum of this sequence. For limit ordinals below ω^ω, this fundamental sequence is given by

$$\begin{aligned} & (\omega^d.\, a_d + \cdots + \omega^r.\, a_r)_n = \\ & \quad \omega^d.\, a_d + \cdots + \omega^{r+1}.\, a_{r+1} + \omega^r.\, (a_r - 1) + \omega^{r-1}.\, (n+1) \end{aligned}$$

assuming $d \geq r$ and $a_r > 0$.

The family of *fast growing functions* $(F_\alpha)_\alpha$, indexed by ordinals $\alpha \leq \omega^\omega$, is defined as follows:

$$\begin{aligned} F_0(n) &= n+1, \\ F_{\alpha+1}(n) &= F_\alpha^{n+1}(n) = \overbrace{F_\alpha(F_\alpha(\cdots F_\alpha}^{n+1 \text{ times}}(n)\cdots)) \\ F_\lambda(n) &= F_{\lambda_n}(n) \quad \text{if } \lambda < \omega^\omega \text{ is a limit ordinal} \\ F_{\omega^\omega}(n) &= F_{\omega^{n+1}}(n) \end{aligned}$$

The value $F_1(n)$ is linear in n, $F_2(n)$ is exponential in n, and $F_3(n)$ is a tower of exponential of height n. The function F_ω is an Ackermannian function. This function is defined by diagonalization over the ordinals $\alpha < \omega$, since $F_\omega(n) = F_{n+1}(n)$. The hyper-Ackermannian function F_{ω^ω} used in this paper is defined in the same fashion, by diagonalization over the ordinals $\alpha < \omega^\omega$.

Example V.1. The above definition induces:

$$\begin{aligned} F_{\omega^2.4}(8) &= F_{\omega^2.3+\omega.9}(8) \\ &= F_{\omega^2.3+\omega.8+9}(8) \\ &= \underbrace{F_{\omega^2.3+\omega.8+8}(\cdots(F_{\omega^2.3+\omega.8+8}}_{9 \text{ times}}(8))\cdots) \end{aligned}$$

In addition to the usual linear order $\leq$ on ordinals, we will also use a partial order, called *embedding* in [9], and defined as follows. For two ordinals α, β such that $\alpha = \omega^d.\, a_d + \cdots + \omega^0.\, a_0$ and $\beta = \omega^d.\, b_d + \cdots + \omega^0.\, b_0$, α *embeds* in β if $a_i \leq b_i$ for all $i \in \{0, \ldots, d\}$. We write $\alpha \sqsubseteq \beta$ when α embeds in

β. We will use the following monotonicity properties of fast growing functions. These properties are shown by induction over ordinals (see, for instance, Lemmas 2.1 and 2.2 in [9]).

Lemma V.2 (Monotonicity, [9]). *For all ordinals $\alpha, \beta < \omega^\omega$ and for all natural numbers $m, n \in \mathbb{N}$,*

$$\alpha \sqsubseteq \beta \text{ and } m \leq n \quad \Rightarrow \quad m < F_\alpha(m) \leq F_\beta(n)$$

VI. Bad Nested Sequences Over $\mathbb{N}^d$

In this section, we derive upper bounds for the length of bad nested sequences over multiple disjoint copies of $\mathbb{N}^d$. The sum of $c \in \mathbb{N}$ distinct copies of $\mathbb{N}^d$ is the Cartesian product $\{1, \ldots, c\} \times \mathbb{N}^d$, denoted by $c \times \mathbb{N}^d$. This set naturally corresponds to states of vector addition systems with c locations and d counters ranging over the natural numbers. The set $c \times \mathbb{N}^d$ is equipped with the wqo $\preceq_{c\times\mathbb{N}^d}$ defined by $(p, \boldsymbol{u}) \preceq_{c\times\mathbb{N}^d} (q, \boldsymbol{v})$ if $p = q$ and $\boldsymbol{u} \leq \boldsymbol{v}$, and the norm $||.||_{c\times\mathbb{N}^d}$ defined by $||(q, \boldsymbol{v})||_{c\times\mathbb{N}^d} = ||\boldsymbol{v}||_\infty$, where $||\boldsymbol{v}||_\infty$ maps $\boldsymbol{v}$ to its largest component (to zero for the empty vector). In this section, we prove the following theorem.

Theorem VI.1. *For every $c, d \geq 1$ and $n \geq 2$, we have:*

$$L_{c\times\mathbb{N}^d}(n) \leq F_{\omega^d.c}(d.n)$$

Remark VI.2. The previous theorem requires $d \geq 1$. When $d = 0$, the norm of any element in $c \times \mathbb{N}^0$ is zero, so the value $L_{c\times\mathbb{N}^0}(n)$ does not depend on n. Let us denote by k_c this value. As the quotient of $c \times \mathbb{N}^0$ by an element $v \in c \times \mathbb{N}^0$ is equal to $(c-1)\times\mathbb{N}^0$, Theorem IV.13 shows that $k_c = 2k_{c-1}+1$ for every $c \geq 1$. From $k_0 = 0$, we derive $k_c = 2^c - 1$ which proves the following equality:

$$L_{c\times\mathbb{N}^0}(n) = 2^c - 1$$

The rest of this section is the technical part for proving Theorem VI.1. In particular, we assume that $d \geq 1$. Our approach is based on the descent equation introduced in Theorem IV.13. This approach introduces sums of multiple disjoint copies of sets $\mathbb{N}^i$ for various dimensions $i \in \{0, \ldots, d\}$. Let us show this property on the normed wqo set $\mathbb{N}^d$ (no multiple copy). The descent equation introduces the quotients $\mathbb{N}^d/\boldsymbol{v}$ where $\boldsymbol{v} \in \mathbb{N}^d$. A vector $\boldsymbol{x}$ in such a quotient satisfies $\boldsymbol{x}(j) \in \{0, \ldots \boldsymbol{v}(j)-1\}$ for some index j. Hence, the quotient $\mathbb{N}^d/\boldsymbol{v}$ can be embedded into $(\boldsymbol{v}(1) + \cdots + \boldsymbol{v}(d)) \times \mathbb{N}^{d-1}$. By applying recursively the descent equation, starting from $c \times \mathbb{N}^d$, finite sums of disjoint copies of sets $\mathbb{N}^i$ for various dimensions $i \in \{0, \ldots, d\}$ are introduced.

Those finite sums are denoted formally thanks to the notion of *types* introduced in [13]. We refer to this paper for further explanations. A type is a function $\tau : \{0, \ldots, d\} \mapsto \mathbb{N}$, also viewed as a multiset. We associate to a type τ the set $\mathbb{N}^\tau$ defined as follows:

$$\mathbb{N}^\tau = \bigcup_{i=0}^{d} \tau(i) \times \mathbb{N}^i$$

The set $\mathbb{N}^\tau$ is equipped with a wqo $\preceq_\tau$ and a norm $||.||_\tau$ by observing that the sets $\tau(i) \times \mathbb{N}^i$ are pairwise disjoint. More formally, the wqo $\preceq_\tau$ on $\mathbb{N}^\tau$ is defined by $u \preceq_\tau v$ if there exists i such that $u, v \in \tau(i) \times \mathbb{N}^i$ and $u \preceq_{\tau(i)\times\mathbb{N}^i} v$, and the norm $||.||_\tau$ is defined by $||v||_\tau = ||v||_{\tau(i)\times\mathbb{N}^i}$ where i is such that $v \in \tau(i) \times \mathbb{N}^i$.

Since the structure $(\mathbb{N}^\tau, \preceq_\tau, ||.||_\tau)$ is a normed wqo, the function $K_{\mathbb{N}^\tau}$ is well defined. To simplify notations, we denote this function by K_τ. In the sequel, we assume that the reader is familiar with multiset notations. In particular, since a type is a multiset of natural numbers in $\{0, \ldots, d\}$, we denote by $\emptyset$ the empty type that maps any i on zero. Given a type τ such that $\tau(i) \geq 1$, we denote by $\tau - \{i\}$ the type ϱ defined by $\varrho(i) = \tau(i) - 1$ and $\varrho(j) = \tau(j)$ if $j \neq i$.

Following the approach introduced in [13] (explained in the beginning of this section on the quotients $\mathbb{N}^d/\boldsymbol{v}$), the quotients $\mathbb{N}^\tau/v$, where $v \in \mathbb{N}^\tau$, can be embedded into $\mathbb{N}^\varrho$ for types ϱ derived from τ and v. These types are defined as follows. Given a type τ, a dimension $i \in \{0, \ldots, d\}$ such that $\tau(i) \geq 1$ and a natural number $n \geq 0$, the derivation type $\partial_n^i(\tau)$ is defined by:

$$\partial_n^i(\tau) = \tau - \{i\} + n. \sum_{0\leq j<i} \{j\}$$

We obtain the following lemma, whose proof (which can be found in the appendix) is similar to that of [13, Lemma A.3]:

Lemma VI.3. *It holds that:*

$$K_{\mathbb{N}^\tau/v}(m) \leq K_{\partial^i_{d.n}(\tau)}(m)$$

for every type τ, for every $i \in \{1, \ldots, d\}$, for every $v \in \tau(i) \times \mathbb{N}^i$, for every $n \geq ||v||_\tau$, and for every $m \in \mathbb{N}$.

Combining this lemma with Theorem IV.13 and the fact that K_τ is monotonic for all τ, we get the following inequalities, which hold for every natural number $n \geq 0$, and for every type $\tau \neq \emptyset$:

$$K_\tau(n) \leq \max_{i:\tau(i)\geq 1} \{K_{\partial^i_{d.n}(\tau)}(K_{\partial^i_{d.n}(\tau)}(n+1))\}$$

For a type τ, we denote by F_τ the function defined by:

$$F_\tau \quad = \quad F_{\omega^d.\tau(d)+\cdots+\omega^0.\tau(0)}$$

We associate to a type $\tau \neq \emptyset$ and a natural number $n \in \mathbb{N}$ the type $\tau_n = \partial_n^i(\tau)$ where i is the minimal index such that $\tau(i) > 0$. An immediate induction shows the following equality for every type $\tau \neq \emptyset$:

$$F_\tau(n) = F_{\tau_n}^{n+1}(n)$$

In the sequel, we prove that for every type τ, for every $n \geq 2$ we have:

$$K_\tau(n) \leq F_\tau(d.n) \tag{1}$$

Note that Theorem VI.1 is a direct consequence of this inequality with the type $\tau = c.\{d\}$.

We write $\tau \sqsubseteq \varrho$ if $\tau(i) \leq \varrho(i)$ for all i. We will use the lexicographic order $<$ on types, with $\tau(i+1)$ being more significant than $\tau(i)$ for all i. This is a well-founded linear

order, which is the same as the order used for types in [11], [13]. We first prove some results on the family (F_τ).

Lemma VI.4. *For every type τ, for every $i \in \{0, \dots, d\}$ such that $\tau(j) > 0$ implies $j \geq i$, and for every $n \geq 0$:*

$$F_{\tau+\{i\}}(n) \geq F_\tau(2n)$$

Proof. The case $n = 0$ comes from $F_\varrho(0) = 1$ for every type ϱ (the equality is obtained by induction on ϱ with the lexicographic order) . We introduce $\varrho = \tau + \{i\}$. Since $\tau(j) > 0$ implies $j \geq i$, we have $\varrho_n = \tau + n.(\{i-1\} + \cdots + \{0\})$. Lemma V.2 provides $F^n_{\varrho_n}(n) \geq F^n_\emptyset(n) = 2n$. We have:

$$\begin{aligned} F_\varrho(n) &= F^{n+1}_{\varrho_n}(n) \\ &\geq F_{\varrho_n}(F^n_\emptyset(n)) && \text{[Lemma V.2]} \\ &\geq F_\tau(2n) && \text{[Lemma V.2]} \end{aligned}$$

This concludes the proof of the lemma. □

Notice that for every type $\tau \neq \emptyset$ and for every $n \geq 0$, the type τ_n is equal to $\partial^i_n(\tau)$ where i is the minimal index such that $\tau(i) > 0$. The following lemma shows that if we consider a larger index, we get a smaller value.

Lemma VI.5. *For every type $\tau \neq \emptyset$, for every i such that $\tau(i) > 0$, and for every $n \geq 2$, for every $k \geq n$, we have:*

$$F_{\partial^i_n(\tau)}(k) \leq F_{\tau_n}(k)$$

Proof. We decompose τ into $\tau = \tau' + \varrho$ such that $\tau'(j) = 0$ if $j < i$ and $\varrho(j) = 0$ if $j \geq i$. Observe that $\partial^i_n(\tau) = \tau'_n + \varrho$. If $\varrho = \emptyset$ then i is the minimal index such that $\tau(i) > 0$. In this case the lemma is immediate. So, we can assume that $\varrho \neq \emptyset$. In this case $\tau_n = \tau' + \varrho_n$. If $\varrho_n = \emptyset$ then $\varrho = \{0\}$ and $F_{\tau_n}(k) = F_{\tau'}(k)$. Thus

$$\begin{aligned} F_{\partial^i_n(\tau)}(k) &= F_{\tau'_n+\varrho}(k) \\ &= F_{\tau'_n+\{0\}}(k) \\ &= F^{k+1}_{\tau'_n}(k) \\ &\leq F^{k+1}_{\tau'_k}(k) && \text{[Lemma V.2]} \\ &= F_{\tau'}(k) = F_{\tau_n}(k) \end{aligned}$$

Hence, we can assume that $\varrho_n \neq \emptyset$. We introduce the norms $||\varrho_n||_1 = \sum_{j=0}^d \varrho_n(j)$ and $||\varrho||_1 = \sum_{j=0}^d \varrho(j)$. Thus $||\varrho_n||_1 \geq 1$. Lemma VI.4 and Lemma V.2 show by induction that $F_{\tau_n}(k) \geq F_{\tau'}(m)$ where $m = k.2^{||\varrho_n||_1}$. Note that $m \geq k.(1+||\varrho_n||_1) \geq k+(k-1).||\varrho_n||_1+||\varrho_n||_1 \geq k+1+||\varrho_n||_1$ from $k-1 \geq 1$ and $||\varrho_n||_1 \geq 1$. As $||\varrho_n||_1 \geq ||\varrho||_1 - 1$, we get $m \geq k + ||\varrho||_1$. Now, just observe that $F_{\tau'}(m) = F^{m+1}_{\tau'_m}(m) \geq F_{\tau'_m}(k)$ from Lemma V.2. For every $j < i$, we have $\partial^i_n(\tau)(j) = \tau(j) + n = \varrho(j) + n \leq ||\varrho||_1 + k \leq m = \tau'_m(j)$. Hence we infer that $\partial^i_n(\tau) \sqsubseteq \tau'_m$ and so Lemma V.2 provides $F_{\partial^i_n(\tau)}(k) \leq F_{\tau'_m}(k)$. The lemma follows from this inequality, and the inequalities $F_{\tau'}(m) \geq F_{\tau'_m}(k)$ and $F_{\tau_n}(k) \geq F_{\tau'}(m)$ previously proved. □

Lemma VI.6. *$x.F_\tau(n) + y \leq F_\tau(x.n + x - 1 + y)$ for every $x \geq 1$, $y, n \geq 0$, and for every type τ.*

Proof. The lemma is proved by induction on types, with the lexicographic order. The base case $\tau = \emptyset$ is immediate since $x.F_\emptyset(n) + y = x(1+n) + y = 1 + x.n + x - 1 + y = F_\emptyset(x.n + x - 1 + y)$. Induction case for a type $\tau \neq \emptyset$. We have $x.F_\tau(n) + y = x.F^{n+1}_{\tau_n}(n) + y$. Since $\tau_n < \tau$, and F_{τ_n} is monotonic, by induction on $r \geq 0$, we get for every $k \geq 0$:

$$x.F^r_{\tau_n}(k) + y = F^r_{\tau_n}(x.k + r(x-1) + y)$$

By introducing $m = x.n + x - 1 + y$, we get:

$$\begin{aligned} x.F^{n+1}_{\tau_n}(n) + y &\leq F^{n+1}_{\tau_n}(m + n.(x-1)) \\ &\leq F^{n+1}_{\tau_m}(m + n.(x-1)) && \text{[Lemma V.2]} \end{aligned}$$

Notice that $F^{n.(x-1)}_\emptyset(m) = m + n.(x-1)$. Thus $m + n.(x-1) \leq F^{n.(x-1)}_{\tau_m}(m)$ from Lemma V.2. As $m - n = n.(x-1) + x - 1 + y \geq 0$, we deduce that $F^{n.(x-1)}_{\tau_m}(m) \leq F^{m-n}_{\tau_m}(m)$ from Lemma V.2. Combined with the previous relations, we get:

$$\begin{aligned} x.F^{n+1}_{\tau_n}(n) + y &\leq F^{n+1+m-n}_{\tau_m}(m) && \text{[Lemma V.2]} \\ &= F^{m+1}_{\tau_m}(m) \\ &= F_\tau(m) \\ &= F_\tau(x.n + x - 1 + y) \end{aligned}$$

We have proved the induction. □

Lemma VI.7. *We have $K_\tau(n) \leq F_\tau(d.n)$ for every $n \geq 2$.*

Proof. We prove the lemma by induction on types with the lexicographic order. The base case $\tau = \emptyset$ is immediate. Let us consider the case $\tau = \{0\}$. In this case $K_\tau(n) \leq K_\emptyset(K_\emptyset(n+1)) = n + 1$. Since $F_\tau(d.n) \geq F_\emptyset(d.n) = d.n + 1$, we are done. Let us prove the induction step with a type τ such that $\tau \neq \emptyset$ and $\tau \neq \{0\}$. Let $n \geq 2$ and i be an index such that $K_\tau(n) \leq K_{\partial^i_{d.n}(\tau)}(K_{\partial^i_{d.n}(\tau)}(n+1))$. We have:

$$\begin{aligned} K_\tau(n) &\leq F_{\partial^i_{d.n}(\tau)}(d.F_{\partial^i_{d.n}(\tau)}(d.(n+1))) \\ &\qquad\qquad \text{[IH and Lemma V.2]} \\ &\leq F_{\tau_{d.n}}(d.F_{\tau_{d.n}}(d.(n+1))) \\ &\qquad\qquad \text{[Lemma VI.5 and Lemma V.2]} \\ &\leq F_{\tau_{d.n}}(F_{\tau_{d.n}}(d^2.(n+1) + d - 1)) \\ &\qquad\qquad \text{[Lemma VI.6 and Lemma V.2]} \end{aligned}$$

Since $\tau \neq \emptyset$ and $\tau \neq \{0\}$, we deduce that $\tau_{d.n} \neq \emptyset$. With an immediate induction, and Lemma VI.4, we get:

$$\begin{aligned} F^{dn-1}_{\tau_{d.n}}(dn) &\geq 2^{dn-1}(dn+1) - 1 \\ &\geq dn(dn+1) - 1 \\ &\geq 2d(dn+1) - 1 \\ &\geq d^2.(n+1) + d - 1 \end{aligned}$$

Therefore $K_\tau(n) \leq F_{\tau_{d.n}}(F_{\tau_{d.n}}(F^{d.n-1}_{\tau_{d.n}}(d.n))) = F^{d.n+1}_{\tau_{d.n}}(d.n) = F_\tau(d.n)$. We have proved the induction step. □

Since $L_\tau(n) \leq K_\tau(n) \leq F_\tau(dn)$, Theorem VI.1 is proved.

VII. Application to Pushdown VASS

We now apply the results of the previous sections to vector addition systems with states (VASS) extended with a pushdown stack. Consider a *dimension* $d \in \mathbb{N}$, with $d \geq 1$. A *pushdown VASS* is a 5-tuple $\mathcal{A} = \langle Q, q_{\text{init}}, \boldsymbol{v}_{\text{init}}, \Gamma, \Delta \rangle$ where Q is a finite set of *locations*, $q_{\text{init}} \in Q$ is the *initial location*, $\boldsymbol{v}_{\text{init}} \in \mathbb{N}^d$ is the *initial vector*, Γ is a finite *stack alphabet*, and $\Delta \subseteq Q \times \mathbb{Z}^d \times \mathtt{Op} \times Q$ is a finite set of transition *rules*. As before, the set $\mathtt{Op}$ of *operations* is given by $\mathtt{Op} = \{\varepsilon\} \cup \{\mathsf{push}(\gamma), \mathsf{pop}(\gamma) \mid \gamma \in \Gamma\}$. Informally, a pushdown VASS can be viewed as a pushdown automaton equipped with d counters ranging over natural numbers. Operations on counters are limited to translations. A rule $(q, \boldsymbol{a}, \mathtt{op}, r)$ means that the pushdown VASS can move from location q to location r by performing the operation $\mathtt{op}$ on the stack and adding the vector $\boldsymbol{a}$ to its counters, provided that they remain nonnegative.

The operational semantics of a pushdown VASS $\mathcal{A}$ is defined in two steps. First, we associate with $\mathcal{A}$ the pushdown system $\mathcal{P}^{\mathcal{A}} = \langle S^{\mathcal{A}}, s^{\mathcal{A}}_{\text{init}}, \Gamma, \Delta^{\mathcal{A}} \rangle$ given by:

$$\begin{array}{rcl} S^{\mathcal{A}} & = & Q \times \mathbb{N}^d \\ s^{\mathcal{A}}_{\text{init}} & = & (q_{\text{init}}, \boldsymbol{v}_{\text{init}}) \\ ((q, \boldsymbol{u}), \mathtt{op}, (r, \boldsymbol{v})) \in \Delta^{\mathcal{A}} & \Leftrightarrow & (q, \boldsymbol{v} - \boldsymbol{u}, \mathtt{op}, r) \in \Delta \end{array}$$

Note that $\mathcal{P}^{\mathcal{A}}$ is finitely-branching. Second, we define the operational semantics of $\mathcal{A}$ to be the operational semantics of $\mathcal{P}^{\mathcal{A}}$ (see Section III). Let us introduce the partial-order $\preceq_{\mathcal{A}}$ on $S^{\mathcal{A}}$ defined by $(q, \boldsymbol{v}) \preceq_{\mathcal{A}} (q', \boldsymbol{v}')$ when $q = q'$ and $\boldsymbol{v} \leq \boldsymbol{v}'$. This partial-order makes $\mathcal{P}^{\mathcal{A}}$ a well-structured pushdown system (see Definition III.1). According to Propositions III.3, III.4 and III.5, the termination and boundedness problems for pushdown VASS can be solved by constructing the reduced reachability tree, which is finite. We show in this section that the worst-case size of this tree is hyper-Ackermannian.

For the remainder of this section, we restrict ourselves to pushdown VASS such that $||\boldsymbol{a}||_\infty \leq 1$ for each rule $(q, \boldsymbol{a}, \mathtt{op}, r)$. The hyper-Ackermannian upper bound that we obtain below can be extended to the general case by splitting each rule $(q, \boldsymbol{a}, \mathtt{op}, r)$ into $||\boldsymbol{a}||_\infty$ basic ones. The *size* of a pushdown VASS $\mathcal{A} = \langle Q, q_{\text{init}}, \boldsymbol{v}_{\text{init}}, \Gamma, \Delta \rangle$ is defined as

$$|\mathcal{A}| \;=\; d + |Q| + d \cdot ||\boldsymbol{v}_{\text{init}}||_\infty + |\Gamma| + d \cdot |\Delta|$$

According to Remark IV.3, every run of $\mathcal{P}^{\mathcal{A}}$ can be seen as a nested sequence over $(S^{\mathcal{A}}, \preceq_{\mathcal{A}})$. Let us introduce the norm $||.||_{S^{\mathcal{A}}}$ on $(S^{\mathcal{A}}, \preceq_{\mathcal{A}})$, defined by $||(q, \boldsymbol{v})||_{S^{\mathcal{A}}} = ||\boldsymbol{v}||_\infty$. This nested sequence is n-controlled, for $n = ||\boldsymbol{v}_{\text{init}}||_\infty + 2$, since the initial state of $\mathcal{P}^{\mathcal{A}}$ is $(q_{\text{init}}, \boldsymbol{v}_{\text{init}})$ and $||\boldsymbol{a}||_\infty \leq 1$ for each rule $(q, \boldsymbol{a}, \mathtt{op}, r)$. Recall that every subsumed node in the reduced reachability tree of $\mathcal{P}^{\mathcal{A}}$ is necessarily a leaf. It follows from Remark IV.7 that the height of this tree is bounded by the maximal length of n-controlled bad nested sequences, namely $L_{S^{\mathcal{A}}}(n)$. Note that $(S^{\mathcal{A}}, \preceq_{\mathcal{A}}, ||.||_{S^{\mathcal{A}}})$ is isomorphic to $(|Q| \times \mathbb{N}^d, \preceq_{|Q| \times \mathbb{N}^d}, ||.||_{|Q| \times \mathbb{N}^d})$. Since $Q \neq \emptyset$, $d \geq 1$ and $n \geq 2$, we get from Theorem VI.1 that the height of the reduced reachability tree is at most $F_{\omega^d \cdot |Q|}(d \cdot n)$. Each node of the reduced reachability tree can have at most $|\Delta|$ children. We derive the following upper bound on its size.

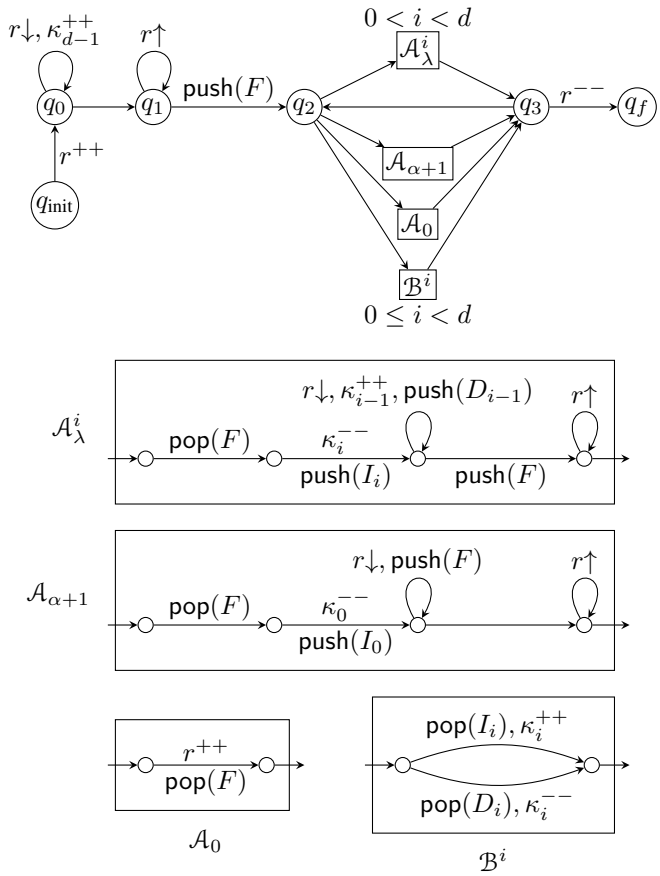

Figure 1. Pushdown VASS $\mathcal{A}_d(n)$ that weakly computes $F_{\omega^d}(n)$.

Theorem VII.1. *The reduced reachability tree of a pushdown VASS $\mathcal{A}$ has at most $F_{\omega^\omega}(|\mathcal{A}|)$ nodes.*

Remark VII.2. For classical VASS (i.e., $\mathtt{op} = \varepsilon$ for each rule $(q, \boldsymbol{a}, \mathtt{op}, r)$), the reduced reachability tree has at most $F_\omega(|\mathcal{A}|)$ nodes [22], [13]. For pushdown finite-state systems, it follows from Remark VI.2 that its size is bounded by $|\Delta|^{2^{|Q|}}$.

We next give an almost matching lower bound for the above-mentioned size of the reduced reachability tree. We introduce a pushdown VASS $\mathcal{A}_d(n)$, depicted in Figure 1, that is parametrized by a "target" dimension $d \geq 1$ and an initial value $n \in \mathbb{N}$. The lower bound is obtained as follows. First, we show that $\mathcal{A}_d(n)$ computes, in a weak sense, the value $F_{\omega^d}(n)$. This computation requires, in $\mathcal{A}_d(n)$, a run of length at least $F_{\omega^d}(n)$. Second, we prove that all runs of $\mathcal{A}_d(n)$ are finite. Therefore, its reduced reachability tree is identical to its reachability tree, and so its height is at least $F_{\omega^d}(n)$. The remainder of this section presents these ideas in detail. To simplify notation, we will shortly write $\mathcal{A}_d$ when the parameter n is understood from the context.

The dimension of $\mathcal{A}_d$ is $d + 2$. But, for clarity, we will use counter names to refer to individual coordinates of the

vectors in the configurations of $\mathcal{A}_d$. The parameter n is the initial value of the first counter, called r, which will also contain the result at the end. For intermediate steps, we use the counters $\overline{r}, \kappa_0, \ldots, \kappa_{d-1}$. The notation $r{\downarrow}$ in Figure 1 is short for $r^{--}, \overline{r}^{++}$, i.e., a decrement of r followed by an increment of $\overline{r}$. Similarly, the notation $r{\uparrow}$ is short for $r^{++}, \overline{r}^{--}$, i.e., an increment of r followed by a decrement of $\overline{r}$. The stack alphabet is $\Gamma = \{F, D_0, I_0, \ldots, D_{d-1}, I_{d-1}\}$. Intuitively, the stack and counters of $\mathcal{A}_d$ are used to represent the recursive definition of $F_{\omega^d}(n)$. As expected, the stack is used to store the pending recursive calls. The innermost $F_\alpha(m)$ computation is encoded as $\alpha = \omega^{d-1}.\kappa_{d-1} + \cdots + \omega^0.\kappa_0$ and $r = m + 1$. For example, $F_{\omega^2.3+\omega.5+2}(8)$ is represented by having F on top of the stack, $\kappa_2 = 3$, $\kappa_1 = 5$, $\kappa_0 = 2$ and $r = 9$. The stack symbols D_i, I_i are used to restore the calling context (i.e., the counters κ_i) of pending recursive calls.

The initial location of $\mathcal{A}_d$ is q_{init}, and its initial vector is $\boldsymbol{v}_{\text{init}} = (n, 0, \ldots, 0)$. This means that the initial value of the counter r is n and all other counters have initial value 0. First, we increment r by one and move to q_0. The loop at q_0 allows us to increment κ_{d-1} to the value $n + 1$ and the loop at q_1 allows us to set r back to $n + 1$. Then we push F on the stack and reach q_2. At this point, the stack contents is $\sigma = F$, the counters $\kappa_{d-1}, \ldots, \kappa_0$ contain $n+1, 0, \ldots, 0$, respectively, and $r = n + 1$. This configuration represents $F_{w^{d-1}.(n+1)}(n)$, which is equal to $F_{\omega^d}(n)$, i.e., the number that we want to compute. From the location q_2, $\mathcal{A}_d$ can either execute $\mathcal{A}^i_\lambda$ or $\mathcal{A}_{\alpha+1}$ or $\mathcal{A}_0$ or $\mathcal{B}^i$ to reach q_3. From the location q_3, $\mathcal{A}_d$ can either go back to q_2 to continue the computation, or move to the final location q_f.

At q_2, if the top of the stack is F and $\kappa_0 = 0$, then it represents $F_{\omega^{d-1}.\kappa_{d-1}+\cdots+\omega^i.\kappa_i}(r-1)$, assuming that $\kappa_i > 0$ and $\kappa_{i-1} = \cdots = \kappa_0 = 0$. In that case, the purpose of $\mathcal{A}^i_\lambda$ is to replace the top of the stack with something that represents $F_{\omega^{d-1}.\kappa_{d-1}+\cdots+\omega^i.(\kappa_i-1)+\omega^{i-1}.r}(r-1)$. Accordingly, $\mathcal{A}^i_\lambda$ first pops F from the stack. Then it decrements κ_i and pushes I_i on the stack to "shield" the part of the stack below from this decreased value of κ_i. Then $\mathcal{A}^i_\lambda$ increments the counter κ_{i-1} r times, also pushing the symbol D_{i-1} each time. The purpose of D_{i-1} is to "guard" the part of the stack below from this increased value of κ_{i-1}. Finally, $\mathcal{A}^i_\lambda$ pushes F on the stack, restores the value of r and exits.

At q_2, if the top of the stack is F and $\kappa_0 > 0$, then it represents $F_{\alpha+1}(r-1)$, where $\alpha = \omega^{d-1}.\kappa_{d-1}+\cdots+\omega^0.(\kappa_0-1)$. In that case, the purpose of $\mathcal{A}_{\alpha+1}$ is to replace the top of the stack with something that represents $F^r_\alpha(r-1)$. Accordingly, $\mathcal{A}_{\alpha+1}$ first pops F from the stack. Then it decrements κ_0 and pushes I_0 on the stack to "shield" the part of the stack below from this decreased value of κ_0. Then it pushes r copies of the symbol F, restores the value of r and exits.

At q_2, if the top of the stack is F and $\kappa_{d-1} = \cdots = \kappa_0 = 0$, then it represents $F_0(r-1)$. Accordingly, $\mathcal{A}_0$ simply pops F from the stack, increments r and exits.

At q_2, if the top of the stack is I_i or D_i, then sometime back, either 1) $\mathcal{A}^i_\lambda$ decremented κ_i, pushed one I_i to shield the bottom part of the stack from this decrement, incremented κ_{i-1} n times and pushed n copies of D_{i-1} to guard the bottom part of the stack from these increments, or 2) $i = 0$, $n = 0$, $\mathcal{A}_{\alpha+1}$ decremented κ_0 once and pushed one I_0 to shield the bottom part of the stack from this decrement. The purpose of $\mathcal{B}^0, \ldots, \mathcal{B}^{d-1}$ is to backtrack and access the bottom part of the stack by removing guards and shields, and restoring the counter values. Accordingly, $\mathcal{B}^i$ either pops the symbol D_i from the top of the stack, decrements κ_i and exits, or it pops I_i, increments κ_i and exits.

Lemma VII.3. *Suppose $\mathcal{A}_d$ is in location q_2, F is on the top of the stack and the counter r has value $m + 1$. Then there is a partial[2] run in $\mathcal{A}_d$ that reaches q_3, removes F from the top of the stack, sets the counter r to the value $F_{\omega^{d-1}.\kappa_{d-1}+\cdots+\omega^0.\kappa_0}(m) + 1$, and does no other change.*

Corollary VII.4. *There is a run in $\mathcal{A}_d(n)$ that reaches q_f with the empty stack and the counter r having the value $F_{\omega^d}(n)$.*

Proof. A witnessing run is provided in the table below. The partial run from q_2 to q_3 is derived from Lemma VII.3.

Loc.	r	κ_{d-1}	Stack	
$(q_{\text{init}},$	$n,$	$0,$	$\varepsilon)$	$\xrightarrow{r^{++}}$
$(q_0,$	$n+1,$	$0,$	$\varepsilon)$	$\xrightarrow{(r{\downarrow},\kappa_{d-1}^{++})^{n+1}}$
$(q_1,$	$0,$	$n+1,$	$\varepsilon)$	$\xrightarrow{(r{\uparrow})^{n+1}\mathsf{push}(F)}$
$(q_2,$	$n+1,$	$n+1,$	$F)$	$\rightarrow^*$
$(q_3,$	$F_{\omega^{d-1}.(n+1)}(n)+1,$	$n+1,$	$\varepsilon)$	$\xrightarrow{r^{--}}$
$(q_f,$	$F_{\omega^{d-1}.(n+1)}(n),$	$n+1,$	$\varepsilon)$	

The observation that $F_{\omega^d}(n) = F_{\omega^{d-1}.(n+1)}(n)$ concludes the proof of the corollary. □

Next, we prove that all runs of $\mathcal{A}_d$ are finite. The intuition is to measure, by an ordinal, the "size" of both the stack contents and the values of the counters $\kappa_{d-1}, \ldots, \kappa_0$, and prove that the size decreases at each step. Consider a vector $\boldsymbol{v} = (v_{d-1}, \ldots, v_0)$ in $\mathbb{N}^d$. The ordinal associated with $\boldsymbol{v}$ is defined as $\mathsf{Ord}(\boldsymbol{v}) = \omega^{d-1}.v_{d-1} + \cdots + \omega^0.v_0$. Let $\sigma \in \Gamma^*$ be a string over the stack alphabet. The ordinal associated with σ and $\boldsymbol{v}$ is defined as follows, by induction on $|\sigma|$.

$$
\begin{aligned}
\mathsf{Ord}(\varepsilon, \boldsymbol{v}) &= 0 \\
\mathsf{Ord}(\sigma F, \boldsymbol{v}) &= \mathsf{Ord}(\sigma, \boldsymbol{v}) + \omega^{\mathsf{Ord}(\boldsymbol{v})} \\
\mathsf{Ord}(\sigma I_i, \boldsymbol{v}) &= \mathsf{Ord}(\sigma, \boldsymbol{v} + \boldsymbol{e}_i) \\
\mathsf{Ord}(\sigma D_i, \boldsymbol{v}) &= \begin{cases} \text{undefined} & \text{if } v_i = 0 \\ \mathsf{Ord}(\sigma, \boldsymbol{v} - \boldsymbol{e}_i) & \text{if } v_i > 0 \end{cases}
\end{aligned}
$$

The ordered pair $\mathsf{dr}(\sigma, \boldsymbol{v}) = (\mathsf{Ord}(\sigma, \boldsymbol{v}), |\sigma|)$ is the *depth of remaining recursion* associated with a configuration of $\mathcal{A}_d$ whose stack contents is σ and where the counter values of $\kappa_{d-1}, \ldots, \kappa_0$ form the vector $\boldsymbol{v}$. We use the lexicographic order on ordered pairs to compare depths of remaining recursion.

[2]By *partial run*, we mean a run that is not required to start from the initial configuration.

Lemma VII.5. *Every partial run of $\mathcal{A}_d(n)$ that starts and ends in q_2, but does not visit q_2 in between, makes the depth of remaining recursion strictly decrease.*

Corollary VII.6. *All runs of $\mathcal{A}_d(n)$ are finite.*

Proof. By contradiction, suppose that there is an infinite run. It is readily seen that, after a finite prefix, the run must stay within q_2, q_3, $\mathcal{A}^i_\lambda$, $\mathcal{A}_{\alpha+1}$, $\mathcal{A}_0$ and $\mathcal{B}^i$. Upon entering into $\mathcal{A}^i_\lambda$, $\mathcal{A}_{\alpha+1}$, $\mathcal{A}_0$ or $\mathcal{B}^i$, the run must exit after a finite number of steps, since the loops inside these can only be executed finitely many times for each entry. Hence, the run must visit q_2 infinitely often. Now from Lemma VII.5, we infer that there is an infinite strictly decreasing sequence of depths of remaining recursion, which is a contradiction. □

We obtain the following lower bound on the size of the reduced reachability tree for pushdown VASSs.

Theorem VII.7. *For every $n \in \mathbb{N}$, there exists a pushdown VASS $\mathcal{A}_n$, of size quadratic in n, such that the reduced reachability tree of $\mathcal{A}_n$ has at least $F_{\omega^\omega}(n)$ nodes.*

Proof. Corollary VII.6, combined with Proposition III.4, shows that the reachability tree of $\mathcal{A}_{n+1}(n)$ contains no subsumed node. Therefore, the reachability tree of $\mathcal{A}_{n+1}(n)$ coincides with its reduced reachability tree, which is finite according to Proposition III.3. The height of this tree is at least $F_{\omega^\omega}(n) = F_{\omega^{n+1}}(n)$ by Corollary VII.4. It is readily seen that $|\mathcal{A}_{n+1}(n)|$ is quadratic in n. □

Remark VII.8. The hyper-Ackermannian lower and upper bounds of Theorems VII.1 and VII.7 also apply to the reachability set when it is finite.

VIII. Conclusion

Reachability sets of bounded VASS (without stack) are known to be Ackermannian large. The lower-bound was provided by [21] thanks to a family of VASS with finite, but large reachability sets (Ackermannian). The upper-bound was first derived in [22] from bounds on the length of bad sequences of vectors of natural numbers. However, the boundedness problem for VASS is decidable in exponential space. This complexity was proved by Rackoff in [23] by observing that unboundedness of a VASS can be witnessed by finite runs of length at most doubly exponential.

In this paper, reachability sets of bounded pushdown VASS are shown to be hyper-Ackermannian large. The lower-bound is obtained thanks to a family of bounded pushdown VASS with finite, but large reachability sets (hyper-Ackermannian). The upper-bound is derived from a precise complexity analysis of an adaptation of the Karp & Miller algorithm. Based on this algorithm, we deduce that the unboundedness of pushdown VASS can be witnessed by finite runs. We think that the precise complexity of the boundedness problem can be obtained by bounding the length of these witnesses. However, adapting Rackoff's techniques to pushdown VASS is a difficult and challenging problem. Note that witnesses have a length that is at least a tower of exponentials, i.e., a length in $F_3(n)$, where n is the size of the pushdown VASS. Such a lower-bound can be easily obtained by adapting [19]. The complexity gap between the lower-bound $F_3(n)$ and the upper-bound $F_{\omega^\omega}(n)$ is an open problem.

References

[1] P. A. Abdulla, M. F. Atig, G. Delzanno, and A. Podelski. Push-down automata with gap-order constraints. In *Selec. FSEN'13*, volume 8161 of *LNCS*, pages 199–216. Springer, 2013.

[2] P. A. Abdulla, K. Čerāns, B. Jonsson, and Y.-K. Tsay. Algorithmic analysis of programs with well quasi-ordered domains. *Inform. Comput.*, 160(1–2):109–127, 2000.

[3] R. Alur and P. Madhusudan. Adding nesting structure to words. *J. ACM*, 56(3):1–16, 2009.

[4] M. F. Atig and P. Ganty. Approximating petri net reachability along context-free traces. In *Proc. FSTTCS'11*, volume 13 of *LIPIcs*, pages 152–163. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2011.

[5] A. Bouajjani and M. Emmi. Analysis of recursively parallel programs. *ACM Trans. Prog. Lang. Syst.*, 35(3):1–49, 2013.

[6] X. Cai and M. Ogawa. Well-structured pushdown systems. In *Proc. CONCUR'13*, volume 8052 of *LNCS*, pages 121–136. Springer, 2013.

[7] R. Chadha and M. Viswanathan. Decidability results for well-structured transition systems with auxiliary storage. In *Proc. CONCUR'07*, volume 4703 of *LNCS*, pages 136–150. Springer, 2007.

[8] R. Chadha and M. Viswanathan. Deciding branching time properties for asynchronous programs. *Theor. Comput. Sci.*, 410(42):4169–4179, 2009.

[9] P. Chambart and Ph. Schnoebelen. The ordinal recursive complexity of lossy channel systems. In *Proc. LICS'08*, pages 205–216. IEEE, 2008.

[10] S. Demri, M. Jurdziński, O. Lachish, and R. Lazić. The covering and boundedness problems for branching vector addition systems. *J. Comput. Syst. Sci.*, 79(1):23–38, 2013.

[11] N. Dershowitz and Z. Manna. Proving termination with multiset orderings. *Commun. ACM*, 22(8):465–476, 1979.

[12] C. Dufourd, A. Finkel, and Ph. Schnoebelen. Reset nets between decidability and undecidability. In *Proc. ICALP'98*, volume 1443 of *LNCS*, pages 103–115. Springer, 1998.

[13] D. Figueira, S. Figueira, S. Schmitz, and Ph. Schnoebelen. Ackermannian and primitive-recursive bounds with Dickson's lemma. In *Proc. LICS'11*, pages 269–278. IEEE, 2011.

[14] A. Finkel. Reduction and covering of infinite reachability trees. *Inform. Comput.*, 89(2):144–179, 1990.

[15] A. Finkel, P. McKenzie, and C. Picaronny. A well-structured framework for analysing Petri net extensions. *Inform. Comput.*, 195(1-2):1–29, 2004.

[16] A. Finkel and Ph. Schnoebelen. Well-structured transition systems everywhere! *Theor. Comput. Sci.*, 256(1–2):63–92, 2001.

[17] P. Ganty and R. Majumdar. Algorithmic verification of asynchronous programs. *ACM Trans. Prog. Lang. Syst.*, 34(1):6:1–6:48, 2012.

[18] C. Haase, S. Schmitz, and Ph. Schnoebelen. The power of priority channel systems. In *Proc. CONCUR'13*, volume 8052 of *LNCS*, pages 319–333. Springer, 2013.

[19] R. Lazic. The reachability problem for vector addition systems with a stack is not elementary. *CoRR*, abs/1310.1767, 2013. Presented at RP'12.

[20] M. Löb and S. Wainer. Hierarchies of number-theoretic functions. I. *Archiv für mathematische Logik und Grundlagenforschung*, 13(1–2):39–51, 1970. http://dx.doi.org/10.1007/BF01967649.

[21] E. Mayr and A. Meyer. The complexity of the finite containment problem for petri nets. *J. ACM*, 28(3):561–576, 1981.

[22] K. McAloon. Petri nets and large finite sets. *Theor. Comput. Sci.*, 32(1–2):173–183, 1984.

[23] C. Rackoff. The covering and boundedness problems for vector addition systems. *Theor. Comput. Sci.*, 6(2):223–231, 1978.

[24] K. Reinhardt. Reachability in petri nets with inhibitor arcs. *Electr. Notes Theor. Comput. Sci.*, 223:239–264, 2008.

[25] S. Schmitz and Ph. Schnoebelen. Multiply-recursive upper bounds with Higman's lemma. In *Proc. ICALP'11*, volume 6756 of *LNCS*, pages 441–452. Springer, 2011.

[26] K. Sen and M. Viswanathan. Model checking multithreaded programs with asynchronous atomic methods. In *Proc. CAV'06*, volume 4144 of *LNCS*, pages 300–314. Springer, 2006.

APPENDIX

A. Proofs of Section IV

Lemma IV.8. *Bad nested sequences over a wqo are finite.*

Proof. We consider an infinite sequence $(s_0, h_0), (s_1, h_1), \ldots$ of elements in $S \times \mathbb{N}$ with $h_0 = 0$. From the infinite sequence $h_{i_0}, h_{i_1}, h_{i_2}, \ldots$ we extract an infinite sequence $i_0, i_1, i_2, \ldots$ defined as follows:

$$\begin{array}{rcl} i_0 & = & 0 \\ i_{k+1} & = & \min\{i > i_k \mid \forall j > i_k, h_i \leq h_j\} \end{array}$$

By definition $h_{i_+k} \leq h_j$ By definition, $h_{i_k} \leq h_j$ for all $j, k \in \mathbb{N}$ with $i_k \leq j$. Since $\preceq$ is a well-quasi-order, there exists $k < l$ such that $s_{i_k} \preceq s_{i_l}$. Thus the sequence $(s_0, h_0), (s_1, h_1), \ldots$ is good. □

B. Proofs of Section VI

Lemma VI.3. *It holds that:*

$$K_{\mathbb{N}^\tau/v}(m) \leq K_{\partial^i_{d.n}(\tau)}(m)$$

for every type τ, for every $i \in \{1, \ldots, d\}$, for every $v \in \tau(i) \times \mathbb{N}^i$, for every $n \geq ||v||_\tau$, and for every $m \in \mathbb{N}$.

Proof. Since $v \in \tau(i) \times \mathbb{N}^i$ there exist $q \in \{1, \ldots, \tau(i)\}$ and $\boldsymbol{v} \in \mathbb{N}^i$ such that $v = (q, \boldsymbol{v})$. By adapting the following proof, we can assume without loss of generality that $q = \tau(i)$. Since $\tau(i) > 0$, the derivation type $\varrho = \partial^i_{d.n}(\tau)$ is well defined.

We introduce the projection function $\pi_r : \mathbb{N}^i \mapsto \mathbb{N}^{i-1}$ that removes the component indexed by r, and formally defined as follows:

$$\pi_r(\boldsymbol{u}) = (\boldsymbol{u}(1), \ldots, \boldsymbol{u}(r-1), \boldsymbol{u}(r+1), \ldots, \boldsymbol{u}(i))$$

Let us observe that for every $\boldsymbol{u} \in \mathbb{N}^i/\boldsymbol{v}$ there exists a minimal $r \in \{1, \ldots, i\}$ such that $\boldsymbol{u}(r) < \boldsymbol{v}(r)$. We denote by $\mu(\boldsymbol{u})$ this minimal index r. Since $||\boldsymbol{v}||_\infty \leq n$ we deduce that $\boldsymbol{u}(r) \in \{0, \ldots, n-1\}$. Notice that this set can be empty if $n = 0$.

We introduce the function $\nu : \mathbb{N}^\tau/v \mapsto \mathbb{N}^\varrho$ defined for every $u = (p, \boldsymbol{u}) \in \tau(j) \times \mathbb{N}^j$ by:

$$\nu(u) = \begin{cases} u & \text{if } j \neq i \ \vee \ (j = i \wedge p < \tau(i)) \\ (\tau(i-1) + r + i.\boldsymbol{u}(r), \pi_r(\boldsymbol{u})) & \text{if } j = i \wedge p = \tau(i) \wedge r = \mu(\boldsymbol{u}) \end{cases}$$

Observe that if $(u_0, h_0) \ldots, (u_k, h_k)$ is a m-controlled bad nested sequence in $\mathbb{N}^\tau/v$ then $(\nu(u_0), h_0), \ldots, (\nu(u_k), h_k)$ is a m-controlled bad nested sequence in $\mathbb{N}^\varrho$. We have proved the lemma. □

C. Proofs of Section VII

Lemma A.1. *For every $n \in \mathbb{N}$, $F_\omega(n) \geq n^{n+1}$.*

Proof. We first show that $F_2^2(n) \geq n^{n+1}$ for every $n \in \mathbb{N}$. Observe that $F_1(n) = 2n + 1$ for all $n \in \mathbb{N}$. An easy induction on k shows that $F_1^k(n) = (n+1)2^k - 1$ for every $k, n \in \mathbb{N}$. It follows that $F_2(n) = (n+1)2^{n+1} - 1$, hence, $F_2(n) \geq n2^n$ for all $n \in \mathbb{N}$. By monotonicity of F_2 (see Lemma V.2), we obtain that:

$$F_2^2(n) \ \geq \ F_2(n2^n) \ \geq \ n2^{n2^n} \ \geq \ n2^{2^n} \ \geq \ nn^n \ = \ n^{n+1}$$

Let us now prove the statement of the lemma, namely that $F_\omega(n) \geq n^{n+1}$ for all $n \in \mathbb{N}$. It is readily seen that this inequality holds for $n \leq 1$, since $F_\omega(0) = F_1(0) = 1$ and $F_\omega(1) = F_2(1) = 7$. Assume for the remainder of the proof that $n \geq 2$. By Lemma V.2, we get that

$$F_\omega(n) \ = \ F_{n+1}(n) \ \geq \ F_3(n) \ = \ F_2^{n+1}(n) \ \geq \ F_2^2(n)$$

Hence, $F_\omega(n) \geq F_2^2(n) \geq n^{n+1}$, which concludes the proof of the lemma. □

Theorem VII.1. *The reduced reachability tree of a pushdown VASS $\mathcal{A}$ has at most $F_{\omega^\omega}(|\mathcal{A}|)$ nodes.*

Proof. Consider a pushdown VASS $\mathcal{A} = \langle Q, q_{\text{init}}, \boldsymbol{v}_{\text{init}}, \Gamma, \Delta \rangle$, and let $\mathcal{T}$ denote its reduced reachability tree. Recall that $d \geq 1$ and $|\mathcal{A}| = |Q| + |\Gamma| + d \cdot (1 + ||\boldsymbol{v}_{\text{init}}||_\infty + |\Delta|)$. The case where Δ is empty is trivial, since $F_{\omega^\omega}(n) = F_{\omega^{n+1}}(n) > n$ for every $n \in \mathbb{N}$ (by Lemma V.2), and $\mathcal{T}$ contains only one node if $\Delta = \emptyset$. So we assume for the remainder of the proof that $\Delta \neq \emptyset$.

We have shown in the main text preceding the theorem that the height h of $\mathcal{T}$ satisfies $h \leq F_{\omega^d.|Q|}(d \cdot (||\boldsymbol{v}_{\text{init}}||_\infty + 2))$. Non-emptiness of Δ entails that $|\mathcal{A}| \geq d \cdot (||\boldsymbol{v}_{\text{init}}||_\infty + 2)$. Hence, by monotonicity of $F_{\omega^d.|Q|}$, we get that

$$h \leq F_{\omega^d.|Q|}(|\mathcal{A}|)$$

Since each node of $\mathcal{T}$ can have at most $|\Delta|$ children, we obtain that $\mathcal{T}$ has at most $|\Delta|^{h+1}$ nodes. To reduce clutter in the inequations below, we introduce $n = |\mathcal{A}|$, $\alpha = \omega^n.\, n + \cdots + \omega^0.\, n$, and $\beta = \omega^d.\, |Q|$. Observe that $n \geq |Q|$ and $n \geq d \geq 1$. It follows that $\alpha \sqsupseteq \beta$ and $\alpha \sqsupseteq \omega$. Recall also that, according to Lemma V.2, for each ordinal $\gamma < \omega^\omega$, F_γ is monotonic and $F_\gamma(m) \geq m$ for all $m \in \mathbb{N}$. We derive that:

$$
\begin{aligned}
F_{\omega^\omega}(n) &= F_{\omega^{n+1}}(n) \\
&= F_{\omega^n.(n+1)}(n) \\
&= F_{\omega^n.n+\cdots+\omega^1.n+\omega^0.(n+1)}(n) \\
&= F_{\alpha+1}(n) \\
&= F_\alpha^{n+1}(n) \\
&\geq F_\alpha^n(F_\beta(n)) && [\alpha \sqsupseteq \beta] \\
&\geq F_\alpha(F_\beta(n)) && [n \geq 1] \\
&\geq F_\omega(F_\beta(n)) && [\alpha \sqsupseteq \omega] \\
&\geq (F_\beta(n))^{F_\beta(n)+1} && [\text{Lemma A.1}] \\
&\geq (F_\beta(n))^{h+1} && [F_\beta(n) \geq h] \\
&\geq |\Delta|^{h+1} && [F_\beta(n) \geq n \geq |\Delta|]
\end{aligned}
$$

We conclude that $\mathcal{T}$ has at most $F_{\omega^\omega}(n)$ nodes. □

Lemma VII.3. *Suppose $\mathcal{A}_d$ is in location q_2, F is on the top of the stack and the counter r has value $m+1$. Then there is a partial*[3] *run in $\mathcal{A}_d$ that reaches q_3, removes F from the top of the stack, sets the counter r to the value $F_{\omega^{d-1}.\kappa_{d-1}+\cdots+\omega^0.\kappa_0}(m)+1$, and does no other change.*

Proof. By assumption, the starting value of r is $m+1$, and the starting location is q_2. Let $c_{d-1}, \ldots, c_0$ denote the starting values of the counters $\kappa_{d-1}, \ldots, \kappa_0$, respectively. The proof is by induction on the ordinal $\omega^{d-1}.\, c_{d-1} + \cdots + \omega_0.\, c_0$. We write configurations as tuples enclosed in parentheses, with the location as first element, followed by the values of $r, \kappa_{d-1}, \ldots, \kappa_0$, and finally the stack content.

- $\omega^{d-1}.\, c_{d-1} + \cdots + \omega^0.\, c_0 = 0$. Recall that $F_0(m) = m+1$.

$$(q_2, r = m+1, 0, \ldots, 0, \sigma F) \xrightarrow{\mathcal{A}_0} (q_3, r = m+2, 0, \ldots, 0, \sigma)$$

- $\omega^{d-1}.\, c_{d-1} + \cdots + \omega^0.\, c_0 = \alpha + 1$, a successor ordinal. This means that $c_0 > 0$. Recall that $F_{\alpha+1}(m) = F_\alpha^{m+1}(m)$.

State	r	κ_{d-1}	$\cdots$	κ_1	κ_0	Stack	
$(q_2,$	$m+1,$	$c_{d-1},$	$\ldots,$	$c_1,$	$c_0,$	$\sigma F)$	$\xrightarrow{\mathcal{A}_{\alpha+1}}$
$(q_3,$	$m+1,$	$c_{d-1},$	$\ldots,$	$c_1,$	$c_0-1,$	$\sigma I_0 F^{m+1})$	$\rightarrow$
$(q_2,$	$m+1,$	$c_{d-1},$	$\ldots,$	$c_1,$	$c_0-1,$	$\sigma I_0 F^{m+1})$	$\rightarrow^*$ IH
$(q_3,$	$F_\alpha(m)+1,$	$c_{d-1},$	$\ldots,$	$c_1,$	$c_0-1,$	$\sigma I_0 F^m)$	$\rightarrow$
$(q_2,$	$F_\alpha(m)+1,$	$c_{d-1},$	$\ldots,$	$c_1,$	$c_0-1,$	$\sigma I_0 F^m)$	$\rightarrow^*$ IH
$(q_3,$	$F_\alpha^2(m)+1,$	$c_{d-1},$	$\ldots,$	$c_1,$	$c_0-1,$	$\sigma I_0 F^{m-1})$	$\rightarrow$
							$\vdots\, (m-1)$ times
$(q_3,$	$F_\alpha^{m+1}(m)+1,$	$c_{d-1},$	$\ldots,$	$c_1,$	$c_0-1,$	$\sigma I_0)$	$\rightarrow$
$(q_2,$	$F_{\alpha+1}(m)+1,$	$c_{d-1},$	$\ldots,$	$c_1,$	$c_0-1,$	$\sigma I_0)$	$\xrightarrow{\mathcal{B}^0}$
$(q_3,$	$F_{\alpha+1}(m)+1,$	$c_{d-1},$	$\ldots,$	$c_1,$	$c_0,$	$\sigma)$	

- $\omega^{d-1}.\, c_{d-1} + \cdots + \omega^0.\, c_0 = \lambda$, a limit ordinal. Suppose $c_i > 0$ and $c_{i-1} = \cdots = c_0 = 0$, with $i > 0$. Then, by definition of

[3] By *partial run*, we mean a run that is not required to start from the initial configuration.

the canonical fundamental sequence, $\lambda_m = \omega^{d-1}.\, c_{d-1} + \cdots + \omega^i.\,(c_i - 1) + \omega^{i-1}.\,(m+1)$. Recall that $F_\lambda(m) = F_{\lambda_m}(m)$.

State	r	κ_{d-1}	$\cdots$	κ_i	κ_{i-1}	Stack	
$(q_2,$	$m+1,$	$c_{d-1},$	$\ldots,$	$c_i,$	$0,$	$\sigma F)$	$\xrightarrow{\mathcal{A}^i_\lambda}$
$(q_3,$	$m+1,$	$c_{d-1},$	$\ldots,$	$c_i - 1,$	$m+1,$	$\sigma I_i D^{m+1}_{i-1} F)$	$\rightarrow$
$(q_2,$	$m+1,$	$c_{d-1},$	$\ldots,$	$c_i - 1,$	$m+1,$	$\sigma I_i D^{m+1}_{i-1} F)$	$\rightarrow$ IH
$(q_3,$	$F_{\lambda_m}(m)+1,$	$c_{d-1},$	$\ldots,$	$c_i - 1,$	$m+1,$	$\sigma I_i D^{m+1}_{i-1})$	$\rightarrow$
$(q_2,$	$F_\lambda(m)+1,$	$c_{d-1},$	$\ldots,$	$c_i - 1,$	$m+1,$	$\sigma I_i D^{m+1}_{i-1})$	$\xrightarrow{\mathcal{B}^{i-1}}$
$(q_3,$	$F_\lambda(m)+1,$	$c_{d-1},$	$\ldots,$	$c_i - 1,$	$m,$	$\sigma I_i D^{m}_{i-1})$	$\rightarrow$
							$\vdots\, m$ times
$(q_3,$	$F_\lambda(m)+1,$	$c_{d-1},$	$\ldots,$	$c_i - 1,$	$0,$	$\sigma I_i)$	$\rightarrow$
$(q_2,$	$F_\lambda(m)+1,$	$c_{d-1},$	$\ldots,$	$c_i - 1,$	$0,$	$\sigma I_i)$	$\xrightarrow{\mathcal{B}^{i}}$
$(q_3,$	$F_\lambda(m)+1,$	$c_{d-1},$	$\ldots,$	$c_i,$	$0,$	$\sigma)$	

This concludes the proof of the lemma. □

Lemma VII.5. *Every partial run of $\mathcal{A}_d(n)$ that starts and ends in q_2, but does not visit q_2 in between, makes the depth of remaining recursion strictly decrease.*

Proof. By case analysis on whether the partial run executes $\mathcal{A}^i_\lambda$, $\mathcal{A}_{\alpha+1}$, $\mathcal{A}_0$ or $\mathcal{B}^i$. We use the following properties of ordinals.

$$\alpha \geq 0, \beta < \gamma \Rightarrow \alpha + \beta < \alpha + \gamma$$
$$\alpha \neq 0, \beta < \gamma \Rightarrow \alpha.\,\beta < \alpha.\,\gamma$$
$$\alpha > 1, \beta < \gamma \Rightarrow \alpha^\beta < \alpha^\gamma$$

In the following, we denote by $\boldsymbol{v} = (v_{d-1}, \ldots, v_0)$ the vector of natural numbers formed by the values of the counters $\kappa_{d-1}, \ldots, \kappa_0$. The contents of the stack is denoted by σ.

- $\mathcal{A}^i_\lambda$: $(\sigma F, \boldsymbol{v}) \xrightarrow{\mathcal{A}^i_\lambda} (\sigma I_i D^m_{i-1} F, \boldsymbol{v} - \boldsymbol{e}_i + m.\,\boldsymbol{e}_{i-1})$, with $m \in \mathbb{N}, i > 0, v_i > 0$

$$\begin{aligned}
&\mathsf{dr}(\sigma I_i D^m_{i-1} F, \boldsymbol{v} - \boldsymbol{e}_i + m.\,\boldsymbol{e}_{i-1}) \\
&= (\mathsf{Ord}(\sigma I_i D^m_{i-1}, \boldsymbol{v} - \boldsymbol{e}_i + m.\,\boldsymbol{e}_{i-1}) + \omega^{\mathsf{Ord}(\boldsymbol{v} - \boldsymbol{e}_i + m.\boldsymbol{e}_{i-1})}, |\sigma I_i D^m_{i-1} F|) \\
&= (\mathsf{Ord}(\sigma, \boldsymbol{v}) + \omega^{\mathsf{Ord}(\boldsymbol{v} - \boldsymbol{e}_i + m.\boldsymbol{e}_{i-1})}, |\sigma I_i D^m_{i-1} F|) \\
&< (\mathsf{Ord}(\sigma, \boldsymbol{v}) + \omega^{\mathsf{Ord}(\boldsymbol{v})}, |\sigma F|) \\
&= (\mathsf{Ord}(\sigma F, \boldsymbol{v}), |\sigma F|) \\
&= \mathsf{dr}(\sigma F, \boldsymbol{v})
\end{aligned}$$

- $\mathcal{A}_{\alpha+1}$: $(\sigma F, \boldsymbol{v}) \xrightarrow{\mathcal{A}_{\alpha+1}} (\sigma I_0 F^m, \boldsymbol{v} - \boldsymbol{e}_0)$, with $m \in \mathbb{N}, v_0 > 0$

$$\begin{aligned}
\mathsf{dr}(\sigma I_0 F^m, \boldsymbol{v} - \boldsymbol{e}_0) &= (\mathsf{Ord}(\sigma I_0 F^m, \boldsymbol{v} - \boldsymbol{e}_0), |\sigma I_0 F^m|) \\
&= (\mathsf{Ord}(\sigma, \boldsymbol{v}) + \omega^{\mathsf{Ord}(\boldsymbol{v} - \boldsymbol{e}_0)}.\, m, |\sigma I_0 F^m|) \\
&< (\mathsf{Ord}(\sigma, \boldsymbol{v}) + \omega^{\mathsf{Ord}(\boldsymbol{v} - \boldsymbol{e}_0)}.\, \omega, |\sigma F|) \\
&= (\mathsf{Ord}(\sigma, \boldsymbol{v}) + \omega^{\mathsf{Ord}(\boldsymbol{v})}, |\sigma F|) \\
&= (\mathsf{Ord}(\sigma F, \boldsymbol{v}), |\sigma F|) \\
&= \mathsf{dr}(\sigma F, \boldsymbol{v})
\end{aligned}$$

- $\mathcal{A}_0$: $(\sigma F, \boldsymbol{v}) \xrightarrow{\mathcal{A}_0} (\sigma, \boldsymbol{v})$

$$\begin{aligned}
\mathsf{dr}(\sigma, \boldsymbol{v}) &= (\mathsf{Ord}(\sigma, \boldsymbol{v}), |\sigma|) \\
&< (\mathsf{Ord}(\sigma, \boldsymbol{v}) + \omega^{\mathsf{Ord}(\boldsymbol{v})}, |\sigma F|) \\
&= (\mathsf{Ord}(\sigma F, \boldsymbol{v}), |\sigma F|) \\
&= \mathsf{dr}(\sigma F, \boldsymbol{v})
\end{aligned}$$

- $\mathcal{B}^i$ with I_i on top: $(\sigma I_i, \boldsymbol{v}) \xrightarrow{\mathcal{B}^i} (\sigma, \boldsymbol{v} + \boldsymbol{e}_i)$

$$\begin{aligned} \mathsf{dr}(\sigma, \boldsymbol{v} + \boldsymbol{e}_i) &= (\mathsf{Ord}(\sigma, \boldsymbol{v} + \boldsymbol{e}_i), |\sigma|) \\ &< (\mathsf{Ord}(\sigma, \boldsymbol{v} + \boldsymbol{e}_i), |\sigma I_i|) \\ &= (\mathsf{Ord}(\sigma I_i, \boldsymbol{v}), |\sigma I_i|) \\ &= \mathsf{dr}(\sigma I_i, \boldsymbol{v}) \end{aligned}$$

- $\mathcal{B}^i$ with D_i on top: $(\sigma D_i, \boldsymbol{v}) \xrightarrow{\mathcal{B}^i} (\sigma, \boldsymbol{v} - \boldsymbol{e}_i)$, with $v_i > 0$

$$\begin{aligned} \mathsf{dr}(\sigma, \boldsymbol{v} - \boldsymbol{e}_i) &= (\mathsf{Ord}(\sigma, \boldsymbol{v} - \boldsymbol{e}_i), |\sigma|) \\ &< (\mathsf{Ord}(\sigma, \boldsymbol{v} - \boldsymbol{e}_i), |\sigma D_i|) \\ &= (\mathsf{Ord}(\sigma D_i, \boldsymbol{v}), |\sigma D_i|) \\ &= \mathsf{dr}(\sigma D_i, \boldsymbol{v}) \end{aligned}$$

We have shown that, in all cases, the partial run makes the depth of remaining recursion strictly decrease. □