# Coverability Trees for Petri Nets with Unordered Data

Piotr Hofman[1,i], Sławomir Lasota[2,ii], Ranko Lazić[3,iii], Jérôme Leroux[4], Sylvain Schmitz[1,iv], and Patrick Totzke[3,iii]

[1] LSV, ENS Cachan & CNRS, Université Paris-Saclay, France
[2] University of Warsaw, Poland
[3] DIMAP, Department of Computer Science, University of Warwick, UK
[4] LaBRI, CNRS, France

**Abstract.** We study an extension of classical Petri nets where tokens carry values from a countable data domain, that can be tested for equality upon firing transitions. These Unordered Data Petri Nets (UDPN) are well-structured and therefore allow generic decision procedures for several verification problems including coverability and boundedness.
We show how to construct a finite representation of the coverability set in terms of its ideal decomposition. This not only provides an alternative method to decide coverability and boundedness, but is also an important step towards deciding the reachability problem. This also allows to answer more precise questions about the reachability set, for instance whether there is a bound on the number of tokens on a given place (place boundedness), or if such a bound exists for the number of different data values carried by tokens (place width boundedness).
We provide matching Hyper-Ackermann bounds on the size of coverability trees and on the running time of the induced decision procedures.

## 1 Introduction

*Unordered data Petri nets* (UDPN [15]) extend Petri nets by decorating tokens with data values taken from some countable data domain $\mathbb{D}$. These values act as pure names: they can only be compared for equality or non-equality upon firing transitions. Such systems can model for instance distributed protocols where process identities need to be taken into account [21]. UDPNs also coincide with the natural generalisation of Petri nets in the framework of sets with atoms [3]. In spite of their high expressiveness, UDPNs fit in the large family of Petri net extensions among the *well-structured* ones [1, 7]. As such, they still enjoy decision

[i] Partially funded by the Polish National Science Centre grant 2013/09/B/ST6/01575 and by Labex Digicosme, Université Paris-Saclay, project VERICONISS.
[ii] Partially funded by the Polish National Science Centre grant 2012/07/B/ST6/01497.
[iii] Partially supported by the EPSRC grant EP/M011801/1.
[iv] Partially supported by the ANR grant ANR-14-CE28-0005 PRODAQ and by the Leverhulme Trust Visiting Professorship VP1-2014-041.

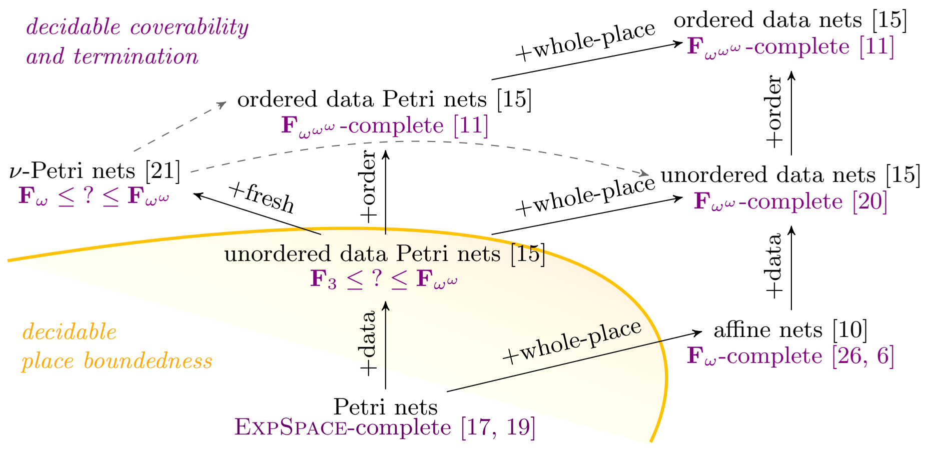
decidable coverability
and termination
ordered data nets [15]
$\mathbf{F}_{\omega^{\omega^\omega}}$-complete [11]
+whole-place
ordered data Petri nets [15]
$\mathbf{F}_{\omega^{\omega^\omega}}$-complete [11]
+order
$\nu$-Petri nets [21]
$\mathbf{F}_\omega \leq ? \leq \mathbf{F}_{\omega^\omega}$
+fresh
+order
+whole-place
unordered data nets [15]
$\mathbf{F}_{\omega^\omega}$-complete [20]
unordered data Petri nets [15]
$\mathbf{F}_3 \leq ? \leq \mathbf{F}_{\omega^\omega}$
+data
+data
decidable
place boundedness
+whole-place
affine nets [10]
$\mathbf{F}_\omega$-complete [26, 6]
Petri nets
ExpSpace-complete [17, 19]

**Fig. 1.** A short taxonomy of some well-structured extensions of Petri nets. Complexities in violet refer to the coverability and termination problems, and can be taken as proxies for expressiveness; the exact complexities of coverability and termination in $\nu$-Petri nets and UDPNs are unknown at the moment. Place boundedness is decidable below the yellow line and undecidable above. As indicated by the dashed arrows, freshness can be enforced using a dense linear order or whole-place operations.

procedures for several verification problems, prominently safety (through the *coverability* problem) and termination.

Unordered data Petri nets have an interesting position in the taxonomy of well-structured Petri net extensions (see Fig. 1). Indeed, all their extensions forgo the decidability of the *reachability* problem (whether a target configuration is reachable) and of the *place boundedness* problem (whether the number of tokens in a given place can be bounded along all runs): this is the case of *$\nu$-Petri nets* [21] that allow to create fresh data values, of *ordered data Petri nets* [15] that posit a dense linear ordering on $\mathbb{D}$, and of *unordered data nets* [15] that allow to perform 'whole-place' operations, which move and/or duplicate all the tokens from a place to another. By contrast, it is currently open whether reachability is decidable in UDPNs, and a consequence of our results in this paper is that place boundedness is decidable—which is a significant first step if we wish to adapt to UDPNs some of the known algorithms for reachability in Petri nets [18, 13].

*Contributions.* In this paper, we show how to construct finite *coverability trees* for UDPNs, adapting the existing construction of Karp and Miller [12] for Petri nets. Such trees are constructed forward from an initial configuration like reachability trees, but approximate the latter by *accelerating* sequences of transitions and explicitly manipulating limits of reachable configurations as downwards-closed sets (see Sec. 4.1). We rely for all this on a general theory for representing downwards-closed sets as finite unions of *ideals* developed by Finkel and Goubault-Larrecq [9] for this exact purpose (see Sec. 3).

Coverability trees contain a wealth of information about the system at hand, and allow to answer various coverability and boundedness questions—allowing us to derive a new result: the place boundedness problem is decidable for UDPNs, and so are its variants, like place width- and place depth boundedness (see Sec. 2). We also establish in Sec. 5 matching 'hyper-Ackermannian' lower and upper bounds on the size of UDPNs coverability trees. This yields $\mathbf{F}_{\omega^\omega}$ upper bounds for the already mentioned decidable problems in UDPNs, in terms of the fast-growing complexity classes $(\mathbf{F}_\alpha)_\alpha$ from [23]. These complexity results rely largely on the work of Rosa-Velardo [20] on the complexity of coverability in unordered data nets. Due to space constraints, most proof details are omitted but can be found online in the full version of the paper available from `https://hal.inria.fr/hal-01252674`.

*Related Work.* A coverability tree construction has already been undertaken by Rosa-Velardo, Martos-Salgado, and de Frutos-Escrig [22] in the case of $\nu$-Petri nets, and inescapably there are many similarities between their work and ours. Our construction does however not merely remove freshness constraints from theirs: (1) we start anew and rely on a strong invariant on the form of ideals in UDPN coverability trees, which leads to significant simplifications but would be markedly difficult to extract from Rosa-Velardo et al.'s construction, and (2) coverability trees are not necessarily finite for $\nu$-Petri nets (place boundedness is indeed undecidable [21]), which means that our termination argument and complexity bounds are entirely new considerations.

Like Rosa-Velardo et al. [22] we rely on the work of Finkel and Goubault-Larrecq [9] on forward analysis of well-structured systems. Finkel and Goubault-Larrecq provide in particular an abstract generic procedure, but without guarantee of termination in general, and their framework needs to be instantiated for each specific class of systems.

## 2 Model

Our presentation of unordered data Petri nets differs from the original one [15] on two counts: we work with an equivalent formalism with more of a *vector addition system* [12] flavour, and because we need to work with ideals we define the syntax and the semantics on extended configurations, which allow for infinitely many different data values and infinite counts.

Let $\mathbb{Z}$ and $\mathbb{N}$ denote the sets of integers and non-negative integers respectively, and complete them as $\mathbb{Z}_\omega \overset{\text{def}}{=} \mathbb{Z} \uplus \{\omega\}$ and $\mathbb{N}_\omega \overset{\text{def}}{=} \mathbb{N} \uplus \{\omega\}$ with a new top element $\omega$ with $\omega > z$ and $z + \omega = \omega + z = \omega$ for all $z$ in $\mathbb{Z}$. Given a dimension $k$ in $\mathbb{N}$, we denote the projection into the $i$th component of a vector $\boldsymbol{v} \in \mathbb{Z}_\omega^k$ by $\boldsymbol{v}[i]$ and define the product ordering and sum over $\mathbb{Z}_\omega^k$ componentwise: $\boldsymbol{u} \leq \boldsymbol{v}$ if $\boldsymbol{u}[i] \leq \boldsymbol{v}[i]$ for all $1 \leq i \leq k$, and $(\boldsymbol{u} + \boldsymbol{v})[i] \overset{\text{def}}{=} \boldsymbol{u}[i] + \boldsymbol{v}[i]$ for all $1 \leq i \leq k$. We write $\mathbf{0}$ for the vector with 0 on all components.

*Data Vectors.* Fix some countable domain $\mathbb{D}$ of data values and a dimension $k$ in $\mathbb{N}$. A *data vector* is a function $f\colon \mathbb{D} \to \mathbb{Z}_\omega^k$.Data vectors can be partially ordered and summed pointwise: $f \leq g$ if $f(d) \leq g(d)$ for all $d$ in $\mathbb{D}$, and $(f+g)(d) \overset{\text{def}}{=} f(d)+g(d)$ for all $d$ in $\mathbb{D}$. As usual we write $f < g$ if $f \leq g$ and $f(d) < g(d)$ for some $d$ in $\mathbb{D}$. For a subset $K \subseteq \{1,\ldots,k\}$ we write $f{\restriction_K}$ for the projection of $f$ into components in $K$: for all $d$ in $\mathbb{D}$, $f{\restriction_K}(d) \overset{\text{def}}{=} (f(d)){\restriction_K}$. The *support* of a data vector $f$ is $Supp_{\mathbf{0}}(f) \overset{\text{def}}{=} \{d \in \mathbb{D} \mid f(d) \neq \mathbf{0}\}$; $f$ is *finitely supported* if $Supp_{\mathbf{0}}(f)$ is finite. We say that a data vector $f$ is *non-negative* if $f(d)$ belongs to $\mathbb{N}_\omega^k$ for all $d$ in $\mathbb{D}$. It is *finite* if $f(d)$ belongs to $\mathbb{Z}^k$ for all $d$ and it is finitely supported.

We call bijections $\sigma : \mathbb{D} \to \mathbb{D}$ *(data) permutations* and write $f\sigma$ for the composition of a permutation $\sigma$ and a data vector $f$. If two data vectors $f, g$ satisfy $f = g\sigma$ for some permutation $\sigma$, we say $f$ and $g$ are equal *up to permutation* and write $f \equiv g$.

In the sequel we will consider sets of data vectors that are *finite up to permutation*: a set $X$ of data vectors is finite up to permutation if there is a finite subset $X'$ of $X$ such that every $f \in X$ is equal up to permutation to some $f' \in X'$. Note that if $X$ is closed under permutations then $X'$ can be used as its finite presentation; any such finite set $X'$ we call *representative* of $X$.

**Definition 2.1 (UDPN).** *An* unordered data Petri net *(UDPN) is a finite set $\mathcal{T}$ of finite data vectors. A* transition *is a data vector $t \overset{\text{def}}{=} f\sigma$, where $f \in \mathcal{T}$ and $\sigma$ is a data permutation. There is a* step *$f \xrightarrow{t} g$ between non-negative data vectors $f, g$ if $g = f + t$ for some transition $t$. Note that this enforces that $f(d)+t(d) \geq 0$ for all $d$ in $\mathbb{D}$ since $g$ is non-negative. We simply write $f \to g$ if $f \xrightarrow{t} g$ for some transition $t$ and let $\xrightarrow{*}$ denote the transitive and reflexive closure of $\to$.*

*A* configuration *is a finite non-negative data vector $f$, i.e. $f(d)$ belongs to $\mathbb{N}^k$ for all $d$ in $\mathbb{D}$, and $f(d) = \mathbf{0}$ for almost all $d$ in $\mathbb{D}$. We write Confs for the set of configurations and note that Confs is closed under UDPN steps. The* reachability *set from a given vector $f$ is defined as*

$$Reach(f) \overset{\text{def}}{=} \{g \in Confs \mid f \xrightarrow{*} g\}. \tag{1}$$

Observe that UDPNs over any domain with cardinality $|\mathbb{D}| = 1$ are classical vector addition systems [12]. Notice also that, the set of transitions in an UDPN is finite up to permutations of $\mathbb{D}$ and that the step relation is closed under permutations: For every non-negative data vector $f$, transition $t$ and permutation $\sigma$ we have that

$$f \xrightarrow{t} g \text{ implies } f\sigma \xrightarrow{t\sigma} g\sigma. \tag{2}$$

*Example 2.2.* For the domain $\mathbb{D} \overset{\text{def}}{=} \mathbb{N}$ and $k \overset{\text{def}}{=} 2$, consider a 2-dimensional UDPN $\mathcal{T} = \{t_1, t_2\}$, with vectors $t_1, t_2$ defined as $t_1 : 0 \mapsto (1,0)$ and $t_1 : n \mapsto (0,0)$ for all $n > 0$, and $t_2 : 0 \mapsto (-1,-1)$, $t_2 : 1 \mapsto (1,1)$ and $t_2 : n \mapsto (0,0)$ for all $n > 1$.

The configuration $f_0$ with $f_0 : 0 \mapsto (1,1)$ and $f_0 : n \mapsto (0,0)$ for $n > 0$, has infinitely many $t_1$-successors. Namely, $g_i \overset{\text{def}}{=} f_0 + t_1\sigma_i$ for every permutation

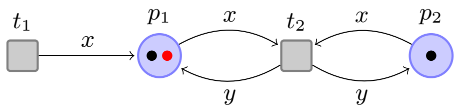

**Fig. 2.** A place/transition representation of the UDPN of Ex. 2.2 in the style of [21, 11]. Different data values are depicted though differently coloured tokens in places (the circles), and through differently named variables in transitions (the boxes and arrows).

$\sigma_i$ that swaps 0 and $i \in \mathbb{D}$. However, there are only two such successors up to permutation because $g_i \equiv g_j$ for all $i, j > 0$. The reachability set of $f_0$ is

$$\mathit{Reach}(f_0) = \left\{ g \;\middle|\; \begin{array}{l} \exists d_1, d_2, \ldots, d_m \in \mathbb{D}\; \exists n_1, n_2, \ldots, n_m \in \mathbb{N} \\ g(d_1) = (n_1, 1) \text{ and } \forall 1 < i \leq m,\; g(d_i) = (n_i, 0), \\ \forall d \in \mathbb{D} \setminus \{d_1, d_2, \ldots, d_m\}\; g(d) = (0,0) \end{array} \right\}. \quad (3)$$

In the sequel, we present UDPNs only up to permutation in matrix form by juxtaposing the vectors from their finite supports: we write $t_1 \overset{\text{def}}{=} \left[\begin{smallmatrix} 1 \\ 0 \end{smallmatrix}\right]$, $t_2 \overset{\text{def}}{=} \left[\begin{smallmatrix} -1 & 1 \\ -1 & 1 \end{smallmatrix}\right]$ and $f_0 \overset{\text{def}}{=} \left[\begin{smallmatrix} 1 \\ 1 \end{smallmatrix}\right]$. The $t_1$-successors of $f_0$ are $g_0 = \left[\begin{smallmatrix} 2 \\ 1 \end{smallmatrix}\right]$ and $g_i = \left[\begin{smallmatrix} 1 \\ 1 \end{smallmatrix}\right] + \left[\begin{smallmatrix} 1 \\ 0 \end{smallmatrix}\right] = \left[\begin{smallmatrix} 1 & 1 \\ 1 & 0 \end{smallmatrix}\right]$ for $i \neq 0$. The latter is depicted in Fig. 2 in the style of coloured place/transition nets. The reachability set of $f_0$ can be written as $\mathit{Reach}(f_0) = \left\{ \left[\begin{smallmatrix} n_0 & n_1 & \cdots & n_m \\ 1 & 0 & \cdots & 0 \end{smallmatrix}\right] \mid m \geq 0, n_0, \ldots, n_m \geq 1 \right\}$.

*Embeddings.* We say that a data vector $f$ *embeds* into a data vector $g$ and write $f \sqsubseteq g$ (resp. $f \sqsubset g$) if there exists an injection $\pi \colon \mathbb{D} \to \mathbb{D}$ such that $f \leq g\pi$ (resp. $f < g\pi$). The injection $\pi$ itself is called an *embedding* (of $f$ into $g$) and a *permutation embedding* in case it is bijective. Given a set of configurations $C$, its *downward-closure* ${\downarrow}C$ is $\{f \in \mathit{Confs} \mid \exists g \in C \,.\, f \sqsubseteq g\}$, and as usual a set $C$ is *downwards-closed* if ${\downarrow}C = C$.

Finitely supported data vectors are isomorphic to finite multisets of vectors in $\mathbb{Z}_\omega^k$ when working up to data permutation. Moreover, on permutation classes of finitely supported data vectors, the embedding ordering coincides with the usual embedding ordering over finite multisets of vectors; in consequence, UDPN configurations are well-quasi-ordered by the embedding ordering. Thus an UDPN defines a quasi-ordered transition system $(\mathit{Confs}, \to, \sqsubseteq)$, which satisfies a (strong) *compatibility* condition as shown in the following lemma. Together with the fact that $(\mathit{Confs}, \sqsubseteq)$ is a wqo, this entails that it is a *well-structured* transition system in the sense of [1, 7].

**Lemma 2.3 (Strong Strict Compatibility).** *Let $f, f', g$ be configurations. If $f \sqsubseteq f'$ (resp. $f \sqsubset f'$) and $f \to g$, then there exists a configuration $g'$ with $f' \to g'$ and $g \sqsubseteq g'$ (resp. $g \sqsubset g'$).*

*Proof.* Consider a finite data vector $t$ such that $f \xrightarrow{t} g$, and a permutation $\pi$ of $\mathbb{D}$ such that $f \leq f'\pi$ (recall that, when working with finitely supported data vectors, embeddings can be assumed to be permutations). We claim that

$g' \overset{\text{def}}{=} f' + t\pi^{-1}$ satisfies $f' \xrightarrow{t\pi^{-1}} g'$ and $g \le g'\pi$. Indeed, for all $d$ in $\mathbb{D}$, noting $e \overset{\text{def}}{=} \pi(d)$,

$$g(d) = f(d) + t(d) \le f'(\pi(d)) + t(d) = f'(e) + t(\pi^{-1}(e)) = g'(e) = g'(\pi(d)) \ .$$

Furthermore, assuming $f < f'\pi$, for at least one $d$ in $\mathbb{D}$ the above inequality is strict. □

*Decision Problems.* For the purpose of verification, we are interested in standard decision problems for UDPN, including *reachability* (does $f \xrightarrow{*} g$ hold for given configurations $f$ and $g$?), *coverability* (given configurations $f, g$, does there exists $g' \sqsupseteq g$ s.t. $f \xrightarrow{*} g'$?), and *boundedness* (is *Reach*$(f)$ finite up to permutation?).

While the decidability of reachability remains open, well-structuredness of UDPN (and some basic effectiveness assumptions) implies that the coverability and boundedness problems are decidable using the generic algorithms from [1, 7]. In fact, decidability holds more generally for ordered data Petri nets [15, Thm. 4.1]. For the coverability problem, Rosa-Velardo [20, Thm. 1] proved an HyperAckermann upper bound ($\mathbf{F}_{\omega^\omega}$ in the hierarchy from [23]; see Sec. 5), while Lazić et al. [15, Thm. 5.2] proved a Tower lower bound ($\mathbf{F}_3$ in the same hierarchy). The complexity of the boundedness problem has not been studied before. Furthermore, the following, more precise variant of boundedness called *place boundedness* was not known to be decidable:

**Input:** An UDPN, a configuration $f$, and a set of coordinates $K \subseteq \{1, \ldots, k\}$.
**Question:** Is $\{g{\restriction}_K \mid g \in \mathit{Reach}(f)\}$ finite up to permutation?

In the presence of an infinite data domain $\mathbb{D}$, place boundedness can be further refined: even if infinitely many configurations are reachable, the system can still be bounded in the sense that there exists a bound on the number of different data values in reachable configurations; Rosa-Velardo and de Frutos-Escrig [21] call this *bounded width.* Similarly, there may exist some bound on the multiplicities with which any data value occurs, while the number of different data values is unbounded; Rosa-Velardo and de Frutos-Escrig [21] call this *bounded depth.*

We formalise the resulting decision problems in our notation as follows. The *place width boundedness* problem is given as:

**Input:** An UDPN, a configuration $f$, and a set of coordinates $K \subseteq \{1, \ldots, k\}$.
**Question:** Is $\{|\mathit{Supp}_{\mathbf{0}}(g{\restriction}_K)| \mid g \in \mathit{Reach}(f)\}$ finite?

The *place depth boundedness* problem is the following:

**Input:** An UDPN, a configuration $f$, and a set of coordinates $K \subseteq \{1, \ldots, k\}$.
**Question:** Is $\{g{\restriction}_K(d) \mid g \in \mathit{Reach}(f), d \in \mathbb{D}\}$ finite?

If the answer to the depth (width) boundedness problem is positive we call those components $i \in K$ depth (width) bounded.

*Example 2.4.* From the initial configuration $f_0 = \left[\begin{smallmatrix}1\\1\end{smallmatrix}\right]$, the UDPN from Ex. 2.2 can reach any configuration of the form $\left[\begin{smallmatrix}n\\1\end{smallmatrix}\right]$ in $n$ many $t_1$-steps, all exercised on the same data value. Similarly, any configuration of the form $\left[\begin{smallmatrix}1 & 1 & \cdots & 1\\1 & 0 & \cdots & 0\end{smallmatrix}\right]$ with a support of size $n+1$ can be reached after a sequence of $n$ transitions $t_1$, each exercised on a different data value. Consequently, the first component is neither depth nor width bounded.

However, any reachable configuration $g$ will satisfy $\sum_{d\in\mathbb{D}} g(d)[2] = 1$. In Petri net parlance, there is always exactly one token in the second place. The system is thus place bounded for $K \overset{\text{def}}{=} \{2\}$.

The main contribution of this paper is the effective computability of a suitable abstraction of the classical coverability tree construction [12] for UDPNs. This provides a way to decide all variants of the boundedness problem mentioned above. We summarise the consequences of our construction below.

**Theorem 2.5.** *In UDPNs, place depth boundedness implies place width boundedness. In consequence, place depth boundedness coincides with place boundedness.*

**Theorem 2.6.** *The boundedness, place boundedness, and place width boundedness problems for UDPNs are in* $\mathbf{F}_{\omega^\omega}$*, i.e. in* HyperAckermann.

Let us emphasise the importance of decidability of place boundedness: first, the problem is undecidable in all the extensions of UDPNs in Fig. 1. Moreover, in the case of Petri nets, the decidability of place boundedness plays a crucial role in the decidability proofs for reachability [18, 13, 14, 16], hence Thm. 2.6 provides one of the basic building blocks for future attempts at proving the decidability of reachability for UDPNs.

## 3 Simple Ideals

A key observation about all decision problems mentioned in the previous section is that they do not require computing the reachability set: they can all be solved given some suitable representation of the *cover* [9], defined as

$$\mathit{Cover}(f) \overset{\text{def}}{=} {\downarrow}\mathit{Reach}(f), \tag{4}$$

for $f$ the initial configuration. Indeed, coverability reduces to checking whether $g \in \mathit{Cover}(f)$, boundedness to checking whether $\mathit{Cover}(f)$ is finite up to permutation of $\mathbb{D}$, and place boundedness to checking whether $\{g{\restriction_K} \mid g \in \mathit{Cover}(f)\}$ is finite up to permutation of $\mathbb{D}$. The main property of the coverability tree we construct in Sec. 4 is that we can extract a suitable representation of $\mathit{Cover}(f)$.

*Ideals and Clovers.* We refer the reader to the work of Finkel and Goubault-Larrecq [8, 9] for details; it suffices to say that downwards-closed sets of configurations can be represented as finite unions of so-called *configuration ideals.* Formally, a configuration ideal $J$ is a non-empty, downwards-closed, and *directed* set of configurations; this last condition means that, if $f$ and $f'$ are configurations in $J$, then there exists $h$ in $J$ with $f \sqsubseteq h$ and $f' \sqsubseteq h$. Crucially for algorithmic considerations, a configuration ideal $J$ can in turn be *represented* as the downward-closure

$$J = {\downarrow}g \overset{\text{def}}{=} \{h \in \mathit{Confs} \mid h \sqsubseteq g\} \tag{5}$$

of a non-negative data vector $g$ having a finite range: $g(\mathbb{D})$ is a finite subset of $\mathbb{N}_\omega^k$. We can check that every such ${\downarrow}g$ is a configuration ideal (see [8, 9] for the converse): it is non-empty and downwards-closed by definition, and we can check it is also directed. Indeed, if $f$, $f'$ are configurations and $\pi$, $\pi'$ are injections with $f \leq g\pi$ and $f' \leq g\pi'$, then since $f$ and $f'$ are finitely supported we can assume $\pi$ and $\pi'$ to be permutations, and we can define a configuration $h \leq g$ such that $f \leq h\pi$ and $f' \leq h\pi'$: set $h$ as the pointwise least upper bound $h(d)[i] \overset{\text{def}}{=} \max(f(\pi^{-1}(d))[i], f'(\pi'^{-1}(d))[i]) \leq g(d)[i]$ for all $d$ and $1 \leq i \leq k$.

$\mathit{Cover}(f)$, being downwards-closed, is represented by a finite set of representations of configuration ideals, called $\mathit{Clover}(f)$ by Finkel and Goubault-Larrecq:

$$\mathit{Cover}(f) = \bigcup\{{\downarrow}g \mid g \in \mathit{Clover}(f)\} .$$

$\mathit{Clover}(f)$ is determined uniquely up to permutation, and contains $\sqsubseteq$-maximal data vectors $g$ satisfying ${\downarrow}g \subseteq \mathit{Cover}(f)$; for further details see [8, 9]. In the following we identify a configuration ideal $J = {\downarrow}g$ with its representation $g$.

*Remark 3.1 (Ideals for Petri nets).* For readers familiar with Karp and Miller's coverability trees for Petri nets, observe that configuration ideal representations generalise the notion of 'extended markings', which are vectors in $\mathbb{N}_\omega^k$. Also, $\mathit{Clover}(f)$ for a Petri net can be computed as the set of vertex labels in its coverability tree—this will also be our case.

*Simple Ideals.* Crucially, it turns out that we do not need general configuration ideals for our coverability trees for UDPNs. We only need to consider the downward-closures ${\downarrow}g$ of non-negative vectors $g$ (cf. (5)), where the set of vectors appearing infinitely often as $g(d)$, when $d$ ranges over $\mathbb{D}$, is a singleton $\{\boldsymbol{I}\}$ for some vector $\boldsymbol{I}$ in $\{0, \omega\}^k$ (instead of a finite subset of $\mathbb{N}_\omega^k$ for general configuration ideals). Put differently, given such a vector $\boldsymbol{I}$, we define the *$\boldsymbol{I}$-support* of a data vector $f$ as $\mathit{Supp}_{\boldsymbol{I}}(f) \overset{\text{def}}{=} \{d \in \mathbb{D} \mid f(d) \neq \boldsymbol{I}\}$, and define an *$\boldsymbol{I}$-simple ideal (representation)* as a non-negative data vector with finite $\boldsymbol{I}$-support. In particular, a finitely supported non-negative data vector is a $\mathbf{0}$-simple ideal. We write $M$, $N$, ... to denote simple ideals. A simple ideal $M$ can be represented concretely as a pair $M = \langle m, \boldsymbol{I}\rangle$ where $m$ is the finite multiset of vectors in $\mathbb{N}_\omega^k$ obtained from $M$ by restriction to its $\boldsymbol{I}$-support.

*Example 3.2.* We represent simple ideals similarly as configurations, using the additional last column for the $\boldsymbol{I}$ part. Continuing with the UDPN of Ex. 2.2, its cover is the downward-closure of a single $\boldsymbol{I}$-simple ideal:

$$\mathit{Clover}(f_0) = \left[\begin{smallmatrix}\omega\\1\end{smallmatrix}\middle|\begin{smallmatrix}\omega\\0\end{smallmatrix}\right] , \qquad \mathit{Cover}(f_0) = {\downarrow}\left[\begin{smallmatrix}\omega\\1\end{smallmatrix}\middle|\begin{smallmatrix}\omega\\0\end{smallmatrix}\right] ,$$

where $\boldsymbol{I} \overset{\text{def}}{=} (\omega, 0)$. The $\boldsymbol{I}$-support of the ideal has one element, mapped to $(\omega, 1)$.

Note that UDPN steps map $\boldsymbol{I}$-simple ideals to $\boldsymbol{I}$-simple ideals. Lemma 3.3 formally states the relation between steps of ideals and steps of configurations in the downward closures. The next lemma shows that $\boldsymbol{I}$-simple ideals can only have finitely many successors up to permutation. This property will later be used to define coverability trees of finite branching degree.

**Lemma 3.3.** *Let $M, M'$ be $\boldsymbol{I}$-simple ideals such that $M \to M'$. Then for every configuration $c' \in \downarrow M'$ there exist configurations $c \in \downarrow M$ and $c'' \in \downarrow M'$ with $c \to c''$ and $c' \sqsubseteq c''$.*

*Proof.* Suppose $M \xrightarrow{t} M'$ for a finite data vector $t$. The data vector $f \overset{\text{def}}{=} c' - t$ satisfies $f \leq M$ but $f(d)$ can possibly be negative for some data value $d$; therefore we can not simply put $c \overset{\text{def}}{=} f$. A way to fix this is to define $c$ by

$$c(d)[i] \overset{\text{def}}{=} \max(0, x(d)[i]), \qquad \text{for all } d \in \mathbb{D} \text{ and all coordinates } i.$$

Thus defined, $c$ satisfies $c \leq M$, and setting $c'' \overset{\text{def}}{=} c + t$ satisfies $c' \leq c''$ as required. □

**Lemma 3.4.** *Let $M$ be a simple ideal. The set $\{N \mid M \to N\}$ of successors of $M$ is finite up to permutation and has a representative with cardinality bounded by $(|Supp_{\boldsymbol{I}}(M)| + \max_{t\in\mathcal{T}} |Supp_{\mathbf{0}}(t)|)! \cdot |\mathcal{T}|^2$.*

*Proof.* Consider an UDPN defined by the finite set $\mathcal{T}$ of data vectors. Fix an $\boldsymbol{I}$-simple ideal $M$, and denote by $S$ the $\boldsymbol{I}$-support of $M$. We will be now considering *$S$-permutations*, by which we mean those data permutations $\pi$ that satisfy $\pi(d) = d$ for all $d \in S$. Equality and finiteness up to $S$-permutation can be defined exactly as for plain permutations.

A crucial but simple observation is that the set of transitions of the UDPN is finite up to $S$-permutation. Indeed, assume wlog. that $S$ is disjoint from the supports of all vectors in $\mathcal{T}$. Consider the finite set $\mathcal{T}'$ that contains all data vectors $t\sigma$, where $t \in \mathcal{T}$ and permutation $\sigma$ swaps some subset of $S$ with some subset of the support of $t$. Then every transition of the UDPN is of the form $t'\pi$, where $t' \in \mathcal{T}'$ and $\pi$ is an $S$-permutation. Regarding the size of this new UDPN, there are at most $\sum_{t\in\mathcal{T}}(|S| + |Supp_{\mathbf{0}}(t)|)!$ such permutations $\sigma$, hence $|\mathcal{T}'| \leq \sum_{t\in\mathcal{T}}(|S| + |Supp_{\mathbf{0}}(t)|)! \cdot |\mathcal{T}|$.

Now we use the extension, to simple ideals, of the closure of the step relation under permutations, cf. Eq. (2), to derive a strengthening of our claim, namely finiteness of the successors of $M$ up to $S$-permutations. Consider an arbitrary step $M \xrightarrow{t'\pi} N$ of $M$; by Eq. (2) we get

$$M = M\pi^{-1} \xrightarrow{t'} N\pi^{-1}$$

(the equality holds as $\pi$ is an $S$-permutation). Therefore $N$ is equal up to $S$-permutation to some $\mathcal{T}'$-successor of $M$. As $\mathcal{T}'$ is finite, the set of $\mathcal{T}'$-successors of $M$ is finite and bounded by $|\mathcal{T}'|$, which implies our claim. □

A consequence of our construction of coverability trees in Sec. 4, and of the complexity analysis conducted in Sec. 5, is the following core result:

**Theorem 3.5.** *Given an UDPN and an initial configuration $f$, an ideal representation $Clover(f)$ of $Cover(f)$ is computable in $\mathbf{F}_{\omega^\omega}$. Furthermore, $Clover(f)$ contains only simple ideals.*

Theorem 3.5, together with the following proposition, easily imply Theorems 2.5 and 2.6 (below $Clover(f)\restriction_K \stackrel{\text{def}}{=} \{g\restriction_K \mid g \in Clover(f)\}$):

**Proposition 3.6.** *Fix $K \subseteq \{1, \ldots, k\}$. An UDPN is width-bounded iff $Clover(f)\restriction_K$ contains only finitely supported vectors. An UDPN is depth-bounded iff $Clover(f)\restriction_K$ contains only finite vectors.*

*Proof.* The former equivalence, as well as the if direction of the latter one, follow by finiteness of $Clover(f)\restriction_K$. It remains to argue that place depth-boundedness forces $Clover(f)\restriction_K$ to contain only finite vectors. Indeed, a non-finite simple ideal has necessarily $\omega$ at some component, which implies depth-unboundedness. □

The remaining part of the paper is devoted to the proof of Thm. 3.5. In Sec. 4, we present an algorithmic construction of the coverability tree, and show its termination and correctness. Then in Sec. 5 we provide upper and lower bounds on the size of the coverability tree.

## 4 Representing a Cover

We will show that, analogously to the classical construction of Karp and Miller [12] for vector addition systems, the cover set of any UDPN configuration can be effectively represented in the form of a finite *coverability tree*, where nodes are labelled by simple ideals.

For a given initial ideal the construction of a coverability tree amounts to iteratively computing successors (up to permutation), applying symbolic *acceleration* steps when a strictly dominating pair $M \sqsubset M'$ appears on a branch, and terminating a branch if a label embeds into one of its ancestors.

### 4.1 Accelerations

The idea behind acceleration steps is that due to monotonicity (Lem. 2.3), any finite sequence of steps

$$M_0 \xrightarrow{t_1} M_1 \xrightarrow{t_2} \ldots \xrightarrow{t_k} M_k \tag{6}$$

such that $M_0 \sqsubset M_k$ can be extended indefinitely. Such an unfolding may have two distinct kinds of effect: Firstly, it may unboundedly increase components in data values already contained in the initial $\boldsymbol{I}$-support (we call this effect *depth acceleration*). Secondly, it may increase an unbounded number of 'fresh' data values, outside of the initial $\boldsymbol{I}$-support (we call this effect *width acceleration*). Our construction only accelerates increasing sequences as above when there is a permutation (i.e. bijective) embedding of $M_0$ into $M_k$.

As a building block we shall use the usual vector acceleration: for two non-negative vectors $\boldsymbol{v}, \boldsymbol{v}' \in \mathbb{N}_\omega^k$ with $\boldsymbol{v}' \le \boldsymbol{v}$, define a new vector $\mathrm{acc}(\boldsymbol{v}', \boldsymbol{v})$, for $1 \le i \le k$ by:

$$\mathrm{acc}(\boldsymbol{v}', \boldsymbol{v})[i] \stackrel{\text{def}}{=} \begin{cases} \boldsymbol{v}[i], & \text{if } \boldsymbol{v}'[i] = \boldsymbol{v}[i], \\ \omega, & \text{if } \boldsymbol{v}'[i] < \boldsymbol{v}[i]. \end{cases}$$

**Definition 4.1 (Depth and Width Acceleration).** *For $\boldsymbol{I}$-simple ideals $M'$, $M$ and a permutation $\pi$ with $M' < M\pi$, or equivalently $M'\pi^{-1} < M$, the* depth acceleration *of $M', M, \pi$ is the $\boldsymbol{I}$-simple ideal defined by*

$$M_{depth}(d) \stackrel{def}{=} acc(M'(\pi^{-1}(d)), M(d)), \qquad \text{for all data values } d \in \mathbb{D}.$$

*For $d \in \mathbb{D}$ such that $M'(\pi^{-1}(d)) = \boldsymbol{I} < M(d)$, put $\boldsymbol{I}_d \stackrel{def}{=} acc(M'(\pi^{-1}(d)), M(d))$; the* width acceleration *of $M', M, \pi, d$ is the $\boldsymbol{I}_d$-simple ideal defined by*

$$M_{width}(d) \stackrel{def}{=} \begin{cases} \boldsymbol{I}_d, & \text{if } M(d) = \boldsymbol{I} \\ M(d), & \text{otherwise,} \end{cases}$$

By definition, $M < M_{\text{depth}}, M_{\text{width}}$.

## 4.2 Coverability Trees

By Lem. 3.4 we can compute for any $\boldsymbol{I}$-simple ideal $M$ a *successor representative*, namely a finite set such that every successor of $M$ is equal up to permutation to some element of this set.

For the sake of simplicity, we choose a conservative policy of application of accelerations: first, a proper nesting is imposed, in the sense that two different accelerated paths are either disjoint, or contained one in the other; second, a depth-accelerated path can not contain another accelerated path, while a width-accelerated path can. However, as width accelerations strictly increase the $\boldsymbol{I}$ part, a width-accelerated path is never contained in another accelerated path. Therefore the only allowed inclusion is when a depth-accelerated path is included in a width-accelerated one.

**Definition 4.2 (Coverability Tree).** *A* coverability tree *is a tree with nodes labelled by simple ideals such that the following criteria are satisfied.*

1. *A node with label $N$ is a leaf iff it has an ancestor with label $N' \sqsupseteq N$.*
2. *Otherwise, suppose an interior node $N$ has an ancestor $N'$ such that both $N'$, $N$ are $\boldsymbol{I}$-simple and $N' \sqsubset N$. Let $\mathcal{P}$ denote the path from $N'$ to $N$ in the tree, including $N'$ and $N$.*
   - (a) *Suppose $N'(\pi^{-1}(d)) = \boldsymbol{I} < N(d)$ for some permutation $\pi$ with $N'\pi < N$ and $d \in \mathbb{D}$; and for every node in $\mathcal{P}$ that is a depth acceleration of some nodes $M', M$, both $M'$ and $M$ belong to $\mathcal{P}$. Then $N$ has exactly one child labelled by the width acceleration of $N', N, \pi, d$.*
   - (b) *Otherwise, if $\mathcal{P}$ contains no acceleration then $N$ has exactly one child labelled by the depth acceleration of $N', N, \pi$, for some permutation $\pi$ with $N'\pi < N$.*

*3. Otherwise, if a node $N$ satisfies none of the above criteria then its set of children is the successor representative of $N$.*

*Remark 4.3.* Note that Def. 4.2 does not determine the coverability tree unambiguously: the choice of a permutation $\pi$ in points 2(a) and 2(b) is not unique.

*Remark 4.4.* The condition in point 1 in Def. 4.2 implies that no branch of a coverability tree contains two different nodes with the same label. We identify a node with its label in the sequel.

*Example 4.5.* We pick some coverability tree for the UDPN from Ex. 2.2 rooted in the configuration $f_0$. There is a branch with labels (up to permutation)

$$f_0=\left[\begin{array}{c|c}1&0\\1&0\end{array}\right],\ f_1=\left[\begin{array}{c|c}2&0\\1&0\end{array}\right],\ f_2=\left[\begin{array}{c|c}\omega&0\\1&0\end{array}\right],\ f_3=\left[\begin{array}{cc|c}\omega&1&0\\0&1&0\end{array}\right],\ f_4=\left[\begin{array}{cc|c}\omega&1&\omega\\0&1&0\end{array}\right],\ f_5=\left[\begin{array}{ccc|c}\omega&1&1&\omega\\0&1&0&0\end{array}\right],$$

where $f_2$ is a depth acceleration (of $f_0, f_1$), $f_4$ is a width acceleration (of $f_0, f_3$), and all other nodes are the result of successor steps from their parent. The node $f_5$ is a leaf because $f_5 \sqsubseteq f_4$.

*Correctness.* A coverability tree is finite (termination), and represents the cover of its root node (completeness and soundness). These required properties are proven in detail in the full paper:

- *Finiteness* is proven by first exhibiting a wqo for the specific type of $\boldsymbol{I}$-simple ideals that appears on coverability trees. This wqo depends on the existence of permutation embeddings, a property that on its own does *not* induce a well-quasi-ordering over the set of all $\boldsymbol{I}$-simple ideals. Our termination argument is further refined to derive complexity bounds; see Sec. 5.2.
- The *completeness* proof relies on the monotonicity of steps over simple ideals, and shows that all the elements in $\mathit{Cover}(f)$ are covered by some simple ideal in any coverability tree.
- *Soundness* is the most delicate property to establish. Its crux is that neither width nor depth accelerations may take us outside the cover of the initial configuration.

## 5 Complexity Bounds

In the section, we prove lower and upper bounds on the resources needed by the construction of the coverability tree. We refer the reader to [25, 24] for gentle introductions to the techniques employed to prove these results. The enormous complexities involved in our construction require to use *fast-growing complexity* classes [23], which we present succinctly in Sec. 5.1 and in more details in the full paper, before showing hyper-Ackermannian upper and lower bounds in sections 5.2 and 5.3.

### 5.1 Fast-Growing Complexity

In order to express the non-elementary functions required for our complexity statements, we shall employ a family of subrecursive functions $(h^\alpha)_\alpha$ indexed by ordinals $\alpha$ known as the *Hardy hierarchy.*

*Ordinal Terms.* We use ordinal terms $\alpha$ in *Cantor Normal Form* (CNF), which can be written as terms $\alpha = \omega^{\alpha_1} + \cdots + \omega^{\alpha_n}$ where $\alpha_1 \geq \cdots \geq \alpha_n$ are themselves written in CNF. Using such notations, we can express any ordinal below $\varepsilon_0$, the minimal fixpoint of $x = \omega^x$. The ordinal 0 is obtained when $n = 0$; otherwise if $\alpha_n = 0$ the ordinal $\alpha$ is a *successor* ordinal $\omega^{\alpha_1} + \cdots + \omega^{\alpha_{n-1}} + 1$, and if $\alpha_n > 0$ the ordinal $\alpha$ is a *limit* ordinal. We usually write $\lambda$ to denote limit ordinals.

*Fundamental Sequences.* For all $x$ in $\mathbb{N}$ and limit ordinals $\lambda$, we use a standard assignment of fundamental sequences $\lambda(0) < \lambda(1) < \cdots < \lambda(x) < \cdots < \lambda$ with supremum $\lambda$. Fundamental sequences are defined by transfinite induction by:

$$(\gamma + \omega^{\beta+1})(x) \overset{\text{def}}{=} \gamma + \omega^{\beta} \cdot (x+1) \ , \qquad (\gamma + \omega^{\lambda'})(x) \overset{\text{def}}{=} \gamma + \omega^{\lambda'(x)} \ . \tag{7}$$

For instance, $\omega(x) = x + 1$, $\omega^2(x) = \omega \cdot (x + 1)$, $\omega^\omega(x) = \omega^{x+1}$, etc.

*The Hardy Hierarchy.* Let $h\colon \mathbb{N} \to \mathbb{N}$ be a strictly increasing function. The Hardy functions $(h^\alpha\colon \mathbb{N} \to \mathbb{N})_\alpha$ are defined by transfinite induction on their ordinal indices by

$$h^0(x) \overset{\text{def}}{=} x \ , \qquad h^{\alpha+1}(x) \overset{\text{def}}{=} h^\alpha(h(x)) \ , \qquad h^\lambda(x) \overset{\text{def}}{=} h^{\lambda(x)}(x) \ . \tag{8}$$

Observe that $h^k(x)$ for a finite $k$ is simply the $k$th iterate of $h$. For limit ordinals $\lambda$, $h^\lambda(x)$ performs a form of diagonalisation: for instance, setting $H(x) \overset{\text{def}}{=} x + 1$ the successor function, $H^\omega(x) = H^{x+1}(x) = 2x + 1$, $H^{\omega^2}(x) = 2^{x+1}(x + 1) - 1$ is a function of exponential growth, while $H^{\omega^3}$ is a non elementary function akin to a tower of exponentials of height $x$, $H^{\omega^\omega}$ is a non primitive-recursive function with growth similar to the Ackermann function, and $H^{\omega^{\omega^\omega}}$ is a non multiply-recursive function characteristic of hyper-Ackermannian complexity.

*Complexity Classes.* Following [23], we can define complexity classes for computations with time or space resources bounded by Hardy functions of the size of the input. We concentrate in this paper on the HyperAckermann complexity class. Let FMR denote the set of multiply-recursive functions and let $h$ be any multiply-recursive strictly increasing function, then [23, Thm. 4.2]:

$$\textsc{HyperAckermann} \overset{\text{def}}{=} \mathbf{F}_{\omega^\omega} = \bigcup_{m \in \text{FMR}} \textsc{DTime}(h^{\omega^{\omega^\omega}}(m(n))) \tag{9}$$

is the set of decision problems solvable with resources bounded by an hyper-Ackermannian function applied to a multiply-recursive function $m$ of the size of the input. This class is closed under multiply-recursive reductions, and several problems are known to be complete for it (see Sec. 6.2 of [23] for a survey), including coverability in unordered data nets [20].

### 5.2 Upper Bounds

We focus on the worst-case *norm* of the constructed simple ideals, from which bounds on the total size of the coverability tree and the complexity upper bound in Thm. 3.5 can both be derived.

*Norms of Simple Ideals.* For a vector $\boldsymbol{u}$ in $\mathbb{Z}_\omega^k$, its *norm* is its maximal finite absolute value: $\|\boldsymbol{v}\| \overset{\text{def}}{=} \max\{|\boldsymbol{v}[i]| \mid 1 \leq i \leq k \wedge \boldsymbol{v}[i] \neq \omega\}$. Observe that, if $\boldsymbol{I}$ is in $\{0,\omega\}^k$, then $\|\boldsymbol{I}\| = 0$. For an $\boldsymbol{I}$-simple ideal $M$, and thus for finitely supported ones in particular, we define its norm as the maximum between the cardinality of its support and the maximal norm of its vectors: $\|M\| \overset{\text{def}}{=} \max\{|Supp_{\boldsymbol{I}}(M)|, \|M(d)\| \mid d \in \mathbb{D}\}$. Note that the vectors for data $d$ outside the support have all norm 0. In the full paper, we exhibit a bound $B \overset{\text{def}}{=} h^{\omega^{\omega^{k+3}}}(\|f_0\|)$ on the norms of all the simple ideals constructed in a coverability tree rooted by $f_0$ as defined in Def. 4.2, where $h$ also depends on the UDPN:

**Theorem 5.1.** *The norms of the simple ideals in a coverability tree rooted in a configuration $f_0$ for a $k$-dimensional UDPN $\mathcal{T}$ are bounded by $h^{\omega^{\omega^{k+3}}}(\|f_0\|)$, where $h(x)$ is an elementary function of $x$, $k$, and $\|\mathcal{T}\|$.*

The main technical ingredients for Thm. 5.1 are combinatorial statements on the lengths of so-called *controlled bad sequences* proven by Rosa-Velardo [20, App. A] for finite multisets of vectors of natural numbers. Our proofs require however a substantial amount of work on top of that of Rosa-Velardo's for two reasons: we work with extended vectors in $\mathbb{N}_\omega^k$, and use permutation embeddings rather than just plain embeddings.

*Relating Norms with Sizes and Complexity.* The norm $\|M\| \leq B$ of a simple ideal $M$ is directly related to the size of its concrete binary representation: the latter needs at most $\|M\| \cdot k \cdot (\lceil \log \|M\| \rceil + 1)$ bits for the $\boldsymbol{I}$ supported part of the ideal and $k$ bits for the $\boldsymbol{I}$ vector itself. We can also bound the length of the branches in our coverability trees: there are indeed at most $(B+2)^{kB} \cdot 2^k$ different simple ideals with norm $\leq B$, and no two interior nodes on a branch are labelled by the same ideal due to condition 1 in Def. 4.2 (see Rem. 4.4). Finally, by Lem. 3.4, the branching degree of the coverability tree is bounded by an exponential function $(B + \|\mathcal{T}\|)! \cdot |\mathcal{T}|^2$ in $B$ and the size of $\mathcal{T}$. These three observations combined allow to bound the size of the coverability tree:

**Theorem 5.2 (Size of Coverability Trees).** *The size of a coverability tree built from an initial configuration $f_0$ for a $k$-dimensional UDPN $\mathcal{T}$ is bounded by an elementary function of $B$, $k$, and the size of $\mathcal{T}$.*

Theorem 5.2 along with Eq. (9) and the completeness and soundness of coverability trees yields the proof of Thm. 3.5, using the fact that $\mathbf{F}_{\omega^\omega}$ is closed under elementary reductions [23, Thm. 4.7].

## 5.3 Lower Bounds

The sheer complexity bounds we just obtained on the size of coverability trees beg the question whether they are the best possible. We show in Thm. 5.3 that, indeed, the size of coverability trees for a family of UDPNs is provably non multiply-recursive, matching essentially the statement of Thm. 5.2:

**Theorem 5.3 (Hyper-Ackermannian Coverability Trees).** *There exists families of $O(k)$-sized UDPNs $(\mathcal{T}_k)_k$ and $O(k+\log n)$-sized initial configurations $(f_{k,n})_{k,n}$, whose coverability trees are of size at least $H^{\omega^{\omega^k}}(n)$.*

*Hardy Computations.* As detailed in the full paper, we prove Thm. 5.3 by 'implementing' the computation of Hardy functions $H^{\omega^{\omega^k}}$ in nets $\mathcal{T}_k$. The main idea, first developed in [11, 24], is to see the equations in (8) for $0 < \alpha$ as rewriting rules operating on pairs $(\alpha, n)$:

$$(\alpha+1, n) \to (\alpha, n+1)\;, \qquad\qquad (\lambda, n) \to (\lambda(n), n)\;. \tag{10}$$

Note that a sequence $(\alpha_0, n_0) \to (\alpha_1, n_1) \to \cdots \to (\alpha_i, n_i) \to \cdots$ of rewriting steps maintains $H^{\alpha_i}(n_i) = H^{\alpha_0}(n_0)$ for all $i$, and must eventually terminate at some rank $\ell$ with $\alpha_\ell = 0$ since $\alpha_i > \alpha_{i+1}$ for all $i$, and then $n_\ell = H^{\alpha_0}(n_0)$.

Using a natural representation of ordinals $\alpha < \omega^{\omega^k}$ as finite multisets of vectors also employed by Rosa-Velardo [20], a pair $(\alpha, n)$ can be encoded as a configuration of $\mathcal{T}_k$, and the rewriting rules of (10) can be implemented on such codes by steps of $\mathcal{T}_k$. This is however not a perfect implementation: many incorrect computations yielding results different from $H^{\omega^{\omega^k}}(n) = H^{\omega^{\omega^{k-1}\cdot(n+1)}}(n)$ are possible. The crucial point is that there exists a *perfect* computation in $\mathcal{T}_k$, of length at least $H^{\omega^{\omega^k}}(n)$. Furthermore, this computation does not allow any acceleration step, and has therefore to occur as such in any coverability tree.

## 6 Concluding Remarks

In this paper, we have presented a procedure to construct coverability trees for UDPNs in the style of Karp and Miller [12]. This yields decision procedures for coverability and several variants of the boundedness problem including place-boundedness, depth- and width place-boundedness. Besides its interest for the formal verification of UDPNs, this paves the way towards future attempts at proving the decidability of reachability along the lines developed for Petri nets in [18, 13, 14, 16].

We have derived hyper-Ackermannian upper bounds on the complexity of our construction, and shown that such enormous complexities are actually attained on some UDPNs. Note that this however does *not* provide a lower bound

– on the size of *Clover*$(f)$, for which the best known bound is an Ackermannian lower bound adapted from the case of Petri nets [4], nor
– on the complexity of the various boundedness problems on UDPNs, for which the best lower bound is hardness for TOWER $= \mathbf{F}_3$, adapted from the coverability problem [15].

We actually suspect that much lower complexities that HYPERACKERMANN could be obtained for the coverability and boundedness problems. For instance, in the case of Petri nets, coverability trees have a worst-case Ackermannian size [4, 6], but coverability, boundedness, and place-boundedness are all EXPSPACE-complete [17, 19, 2, 5].

## References

1. Abdulla, P.A., Čerāns, K., Jonsson, B., Tsay, Y.K.: Algorithmic analysis of programs with well quasi-ordered domains. Inform. and Comput. 160(1–2), 109–127 (2000)
2. Blockelet, M., Schmitz, S.: Model-checking coverability graphs of vector addition systems. MFCS 2011. LNCS, vol. 6907, pp. 108–119. Springer (2011)
3. Bojańczyk, M., Klin, B., Lasota, S.: Automata theory in nominal sets. Logic. Meth. in Comput. Sci. 10(3:4), 1–44 (2014)
4. Cardoza, E., Lipton, R.J., Meyer, A.R.: Exponential space complete problems for Petri nets and commutative semigroups: Preliminary report. STOC'76. pp. 50–54. ACM (1976)
5. Demri, S.: On selective unboundedness of VASS. J. Comput. Syst. Sci. 79(5), 689–713 (2013)
6. Figueira, D., Figueira, S., Schmitz, S., Schnoebelen, Ph.: Ackermannian and primitive-recursive bounds with Dickson's Lemma. LICS 2011. pp. 269–278. IEEE Press (2011)
7. Finkel, A., Schnoebelen, Ph.: Well-structured transition systems everywhere! Theor. Comput. Sci. 256(1–2), 63–92 (2001)
8. Finkel, A., Goubault-Larrecq, J.: Forward analysis for WSTS, part I: Completions. STACS 2009. LIPIcs, vol. 3, pp. 433–444. LZI (2009)
9. Finkel, A., Goubault-Larrecq, J.: Forward analysis for WSTS, part II: Complete WSTS. Logic. Meth. in Comput. Sci. 8(3:28), 1–35 (2012)
10. Finkel, A., McKenzie, P., Picaronny, C.: A well-structured framework for analysing Petri net extensions. Inform. and Comput. 195(1–2), 1–29 (2004)
11. Haddad, S., Schmitz, S., Schnoebelen, Ph.: The ordinal recursive complexity of timed-arc Petri nets, data nets, and other enriched nets. LICS 2012. pp. 355–364. IEEE Press (2012)
12. Karp, R.M., Miller, R.E.: Parallel program schemata. J. Comput. Syst. Sci. 3(2), 147–195 (1969)
13. Kosaraju, S.R.: Decidability of reachability in vector addition systems. Proc. STOC'82. pp. 267–281. ACM (1982)
14. Lambert, J.L.: A structure to decide reachability in Petri nets. Theor. Comput. Sci. 99(1), 79–104 (1992)
15. Lazić, R., Newcomb, T., Ouaknine, J., Roscoe, A., Worrell, J.: Nets with tokens which carry data. Fund. Inform. 88(3), 251–274 (2008)
16. Leroux, J., Schmitz, S.: Demystifying reachability in vector addition systems. LICS 2015. pp. 56–67. IEEE Press (2015)
17. Lipton, R.: The reachability problem requires exponential space. Tech. Rep. 62, Yale University (1976)
18. Mayr, E.W.: An algorithm for the general Petri net reachability problem. Proc. STOC'81. pp. 238–246. ACM (1981)
19. Rackoff, C.: The covering and boundedness problems for vector addition systems. Theor. Comput. Sci. 6(2), 223–231 (1978)
20. Rosa-Velardo, F.: Ordinal recursive complexity of unordered data nets. Tech. Rep. TR-4-14, Departamento de Sistemas Informáticos y Computación, Universidad Complutense de Madrid (2014), `http://antares.sip.ucm.es/frosa/docs/complexityUDN.pdf`
21. Rosa-Velardo, F., de Frutos-Escrig, D.: Decidability and complexity of Petri nets with unordered data. Theor. Comput. Sci. 412(34), 4439–4451 (2011)

22. Rosa-Velardo, F., Martos-Salgado, M., de Frutos-Escrig, D.: Accelerations for the coverability set of Petri nets with names. Fund. Inform. 113(3–4), 313–341 (2011)
23. Schmitz, S.: Complexity hierarchies beyond Elementary. ACM Trans. Comput. Theory (2016), `http://arxiv.org/abs/1312.5686`, to appear
24. Schmitz, S., Schnoebelen, Ph.: Algorithmic aspects of WQO theory. Lecture notes (2012), `http://cel.archives-ouvertes.fr/cel-00727025`
25. Schmitz, S., Schnoebelen, Ph.: The power of well-structured systems. Concur 2013. LNCS, vol. 8052, pp. 5–24. Springer (2013)
26. Schnoebelen, Ph.: Revisiting Ackermann-hardness for lossy counter machines and reset Petri nets. MFCS 2010. LNCS, vol. 6281, pp. 616–628. Springer (2010)