Bounds on Mobility

Reiner Hüchting[1], Rupak Majumdar[2], and Roland Meyer[1]

[1]University of Kaiserslautern [2]MPI-SWS

Abstract. We study natural semantic fragments of the π-calculus: depth-bounded processes (there is a bound on the longest communication path), breadth-bounded processes (there is a bound on the number of parallel processes sharing a name), and name-bounded processes (there is a bound on the number of shared names). We give a complete characterization of the decidability frontier for checking if a π-calculus process in one subclass belongs to another. Our main construction is a general acceleration scheme for π-calculus processes. Based on this acceleration, we define a Karp and Miller (KM) tree construction for the depth-bounded π-calculus. The KM tree can be used to decide if a depth-bounded process is name-bounded, if a depth-bounded process is breadth-bounded by a constant k, and if a name-bounded process is additionally breadth-bounded. Moreover, we give a procedure that decides whether an arbitrary process is bounded in depth by a given k.

We complement our positive results with undecidability results for the remaining cases. While depth- and name-boundedness are known to be Σ_1-complete, we show that breadth-boundedness is Σ_2-complete, and checking if a process has a breadth bound at most k is Π_1-complete, even when the input process is promised to be breadth-bounded.

1 Introduction

The π-calculus is an expressive formalism for modelling and reasoning about concurrent systems. The full π-calculus is Turing-complete. From a verification perspective, much research has therefore focused on defining semantic fragments which have decidable analysis questions but retain enough expressiveness to capture practical systems. π-calculus processes model communication between components along channels. Natural restrictions on the use of these channels give rise to natural semantic fragments, like bounding the depth of communication (the longest communication chain between processes), bounding the degree of sharing (the number of processes sharing a channel), or bounding the number of channels used concurrently. These restrictions model natural resource constraints in implementations, and indeed have all been studied previously: bounds on depth lead to *depth-bounded* processes [11], bounds on sharing lead to *breadth-bounded* processes [12], and bounds on the number of concurrent channels lead to *name-bounded* processes [9]. While these bounds are defined by induction on the syntax, the restricted classes they define are *semantic*: for example, a process P is breadth-bounded if there is a bound $k \geq 0$ such that in every process reachable from P, every channel is shared by at most k processes. The semantic fragments are still very expressive. For example, name-bounded processes can simulate Petri nets, and depth- or breadth-bounded processes can simulate extensions of Petri nets with reset operations. At the same time, they enable algorithmic verification, e.g. coverability is decidable for depth-bounded processes [18].

Little is known about the relation between semantic fragments, beyond the fact that name-bounded processes are also depth-bounded, and the incomparability of the depth- and breadth-bounded fragments. In particular, the *classification problem* —given a process from one (sub)class, does it also belong to another?— has not been studied. By an analogue of Rice's theorem, checking if a π-calculus process belongs to a semantic fragment is likely to be undecidable. However, the status of natural classification questions, such as whether a given depth-bounded process is actually name-bounded, remains open.

The classification problem has various applications in verification. As stand-alone analysis, classification can judge the resource requirements of a system. For example, to check whether a system is implementable on a given platform, one may check whether it is depth-, breadth-, or name-bounded by a suitable constant k. In turn, a heap-manipulating program that is shown to violate name boundedness may have a memory leak. Within a verification effort, classification serves as a type check that precedes the actual analysis. If the check establishes certain bounds, then the following verification may employ specialized algorithms that make use of this knowledge. For example, to check coverability for general depth-bounded systems, there is only a forward procedure based on iterative refinement. However, if the system is additionally known to be breadth- or name-bounded then more efficient acceleration schemes apply. Similarly, if a bound on the depth is known, then one can use a backwards search.

In this paper, we study and completely characterize the decidability frontier for classification of π-calculus processes into the depth, breadth, and name-bounded semantic fragments, as well as their "k-restricted" versions consisting of all processes where the bound k is given explicitly. Our main construction is a general acceleration scheme for π-calculus processes, and a π-calculus analogue of the Karp-Miller construction for vector addition systems. We characterize the limits of the acceleration for depth-bounded systems and show that —using suitable break conditions— the Karp-Miller construction can decide classification questions such as "given depth-bounded P, is P name-bounded?" and "given depth-bounded P and $k \geq 0$, is P breadth-bounded by k?".

Figure 1 shows a summary of the classification problems, and highlights the results of this paper that complete the picture. The question "given a general process P and $k \geq 0$, is P depth-bounded by k?" was also open (and conjectured to be undecidable). We prove it decidable by showing the existence of a small witness in case the depth bound k is violated. Then we reduce to the coverability problem for depth-bounded processes, which is known to be decidable [18].

We complement our decidability results with undecidability results in the remaining cases. We show that checking if a breadth-bounded process is also depth- or name-bounded is Σ_1-complete, and that checking if a breadth-bounded process has breadth-bound at most a given k is already Π_1-complete. Additionally, while

	$\in$ Breadth	$\in$ Breadth(k)	$\in$ Depth	$\in$ Depth(k)	$\in$ Name	$\in$ Name(k)
Proc	Σ_2	Π_1	(Σ_1)	$\surd$	(Σ_1)	$\surd$ [9]
Breadth		Π_1	(Σ_1)	$(\surd)$	Σ_1	$(\surd)$
Depth	Σ_1	$\surd$		$(\surd)$	$\surd$	$(\surd)$
Name	$\surd$	$(\surd)$		$(\surd)$		$(\surd)$

$\surd$: decidable Σ_i/Π_i : undecidable $(-)$: follows from another result
shaded box : results in this paper empty box : trivial

Fig. 1. Decidability frontier for the classification problem.

checking if a process is depth- or name-bounded is only Σ_1-complete, checking breadth boundedness is actually Σ_2-complete. Our lower bounds follow from simulations of Turing machines by depth- or breadth-bounded processes.

Related Work The Karp-Miller construction, originally developed for Petri nets [10], is generalized to WSTS in [4, 5]. The use of Karp-Miller procedures in model checking is assessed e.g. in [3]. In [6–8], Finkel and Goubault-Larrecq provide a general algorithmic concept that abstracts from the actual tree structure. They devise a class of systems for which this type of algorithm is guaranteed to terminate. One can prove that our domain of limits is the ideal completion of the partially ordered set of processes in the sense of [6]. Wies et al. make a similar observation in [18]. These results make the general acceleration-based algorithms developed in [7, 8] applicable to depth-bounded processes. In contrast to [7, 8], we explicitly construct a Karp-Miller tree in order to decide properties like name and breadth boundedness. Recently, a specific acceleration scheme was developed for name-bounded processes [9]. Our acceleration scheme applies to any process, not restricted to depth-bounded systems and subsumes the results in [9].

The key observation in our acceleration is that repeating transition sequences leads to a cyclic behaviour in the use of restricted names. A similar observation is made for ν-APNs (an extension of Petri nets using names as tokens) in [14], where it is used to establish decidability of so-called width boundedness. The property asks for a bound on the number of names that holds in all reachable ν-APN markings and is related to name boundedness. However, our decision procedure handles the more general depth-bounded systems. The notion of depth applies to other concurrency models with name creation (e.g. ν-MSR in [15]), but we are not aware of any results for the classification problem.

2 The π-calculus

We recall the basics on π-calculus [13, 16], a formalism to encode computation using processes that exchange messages over channels. Messages and channels are represented uniformly by *names* from a countable set $\mathcal{N}$. Processes communicate by synchronising on *prefixes* π that *send* ($\overline{x}\langle y\rangle$) or *receive* ($x(y)$) message y on channel x. Using these communication primitives, we construct processes using choice ($+$), parallel composition ($|$), restriction (νa), and calls ($K\lfloor\tilde{a}\rfloor$). Process identifiers K in calls are associated with defining equations $K(\tilde{x}) := P$ where P is a process and $\tilde{x}$ is a vector of distinct names. Each process depends on finitely many defining equations. Formally, *π-calculus processes* P, Q, R are defined by

$$M ::= \mathbf{0} \;\mid\; \pi.P \;\mid\; M_1 + M_2 \qquad\qquad P ::= M \;\mid\; K\lfloor\tilde{a}\rfloor \;\mid\; P_1 \mid P_2 \;\mid\; \nu a.P \enspace .$$

We write Proc for the set of all processes. Processes M and $K\lfloor\tilde{a}\rfloor$ are called *sequential*, and we use S to indicate that a process is sequential. We denote parallel compositions by products ($\prod$), and use P^k for the k-fold parallel composition of P. Multiple restrictions $\nu a_1 \ldots \nu a_k.P$ are written as a vector $\nu\tilde{a}.P$.

A name a that is neither bound by a restriction νa nor by an input prefix $x(a)$ is called *free*. Names bound by ν are called *restricted*. A restricted name νa is *active* if it is not covered by a prefix. For instance, in the process $\nu a.a(y).\nu b.P\lfloor y\rfloor$, names a and b are restricted, a is active, b is not active, and y is bound but not restricted. We denote the set of free names in P by $fn(P)$ and the active restricted names by $arn(P)$. Since we will be able to α-convert bound names, we assume active restrictions to be unique within a process, and to be disjoint from the free names. By $P\{\tilde{a}/\tilde{x}\}$ we mean the substitution of the free names $\tilde{x}$ in P by $\tilde{a}$. We only apply substitutions that do not clash with the restricted names.

The π-calculus semantics relies on the *structural congruence* relation $\equiv$, the smallest relation that allows for α-conversion of bound names, where $+$ and $|$ are associative and commutative with neutral element $\mathbf{0}$, and that satisfies

$$\nu a.\mathbf{0} \equiv \mathbf{0} \qquad \nu a.\nu b.P \equiv \nu b.\nu a.P \qquad \nu a.(P \mid Q) \equiv P \mid \nu a.Q \quad \text{if } a \notin fn(P) \enspace .$$

The behaviour of processes is given by the *reaction relation*, the smallest relation closed under $|$, ν, and structural congruence, and that satisfies the rules

$$x(z).P + M \mid \overline{x}\langle y\rangle.Q + N \to P\{y/z\} \mid Q \qquad \text{and} \qquad K\lfloor\tilde{a}\rfloor \to P\{\tilde{a}/\tilde{x}\}$$

with $K(\tilde{x}) := P$ a defining equation. Up to $\equiv$, processes have only finitely many successors. The set of all processes reachable from P is the *reachability set* $\mathcal{R}(P)$.

We define the *embedding* ordering $\preceq$ on processes to satisfy $\nu\tilde{a}.P \preceq \nu\tilde{a}.(P \mid Q)$ and to be closed under structural congruence. We use $P{\downarrow} := \{Q \in \mathsf{Proc} \mid Q \preceq P\}$ to denote the downward closure of P wrt. embedding.

Our development relies on two normal forms for processes. A process $\nu\tilde{a}.P$ with $P = S_1 \mid \ldots \mid S_n$ and $\tilde{a} \subseteq fn(P)$ is in *standard form* [13], which can be obtained by maximising the scope of restricted names. Similarly, a process is rewritten to *restricted form* by minimising the scope of restricted names [12]. We write $P \equiv F_1 \mid \ldots \mid F_n$ where each F_i is sequential or of the form $\nu a.P'$ with P' again in restricted form and $a \in fn(F_i)$ for all i. The F_i are called *fragments*.

The restricted form of a process P can be used to determine its *depth*, defined as the minimal nesting depth of restrictions in its congruence class: $|P|_{\mathcal{D}} := min\{nest_\nu(Q) \mid P \equiv Q \text{ in restricted form}\}$ with $nest_\nu$ defined inductively as $nest_\nu(S) := 0$, $nest_\nu(P_1 \mid P_2) := max\{nest_\nu(P_1), nest_\nu(P_2)\}$, and $nest_\nu(\nu a.P) := 1 + nest_\nu(P)$. Similarly, the breadth $|P|_{\mathcal{B}}$ is the maximal number of sequential processes composed in parallel underneath an active restricted name.

We define the following subclasses of the set Proc of π-calculus processes. The class $\mathsf{Depth}(k)$ contains all processes P that are *bounded in depth* by k. Formally, for all $Q \in \mathcal{R}(P)$ we have $|Q|_{\mathcal{D}} \leq k$. Then the class Depth of all depth-bounded processes is the union $\cup_{k \geq 0}\mathsf{Depth}(k)$. The classes $\mathsf{Breadth}(k)$ and $\mathsf{Breadth}$ are defined analogously. Finally, $\mathsf{Name}(k)$ contains all processes P so that every process in $\mathcal{R}(P)$ has at most k active restricted names, and $\mathsf{Name} := \cup_{k \geq 0}\mathsf{Name}(k)$. Clearly, $\mathsf{Name}(k) \subseteq \mathsf{Depth}(k)$ for each $k \geq 0$ and so $\mathsf{Name} \subseteq \mathsf{Depth}$. It is well known that Depth and $\mathsf{Breadth}$ are incomparable.

Our results rely on the fact that depth-, breadth-, and name-bounded processes can be represented in a finite way:

Lemma 1. *Let $P \in \mathsf{Name}(k)$. There is a finite set of sequential processes $\{S_1, \ldots S_n\}$ so that every $Q \in \mathcal{R}(P)$ satisfies*

$$Q \equiv \nu a_1 \ldots a_k.(S_1^{m_i} \mid \ldots \mid S_n^{m_n}) \quad \textit{with} \quad m_1, \ldots, m_n \in \mathbb{N} \ .$$

Similarly, given a depth-bounded process $P \in \mathsf{Depth}$, all reachable $Q \in \mathcal{R}(P)$ can be written using finitely many sequential processes [11]. Indeed, with depth bound k, we rewrite Q to restricted form with at most k nested restrictions. Then we choose a distinguished name a_k for each nesting level:

$$Q \equiv \nu a_1.(\nu a_2.(\ldots) \mid \ldots \mid \nu a_2(\ldots)) \ .$$

After this modification, the sequential processes take the form $D\sigma$ with finitely many D and finitely many substitutions σ. For $\mathsf{Depth} \cap \mathsf{Breadth}$, we obtain a finite representation using fragments [12].

Lemma 2. *Let $P \in \mathsf{Depth} \cap \mathsf{Breadth}$. There is a finite set of fragments $\{F_1, \ldots F_n\}$ so that every $Q \in \mathcal{R}(P)$ satisfies*

$$Q \equiv F_1^{m_i} \mid \ldots \mid F_n^{m_n} \quad \textit{with} \quad m_1, \ldots, m_n \in \mathbb{N} \ .$$

Decision Problems. We study decision problems of the form $(\mathcal{C}_1, \mathcal{C}_2)$ for classes of processes $\mathcal{C}_1$ and $\mathcal{C}_2$, asking for a process from class $\mathcal{C}_1$, is it also in $\mathcal{C}_2$.

3 Karp and Miller for Bounded Depth

We now describe an adaptation of the Karp-Miller algorithm for Petri nets [10] to the π-calculus. The Karp-Miller algorithm determines a finite representation of (the downward closure of) a system's reachability set. It unwinds the state space until it detects a transition sequence between comparable states. The key ingredient is then an acceleration theorem that characterizes the states reachable with arbitrary repetitions of this sequence in a symbolic way. When transferring the idea to depth-bounded processes, the unwinding finds $Q_1 \rightarrow^* Q_2$ with $Q_1 \prec Q_2$. By monotonicity, this can be repeated as

$$Q_1 \rightarrow^* Q_2 \rightarrow^* Q_3 \rightarrow^* \ldots \qquad \text{with} \qquad Q_1 \prec Q_2 \prec Q_3 \prec \ldots \ .$$

Our main result is an acceleration theorem that characterizes $\{Q_i \mid i \in \mathbb{N}\}$. Acceleration for the π-calculus is more involved than for Petri nets, because we have to take the identities of restricted names into account. Consider the reaction

$$\nu a.\nu b.K_1\lfloor a,b\rfloor \to \nu a.\nu b.\,(K_1\lfloor a,b\rfloor \mid K_2\lfloor b\rfloor)$$

with $K_1(x,y) := \nu z.(K_1\lfloor z,x\rfloor \mid K_2\lfloor x\rfloor)$ and $K_2(x) := \mathbf{0}$. At first glance, k repetitions of this reaction should lead to processes $\nu a.\nu b.\,\big(K_1\lfloor a,b\rfloor \mid (K_2\lfloor b\rfloor)^k\big)$. However, due to α-conversion, a and b in source and target process are different. Since a has been renamed to b and a fresh a has been created, repetitions will lead to terms $\nu c.K_2\lfloor c\rfloor$. To see this, we repeat the reaction without renaming:

$$\begin{aligned}\nu a.\nu b.K_1\lfloor a,b\rfloor &\to \nu c.\nu a.(K_1\lfloor c,a\rfloor \mid K_2\lfloor a\rfloor)\\ &\to \nu d.\nu c.(K_1\lfloor d,c\rfloor \mid K_2\lfloor c\rfloor \mid \nu a.K_2\lfloor a\rfloor)\ .\end{aligned}$$

The example illustrates that we have to track the identity of restricted names over transitions. We have to determine if a restricted name is *stable* in the sense that it remains in the original process, or it is *fragile*, i.e. it is eventually forgotten and moves to the accelerated part. The following subsection develops a suitable notion of identity relations, afterwards we turn to the actual acceleration.

3.1 Identity Relations

To track the identity of active restricted names over transitions, we extend the reaction relation. Recall that we assume active restrictions to be unique within processes. With each reaction $P \to Q$, we associate an *identity relation* $I \subseteq (\mathcal{N} \cup \{\star\}) \times \mathcal{N}$ relating the active restrictions in P and Q. Formally, I is the smallest set that satisfies the following conditions. If reaction $P \to Q$ α-converts $a \in arn(P)$ into $b \in arn(Q)$, then we have $(a,b) \in I$. Moreover, if νc becomes active in Q, we have $(\star, c) \in I$. We write $P \to_I Q$ for these *extended reactions* in $\mathsf{Proc} \times \mathbb{P}((\mathcal{N} \cup \{\star\}) \times \mathcal{N}) \times \mathsf{Proc}$. The definition generalizes to sequences of extended reactions by composing the identity relations. Such a sequence is *faithful* if it does not use α-conversion. Formally, $I \cap (\mathcal{N} \times \mathcal{N})$ is the identity.

For the purpose of acceleration, we focus on the special case that I acts on a finite subset $\tilde{a} \subseteq \mathcal{N}$ of names, $I \subseteq (\{\star\} \cup \tilde{a}) \times \tilde{a}$. In this case, I induces a partition $\tilde{a} = \tilde{f} \uplus \tilde{s}$ as follows. The set $\tilde{f}$ of *fragile names* contains all names that are recreated in repeated applications of I. Technically, these names form a least fixed point. If $(\star, a) \in I$, then $a \in \tilde{f}$. If $a \in \tilde{f}$ and $(a,b) \in I$, then $b \in \tilde{f}$. The remaining names $\tilde{s} := \tilde{a} \setminus \tilde{f}$ are called *stable.* We use σ_I for $I \cap (\tilde{s} \times \tilde{s})$.

Lemma 3. *$\sigma_I : \tilde{s} \to \tilde{s}$ is a bijection.*

Since σ_I is a bijection, repeated applications of σ_I become periodic. The *period* of σ_I is the smallest $p \geq 1$ so that $\sigma_I^p = id$. Now $\sigma_I^x = \sigma_I^{x \bmod p}$ for all $x \in \mathbb{N}$.

We call an identity relation I *forgetful* if it immediately recreates all fragile names, as opposed to needing several applications to recreate them. Formally, I is forgetful if the fragile names are precisely the names $(\star, a)$ in I. We also call an extended reaction sequence $P \rightarrow_I^* Q$ forgetful if I is. To give an example, consider $\tilde{a} = a.b.c.d$ and $I = \{(\star, a), (a, b), (d, c), (c, d)\}$. Then $\tilde{f} = a.b$, $\tilde{s} = c.d$, $\sigma_I(c) = d$, and $\sigma_I(d) = c$. The period of σ_I is $p = 2$. Relation I is not forgetful, but $I^2 = \{(\star, a), (\star, b), (c, c), (d, d)\}$ is.

3.2 Acceleration for π-calculus

We make precise what it means to repeat a reaction sequence. Embedding $P \preceq Q_1$ ensures $P \equiv \nu\tilde{a}.P'$ and $Q_1 \equiv \nu\tilde{a}.(P' \mid Q')$. Hence, Q_1 contains all the sequential processes of P. So if $P \rightarrow^* Q_1$ then also $Q_1 \rightarrow^* Q_2$ with reactions between the same sequential processes. We use the syntax $P \xrightarrow{\rho} Q_1$ and $Q_1 \xrightarrow{\rho} Q_2$ to indicate that the sequences rely on the same reactions ρ.

Our main result characterizes the shape of Q_k with $P \xrightarrow{\rho^k} Q_k$. The idea is to compose additional copies of Q' in parallel with the repeating process P'. These copies keep track of (i) the fragile names forgotten by P' and (ii) repeated applications of σ_I to the stable names. We state the precise result for forgetful sequences. The general case is more involved, but does not add ideas.

Theorem 1 (Acceleration). *Let $\nu\tilde{s}.\nu\tilde{f}.P \xrightarrow{\rho'}_{I'} \nu\tilde{s}'.\nu\tilde{f}'.(P' \mid \nu\tilde{f}.Q)$ be a faithful sequence that, using α-conversion, gives rise to the forgetful sequence*

$$\nu\tilde{s}.\nu\tilde{f}.P \xrightarrow{\rho}_I \nu\tilde{s}.\nu\tilde{f}.(P \mid \nu\tilde{f}_1.Q\{\tilde{f}_1/\tilde{f}\}\{\tilde{f}/\tilde{f}'\}\sigma_I) \ .$$

Then for all $k \in \mathbb{N}$ we have

$$\nu\tilde{s}.\nu\tilde{f}.P \xrightarrow{\rho^k} \nu\tilde{s}.\nu\tilde{f}.(P \mid \nu\tilde{f}_k.(Q_k \mid \ldots \nu\tilde{f}_2.(Q_2 \mid \nu\tilde{f}_1.Q_1)\ldots))$$

where $Q_i := Q\{\tilde{f}_i/\tilde{f}\}\{\tilde{f}_{i+1}/\tilde{f}'\}\sigma_I^{k-i+1}$ and $\tilde{f}_{k+1} := \tilde{f}$.

Proof. We use an induction on the number of repetitions of ρ. In the base case, the term $\nu\tilde{f}_k.(\ldots)$ is missing. Now assume ρ^k behaves as required. We can repeat ρ once more since embedding is a simulation [11]. We use a faithful repetition that avoids α-conversion and instantiates restricted names to fresh names:

$$\begin{aligned}\nu\tilde{s}.\nu\tilde{f}.P \xrightarrow{\rho^k}\ & \nu\tilde{s}.\nu\tilde{f}.(P \mid \nu\tilde{f}_k.(Q_k \mid \ldots \nu\tilde{f}_2.(Q_2 \mid \nu\tilde{f}_1.Q_1)\ldots)) \\ \xrightarrow{\rho'}_{I'}\ & \nu\tilde{s}'.\nu\tilde{f}.(\nu\tilde{f}'.(P' \mid Q) \mid \nu\tilde{f}_k.(Q_k \mid \ldots \nu\tilde{f}_2.(Q_2 \mid \nu\tilde{f}_1.Q_1)\ldots)) \ .\end{aligned}$$

Since the names $\tilde{f}'$ are fresh and do not occur free in $\nu\tilde{f}_k.(\ldots)$, we can extend the scope of $\nu\tilde{f}'$. Process P forgets the names $\tilde{f}$ in the reaction sequence ρ. After swapping $\nu\tilde{f}$ and $\nu\tilde{f}'$, we can therefore restrict the scope of $\nu\tilde{f}$:

$$\begin{aligned}& \nu\tilde{s}'.\nu\tilde{f}.[\nu\tilde{f}'.(P' \mid Q) \mid \nu\tilde{f}_k.(Q_k \mid \ldots \nu\tilde{f}_2.(Q_2 \mid \nu\tilde{f}_1.Q_1)\ldots)] \\ \equiv\ & \nu\tilde{s}'.\nu\tilde{f}.\nu\tilde{f}'.(P' \mid Q \mid \nu\tilde{f}_k.(Q_k \mid \ldots \nu\tilde{f}_2.(Q_2 \mid \nu\tilde{f}_1.Q_1)\ldots)) \\ \equiv\ & \nu\tilde{s}'.\nu\tilde{f}'.\nu\tilde{f}.(P' \mid Q \mid \nu\tilde{f}_k.(Q_k \mid \ldots \nu\tilde{f}_2.(Q_2 \mid \nu\tilde{f}_1.Q_1)\ldots)) \\ \equiv\ & \nu\tilde{s}'.\nu\tilde{f}'.(P' \mid \nu\tilde{f}.[Q \mid \nu\tilde{f}_k.(Q_k \mid \ldots \nu\tilde{f}_2.(Q_2 \mid \nu\tilde{f}_1.Q_1)\ldots)]) \ .\end{aligned}$$

We rename $\tilde{f}$ to fresh names $\tilde{f}_{k+1}$ and afterwards $\tilde{f}'$ to $\tilde{f}$. The last step is to rename $\tilde{s}'$ to $\tilde{s}$ using σ_I. The resulting processes have the required form. $\square$

Depth boundedness of $\nu\tilde{s}.\nu\tilde{f}.P$ allows us to draw further conclusions about the shape of the processes resulting from acceleration. We now characterize this shape. We start with the assumption that $\nu\tilde{s}.\nu\tilde{f}.P$ is even name-bounded. This case explains well the finiteness obtained from periodicity of σ_I. Moreover, it illustrates the shape of limit processes in the decision procedure for name boundedness of depth-bounded processes. Under the assumption of name boundedness, the accelerated processes Q_i cannot contain the fragile names $\tilde{f}_i$ and $\tilde{f}_{i+1}$. As a consequence, these processes can be written as $Q\sigma^i$.

Lemma 4. *Under the conditions in Theorem 1 and the additional assumption that $\nu\tilde{s}.\nu\tilde{f}.P$ is name-bounded, acceleration yields*

$$\nu\tilde{s}.\nu\tilde{f}.P \xrightarrow{\rho^k} \nu\tilde{s}.(\nu\tilde{f}.P \mid \prod_{i=1}^{p}(Q\sigma_I^i)^{\lfloor \frac{k}{p} \rfloor + z_i}) \ .$$

Here, p is the period of σ_I and $z_i = 1$ if $i \leq k$ mod p and $z_i = 0$ otherwise.

Proof. We skip the substitutions $\{\tilde{f}_i/\tilde{f}\}\{\tilde{f}_{i+1}/\tilde{f}'\}$ and remove the corresponding restrictions from Q_i. This leaves us with an i-fold application of σ_I:

$$\nu\tilde{s}.\nu\tilde{f}.(P \mid \nu\tilde{f}_k.(Q_k \mid \ \ldots\nu\tilde{f}_2.(Q_2 \mid \nu\tilde{f}_1.Q_1)\ldots)) \equiv \ \nu\tilde{s}.(\nu\tilde{f}.P \mid \prod_{i=1}^{k} Q\sigma_I^i) \ .$$

The claim then follows from $\sigma^x = \sigma^{x \ mod \ p}$ for all $x \in \mathbb{N}$ and the fact that $k = \lfloor \frac{k}{p} \rfloor p + r$ where $r = k \ mod \ p$. $\square$

In case $\nu\tilde{s}.\nu\tilde{f}.P$ is depth-bounded, the processes Q_i in

$$\nu\tilde{s}.\nu\tilde{f}.(P \mid \nu\tilde{f}_k.(Q_k \mid \ \ldots\nu\tilde{f}_2.(Q_2 \mid \nu\tilde{f}_1.Q_1)\ldots))$$

cannot connect all fragile names $\tilde{f}_{k+1}$ to $\tilde{f}_1$. Instead, we now argue that process $\nu\tilde{f}_k.(\ldots)$ falls apart into a composition of fragments that stem from a finite set.

We first note that Q induces a relation $D \subseteq \tilde{f} \times \tilde{f}$ among the fragile names. Intuitively, $(f_1, f_2) \in D$ indicates that name f_2 from some execution of ρ is connected with name f_1 from the next execution via a fragment in Q. For the formal definition, recall that $Q_i = Q\{\tilde{f}_i/\tilde{f}\}\{\tilde{f}_{i+1}/\tilde{f}'\}\sigma_I^{k-i+1}$. Substitution σ_I is a bijection among the stable names. When studying the relation among the fragile names, we can drop it. Moreover, $\{\tilde{f}_i/\tilde{f}\}$ and $\{\tilde{f}_{i+1}/\tilde{f}'\}$ are bijective renamings of $\tilde{f}$ that we also avoid. To define $D \subseteq \tilde{f} \times \tilde{f}$, we turn the remaining process Q into restricted form. The restricted form is a parallel composition $G_1 \mid \ \ldots \ \mid G_n$ of fragments. Relation $D \subseteq \tilde{f} \times \tilde{f}$ is defined to contain the pair (f_1, f_2) if there is a fragment G_i that has $f_1' \in \tilde{f}'$ and $f_2 \in \tilde{f}$ as free names. To give an example, consider $Q = G_1 \mid G_2$ where $G_1 = \nu a.(K_1\lfloor f_1', a\rfloor \mid K_2\lfloor a, f_2\rfloor)$ and $G_2 = K_3\lfloor f_2', f_3\rfloor$. Then we have $D = \{(f_1, f_2), (f_2, f_3)\}$.

If the process of interest is depth-bounded, relation D is guaranteed to vanish. Otherwise, the connected fragments G_i would witness unbounded depth.

Lemma 5. *If $\nu\tilde{s}.\nu\tilde{f}.P \in \mathsf{Depth}$, then there is $b \leq |\tilde{f}|$ so that $D^b = \emptyset$.*

The lemma shows that fragments in $Q_{i+(b-2)}$ may only share fragile names with fragments in $Q_{i+(b-3)}$, which only share names with fragments in $Q_{i+(b-4)}$ up to Q_i. Combined with periodicity of σ_I, it follows that process $\nu\tilde{f_k}.(\ldots)$ falls apart into fragments from a finite set $\{F_1, \ldots, F_t\}$. In the example, we have $b = 3$ as $D^3 = \emptyset$. So fragments in Q_{i+1} may only share fragile names with fragments in Q_i, but these fragments will not share fragile names with fragments in Q_{i-1}. Hence, we may observe $\nu f_{1,i+2}.f_{2,i+1}.f_{3,i}.(G_{1,i+1} \mid G_{2,i})$ but $G_{2,i} = K_3\lfloor f_{2,i+1}, f_{3,i}\rfloor$ will not share $f_{3,i}$ with another fragment.

Lemma 6. *If $\nu\tilde{s}.\nu\tilde{f}.P$ is depth-bounded, then acceleration yields*

$$\nu\tilde{s}.\nu\tilde{f}.P \xrightarrow{\rho^k} \nu\tilde{s}.(\nu\tilde{f}.(P \mid R_k) \mid \prod_{i=1}^{t} F_i^{n_i})$$

where the R_k are periodic, $t \in \mathbb{N}$ is the number of fragments, and $n_1, \ldots n_t \in \mathbb{N}$ grow unboundedly with k.

To actually compute the fragments $F_1, \ldots, F_t$, we iterate ρ until the fragments repeat, due to periodicity of σ_I and modulo α-conversion of fragile names. The time it takes until repetition depends on the period of σ_I and $|\tilde{f}|$. However, termination does not rely on the precise depth bound.

In the decision procedure for restricted breadth boundedness, we apply Lemma 6 in a setting where also the breadth is bounded. In this case, the stable names are guaranteed not to occur in the accelerated fragments. We derive

$$\nu\tilde{s}.\nu\tilde{f}.P \xrightarrow{\rho^k} \nu\tilde{f}.(\nu\tilde{s}.P \mid R_k) \mid \prod_{i=1}^{t} F_i^{n_i} \quad .$$

3.3 Karp and Miller Trees for Bounded Depth

We now apply these insights about acceleration to devise a Karp and Miller algorithm for depth-bounded processes. Our goal is to separate soundness and completeness of the construction from termination. We will guarantee that the tree represents precisely the downward closure of the reachability set, but the tree computation may not terminate. Termination is studied independently in the next section where we draw conclusions about decidability. In Section 5, we show that in general the coverability set is not computable for depth-bounded systems. Hence, no Karp-Miller algorithm for depth-bounded processes can be both, sound and complete as well as terminating.

To represent the downward closure of the state space, we use *limit processes*:

$$L ::= S \;\;\shortmid\;\; L_1 \mid L_2 \;\;\shortmid\;\; \nu a.L \;\;\shortmid\;\; L^\omega \quad .$$

Structural congruence is extended to limits by

$$\mathbf{0}^\omega \equiv \mathbf{0} \quad L^\omega \equiv L \mid L^\omega \quad (L^\omega)^\omega \equiv L^\omega \quad (L_1 \mid L_2)^\omega \equiv L_1^\omega \mid L_2^\omega \quad L^\omega \mid L^\omega \equiv L^\omega \quad .$$

Algorithm 1 Generic Karp & Miller Tree Construction

```
1:  $V := \{\mathsf{root} : P\}$;     $\rightsquigarrow := \emptyset$;     $work := \mathsf{root} : P$;
2:  while $work$ not empty do
3:      pop $\mathsf{n}_1 : L_1$ from $work$;
4:      Break condition;
5:      for all $L_1 \to_I L_2$ up to $\equiv$ do
6:          if there is $\mathsf{n} : L_p \rightsquigarrow^*_{I_1} \mathsf{n}_1 : L_1$ since last acceleration so that
                    $L_p \equiv \nu\tilde{s}.\tilde{f}.L$ and $L_2 \equiv \nu\tilde{s}.\tilde{f}.(L \mid Q)$ then
7:              $L_2 := \mathbf{Accelerate}(L_p, L_2)$;
8:          let $\mathsf{n}_2$ fresh;   $V := V \cup \{\mathsf{n}_2 : L_2\}$;   $\rightsquigarrow := \rightsquigarrow \cup \{\mathsf{n}_1 : L_1 \rightsquigarrow_I \mathsf{n}_2 : L_2\}$;
9:          $work := work \cdot (\mathsf{n}_2 : L_2)$ provided $L_2$ does not occur on the path;
10: return $(V, \rightsquigarrow, \mathsf{root} : P)$;
```

With this, the reaction relation and the embedding order carry over to limit processes. To generalize the extended reaction relation to limits, we do not consider a restricted name as active if it is covered by an ω. If a reaction takes a restriction $\nu b.L\{b/a\}$ out of a term $(\nu a.L)^\omega$, we add $(\star, b)$ to the identity relation.

Our Karp-Miller construction is stated as Algorithm 1. Understand **break condition** as a no-op for the moment. The command will be instantiated in Section 4. The algorithm constructs a tree $\mathrm{KM}(P) := (V, \rightsquigarrow, \mathsf{root} : P)$ where the nodes are labelled by limit processes, starting from the root that is labelled by the given process P. To construct the tree, the algorithm maintains a worklist of nodes that have not yet been processed. As long as the worklist is not empty, the algorithm pops nodes $\mathsf{n}_1 : L_1$ and computes all successors $L_1 \to_I L_2$. In contrast to a standard state space computation, the algorithm does not immediately add L_2 to the tree, but it checks for a predecessor L_p that is strictly smaller in the following sense: $L_p \equiv \nu\tilde{s}.\tilde{f}.L$ and $L_2 \equiv \nu\tilde{s}.\tilde{f}.(L \mid Q)$. If such an L_p exists, the algorithm accelerates subterms in L_2 to ω and adds the result to the tree.

We use an acceleration scheme that is flat in the following sense. We only look for predecessors L_p starting from the last accelerated process on the path from the root. This ensures that the Karp-Miller transitions between L_p and L_2 are actually reactions. As a consequence, the additional process Q in L_2 can be assumed to be concrete because the reaction relation does not introduce ω.

To define the acceleration, assume $\nu\tilde{s}.\tilde{f}.L \xrightarrow{\rho}_I \nu\tilde{s}.\tilde{f}.(L \mid Q)$. We observe that Theorem 1 carries over to limits. As the input process is depth-bounded, Lemma 6 shows that $k \in \mathbb{N}$ repetitions of the reaction sequence yield

$$\nu\tilde{s}.\nu\tilde{f}.L \xrightarrow{\rho^k} \nu\tilde{s}.(\nu\tilde{f}.(L \mid R_k) \mid \prod_{i=1}^{t} F_i^{n_i})$$

where the R_k are periodic and $n_1, \ldots, n_t \in \mathbb{N}$ grow with k. We set

$$\mathbf{Accelerate}(\nu\tilde{s}.\tilde{f}.L, \nu\tilde{s}.\tilde{f}.(L \mid Q)) := \nu\tilde{s}.(\nu\tilde{f}.(L \mid R_1) \mid \prod_{i=1}^{t} F_i^{\omega}) \ .$$

The following theorem states that the limit processes in $\mathrm{KM}(P)$ are a sound and complete representation of the state space.

Theorem 2. *Let $P \in \mathsf{Depth}$. Then $\mathcal{R}(P){\downarrow} = \mathrm{KM}(P){\downarrow} \cap \mathsf{Proc}$.*

Completeness means every $Q \in \mathcal{R}(P)$ satisfies $Q \in \mathrm{KM}(P){\downarrow}$. This holds because acceleration only returns a larger process and embedding is a simulation. Soundness states that every process $Q \preceq L \in \mathrm{KM}(P)$ is covered by a reachable process, $Q \preceq R \in \mathcal{R}(P)$. The acceleration results show that for every limit $L \in \mathrm{KM}(P)$ and $k \in \mathbb{N}$ there is a process $R_{L,k} \in \mathcal{R}(P)$ that coincides with L in the concrete part and yields more than k parallel compositions of the ω terms.

4 Decidability Results

4.1 Given $P \in \mathsf{Depth}$, is $P \in \mathsf{Name}$?

To decide name boundedness, we instantiate **break condition** as follows:

```
if there is a predecessor $\mathsf{n} : L_p \rightsquigarrow^* \mathsf{n_1} : L_1$ so that $\mathsf{n} : L_p$ is not inside
            an acceleration, $L_p \prec L_1$, and $|arn(L_p)| < |arn(L_1)|$ then
    return not name-bounded;
```

That predecessor $\mathsf{n} : L_p$ is not inside an acceleration, means there is no pair of nodes $\mathsf{n_x} : L_x \rightsquigarrow^+ \mathsf{n} : L_p \rightsquigarrow^+ \mathsf{n_y} : L_y$ that have been used to accelerate L_y. The condition guarantees that the sequence $\mathsf{n} : L_p \rightsquigarrow^* \mathsf{n_1} : L_1$ can be accelerated although it may be nested. The trick is that nested sequences inside have all the processes required to pump.

We say that Algorithm 1 *succeeds*, if it returns the tree (Line 10) and it *breaks*, if it enters the break condition (Line 4). The next observation shows that acceleration never introduces ω over an active restriction.

Lemma 7. *If $\mathrm{KM}(P)$ accelerates an active restriction, it breaks.*

Proof. Let L_1 be the first limit on a path with some $(\nu a.(\ldots))^\omega$. The acceleration is due to some $\mathsf{n} : L_p \rightsquigarrow^* \mathsf{n_1} : L_1$ that occurs after the last acceleration and satisfies $L_p \prec L_1$. As L_1 is minimal, $|arn(L_p)| \in \mathbb{N}$ and $|arn(L_1)| := \omega$. □

We now show Algorithm 1 is partially correct. If $\mathrm{KM}(P)$ succeeds, it will not accelerate active restrictions by Lemma 7. Moreover, the constructed tree is finite and the maximum $max\{|arn(L)| \mid L \in \mathrm{KM}(P)\}$ is a natural number and an optimal name bound. If $\mathrm{KM}(P)$ breaks, we obtain name unboundedness with the above argumentation on nested acceleration and Lemma 6.

Lemma 8. *If $\mathrm{KM}(P)$ succeeds, $P \in \mathsf{Name}$. If $\mathrm{KM}(P)$ breaks, $P \notin \mathsf{Name}$.*

It remains to show that Algorithm 1 is guaranteed to terminate. First assume P is name-bounded. By Lemma 8, the algorithm does not break. To show that the tree construction succeeds, we apply König's lemma and argue that the paths in the tree are finite. Towards a contradiction, assume there was an infinite path

$$\mathsf{root} : P = \mathsf{n}_0 : L_0 \rightsquigarrow \mathsf{n}_1 : L_1 \rightsquigarrow \mathsf{n}_2 : L_2 \rightsquigarrow \ldots$$

Since the Karp-Miller construction is sound (Theorem 2) and $P \in \mathsf{Name}$, not only the reachable processes but also the limits L_i take the shape from Lemma 1:

$$L_i \equiv \nu a_1 \ldots a_k.(S_1^{m_i} \mid \ldots \mid S_n^{m_n})$$

where $n \in \mathbb{N}$ is fixed and potentially some m_i are accelerated to ω. We can understand the limits as vectors in $(\mathbb{N} \cup \{\omega\})^n$. The set of vectors $(\mathbb{N} \cup \{\omega\})^n$ with the usual ordering on $\mathbb{N} \cup \{\omega\}$ (applied component-wise) is well-quasi-ordered. Hence, the infinite path contains an infinite non-decreasing subsequence

$$L_{i_1} \preceq L_{i_2} \preceq \ldots \ .$$

This sequence is strict, as Algorithm 1 never adds a process to the worklist that already occurs on the path. Since there are only n sequential processes and ωs are never removed, the number of S^ω stabilizes in some node in the path, say $\mathsf{n}_{\mathsf{i_k}} : L_{i_k}$. Then there are no more accelerations from L_{i_k}, and $L_{i_{k+1}} \succ L_{i_k}$ can be accelerated. A contradiction to the fact that no ω is added to $L_{i_{k+1}}$.

Lemma 9. *If* $P \in \mathsf{Name}$*, then* $\mathrm{KM}(P)$ *succeeds.*

Now assume that $P \notin \mathsf{Name}$. By Lemma 8, the algorithm does not succeed. To show that it always breaks, assume this is not the case. With Lemma 7, we never accelerate an active restriction. Since Algorithm 1 is complete and $P \notin \mathsf{Name}$, for every $k \in \mathbb{N}$ the tree contains a limit L_k with more than k active restrictions. Thus, the tree is infinite and by König's lemma contains an infinite path

$$\mathsf{root} : P = \mathsf{n}_0 : L_0 \rightsquigarrow \mathsf{n}_1 : L_1 \rightsquigarrow \mathsf{n}_2 : L_2 \rightsquigarrow \ldots \ .$$

We isolate the subsequence $L_{i_1}, L_{i_2}, \ldots$ of limits that (i) form the lhs of an acceleration, like $\mathsf{n}_\mathsf{x} : L_x$ above, or (ii) that lie strictly between accelerations.

We now show that the limits computed without a break form a wqo with $\preceq$. Since we never accelerate a restriction, the limits L_i are term trees over sequential processes S and S^ω. Due to soundness, the limits are bounded in depth by some $k \in \mathbb{N}$. It remains to show that S stems from a finite set. With the depth bound and the observation following Lemma 1, we can assume that the limits are flat processes with at most k nested restrictions. They can be written as

$$L_i \equiv \nu a_1.(\nu a_2.(\ldots) \mid \ldots \mid \nu a_2(\ldots)) \ .$$

This in turn means $S \equiv D\sigma$ with $\sigma : \mathit{fn}(D) \to \mathit{fn}(P) \cup \{a_1, \ldots, a_k\}$, where D is a syntactic subterm of P and σ is taken from a finite set of substitutions.

As the limits on the path are wqo, there is an infinite non-decreasing subsequence $L_{j_1} \preceq L_{j_2} \preceq \ldots$ of the sequence $L_{i_1}, L_{i_2}, \ldots$ we just considered. Again, Algorithm 1 does not add copies and the sequence is strict. Either the number of active restricted names in the limits along this path is bounded, or it grows indefinitely. If it is bounded, this contradicts infinity of the path as in Lemma 9. Otherwise, there are $L_{j_k} \prec L_{j_l}$ with $|arn(L_{j_k})| < |arn(L_{j_l})|$ and the break condition applies.

Lemma 10. *If $P \notin \mathsf{Name}$, then $\mathrm{KM}(P)$ breaks.*

Theorem 3. *Given $P \in \mathsf{Depth}$, it is decidable if $P \in \mathsf{Name}$.*

Name-bounded processes have finite Karp-Miller trees by Lemma 9. We can now decide breadth boundedness for $P \in \mathsf{Name}$ by checking if the tree contains a limit $\nu\tilde{a}.(S_1^{m_1} \mid \ldots \mid S_n^{m_n})$ where $a \in fn(S_i)$ for some $a \in \tilde{a}$ and S_i with $m_i = \omega$.

Corollary 1. *Given $P \in \mathsf{Name}$, it is decidable if $P \in \mathsf{Breadth}$.*

4.2 Given $P \in$ Depth and k, is $P \in$ Breadth(k)?

To derive a decision procedure, we use the following **break condition**:

if $|L_1|_{\mathcal{B}} > k$ **then return** not breadth-bounded by k;

Lemma 11. *If $\mathrm{KM}(P)$ succeeds, $P \in \mathsf{Breadth}(k)$. If $\mathrm{KM}(P)$ breaks, $P \notin \mathsf{Breadth}(k)$.*

We now show that Algorithm 1 is guaranteed to terminate. If $P \in \mathsf{Breadth}(k)$, by Lemma 11, the break condition will not apply. By soundness of the construction, limits are composed of finitely many fragments (Lemma 2). Then an infinite path yields a contradiction to the acceleration behaviour as for name boundedness. If $P \notin \mathsf{Breadth}(k)$, by Lemma 11, the tree construction will not succeed. We show that we enter the break condition. Since $P \notin \mathsf{Breadth}(k)$, there is a process $Q \in \mathcal{R}(P)$ with $|Q|_{\mathcal{B}} > k$. As Algorithm 1 is complete, we eventually find a limit L with $Q \preceq L$ and $k < |Q|_{\mathcal{B}} \leq |L|_{\mathcal{B}}$. The break condition applies for L.

Lemma 12. *If $P \in \mathsf{Breadth}(k)$, then $\mathrm{KM}(P)$ succeeds. If $P \notin \mathsf{Breadth}(k)$, then $\mathrm{KM}(P)$ breaks.*

Theorem 4. *Given $P \in \mathsf{Depth}$ and $k \in \mathbb{N}$, it is decidable if $P \in \mathsf{Breadth}(k)$.*

Note that we cannot semi-decide breadth unboundedness. We show in Section 5 that breadth boundedness, despite the seeming similarity with name boundedness, is undecidable for depth-bounded processes.

4.3 Given $P \in$ Proc and k, is $P \in$ Depth(k)?

By definition, $P \in \mathsf{Depth}(k)$ iff all processes in $\mathcal{R}(P)\!\downarrow$ are in $\mathsf{Depth}(k)$.

If $\mathcal{R}(P)\!\downarrow$ is not in $\mathsf{Depth}(k)$, we claim that there is some process Q in $\mathcal{R}(P)\!\downarrow$ for which $k < |Q|_{\mathcal{D}} \leq |P| + 2k$. Let $P \to^* Q' \to Q$, such that (1) $|Q|_{\mathcal{D}} > k$, and (2) all processes in the path $P \to^* Q'$ have depth less than or equal to k. So Q is a minimal process whose depth exceeds k. Note that Q need not be unique, we arbitrarily pick a minimal process. Consider the step $Q' \to Q$. By case analysis on the possible steps, Q' either unfolds a recursive call and Q has its depth bounded by $|P| + |Q'|_{\mathcal{D}}$, or Q results from a communication and has fragments whose depth is bounded above by $|P| + 2|Q'|_{\mathcal{D}}$. In both cases, $|P|$ takes care of active restrictions that are freed in the reaction. As $|Q'|_{\mathcal{D}} \leq k$ by assumption, we have that there is a process Q in $\mathcal{R}(P)\!\downarrow$ whose depth is at most $|P| + 2k$.

Now, consider the set $\mathsf{Proc}_w := \{Q \in \mathsf{Proc} \mid k < |Q|_{\mathcal{D}} \leq |P| + 2k\}$. We work in the well-quasi-order of $|P| + 2k$ depth-bounded processes with the ordering $\preceq$. If $\mathcal{R}(P)\!\downarrow$ intersects this set, a minimal element of this set must be covered by process P. We enumerate the finitely many minimal elements of this set, and for each minimal element Q, we check if $\mathcal{R}(P)\!\downarrow$ intersects Q. This problem is a coverability question. Since the coverability problem is decidable for depth-bounded systems [18], we get a decision procedure to check if $P \in \mathsf{Depth}(k)$.

Theorem 5. *Given $P \in \mathsf{Proc}$ and $k \in \mathbb{N}$, it is decidable if $P \in \mathsf{Depth}(k)$.*

5 Undecidability Results

We complement our decidability results with undecidability results for the remaining questions. Recall that a problem L is in the class Σ_1 if it is semi-decidable, and in Σ_2 if it is semi-decidable using a Σ_1 oracle. A problem is Π_1 if it is the complement of a Σ_1 problem, and L is $\mathcal{C}$-complete if it is in the class $\mathcal{C}$ and there is a recursive reduction from every $L' \in \mathcal{C}$ to L. The halting problem is Σ_1-complete, the emptiness problem is Π_1-complete, and the finiteness problem is Σ_2-complete [17]. Boundedness for reset Petri nets is also Σ_1-complete [1, 2].

Theorem 6 (Undecidability results).

1. *Given $P \in \mathsf{Proc}$, checking if $P \in \mathsf{Name}$ or $P \in \mathsf{Depth}$ are both Σ_1-complete.*
2. *Given $P \in \mathsf{Proc}$ and $k \in \mathbb{N}$, checking if $P \in \mathsf{Breadth}(k)$ is Π_1-complete.*
3. *Given $P \in \mathsf{Proc}$, checking if $P \in \mathsf{Breadth}$ is Σ_2-complete.*
4. *Given $P \in \mathsf{Depth}$, checking if $P \in \mathsf{Breadth}$ is Σ_1-complete.*

Given a process P, since checking $P \in \mathsf{Depth}(k)$ and $P \in \mathsf{Name}(k)$ is decidable, $P \in \mathsf{Depth}$ and $P \in \mathsf{Name}$ are both Σ_1. For the problem $P \in \mathsf{Breadth}(k)$ to be in Π_1, we semi-decide the complement. We enumerate processes until we find $Q \in \mathcal{R}(P)$ with $|Q|_{\mathcal{B}} > k$. With an oracle for this, we can semi-decide $P \in \mathsf{Breadth}$ by enumerating the possible bounds k and querying the oracle.

We show hardness for these problems. For a Turing machine M with input x, we construct a process $P(M, x)$ that simulates M and in each step increases the number of names and the depth. If M halts on x, then $P(M, x)$ is name-, and also depth-bounded. If M does not halt on x, then $P(M, x)$ is not depth-bounded.

For problems related to breadth boundedness, we show that the computation of a Turing machine M simultaneously on all possible inputs can be simulated by a process $P(M)$ in constant breadth. We increment the breadth of $P(M)$ every time M accepts an input. Restricted breadth boundedness is Π_1-hard, because we can reduce emptiness of M to this question. Breadth boundedness is Σ_2-hard, since the simulating process has bounded breadth iff $L(M)$ is finite.

To show that boundedness in breadth is Σ_1-complete even for $P \in \mathsf{Depth}$, we reduce the boundedness problem for reset Petri nets to breadth boundedness. We simulate net N with a depth-bounded $P(N)$. Tokens are send actions, consumed and produced when transitions fire. To reset a place, we create a fresh name for its tokens. The breadth of $P(N)$ is related to the token count in N, and $P(N)$ is bounded in breadth if and only if N is bounded.

This shows that the coverability set is not computable for $P \in \mathsf{Depth}$. The coverability set of $P(N)$ would allow to decide boundedness of the reset net N.

References

1. C. Dufourd, A. Finkel, and Ph. Schnoebelen. Reset nets between decidability and undecidability. In *ICALP*, volume 1443 of *LNCS*, pages 103–115. Springer, 1998.
2. C. Dufourd, P. Jancar, and Ph. Schnoebelen. Boundedness of reset p/t nets. In *ICALP*, volume 1644 of *LNCS*, pages 301–310. Springer, 1999.
3. E.A. Emerson and K.S. Namjoshi. On model checking for non-deterministic infinite-state systems. In *LICS*, pages 70–80, Jun 1998.
4. A. Finkel. A generalization of the procedure of Karp and Miller to well structured transition systems. In *ICALP*, volume 267 of *LNCS*, pages 499–508. Springer, 1987.
5. A. Finkel. Reduction and covering of infinite reachability trees. *Inf. Comp.*, 89(2):144–179, 1990.
6. A. Finkel and J. Goubault-Larrecq. Forward analysis for WSTS, part I: Completions. In *STACS*, pages 433–444, 2009.
7. A. Finkel and J. Goubault-Larrecq. Forward analysis for WSTS, part II: Complete WSTS. In *ICALP*, volume 5556 of *LNCS*, pages 188–199. Springer, 2009.
8. A. Finkel and J. Goubault-Larrecq. The theory of WSTS: The case of complete WSTS. In *ATPN*, LNCS 7347, pages 3–31. Springer, 2012.
9. R. Hüchting, R. Majumdar, and R. Meyer. A theory of name boundedness. In *CONCUR*, volume 8052 of *LNCS*, pages 182–196. Springer, 2013.
10. R. M. Karp and R. E. Miller. Parallel program schemata. *J. Comput. Syst. Sci.*, 3(2):147–195, 1969.
11. R. Meyer. On boundedness in depth in the π-calculus. In *IFIP TCS*, volume 273 of *IFIP*, pages 477–489. Springer, 2008.
12. R. Meyer. A theory of structural stationarity in the π-calculus. *Acta Inf.*, 46(2):87–137, 2009.
13. R. Milner. *Communicating and Mobile Systems: the π-Calculus.* CUP, 1999.
14. F. Rosa-Velardo and D. de Frutos-Escrig. Forward analysis for Petri nets with name creation. In *ATPN*, pages 185–205. Springer, 2010.
15. F. Rosa-Velardo and M. Martos-Salgado. Multiset rewriting for the verification of depth-bounded processes with name binding. *Inf. Comput.*, 215:68–87, June 2012.
16. D. Sangiorgi and D. Walker. *The π-calculus: a Theory of Mobile Processes.* CUP, 2001.
17. R.I. Soare. *Recursively enumerable sets and degrees.* Springer, 1980.
18. T. Wies, D. Zufferey, and T. A. Henzinger. Forward analysis of depth-bounded processes. In *FOSSACS*, volume 6014 of *LNCS*, pages 94–108. Springer, 2010.