


Complexity of Coverability in Depth-Bounded Processes

A. R. Balasubramanian   

Technische Universität München, Germany

Abstract

We consider the class of depth-bounded processes in π -calculus. These processes are the most expressive fragment of π -calculus, for which verification problems are known to be decidable. The decidability of the coverability problem for this class has been achieved by means of well-quasi orders. (Meyer, IFIP TCS 2008; Wies, Zufferey and Henzinger, FoSSaCS 2010). However, the precise complexity of this problem has not been known so far, with only a known EXPSPACE-lower bound.

In this paper, we prove that coverability for depth-bounded processes is \mathbf{F}_{e_0} -complete, where \mathbf{F}_{e_0} is a class in the fast-growing hierarchy of complexity classes. This solves an open problem mentioned by Haase, Schmitz, and Schnoebelen (LMCS, Vol 10, Issue 4) and also addresses a question raised by Wies, Zufferey and Henzinger (FoSSaCS 2010).

2012 ACM Subject Classification Theory of computation \rightarrow Problems, reductions and completeness; Theory of computation \rightarrow Distributed computing models

Keywords and phrases π -calculus, Depth-bounded processes, Fast-growing complexity classes

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2022.17

Funding A. R. Balasubramanian: Supported by funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement No 787367 (PaVeS).

Acknowledgements I am grateful to the reviewers and Prof. Javier Esparza for their useful comments and suggestions.

1 Introduction

The π -calculus [21, 22] is a well-known formalism for describing concurrent message-passing systems admitting unbounded process creation and mobility of agents. Intuitively speaking, a configuration of such a system is a graph in which each vertex is a process labelled by its current state and there is an edge between two processes if they share a channel using which they can pass messages. The flexibility of π -calculus lies in the fact that processes can transmit the names of channels using channels themselves, allowing reconfiguration of channels using process definitions itself. Due to its immense expressive power, all interesting verification problems quickly become undecidable for π -calculus processes.

Consequently, research on π -calculus has been focused on finding fragments for which certain problems are decidable. The most expressive fragment of π -calculus for which some verification problems still remain decidable is the class of depth-bounded processes [20]. Intuitively, depth-bounded processes are those in which the length of simple paths in the set of reachable configurations is bounded by a constant. It is known that depth-bounded processes can be viewed as well-structured transition systems (WSTS) [20]. This implies that the coverability problem for such systems is decidable [20, 27]. Intuitively, coverability consists of deciding if a given system can reach a configuration where some process is in an error state.

However, despite the positive decidability results known regarding this problem, the exact complexity of this problem has remained open so far. To the best of our knowledge, only an EXPSPACE-hardness result is known for this problem [27]. In this paper, we



© A. R. Balasubramanian;

licensed under Creative Commons License CC-BY 4.0

33rd International Conference on Concurrency Theory (CONCUR 2022).

Editors: Bartek Klin, Slawomir Lasota, and Anca Muscholl; Article No. 17; pp. 17:1–17:19

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

provide complexity-theoretic completeness results for this problem. More specifically, we prove that the coverability problem for depth-bounded processes is \mathbf{F}_{ϵ_0} -complete, where \mathbf{F}_{ϵ_0} is a complexity class in the *fast-growing hierarchy* of complexity classes [24]. This is a hierarchy of complexity classes which allows for a finer classification of problems that do not admit any elementary-time algorithms, i.e., problems which do not have algorithms whose running times can be upper bounded by a fixed tower of exponentials in the input size. In particular, our result proves that the coverability problem for depth-bounded processes is not primitive-recursive and indeed is harder than even problems complete for the Ackermann complexity class.

The complexity-theoretic classification of problems which are non-elementary has attracted a lot of attention in the recent years, with various techniques developed for proving both lower and upper bounds [13, 6, 25, 24, 1, 23, 8, 19, 7, 18]. While these results are obviously negative from a tractability perspective, understanding the precise complexity of a problem may help us to solve it in practice by reducing it to other well-studied problems for which tools and heuristics have been developed, like the satisfiability problem for weak S1S or the Petri net reachability problem [3, 12, 15, 4, 5, 16, 10]. The fast-growing hierarchy is of great assistance in this task. Adding new complete problems for classes in this hierarchy can help us prove hardness results for other problems in the future, without having to resort to coming up with reductions from scratch, i.e., from Turing machines or counter machines.

Our result significantly improves upon the existing lower bound of EXPSPACE-hardness, which is inherited from the coverability problem for Petri nets. Further, it settles a conjecture raised by Hasse, Schmitz and Schnoebelen (Section 8.3 of [17]) and also addresses a question raised by Wies, Zufferey and Henzinger (Section 5 of [27]).¹ To prove the lower bound, we introduce a new model of computation called *nested counter systems with levels*, which (in a manner) simplifies the already existing model of *nested counter systems* [8], while preserving the hardness of that model.

The techniques used in this paper are similar to the ones presented in [2], in order to prove \mathbf{F}_{ϵ_0} -completeness for parameterized coverability of *bounded-depth broadcast networks*. While some of the ideas between these two papers are similar, there are some differences between the models considered in these two papers. First, as the name suggests, broadcast networks allow for a process to *broadcast* to its set of neighbors, whereas processes in π -calculus interact in a manner akin to *rendez-vous* communication. One might expect that there is a drop in complexity when the communication mechanism goes from broadcast to rendez-vous. For instance, as mentioned in [11], coverability for networks with (unrestricted) broadcast communication is Ackermann-complete, while the same problem for rendez-vous networks is (only) EXPSPACE-complete. Our result suggests that this drop in complexity need not always be the case. Further, in broadcast networks, there is no process creation nor dynamic reconfiguration of channels, whereas π -calculus has both. Finally, for the lower bound construction in this paper, we also need to prove depth-boundedness of any reachable configuration in the process constructed for the reduction, whereas no such property needs to be proven for the lower bound construction for broadcast networks. We also believe that the newly introduced model of nested counter systems with levels (whose hardness we prove by using ideas from [2]), makes the proof of the lower bound for π -calculus cleaner when compared with giving a direct reduction from nested counter systems as was done in [2].

¹ The version of the problem that the authors of [27] consider does not assume that a bound on the depth of the process is given as part of the input, whereas in our setting we take this to be the case, in order to prove the upper bound. However, our lower bound result does not require this assumption.

2 Preliminaries

We first present the syntax and the semantics of the version of π -calculus that we will use. The definitions here are taken from the ones given in [27].

2.1 The π -calculus

We assume that there is a countable collection of *names* (denoted by x, y, \dots) and a countable collection of *process identifiers* (denoted by A, B, \dots). Each name and identifier has an associated *arity* in \mathbb{N} . We use boldface letters like \mathbf{x}, \mathbf{y} to denote (possibly empty) vectors over names and denote substitution of names by $[\mathbf{x}/\mathbf{y}]$, i.e., if $\mathbf{x} = x_1, \dots, x_n$ and $\mathbf{y} = y_1, \dots, y_n$, then $[\mathbf{x}/\mathbf{y}]$ denotes a mapping in which each y_i is mapped to x_i and every other name is mapped to itself.

A *process term* (or simply a term) P is either the unit process 0 , or a parameterized process identifier $A(\mathbf{x})$, or any term obtained by the standard operations of parallel composition $P_1 \mid P_2$, external choice $\pi_1 \cdot P_1 + \pi_2 \cdot P_2$ and name restriction $(\nu x)P_1$. Here P_1 and P_2 are themselves terms and π_1 and π_2 are prefixes which can either be an *input prefix* $x(\mathbf{y})$ or an *output prefix* $\bar{x}(\mathbf{y})$ or the empty string. All parameter vectors occurring in a parameterized process identifier or a prefix must respect the arity of the names and identifiers. A *thread* is a term of the form $A(\mathbf{x})$. We use Π and Σ to denote (indexed) parallel composition and external choice. We further use $(\nu \mathbf{x})$ to denote $(\nu x_1)(\nu x_2) \dots (\nu x_n)$ where $\mathbf{x} = x_1, \dots, x_n$. The application of a substitution of names σ to a term P , denoted by $\sigma(P)$, is defined in the usual way.

An occurrence of a name x in a term P is called *free* if it is not below a (νx) or an input prefix $y(x)$. We let $\text{fn}(P)$ denote the set of free names of P . A *bound name* of P is a name of P which is not free. We say that P is *closed* if $\text{fn}(P) = \emptyset$. We use the usual *structural congruence relation* $P \equiv Q$ on process terms, i.e., $P \equiv Q$ if P is syntactically equal to Q upto renaming and reordering of bound names, associativity and commutativity of parallel composition and external choice, elimination of units $((P \mid 0) \equiv P, (\nu x)0 \equiv 0)$ and *scope extrusion* $((\nu x)(P \mid Q) \equiv (\nu x)P \mid Q$ if $x \notin \text{fn}(Q)$).

A *configuration* is a closed term of the form $(\nu \mathbf{x})(\Pi_{i \in I} A_i(\mathbf{x}_i))$. A *process* \mathcal{P} is a pair (I, \mathcal{E}) where I is an *initial configuration* and \mathcal{E} is a set of *parametric equations* of the form $A(\mathbf{x}) = P$ where A is an identifier and P is a term such that 1) every identifier in \mathcal{P} is defined by exactly one equation in \mathcal{E} and 2) if $A(\mathbf{x}) = P$ is an equation, then $\text{fn}(P) \subseteq \{\mathbf{x}\}$. We assume that all the equations are given in the following form:

$$A(\mathbf{x}) = \sum_{i \in I} \pi_i \cdot (\nu \mathbf{x}_i) \left(\prod_{j \in J_i} A_j(\mathbf{x}_j) \right)$$

Operational semantics

Let $\mathcal{P} = (I, \mathcal{E})$ be a process. We define a transition relation on the set of configurations using \mathcal{E} as follows. Let P and Q be configurations. Then $P \rightarrow Q$ iff the following conditions are satisfied:

- $P \equiv (\nu \mathbf{u})(A(\mathbf{v}) \mid B(\mathbf{w}) \mid P')$,
- The defining equation of A in \mathcal{E} is of the form $A(\mathbf{x}) = x(\mathbf{x}') \cdot (\nu \mathbf{x}'')(M) + M'$,
- The defining equation of B in \mathcal{E} is of the form $B(\mathbf{y}) = \bar{y}(\mathbf{y}') \cdot (\nu \mathbf{y}'')(N) + N'$,
- $\sigma = [\mathbf{v}/\mathbf{x}, \mathbf{w}/\mathbf{y}, \mathbf{w}'/\mathbf{x}', \mathbf{z}_A/\mathbf{x}'', \mathbf{z}_B/\mathbf{y}'']$ where $\mathbf{z}_A, \mathbf{z}_B$ are fresh names and \mathbf{w}' is the set of names assigned to \mathbf{y}' under the mapping $[\mathbf{w}/\mathbf{y}]$.
- $\sigma(x) = \sigma(y)$ and
- $Q \equiv (\nu \mathbf{u}, \mathbf{z}_A, \mathbf{z}_B)(\sigma(M) \mid \sigma(N) \mid P')$

17:4 Complexity of Coverability in Depth-Bounded Processes

We denote such a step by $P \xrightarrow{A(\mathbf{v}),\sigma(x),B(\mathbf{w})} Q$ or simply by $P \rightarrow Q$. We can then define the reachability relation $\xrightarrow{*}$ as the reflexive and transitive closure of \rightarrow . We say that a configuration P is reachable in \mathcal{P} iff $I \xrightarrow{*} P$. We further say that P is coverable if $P \equiv (\nu \mathbf{x})P'$ and there exists $Q \equiv (\nu \mathbf{x})(P' \mid R)$ such that $I \xrightarrow{*} Q$. The coverability problem is to decide if a given configuration P is coverable in a given process \mathcal{P} .

Depth-bounded processes

We now define the class of depth-bounded processes. The nesting of restrictions $nest$ of a term P is defined inductively as follows: $nest(0) = nest(A(\mathbf{x})) = nest(\pi_1 \cdot P_1 + \pi_2 \cdot P_2) = 0$, $nest((\nu x)P) = 1 + nest(P)$ and $nest(P_1 \mid P_2) = \max\{nest(P_1), nest(P_2)\}$. The *depth* of a term P is the minimal nesting of restrictions of terms in the congruence class of P :

$$depth(P) := \min\{nest(Q) : Q \equiv P\}$$

► **Definition 1.** A set of configurations \mathcal{C} is called *k-depth-bounded* if the depth of all configurations in \mathcal{C} is at most k . \mathcal{C} is called *depth-bounded* if there is some k such that it is *k-depth-bounded*. A process P is called *(k-)depth-bounded* if its set of reachable configurations is *(k-)depth-bounded*.

► **Example 2.** The following example intuitively demonstrates a system in which there is one “level 0” thread which can spawn “level 1” threads by using a “New1” thread. Then, each level 1 thread can itself spawn “level 2” threads by using their own “New2” threads.

$$Level0(x) = \bar{x}().Level0(x) \quad New1(x) = x().((\nu y)(New1(x) \mid Level1(x, y) \mid New2(y)))$$

$$Level1(x, y) = \bar{y}().Level1(x, y) \quad New2(y) = y().((\nu z)(New2(y) \mid Level2(y, z) \mid New3(z)))$$

$$Level2(y, z) = \bar{z}().Level2(y, z) \quad New3(z) = z().New3(z)$$

Suppose we set $I = (\nu x)(Level0(x) \mid New1(x))$. Then the following is a valid run:

$$\begin{aligned} I &\rightarrow (\nu x)(Level0(x) \mid New1(x) \mid (\nu y)(Level1(x, y) \mid New2(y))) \\ &\rightarrow (\nu x)(Level0(x) \mid New1(x) \mid (\nu y)(Level1(x, y) \mid New2(y) \mid (\nu z)(Level2(y, z) \mid New3(z)))) \end{aligned}$$

We note that the depth of the last configuration in this run is 3. Indeed, we can show that the depth of any reachable configuration from I is at most 3. Later on, we will see that some of the ideas behind this example are relevant to our lower bound construction.

Our main theorem of the paper is that,

► **Theorem 3.** *The coverability problem for depth-bounded processes is \mathbf{F}_{ϵ_0} -complete.*

Here, we assume that the input consists of a process \mathcal{P} and a number k such that \mathcal{P} is *k-depth-bounded*. Further, \mathbf{F}_{ϵ_0} is a complexity class in the *fast-growing hierarchy* of complexity classes [24]. Due to lack of space, we do not define it here. The lower bound behind this theorem is accomplished by giving a log-space reduction from a \mathbf{F}_{ϵ_0} -hard problem. The upper bound is obtained by using results on the length of *controlled bad sequences* over a suitable well-quasi ordering.

We first explain the proof of the lower bound. To do this, we first introduce a model called *nested counter systems with levels* (NCSL) and show that the coverability problem for this model is \mathbf{F}_{ϵ_0} -hard. We then give a reduction from this problem to the coverability problem for depth-bounded processes, thereby proving the lower bound of Theorem 3.

3 Nested counter systems with levels (NCSL)

We now introduce a new model of computation called nested counter systems with levels (NCSL) and prove \mathbf{F}_{ϵ_0} -hardness of coverability for this model. NCSL are closely related to the so-called nested counter systems (NCS) [8]. Indeed, in Section 4, we will recall NCS and prove the hardness result for NCSL by giving a reduction from the coverability problem for NCS.

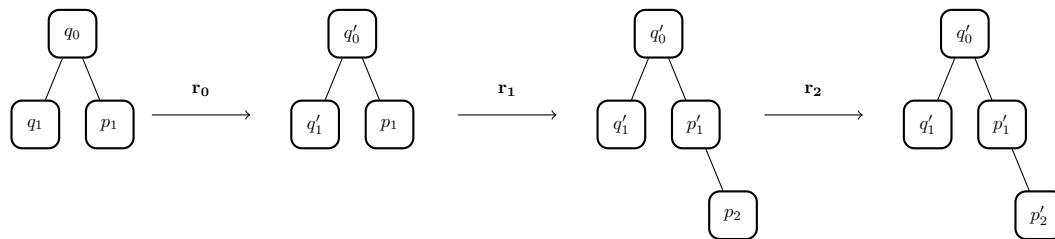
Before describing NCSL in a formal manner, we give some intuition. A k -NCSL is a generalisation of a usual counter system with *higher-order counters*. Intuitively, a 1-dimensional counter is a usual counter which can add or subtract 1. A 2-dimensional counter can add or subtract 1-dimensional counters, a 3-dimensional counter can add or subtract 2-dimensional counters and so on. A k -NCSL can produce up to k -dimensional counters and then manipulate these counters using “local” rules, i.e., rules which update at most 2 counters at a time. Later on, we will consider the NCS model [8], which allows to update multiple counters in a single step.

Formally, a k -nested counter system with levels (k -NCSL) is a tuple $\mathcal{N} = (Q, \delta_0, \dots, \delta_{k-1}, \delta_k)$ where Q is a finite set of *states* and each δ_l is a set of *level- l rules* such that $\delta_l \subseteq \bigcup_{1 \leq i \leq j \leq 2} (Q^i \times Q^j)$. We further enforce that if $l = k$ then $\delta_l \subseteq Q \times Q$. The set $\mathcal{C}_{\mathcal{N}}$ of *configurations* of \mathcal{N} is defined to be the set of all labelled rooted trees of height at most k , with labels from the set Q .

The operational semantics of \mathcal{N} is defined in terms of the following transition relation $\rightarrow \subseteq \mathcal{C}_{\mathcal{N}} \times \mathcal{C}_{\mathcal{N}}$ on configurations: Let $r := ((q_0, \dots, q_i), (q'_0, \dots, q'_j)) \in \delta_l$ be a level- l rule with $l \leq k$ and $0 \leq i \leq j \leq 1$. We say that a configuration C can move to the configuration C' using the rule r (denoted by $C \xrightarrow{r} C'$) if *there is a node v_0 at depth l in C with label q_0 and the following holds*.

- **Creation.** Suppose $r = ((q_0), (q'_0, q'_1))$. Then C' is obtained from C by changing the label of v_0 to q'_0 , creating a new vertex v_1 with label q'_1 and adding it as child to v_0 .
- **1-Preservation.** Suppose $r = ((q_0), (q'_0))$. Then C' is obtained from C by changing the label of v_0 to q'_0 .
- **2-Preservation.** Suppose $r = ((q_0, q_1), (q'_0, q'_1))$. Then there is a child v_1 of v_0 in C with label q_1 and C' is obtained from C by changing the labels of v_0 and v_1 to q'_0 and q'_1 respectively.

► **Example 4.** Let us consider the 2-NCSL \mathcal{N} given by the states $Q = \{p_i, p'_i, q_i, q'_i : 0 \leq i \leq 4\}$ and consisting of the rules $r_0 \in \delta_0, r_1 \in \delta_1, r_2 \in \delta_2$ where $r_0 = ((q_0, q_1), (q'_0, q'_1)), r_1 = ((p_1), (p'_1, p_2)), r_2 = ((p_2), (p'_2))$. In Figure 1, we illustrate the application of these rules to a configuration of \mathcal{N} .



■ **Figure 1** Application of the rules r_0, r_1 and r_2 to a configuration of \mathcal{N} , which is described in Example 4.

We say that $C \rightarrow C'$ if $C \xrightarrow{r} C'$ for some rule r . We can then define the reachability relation $\xrightarrow{*}$ in a standard manner. Given two states $q_{in}, q_f \in Q$, we say that q_{in} can cover q_f if the (unique) configuration consisting of the single root vertex labelled with q_{in} (also called the initial configuration of \mathcal{N}) can reach *some* configuration where the root is labelled by q_f . The coverability problem for an NCSL is then the following: Given an NCSL \mathcal{N} and two states q_{in}, q_f , can q_{in} cover q_f ? We prove that

► **Theorem 5.** *The coverability problem for NCSL is \mathbf{F}_{ϵ_0} -hard, even when restricted to NCSL which only have creation and 2-preservation rules.*

The proof of Theorem 5 is deferred to Section 4. We shall assume this theorem and first prove the main result of this paper (Theorem 3), i.e., that coverability for depth-bounded π -calculus processes is \mathbf{F}_{ϵ_0} -hard.

3.1 Hardness of coverability for depth-bounded π -calculus processes

Throughout this subsection, we let $\mathcal{N} = (Q, \delta_0, \dots, \delta_{k-1}, \delta_k)$ be a fixed k -NCSL which only has creation and 2-preservation rules. Note that since there are no 1-preservation rules, by definition of a k -NCSL, δ_k is empty and so we will ignore δ_k everywhere in this section. Let q_{in} and q_f be two fixed states of \mathcal{N} . We will now construct a depth-bounded process \mathcal{P} and a configuration C of \mathcal{P} such that C can be covered in \mathcal{P} iff q_f can be covered from q_{in} in \mathcal{N} .

Process identifiers, names and the initial configuration

To construct \mathcal{P} , we have to define an initial configuration and a set of parametric equations. We begin by specifying the set of names and the process identifiers that we shall use in the equations. Based on these names and identifiers, we define the initial configuration and also introduce an injective mapping \mathbb{B} from the set of configurations of \mathcal{N} to the set of configurations of \mathcal{P} . This map will be useful to prove the correctness of our reduction.

Process identifiers and names. For each $1 \leq i \leq k$, we will have a process identifier $start[i]$. For each $0 \leq i \leq k$ and each state q of \mathcal{N} , we will have an identifier $q[i]$. Notice that each process identifier is of the form $a[b]$ where $a \in Q \cup \{start\}$ and $0 \leq b \leq k$. The first part “ a ” will be called the *base* of the identifier and the second part “ b ” will be called the *grade* of the identifier. The arities of the identifiers are as follows: The arity of each $start[i]$ will be $|\delta_{i-1}|$. For every state q of \mathcal{N} , the arity of $q[0]$ will be $|\delta_0|$, the arity of $q[k]$ will be $|\delta_{k-1}|$ and the arity of every other $q[i]$ will be $|\delta_{i-1}| + |\delta_i|$.

The set of names that we will be using in the equations will be the set of rules of \mathcal{N} , i.e., $\delta_0 \cup \delta_1 \cup \dots \cup \delta_{k-1}$. For each δ_i , we let \mathbf{n}_i denote some fixed vector comprising all the names from δ_i . We also assume that there is another countably infinite set of names needed to describe the configurations of \mathcal{P} . We note that this latter set is not part of the input.

A mapping. We now introduce an injective map from the set of configurations of \mathcal{N} to the set of configurations of \mathcal{P} . Let C be a configuration of the NCSL \mathcal{N} . To C , we assign a unique configuration of \mathcal{P} (denoted by $\mathbb{B}(C)$) as follows: Let the set of vertices of C be V and let the set of internal vertices of C (the root and the other non-leaf vertices) be IV . $\mathbb{B}(C)$ is then defined as the configuration

$$(\nu \mathbf{z}) (\prod_{v \in V} A_v(\mathbf{x}_v, \mathbf{y}_v) \mid \prod_{v \in IV} B_v(\mathbf{y}_v))$$

where $\{\mathbf{z}\} = \cup_{v \in V} \{\mathbf{x}_v, \mathbf{y}_v\}$ and for each v ,

- $\{\mathbf{x}_v\} \cap \{\mathbf{y}_v\} = \emptyset$,
- If the label of v in C is q and v is at depth l , then $A_v = q[l]$ and $B_v = \text{start}[l + 1]$,
- If v is the root, then \mathbf{x}_v is the empty vector. If v is a leaf, then \mathbf{y}_v is the empty vector. Otherwise, if v is at depth l , then \mathbf{x}_v is of size $|\delta_{l-1}|$ and \mathbf{y}_v is of size $|\delta_l|$.
- For any v' , if v' is a child of v , then $\mathbf{x}_{v'} = \mathbf{y}_v$ and $\{\mathbf{y}_{v'}\} \cap \{\mathbf{x}_v\} = \emptyset$; if v' is a sibling of v , then $\mathbf{x}_{v'} = \mathbf{x}_v$ and $\{\mathbf{y}_{v'}\} \cap \{\mathbf{y}_v\} = \emptyset$; otherwise, $\{\mathbf{x}_v, \mathbf{y}_v\} \cap \{\mathbf{x}_{v'}, \mathbf{y}_{v'}\} = \emptyset$.

To give an intuition behind this mapping, let us look at $\mathbb{B}(C)$ from the perspective of graphs. We construct a graph where there is a vertex for each $A_v(\mathbf{x}_v, \mathbf{y}_v)$ and each $B_v(\mathbf{y}_v)$ and we connect two such vertices by an edge if they share at least one free name and the corresponding identifiers have different grades. By the requirements given above, this would imply that the graph that we get is a tree which has a “copy” of C as a subgraph, along with a new leaf vertex added to every internal vertex of C . Ignoring the new leaf vertices for now, this means that $\mathbb{B}(C)$ can be thought of as a “representation” of C in the process \mathcal{P} . The parametric equations that we shall construct will make sure that if $\mathbb{B}(C)$ can move to a new configuration P , then P will be a representation of C' for some C' such that $C \rightarrow C'$ in the NCSL \mathcal{N} .

We now have the following lemma which proves depth-boundedness of any configuration of the form $\mathbb{B}(C)$. The intuition behind this lemma is that the “graph” of $\mathbb{B}(C)$ contains a copy of C as a subgraph along with some other additional leaf vertices. Hence, since the depth of C is bounded by k , we can expect that the depth of $\mathbb{B}(C)$ is also bounded.

► **Lemma 6** (Depth-boundedness). *For any configuration C , the depth of $\mathbb{B}(C)$ is at most $\sum_{l=0}^{k-1} |\delta_l|$.*

Proof. Let V and IV be the set of vertices and internal vertices of C respectively. For any vertex \mathbf{n} , let $C_{\mathbf{n}}$ be the (labelled) subtree of C rooted at \mathbf{n} and let $V_{\mathbf{n}}$ and $IV_{\mathbf{n}}$ be the set of vertices and internal vertices of $C_{\mathbf{n}}$ respectively.

We know that $\mathbb{B}(C)$ is of the form $(\nu \mathbf{z}) (\prod_{v \in V} A_v(\mathbf{x}_v, \mathbf{y}_v) \mid \prod_{v \in IV} B_v(\mathbf{y}_v))$. Let $\mathbb{B}(C_{\mathbf{n}})$ be the sub-process term of $\mathbb{B}(C)$ given by $\prod_{v \in V_{\mathbf{n}}} A_v(\mathbf{x}_v, \mathbf{y}_v) \mid \prod_{v \in IV_{\mathbf{n}}} B_v(\mathbf{y}_v)$ and let $\{\mathbf{z}_{\mathbf{n}}\} = \cup_{v \in V_{\mathbf{n}}} \{\mathbf{x}_v, \mathbf{y}_v\}$.

By induction on the height h of the vertex \mathbf{n} in the tree C , we will now show that the depth of $(\nu \mathbf{z}_{\mathbf{n}}) \mathbb{B}(C_{\mathbf{n}})$ is at most $\sum_{l=\max\{k-1-h, 0\}}^{k-1} |\delta_l|$. For the base case, when \mathbf{n} is a leaf and $C_{\mathbf{n}}$ is a tree with a single node, we have that $(\nu \mathbf{z}_{\mathbf{n}}) \mathbb{B}(C_{\mathbf{n}}) \equiv (\nu \mathbf{x}_{\mathbf{n}}) q[k](\mathbf{x}_{\mathbf{n}})$ for some q and some vector $\mathbf{x}_{\mathbf{n}}$ of size $|\delta_{k-1}|$. This shows that the claim is true for the base case.

For the induction step, let $Ch(\mathbf{n})$ be the children of \mathbf{n} . By the requirements imposed upon $\mathbb{B}(C)$, we can use the scope extrusion rule to write $(\nu \mathbf{z}_{\mathbf{n}}) \mathbb{B}(C_{\mathbf{n}})$ as $(\nu \mathbf{x}_{\mathbf{n}}, \mathbf{y}_{\mathbf{n}}) (A_{\mathbf{n}}(\mathbf{x}_{\mathbf{n}}, \mathbf{y}_{\mathbf{n}}) \mid B_{\mathbf{n}}(\mathbf{y}_{\mathbf{n}}) \mid \prod_{v \in Ch(\mathbf{n})} (\nu(\mathbf{z}_v \setminus \mathbf{y}_{\mathbf{n}})) \mathbb{B}(C_v))$. By induction hypothesis, we have that the depth of each $(\nu \mathbf{z}_v) \mathbb{B}(C_v)$ is $\sum_{l=k-h}^{k-1} |\delta_l|$. This then implies that the depth of $(\nu \mathbf{x}_{\mathbf{n}}, \mathbf{y}_{\mathbf{n}}) (A_{\mathbf{n}}(\mathbf{x}_{\mathbf{n}}, \mathbf{y}_{\mathbf{n}}) \mid B_{\mathbf{n}}(\mathbf{y}_{\mathbf{n}}) \mid \prod_{v \in Ch(\mathbf{n})} (\nu(\mathbf{z}_v \setminus \mathbf{y}_{\mathbf{n}})) \mathbb{B}(C_v))$ is at most $\sum_{l=k-1-h}^{k-1} |\delta_l|$ if \mathbf{n} is not the root. If \mathbf{n} is the root, then the depth becomes at most $\sum_{l=0}^{k-1} |\delta_l|$ because $\mathbf{x}_{\mathbf{n}} = \emptyset$. Hence, the induction step is complete.

Since $\mathbb{B}(C) \equiv (\nu \mathbf{z}_{\mathbf{n}}) \mathbb{B}(C_{\mathbf{n}})$ where \mathbf{n} is the root, it follows that the depth of $\mathbb{B}(C)$ is at most $\sum_{l=0}^{k-1} |\delta_l|$. ◀

Initial configuration. Recall that for each $i \in \{0, \dots, k-1\}$, we let \mathbf{n}_i denote some fixed vector comprising all the names from δ_i . We then take the initial configuration of \mathcal{P} to be $(\nu \mathbf{n}_0)(q_{in}[0](\mathbf{n}_0) \mid \text{start}[1](\mathbf{n}_0))$. Note that the initial configuration of \mathcal{P} is the image of the initial configuration of \mathcal{N} under the \mathbb{B} mapping.

Parametric equations

Before we describe the parametric equations, we set up some notation. Let $r = ((q_0, \dots, q_i), (q'_0, \dots, q'_j))$ be a rule of the NCSL \mathcal{N} . By definition of creation and 2-preservation rules, it has to be the case that $i \leq 1$ and $j = 1$. In the sequel, for the sake of uniformity across all rules, we adopt the following nomenclature: If $i = 0$, we let $q_1 = \text{start}$. In this way, we can always associate a (unique) tuple $((q_0, q_1), (q'_0, q'_1))$ with any rule r .

Let $r = ((p, q), (p', q'))$ be a rule of \mathcal{N} . We say that the tuple (p, q) (resp. (p', q')) is the *precondition* (resp. *postcondition*) of r and we let $\text{pre}_{\text{fi}}^r := p$, $\text{pre}_{\text{se}}^r := q$, $\text{post}_{\text{fi}}^r := p'$ and $\text{post}_{\text{se}}^r := q'$.

We will set up the parametric equations in such a way so that $C \rightarrow C'$ is a step in \mathcal{N} iff $\mathbb{B}(C) \rightarrow \mathbb{B}(C')$. Intuitively this is accomplished by ensuring that if $r = ((p, q), (p', q')) \in \delta_l$ is a rule of \mathcal{N} , then a thread with identifier $p[l]$ can output along a name and go to $p'[l]$ and a thread with identifier $q[l+1]$ can receive along the same name and go to $q'[l+1]$.

Equations for identifiers of grade 0. For any $q \in Q$, the equation for $q[0]$ is,

$$q[0](\mathbf{n}_0) := \sum_{r \in \delta_0, \text{pre}_{\text{fi}}^r = q} \bar{r}(). \text{post}_{\text{fi}}^r[0](\mathbf{n}_0)$$

Intuitively, this equation corresponds to a thread with identifier $q[0]$ trying to execute some rule $r \in \delta_0$ for which $q = \text{pre}_{\text{fi}}^r$ and then becoming $\text{post}_{\text{fi}}^r[0]$.

Equations for identifiers of grade $1 \leq i \leq k-1$. Recall that the arity of any such identifier is $|\delta_{i-1}| + |\delta_i|$, except for identifiers with base *start*, for which it is $|\delta_{i-1}|$.

■ For any $q \in Q$, we have

$$q[i](\mathbf{n}_{i-1}, \mathbf{n}_i) := \sum_{r \in \delta_i, \text{pre}_{\text{fi}}^r = q} \bar{r}(). \text{post}_{\text{fi}}^r[i](\mathbf{n}_{i-1}, \mathbf{n}_i) + \sum_{r \in \delta_{i-1}, \text{pre}_{\text{se}}^r = q} r(). \text{post}_{\text{se}}^r[i](\mathbf{n}_{i-1}, \mathbf{n}_i)$$

Intuitively, the first summand of the equation corresponds to a thread with identifier $q[i]$ trying to execute some rule $r \in \delta_i$ for which $q = \text{pre}_{\text{fi}}^r$ and then becoming $\text{post}_{\text{fi}}^r[i]$. The second summand corresponds to a thread with identifier $q[i]$ trying to execute some rule $r \in \delta_{i-1}$ for which $q = \text{pre}_{\text{se}}^r$ and then becoming $\text{post}_{\text{se}}^r[i]$.

■ For the *start* base, we have

$$\text{start}[i](\mathbf{n}_{i-1}) := \sum_{r \in \delta_{i-1}, \text{pre}_{\text{se}}^r = \text{start}} r(). \left((\nu \mathbf{n}_i) \text{start}[i](\mathbf{n}_{i-1}) \mid \text{post}_{\text{se}}^r[i](\mathbf{n}_{i-1}, \mathbf{n}_i) \mid \text{start}[i+1](\mathbf{n}_i) \right)$$

Intuitively, this equation is responsible for spawning new threads of grade i with base in Q , when an appropriate output action is taken by some thread of grade $i-1$ with base in Q . First, if a thread with identifier $\text{start}[i]$ receives a message along some channel corresponding to some rule $r \in \delta_{i-1}$ with $\text{pre}_{\text{se}}^r = \text{start}$, then a fresh set of names (denoted by \mathbf{n}_i) are created. After that, the thread retains its identifier and two new threads are spawned, $\text{post}_{\text{se}}^r[i](\mathbf{n}_{i-1}, \mathbf{n}_i)$ and $\text{start}[i+1](\mathbf{n}_i)$. We note that these equations have a similar flavor to that of the equations for *New1* and *New2* given in Example 2.

Equations for identifiers of grade k . Recall that the arity of any identifier with grade k is $|\delta_{k-1}|$.

- For any $q \in Q$, we have

$$q[k](\mathbf{n}_{k-1}) := \sum_{r \in \delta_{k-1}, \text{pre}_{\text{se}}^r = q} r(). \text{post}_{\text{se}}^r[k](\mathbf{n}_{k-1})$$

- For the *start* base, we have

$$\text{start}[k](\mathbf{n}_{k-1}) := \sum_{r \in \delta_{k-1}, \text{pre}_{\text{se}}^r = \text{start}} r(). (\text{post}_{\text{se}}^r[k](\mathbf{n}_{k-1}) \mid \text{start}[k](\mathbf{n}_{k-1}))$$

The intuitions behind these equations are the same as the one for the previous case.

3.2 Proof of correctness

We now formally show the proof of correctness of our reduction. We begin with a lemma which shows that the constructed process \mathcal{P} can simulate the NCSL \mathcal{N} .

► **Lemma 7** (\mathcal{P} simulates \mathcal{N}). *Suppose $C \rightarrow C'$ is a step in \mathcal{N} . Then $\mathbb{B}(C) \rightarrow \mathbb{B}(C')$.*

Proof. Let $r = ((p, q), (p', q')) \in \delta_l$ for some $0 \leq l \leq k-1$ such that $C \xrightarrow{r} C'$. Let V be the set of vertices of C and let IV be the set of internal vertices of C . This means that there is a vertex \mathbf{n} in C at depth l such that the label of \mathbf{n} in C is p .

Let $\mathbb{B}(C) \equiv (\nu \mathbf{z}) (\prod_{v \in V} A_v(\mathbf{x}_v, \mathbf{y}_v) \mid \prod_{v \in IV} B_v(\mathbf{y}_v))$. By definition of the map \mathbb{B} , it has to be the case that $A_{\mathbf{n}} = p[l]$. We have two cases:

- Suppose $q \neq \text{start}$. Then there has to be a child \mathbf{n}' of \mathbf{n} in C such that its label in C is q . Hence, $A_{\mathbf{n}'} = q[l+1]$. Further, $\mathbf{y}_{\mathbf{n}} = \mathbf{x}_{\mathbf{n}'}$. By construction of the parametric equations, this means that $\mathbb{B}(C)$ can reach P where

$$P \equiv (\nu \mathbf{z}) (\prod_{v \in V \setminus \{\mathbf{n}, \mathbf{n}'\}} A_v(\mathbf{x}_v, \mathbf{y}_v) \mid p'[l](\mathbf{x}_{\mathbf{n}}, \mathbf{y}_{\mathbf{n}}) \mid q'[l+1](\mathbf{x}_{\mathbf{n}'}, \mathbf{y}_{\mathbf{n}'}) \mid \prod_{v \in IV} B_v(\mathbf{y}_v))$$

It is then easy to see that $P \equiv \mathbb{B}(C')$.

- Suppose $q = \text{start}$. Then $B_{\mathbf{n}}(\mathbf{y}_{\mathbf{n}}) = \text{start}[l+1](\mathbf{y}_{\mathbf{n}})$. By construction of the parametric equations, this means that $\mathbb{B}(C)$ can reach P given by

$$P \equiv (\nu \mathbf{z}, \mathbf{z}') (\prod_{v \in V \setminus \{\mathbf{n}\}} A_v(\mathbf{x}_v, \mathbf{y}_v) \mid p'[l](\mathbf{x}_{\mathbf{n}}, \mathbf{y}_{\mathbf{n}}) \mid \prod_{v \in IV} B_v(\mathbf{y}_v) \mid q'[l+1](\mathbf{y}_{\mathbf{n}}, \mathbf{z}') \mid \text{start}[l+2](\mathbf{z}'))$$

where the last term $\text{start}[l+2](\mathbf{z}')$ is not present if $l = k-1$. It is then easy to see that $P \equiv \mathbb{B}(C')$. ◀

Next we show that \mathcal{N} can also simulate \mathcal{P} .

► **Lemma 8** (\mathcal{N} simulates \mathcal{P}). *Suppose $\mathbb{B}(C) \rightarrow P$. Then there exists a configuration C' of \mathcal{N} such that $C \rightarrow C'$ and $P \equiv \mathbb{B}(C')$.*

Proof. Let V be the vertices of C and let IV be the set of internal vertices of C . Let $\mathbb{B}(C) \equiv (\nu \mathbf{z}) (\prod_{v \in V} A_v(\mathbf{x}_v, \mathbf{y}_v) \mid \prod_{v \in IV} B_v(\mathbf{y}_v))$ and let $\mathbb{B}(C) \xrightarrow{T_v(\mathbf{w}_v), c, T_{v'}(\mathbf{w}_{v'})} P$.

By construction of the parametric equations, it must be the case that $T_v(\mathbf{w}_v) = A_{\mathbf{n}}(\mathbf{x}_{\mathbf{n}}, \mathbf{y}_{\mathbf{n}})$ for some node \mathbf{n} and c must belong to $\{\mathbf{y}_{\mathbf{n}}\}$. Let $A_{\mathbf{n}} = p[l]$. Since $c \in \{\mathbf{y}_{\mathbf{n}}\}$, by definition of $\mathbb{B}(C)$, c can only be shared among the free names of the threads in $\{A_{\mathbf{n}'}(\mathbf{x}_{\mathbf{n}'}, \mathbf{y}_{\mathbf{n}'}) : \mathbf{n}' \text{ is a child of } \mathbf{n}\} \cup \{B_{\mathbf{n}}(\mathbf{y}_{\mathbf{n}})\}$. We now consider two cases:

17:10 Complexity of Coverability in Depth-Bounded Processes

- Suppose $T_{v'}(\mathbf{w}_{v'}) = A_{n'}(\mathbf{x}_{n'}, \mathbf{y}_{n'})$ for some n' which is a child of n . Let $A_{n'} = q[l+1]$. Since we have $\mathbb{B}(C) \xrightarrow{T_v(\mathbf{w}_v), c, T_{v'}(\mathbf{w}_{v'})} P$, by construction of the equations it has to be the case that there is a rule $r \in \delta_l$ of \mathcal{N} such that $\mathbf{pre}_{\mathbf{f}_i}^r = p$, $\mathbf{pre}_{\mathbf{se}}^r = q$ and

$$P \equiv (\nu \mathbf{z}) (\Pi_{v \in V \setminus \{n, n'\}} A_v(\mathbf{x}_v, \mathbf{y}_v) \mid p'[l](\mathbf{x}_n, \mathbf{y}_n) \mid q'[l+1](\mathbf{x}_{n'}, \mathbf{y}_{n'}) \mid \Pi_{v \in IV} B_v(\mathbf{y}_v))$$

where $p' = \mathbf{post}_{\mathbf{f}_i}^r$ and $q' = \mathbf{post}_{\mathbf{se}}^r$ respectively. Since $A_n = p[l]$ and $A_{n'} = q[l+1]$, it must be the case that the depth of n in C is l and the labels of n and n' in C are p and q respectively. It follows that there exists C' such that $C \xrightarrow{r} C'$. It is then easy to verify that $\mathbb{B}(C') \equiv P$.

- Suppose $T_{v'}(\mathbf{w}_{v'}) = B_n(\mathbf{y}_n)$. We know that $B_n = \mathit{start}[l+1]$. Since it is the case that $\mathbb{B}(C) \xrightarrow{T_v(\mathbf{w}_v), c, T_{v'}(\mathbf{w}_{v'})} P$, by construction of the parametric equations it must be that there is a rule $r \in \delta_l$ of \mathcal{N} such that $\mathbf{pre}_{\mathbf{f}_i}^r = p$, $\mathbf{pre}_{\mathbf{se}}^r = \mathit{start}$ and

$$P \equiv (\nu \mathbf{z}, \mathbf{z}') (\Pi_{v \in V \setminus \{n\}} A_v(\mathbf{x}_v, \mathbf{y}_v) \mid p'[l](\mathbf{x}_n, \mathbf{y}_n) \mid \Pi_{v \in IV} B_v(\mathbf{y}_v) \mid q'[l+1](\mathbf{y}_n, \mathbf{z}') \mid \mathit{start}[l+2](\mathbf{z}'))$$

where the last term $\mathit{start}[l+2](\mathbf{z}')$ is not present if $l = k-1$ and $p' = \mathbf{post}_{\mathbf{f}_i}^r$, $q' = \mathbf{post}_{\mathbf{se}}^r$ respectively. Since $A_n = p[l]$, it must be the case that the depth of n in C is l and the label of n in C is p . It follows then that there exists C' such that $C \xrightarrow{r} C'$. It is then easy to verify that $\mathbb{B}(C') \equiv P$. ◀

Note that the initial configuration I of \mathcal{P} is simply the image of the initial configuration of \mathcal{N} under the map \mathbb{B} . Hence, using Lemmas 6 and 8, we can conclude that

► **Corollary 9.** *The process \mathcal{P} is K -depth-bounded where $K = \sum_{l=0}^{k-1} |\delta_l|$.*

We then get the following theorem, whose proof follows in a straightforward manner by combining Lemmas 7 and 8.

► **Theorem 10.** *$C \xrightarrow{*} C'$ is a run in \mathcal{N} iff $\mathbb{B}(C) \xrightarrow{*} \mathbb{B}(C')$ is a run in the process \mathcal{P} . Consequently q_{in} can cover q_f in \mathcal{N} iff $(\nu \mathbf{n}_0) (q_f[0](\mathbf{n}_0))$ can be covered from the initial configuration I of \mathcal{P} .*

Hence, we have

► **Corollary 11.** *Coverability of depth-bounded processes is \mathbf{F}_{ϵ_0} -hard.*

4 Nested counter systems (NCS)

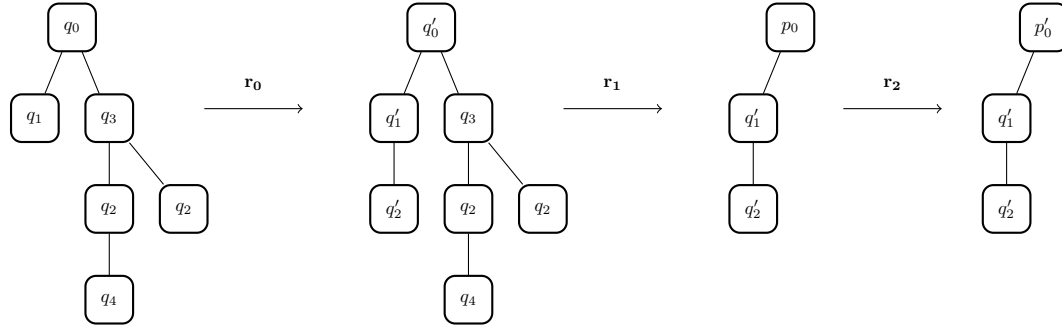
We now prove Theorem 5, by giving a reduction from the coverability problem for *nested counter systems* (NCS) which is known to be \mathbf{F}_{ϵ_0} -hard. We first recall the definition of NCS, which we present in a way that is akin to [2].

A k -nested counter system (k -NCS) is a tuple $\mathcal{N} = (Q, \delta)$ where Q is a finite set of *states* and $\delta \subseteq \bigcup_{1 \leq i, j \leq k+1} (Q^i \times Q^j)$ is a set of *rules*. The set $\mathcal{C}_{\mathcal{N}}$ of *configurations* of \mathcal{N} is defined to be the set of all labelled rooted trees of height at most k , with labels from the set Q .

The operational semantics of \mathcal{N} is defined in terms of the following transition relation $\rightarrow \subseteq \mathcal{C}_{\mathcal{N}} \times \mathcal{C}_{\mathcal{N}}$ on configurations: Let $r := ((q_0, \dots, q_i), (q'_0, \dots, q'_j)) \in \delta$ be a rule with $i \leq j \leq k$. We say that a configuration C can move to the configuration C' using the rule r (denoted by $C \xrightarrow{r} C'$), if there is a path v_0, v_1, \dots, v_i in C starting at the root such that for every $0 \leq l \leq i$, the label of v_l is q_l and, C' is obtained from C by 1) for every $0 \leq l \leq i$, changing the label of each v_l to q'_l and 2) for every $i+1 \leq l \leq j$, creating a new vertex v_l with label q'_l and adding it as a child to v_{l-1} .

Similarly, suppose $r := ((q_0, \dots, q_i), (q'_0, \dots, q'_j)) \in \delta$ is a rule with $j < i \leq k$. Then $C \xrightarrow{r} C'$ if there is a path v_0, v_1, \dots, v_i in C starting at the root such that for every $0 \leq l \leq i$, the label of v_l is q_l and, C' is obtained from C by 1) for every $0 \leq l \leq j$, changing the label of each v_l to q'_l and 2) removing the subtree rooted at the node v_{j+1} .

► **Example 12** (Example from [2]). Let us consider the NCS \mathcal{N} given by the states $Q = \{p_i, p'_i, q_i, q'_i : 0 \leq i \leq 4\}$ and consisting of the following rules: $r_0 = ((q_0, q_1), (q'_0, q'_1, q'_2))$, $r_1 = ((q'_0, q_3, q_2), (p_0))$, $r_2 = ((p_0), (p'_0))$. In Figure 2, we illustrate the application of these rules to a configuration of \mathcal{N} .



■ **Figure 2** Application of the rules r_0, r_1 and r_2 to a configuration of \mathcal{N} , which is described in Example 12.

Similar to NCSL, we can define the notions of $C \rightarrow C'$, $C \xrightarrow{*} C'$ and a state q_{in} covering another state q_f . It is known that the coverability problem for NCS is \mathbf{F}_{e_0} -hard (Theorem 7 of [8]).

We note that the rules of an NCS act “globally”, in the sense that it allows to update the value of (potentially) k many counters in one step. This is in contrast to NCSL, where we can update the value of at most two counters at a time. While it is not particularly surprising that this “global” update can be replaced by a series of “local” updates (hence giving a reduction from NCS to NCSL), the construction is not entirely trivial and requires some intricate arguments in order to prove its correctness.

A special case of NCS

We make a small remark which will help us simplify our reduction later on. Let $\mathcal{N} = (Q, \delta)$ be a k -NCS and let $q_{in}, q_f \in Q$. From \mathcal{N} , we construct a new k -NCS \mathcal{N}' as follows: First we add a new state end . Then, if $r = ((q_0, \dots, q_i), (q'_0, \dots, q'_j)) \in \delta$ with $j < i \leq k$, we replace r with the rule $r' := ((q_0, \dots, q_i), (q'_0, \dots, q'_j, \underbrace{end, \dots, end}_{i-j \text{ times}}))$. Intuitively, we are replacing

all rules which destroy some counters with corresponding rules that simply convert those counters to the state end . It can be easily verified that coverability of q_f from q_{in} is preserved while doing this operation. Hence, from here on, we assume that whenever $\mathcal{N} = (Q, \delta)$ is a k -NCS and $r = ((q_0, \dots, q_i), (q'_0, \dots, q'_j)) \in \delta$ then $i \leq j$.

4.1 Hardness of coverability for NCSL

We shall prove Theorem 5 by giving a reduction from the coverability problem for NCS. Let $k \geq 1$ and let $\mathcal{N} = (Q, \delta)$ be a k -NCS with two fixed states q_{in} and q_f . By the argument given in the previous paragraph, we can assume that if $r = ((q_0, \dots, q_i), (q'_0, \dots, q'_j)) \in \delta$

17:12 Complexity of Coverability in Depth-Bounded Processes

then $i \leq j$. We shall now construct a k -NCSL $\mathcal{N}' = (Q', \delta'_0, \dots, \delta'_{k-1})$ and two states q'_{in} and q'_f of \mathcal{N}' such that q'_{in} can cover q'_f in \mathcal{N}' iff q_{in} can cover q_f in \mathcal{N} . This will then prove Theorem 5. We begin by describing the states of \mathcal{N}' .

States of \mathcal{N}' . For every state q of \mathcal{N} , we will have two states $q[\top]$ and $q[\perp]$. Further, for every rule r of \mathcal{N} , we will have four states $\mathbf{rec}^r[\top], \mathbf{rec}^r[\perp], \mathbf{fwd}^r[\top]$ and $\mathbf{fwd}^r[\perp]$. Notice that each state of \mathcal{N}' is of the form $a[b]$ where $a \in Q \cup \{\mathbf{rec}^r, \mathbf{fwd}^r : r \in \delta\}$ and $b \in \{\top, \perp\}$. If a node v in a configuration C has as its label $a[b]$, then ‘a’ will be called its *base*. Further, if $b = \top$ (resp. $b = \perp$), then v will be called as a *leader node* (resp. *follower node*).

Good configurations of \mathcal{N}' . A configuration C is called good if the root of C is a leader, all other nodes are followers and the base of all the nodes of C belong to Q . Notice that there is a straightforward bijection between the set of all configurations of \mathcal{N} and the set of all good configurations of \mathcal{N}' . This bijection will be denoted by \mathbb{M} .

Rules of \mathcal{N}' . Before we describe the rules of \mathcal{N}' , we will state two invariants that will always be maintained by our construction. The first one is that, in any configuration reachable from a good configuration, exactly one node will be a leader. The second invariant is that, every rule of \mathcal{N}' will have a leader state in its precondition. Combined with the first invariant, this will intuitively ensure that the rules that can be fired from reachable configurations are limited and will help us simplify the proof of correctness of our reduction.

We now describe the rules of \mathcal{N}' . Let $r = ((q_0, \dots, q_i), (q'_0, \dots, q'_j))$ be a rule of \mathcal{N} . Corresponding to rule r , we will have the following set of rules in \mathcal{N}' . (In the following, we adopt the convention that if the name of a rule has a subscript $0 \leq l \leq j$, then that rule belongs to δ'_l).

- $start_0^r := ((q_0[\top]), (\mathbf{rec}^r[\top]))$.
- For every $0 \leq l \leq i - 1$, we have a rule $begin_l^r := ((\mathbf{rec}^r[\top], q_{l+1}[\perp]), (\mathbf{fwd}^r[\perp], \mathbf{rec}^r[\top]))$.
- For every $i \leq l \leq j - 1$, we have a rule $begin_l^r := ((\mathbf{rec}^r[\top]), (\mathbf{fwd}^r[\perp], \mathbf{rec}^r[\top]))$.
- $middle_j^r := ((\mathbf{rec}^r[\top]), (\mathbf{fwd}^r[\top]))$.
- For every $0 \leq l \leq j - 1$, we have a rule $end_l^r := ((\mathbf{fwd}^r[\perp], \mathbf{fwd}^r[\top]), (\mathbf{fwd}^r[\top], q'_{l+1}[\perp]))$.
- $finish_0^r := ((\mathbf{fwd}^r[\top]), q'_0[\top])$.

4.2 Proof of correctness

The intuitive idea behind the above gadget is given by the run demonstrated in the following lemma.

► **Lemma 13** (\mathcal{N}' simulates \mathcal{N}). *Suppose $C \xrightarrow{r} C'$ is a step in the NCS \mathcal{N} . Then, there is a run $\mathbb{M}(C) \xrightarrow{*} \mathbb{M}(C')$ in the NCSL \mathcal{N}' .*

Proof. Let $r = ((q_0, \dots, q_i), (q'_0, \dots, q'_j))$. Since $C \xrightarrow{r} C'$ is a step in \mathcal{N} , it follows that there is a path starting at the root of C labelled by q_0, \dots, q_i . It follows that in $\mathbb{M}(C)$ there is a path P starting at the root labelled by $q_0[\top], q_1[\perp], q_2[\perp], \dots, q_i[\perp]$. We now execute a sequence of rules according to the gadget for r as follows:

- First, using $start_0^r$, we change the label of the root from $q_0[\top]$ to $\mathbf{rec}^r[\top]$.
- Next, by firing $begin_0^r, \dots, begin_{i-1}^r$ in this order, we change the labels of the nodes in the path P to $\underbrace{\mathbf{fwd}^r[\perp], \dots, \mathbf{fwd}^r[\perp]}_{i \text{ times}}, \mathbf{rec}^r[\top]$.

- Then, by firing $begin_i^r, \dots, begin_{j-1}^r$ in this order, we add $j - i$ new nodes to the path P and get a new path P' of length $j + 1$ whose labels are $\underbrace{\mathbf{fwd}^r[\perp], \dots, \mathbf{fwd}^r[\perp]}_{j \text{ times}}, \mathbf{rec}^r[\top]$.
 - We use $middle_j^r$ to change the label of the last node in P' from $\mathbf{rec}^r[\top]$ to $\mathbf{fwd}^r[\top]$.
 - Then, by firing $end_{j-1}^r, \dots, end_0^r$ in this order, we change the labels of the nodes in the path P' to $\mathbf{fwd}^r[\top], q'_1[\perp], \dots, q'_j[\perp]$.
 - Finally, we use $finish_0^r$ to change the label of the root from $\mathbf{fwd}^r[\top]$ to $q'_0[\top]$.
- It can be easily verified that the resulting configuration D is such that $D = \mathbb{M}(C')$. ◀

We now present a converse to the above lemma which shows that a simulation in the other direction is also possible.

► **Lemma 14** (\mathcal{N} simulates \mathcal{N}'). *Suppose $C \xrightarrow{*} C'$ is a path of non-zero length in \mathcal{N}' such that 1) C is a good configuration and 2) in all the configurations between C and C' , the base of the root is not in Q . Then, C' is a good configuration and there is a rule r such that $\mathbb{M}^{-1}(C) \xrightarrow{r} \mathbb{M}^{-1}(C')$.*

Proof sketch. Let $P := C \rightarrow \gamma_0 \rightarrow \gamma_1 \dots \rightarrow C'$. The essential idea behind this lemma is that since C is a good configuration, the root node is a leader node and by the construction of the rules it must be the case that the first step must be of the form $C \xrightarrow{start_0^r} \gamma_0$ for some rule r . Then, by using the invariant that exactly one node is leader at all times and by using the construction of the rules, we can essentially show that P must be a path of the same form as the one given in the proof of Lemma 13. Having proved that, we can then show that in the NCS \mathcal{N} , $\mathbb{M}^{-1}(C) \xrightarrow{r} \mathbb{M}^{-1}(C')$. ◀

Because of these two “simulation” lemmas, we then get

► **Theorem 15.** q_{in} can cover q_f in \mathcal{N} iff $q_{in}[\top]$ can cover $q_f[\top]$ in \mathcal{N}' .

4.3 Wrapping up

The previous theorem implies that coverability for NCSL is \mathbf{F}_{ϵ_0} -hard. To prove Theorem 5, we need to show the same for NCSL with only creation and 2-preservation rules. We now show that 1-preservation rules can be replaced with creation rules in an NCSL while maintaining coverability.

Given a k -NCSL \mathcal{N} with two states q_{in}, q_f , we can remove all 1-preservation rules whilst preserving coverability as follows: We first add a new state end . Then if $r = ((q_0), (q'_0))$ is a 1-preservation rule in \mathcal{N} , we replace r with $r = ((q_0), (q'_0, end))$. It can be easily seen that doing this procedure gives us a $(k + 1)$ -NCSL \mathcal{N}' such that q_{in} can cover q_f in \mathcal{N}' iff q_{in} can cover q_f in \mathcal{N} . Hence Theorem 5 follows.

5 Upper bound for coverability of depth-bounded processes

We now prove the upper bound claim made in Theorem 3. Let $\mathcal{P} = (I, \mathcal{E})$ be a fixed k -depth-bounded process. By introducing new identifiers and equations if necessary, we can assume that at most one name or thread is created during a step between two configurations of \mathcal{P} . Let us consider the following order on the set of configurations: $P \preceq Q$ iff $P \equiv (\nu \mathbf{x})P'$ and $Q \equiv (\nu \mathbf{x})(P' \mid R)$ for some term R . It is known that this is a well-quasi order (wqo) for the set of all k -depth-bounded configurations [20, 27]. Using this fact, we can show that the set of k -depth-bounded configurations of \mathcal{P} , forms a well-structured transition system (WSTS) under the \preceq ordering and then apply the generic backward exploration algorithm for

WSTS [13, 25]. Using the standard and generic complexity arguments for WSTS [26, 13, 25], an upper bound on the the running time of this procedure simply boils down to estimating the length of *controlled bad sequences* of k -depth-bounded configurations under the \preceq order.

Let the size of a configuration C be the number of names and threads that appear in C . Let $H : \mathbb{N} \rightarrow \mathbb{N}$ be the successor function and let $n \in \mathbb{N}$. For each $i \in \mathbb{N}$, we let H^i denote the i -fold application of H to itself i times, with H^0 being the identity function.

► **Definition 16.** A sequence C_0, C_1, \dots , of configurations is called (H, n) -controlled bad if the size of each C_i is at most $H^i(n)$ and $C_i \not\preceq C_j$ for any $i < j$.

To estimate an upper bound on the length of controlled bad sequences of configurations, we first recall the *induced subgraph ordering* on bounded-depth trees.

► **Definition 17.** Let $T_1 = (V_1, E_1, L_1)$ and $T_2 = (V_2, E_2, L_2)$ be two labelled trees with labelling functions $L_1 : V_1 \rightarrow A$ and $L_2 : V_2 \rightarrow A$ for some finite set A . We say that T_1 is an induced subgraph of T_2 , if there is a label preserving injection h from V_1 to V_2 such that $(v, v') \in E_1 \iff (h(v), h(v')) \in E_2$.

It is known that for any $K \geq 1$ and for any finite set A , the set of all labelled trees of depth at most K is well-quasi ordered under the induced subgraph relation (Theorem 2.2 of [9]). Similar to configurations, we can also define controlled bad sequences of labelled bounded-depth trees.

By the arguments given in [20], it follows that the length of controlled bad sequences of k -depth-bounded configurations of \mathcal{P} under the \preceq order can be upper bounded by the length of controlled bad sequences of K -bounded-depth trees with labels from a set A , for some A and K whose sizes are primitive recursive in the size of \mathcal{P} . By the known bounds for controlled bad sequences for labelled bounded-depth trees [2, 17], it follows that

► **Theorem 18.** The length of (H, n) -controlled bad sequences for k -depth-bounded configurations of \mathcal{P} is upper bounded by the function $F_{\epsilon_0}(p(|\mathcal{P}|, k, n))$.

Here F_{ϵ_0} is the *fast-growing function* at level ϵ_0 and p is some primitive recursive function. For our purposes, we do not need the actual definition of F_{ϵ_0} , but we only need to know that \mathbf{F}_{ϵ_0} consists of problems whose running time is upper bounded by the function F_{ϵ_0} composed with any primitive recursive function (See [24]). It follows that,

► **Theorem 19.** The coverability problem for depth-bounded processes is in \mathbf{F}_{ϵ_0} and hence \mathbf{F}_{ϵ_0} -complete.

6 Conclusion

We have shown that the coverability problem for depth-bounded processes in π -calculus is \mathbf{F}_{ϵ_0} -complete. This settles the complexity of the problem and solves an open problem raised in [17] and also in [27]. However, our proof does not give any results regarding the *parameterized complexity* of this problem when the depth k is taken as a parameter, which we plan to investigate as part of future work.

References

- 1 Sergio Abriola, Santiago Figueira, and Gabriel Senno. Linearizing well quasi-orders and bounding the length of bad sequences. *Theor. Comput. Sci.*, 603:3–22, 2015. doi:10.1016/j.tcs.2015.07.012.

- 2 A. R. Balasubramanian. Complexity of coverability in bounded path broadcast networks. In Mikolaj Bojanczyk and Chandra Chekuri, editors, *41st IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2021, December 15-17, 2021, Virtual Conference*, volume 213 of *LIPICs*, pages 35:1–35:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICs.FSTTCS.2021.35.
- 3 Michael Blondin. The ABCs of petri net reachability relaxations. *ACM SIGLOG News*, 7(3):29–43, 2020. doi:10.1145/3436980.3436984.
- 4 Michael Blondin, Alain Finkel, Christoph Haase, and Serge Haddad. The logical view on continuous Petri nets. *ACM Trans. Comput. Log.*, 18(3):24:1–24:28, 2017. doi:10.1145/3105908.
- 5 Michael Blondin and Christoph Haase. Logics for continuous reachability in Petri nets and vector addition systems with states. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017*, pages 1–12. IEEE Computer Society, 2017. doi:10.1109/LICS.2017.8005068.
- 6 Pierre Chambart and Philippe Schnoebelen. The ordinal recursive complexity of lossy channel systems. In *Proceedings of the Twenty-Third Annual IEEE Symposium on Logic in Computer Science, LICS 2008, 24-27 June 2008, Pittsburgh, PA, USA*, pages 205–216, 2008. doi:10.1109/LICS.2008.47.
- 7 Wojciech Czerwinski and Lukasz Orlikowski. Reachability in vector addition systems is Ackermann-complete. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 1229–1240. IEEE, 2021. doi:10.1109/FOCS52979.2021.00120.
- 8 Normann Decker and Daniel Thoma. On freeze LTL with ordered attributes. In Bart Jacobs and Christof Löding, editors, *Foundations of Software Science and Computation Structures - 19th International Conference, FOSSACS 2016, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 2-8, 2016, Proceedings*, volume 9634 of *Lecture Notes in Computer Science*, pages 269–284. Springer, 2016. doi:10.1007/978-3-662-49630-5_16.
- 9 Guoli Ding. Subgraphs and well-quasi-ordering. *Journal of Graph Theory*, 16(5):489–502, 1992. doi:10.1002/jgt.3190160509.
- 10 Jacob Elgaard, Nils Klarlund, and Anders Møller. MONA 1.x: New techniques for WS1S and WS2S. In Alan J. Hu and Moshe Y. Vardi, editors, *Computer Aided Verification*, pages 516–520, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.
- 11 Javier Esparza. Keeping a crowd safe: On the complexity of parameterized verification (invited talk). In Ernst W. Mayr and Natacha Portier, editors, *31st International Symposium on Theoretical Aspects of Computer Science (STACS 2014), STACS 2014, March 5-8, 2014, Lyon, France*, volume 25 of *LIPICs*, pages 1–10. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2014. doi:10.4230/LIPICs.STACS.2014.1.
- 12 Javier Esparza, Ruslán Ledesma-Garza, Rupak Majumdar, Philipp J. Meyer, and Filip Nikišić. An SMT-based approach to coverability analysis. In Armin Biere and Roderick Bloem, editors, *Computer Aided Verification - 26th International Conference, CAV 2014, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 18-22, 2014. Proceedings*, volume 8559 of *Lecture Notes in Computer Science*, pages 603–619. Springer, 2014. doi:10.1007/978-3-319-08867-9_40.
- 13 Diego Figueira, Santiago Figueira, Sylvain Schmitz, and Philippe Schnoebelen. Ackermannian and primitive-recursive bounds with Dickson’s lemma. In *Proceedings of the 26th Annual IEEE Symposium on Logic in Computer Science*, pages 269–278, 2011. doi:10.1109/LICS.2011.39.
- 14 Alain Finkel and Philippe Schnoebelen. Well-structured transition systems everywhere! *Theor. Comput. Sci.*, 256(1-2):63–92, 2001. doi:10.1016/S0304-3975(00)00102-X.
- 15 Estibaliz Fraca and Serge Haddad. Complexity analysis of continuous Petri nets. *Fundam. Informaticae*, 137(1):1–28, 2015. doi:10.3233/FI-2015-1168.

- 16 Christoph Haase and Simon Halfon. Integer vector addition systems with states. In Joël Ouaknine, Igor Potapov, and James Worrell, editors, *Reachability Problems - 8th International Workshop, RP 2014, Oxford, UK, September 22-24, 2014. Proceedings*, volume 8762 of *Lecture Notes in Computer Science*, pages 112–124. Springer, 2014. doi:10.1007/978-3-319-11439-2_9.
- 17 Christoph Haase, Sylvain Schmitz, and Philippe Schnoebelen. The power of priority channel systems. *Log. Methods Comput. Sci.*, 10(4), 2014. doi:10.2168/LMCS-10(4:4)2014.
- 18 Sławomir Lasota. Improved Ackermannian lower bound for the petri nets reachability problem. In Petra Berenbrink and Benjamin Monmege, editors, *39th International Symposium on Theoretical Aspects of Computer Science, STACS 2022, March 15-18, 2022, Marseille, France (Virtual Conference)*, volume 219 of *LIPICs*, pages 46:1–46:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICs.STACS.2022.46.
- 19 Jérôme Leroux. The reachability problem for petri nets is not primitive recursive. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 1241–1252. IEEE, 2021. doi:10.1109/FOCS52979.2021.00121.
- 20 Roland Meyer. On boundedness in depth in the pi-calculus. In Giorgio Ausiello, Juhani Karhumäki, Giancarlo Mauri, and C.-H. Luke Ong, editors, *Fifth IFIP International Conference On Theoretical Computer Science – TCS 2008, IFIP 20th World Computer Congress, TC 1, Foundations of Computer Science, September 7-10, 2008, Milano, Italy*, volume 273 of *IFIP*, pages 477–489. Springer, 2008. doi:10.1007/978-0-387-09680-3_32.
- 21 Robin Milner, Joachim Parrow, and David Walker. A calculus of mobile processes, I. *Inf. Comput.*, 100(1):1–40, 1992. doi:10.1016/0890-5401(92)90008-4.
- 22 Robin Milner, Joachim Parrow, and David Walker. A calculus of mobile processes, II. *Inf. Comput.*, 100(1):41–77, 1992. doi:10.1016/0890-5401(92)90009-5.
- 23 Sylvain Schmitz. Complexity bounds for ordinal-based termination - (invited talk). In *Reachability Problems - 8th International Workshop, RP 2014*, pages 1–19, 2014. doi:10.1007/978-3-319-11439-2_1.
- 24 Sylvain Schmitz. Complexity hierarchies beyond elementary. *ACM Trans. Comput. Theory*, 8(1):3:1–3:36, 2016. doi:10.1145/2858784.
- 25 Sylvain Schmitz and Philippe Schnoebelen. Multiply-recursive upper bounds with Higman’s lemma. In *Automata, Languages and Programming – 38th International Colloquium, ICALP 2011*, pages 441–452, 2011. doi:10.1007/978-3-642-22012-8_35.
- 26 Sylvain Schmitz and Philippe Schnoebelen. The power of well-structured systems. In Pedro R. D’Argenio and Hernán C. Melgratti, editors, *CONCUR 2013 – Concurrency Theory – 24th International Conference, CONCUR 2013, Buenos Aires, Argentina, August 27-30, 2013. Proceedings*, volume 8052 of *Lecture Notes in Computer Science*, pages 5–24. Springer, 2013. doi:10.1007/978-3-642-40184-8_2.
- 27 Thomas Wies, Damien Zufferey, and Thomas A. Henzinger. Forward analysis of depth-bounded processes. In C.-H. Luke Ong, editor, *Foundations of Software Science and Computational Structures, 13th International Conference, FOSSACS 2010, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2010, Paphos, Cyprus, March 20-28, 2010. Proceedings*, volume 6014 of *Lecture Notes in Computer Science*, pages 94–108. Springer, 2010. doi:10.1007/978-3-642-12032-9_8.

A Appendix

A.1 Proofs for subsection 4.2

► **Lemma 14** (\mathcal{N} simulates \mathcal{N}'). *Suppose $C \xrightarrow{*} C'$ is a path of non-zero length in \mathcal{N}' such that 1) C is a good configuration and 2) in all the configurations between C and C' , the base of the root is not in Q . Then, C' is a good configuration and there is a rule r such that $\mathbb{M}^{-1}(C) \xrightarrow{r} \mathbb{M}^{-1}(C')$.*

Proof. Let $P := C \rightarrow \gamma_0 \rightarrow \gamma_1 \dots \rightarrow \gamma_m \rightarrow C'$ be a path in \mathcal{N}' . We split the proof into various steps.

Step 1. Since C is a good configuration, the only node which is a leader is the root, whose base must belong to Q . By construction of the rules of \mathcal{N}' , this implies that the step $C \rightarrow \gamma_0$ must be of the form $C \xrightarrow{start_0^r} \gamma_0$ for some rule r of \mathcal{N} . Let $r = ((q_0, \dots, q_i), (q'_0, \dots, q'_j))$. This implies that the label of the root in C is $q_0[\top]$ and its label in γ_0 is $\mathbf{rec}^r[\top]$.

Step 2. Now, for each $0 \leq l \leq i$, we state two claims:

- Claim A_l : There is a path $P_l := v_l^0, \dots, v_l^l$ starting at the root in C with labels $q_0[\top], q_1[\perp], \dots, q_l[\perp]$ such that γ_l is the same as C , except now the labels along P_l are $\underbrace{\mathbf{fwd}^r[\perp], \dots, \mathbf{fwd}^r[\perp]}_{l \text{ times}}, \mathbf{rec}^r[\top]$.
- Claim B_l : If $l \neq 0$, then $\gamma_{l-1} \xrightarrow{begin_{l-1}^r} \gamma_l$.

We have already shown that claim A_0 is true in step 1. Now, for each $0 \leq l \leq i-1$, assuming claim A_l is true, we shall prove that claims A_{l+1} and B_{l+1} are true.

Because of claim A_l and because C is a good configuration, it follows that the only node which is a leader in γ_l is v_l^l . Further, the base of v_l^l is \mathbf{rec}^r . By construction of the rules in \mathcal{N}' , this implies that the only rule that can be fired from γ_l is $begin_l^r$. Hence, it must be the case that $\gamma_l \xrightarrow{begin_l^r} \gamma_{l+1}$, proving claim B_{l+1} . Further, since v_l^l is the only node which is a leader, firing this rule transforms the state of v_l^l to $\mathbf{fwd}^r[\perp]$ and transforms the state of a child of v_l^l (say v') from $q_{l+1}[\perp]$ to $\mathbf{rec}^r[\top]$. Taking P_{l+1} to be v_l^0, \dots, v_l^l, v' proves claim A_{l+1} .

In particular claim A_i implies that there is a path $\mathbf{path} := v^0, \dots, v^i$ starting at the root such that γ_i is the same as C , except that the labels of \mathbf{path} in C and γ_i are $q_0[\top], \dots, q_i[\perp]$ and $\underbrace{\mathbf{fwd}^r[\perp], \dots, \mathbf{fwd}^r[\perp]}_{i \text{ times}}, \mathbf{rec}^r[\top]$ respectively.

Step 3. For each $i \leq l \leq j$, we state two claims:

- Claim A_l : γ_l is the same as γ_i , except that \mathbf{path} is extended to include $l-i$ new nodes and the labels along this extended path in γ_l is $\underbrace{\mathbf{fwd}^r[\perp], \dots, \mathbf{fwd}^r[\perp]}_{l \text{ times}}, \mathbf{rec}^r[\top]$.
- Claim B_l : If $i \neq l$, then $\gamma_{l-1} \xrightarrow{begin_{l-1}^r} \gamma_l$.

We have already shown that claim A_i is true in step 2. Similar to the arguments given in step 2, we can prove that these new claims are also true.

Step 4. By claim A_j it follows that there is a path $\mathbf{ext-path} := n^0, \dots, n^j$ starting at the root in γ_j such that the labels along $\mathbf{ext-path}$ is $\underbrace{\mathbf{fwd}^r[\perp], \dots, \mathbf{fwd}^r[\perp]}_{j \text{ times}}, \mathbf{rec}^r[\top]$. Further, n^j is the only node which is a leader in γ_j . Hence, the only rule which can be fired from γ_j is $middle_j^r$ and so we have $\gamma_j \xrightarrow{middle_j^r} \gamma_{j+1}$. Notice that the only change that has occurred because of this step is that the label of n^j has been changed to $\mathbf{fwd}^r[\top]$.

Step 5. For each $1 \leq l \leq j$, we state two claims:

- Claim A'_l : γ_{j+l} is the same as γ_j , except that the labels along **ext-path** in γ_{j+l} is $\underbrace{\text{fwd}^r[\perp], \dots, \text{fwd}^r[\perp]}_{j-l+1 \text{ times}}, \text{fwd}^r[\top], q'_{j-l+2}[\perp], \dots, q'_j[\perp]$.
- Claim B'_l : $\gamma_{j+l} \xrightarrow{\text{end}^r_{j-l}} \gamma_{j+l+1}$.

The proof of this is accomplished by similar arguments as given in step 2.

Step 6. By claim A'_j , it follows that γ_{2j} is the same as γ_j , except that the labels along **ext-path** is now $\text{fwd}^r[\top], q'_1[\perp], \dots, q'_j[\perp]$. It follows that the only rule which can be fired from γ_{2j} is finish^r_0 , and so it follows that $\gamma_{2j} \xrightarrow{\text{finish}^r_0} \gamma_{2j+1}$, where the only difference between γ_{2j+1} and γ_{2j} is that the label of the root in γ_{2j+1} is $q'_0[\top]$. Hence, by assumption of the run P , it follows that $\gamma_{2j+1} = C'$.

By combining the arguments given above, it follows then that γ_{2j+1} is a good configuration and also that $\mathbb{M}^{-1}(C) \xrightarrow{r} \mathbb{M}^{-1}(C')$. ◀

► **Theorem 15.** q_{in} can cover q_f in \mathcal{N} iff $q_{in}[\top]$ can cover $q_f[\top]$ in \mathcal{N}' .

Proof. Suppose q_{in} can cover q_f in \mathcal{N} . Let $C_0 \rightarrow C_1 \rightarrow \dots \rightarrow C_m$ be a run in \mathcal{N} where C_0 is the initial configuration and the root of C_m is q_f . By Lemma 13, it follows that $\mathbb{M}(C_0) \xrightarrow{*} \mathbb{M}(C_1) \xrightarrow{*} \dots \xrightarrow{*} \mathbb{M}(C_m)$ and so $q_{in}[\top]$ can cover $q_f[\top]$ in \mathcal{N}' .

Suppose $C \xrightarrow{*} C'$ is a run in \mathcal{N}' such that C is the (unique good) configuration consisting of the single root vertex labelled by $q_{in}[\top]$ and C' is some configuration where the root is labelled by $q_f[\top]$. We split the run into parts of the form $C = C_0 \xrightarrow{*} C_1 \xrightarrow{*} C_2 \dots \xrightarrow{*} C_m = C'$ such that for each $1 \leq l \leq m$, C_l is the first configuration after C_{l-1} where the base of the root is in Q . By Lemma 14, it follows that each C_l is a good configuration and also that $\mathbb{M}^{-1}(C_0) \rightarrow \mathbb{M}^{-1}(C_1) \rightarrow \dots \rightarrow \mathbb{M}^{-1}(C_m)$. Hence, it follows that q_{in} can cover q_f in \mathcal{N} . ◀

A.2 Proofs for Section 5

We now give a proof of Theorem 19. We recall the backward exploration algorithm for well-structured transition systems (WSTS) here, adapted to the coverability problem for depth-bounded processes. Let $\mathcal{P} = (I, \mathcal{E})$ be a k -depth-bounded process and let P be some k -depth-bounded configuration, which we want to check is coverable in \mathcal{P} . Without loss of generality, we can assume that at most one name or thread is created during a step between two configurations of \mathcal{P} . Let \mathcal{C}_k be the set of all k -depth-bounded configurations.

Given a set S of \mathcal{C}_k we let $\uparrow S := \{\gamma' : \exists \gamma \in S, \gamma \preceq \gamma'\}$. A set S is called *upward-closed* if $S = \uparrow S$. Because \preceq is a wqo and because of the definition of the operational semantics of \mathcal{P} , we have that:

- If S is upward-closed, then there exists a finite set B such that $\uparrow B = S$. Such a B will be called the basis of S .
- If S is upward-closed and if $\text{Pre}(S)$ is the set of all configurations $\gamma' \in \mathcal{C}_k$ such that there is a configuration $\gamma \in S$ with $\gamma' \rightarrow \gamma$, then $S \cup \text{Pre}(S)$ is upward-closed. Moreover, given a basis B of S , we can compute a basis B' of $S \cup \text{Pre}(S)$ such that the size of each configuration in B' is at most one more than the maximum size of any configuration of B .

Hence, by the generic backward exploration algorithm for WSTS [14], we get that the following algorithm terminates and decides coverability: Construct a sequence of finite sets B_0, B_1, \dots , such that each $B_i \subseteq \mathcal{C}_k$, B_0 is simply $\{P\}$ and B_{i+1} is a basis for $\uparrow B_i \cup \text{Pre}(\uparrow B_i)$.

Then find the first m such that $\uparrow B_m = \uparrow B_{m+1}$ and check if there is an initial configuration in $\uparrow B_m$. If it is true, then P is coverable; otherwise P is not coverable.

The running time complexity of the algorithm is mainly dominated by the length of the sequence B_0, B_1, \dots, B_m . Since m is the first index such that $\uparrow B_m = \uparrow B_{m+1}$, we can find a minimal element $\gamma_i \in \uparrow B_{i+1} \setminus \uparrow B_i$ for each $i < m$.

Consider the sequence $\gamma_0, \dots, \gamma_{m-1}$. Notice that $\gamma_i \not\leq \gamma_j$ for any $j > i$ and further the size of each γ_i is at most $H^i(n)$, where H is the successor function and n is the size of P . It follows that $\gamma_0, \dots, \gamma_{m-1}$ is a (H, n) -controlled bad sequence. By the arguments given in [20], it follows that the length of controlled bad sequences of \mathcal{C}_k under the \leq order can be upper bounded by the length of controlled bad sequences of K -bounded-depth trees with labels from a set A , for some A and K whose sizes are primitive recursive in the size of \mathcal{P} . By the known bounds for controlled bad sequences for labelled bounded-depth trees [2, 17], it follows that

► **Theorem 18.** *The length of (H, n) -controlled bad sequences for k -depth-bounded configurations of \mathcal{P} is upper bounded by the function $F_{\epsilon_0}(p(|\mathcal{P}|, k, n))$.*

Here F_{ϵ_0} is the *fast-growing function* at level ϵ_0 and p is some primitive recursive function. For our purposes, we do not need the actual definition of F_{ϵ_0} , but we only need to know that \mathbf{F}_{ϵ_0} consists of problems whose running time is upper bounded by the function F_{ϵ_0} composed with any primitive recursive function (See [24]). Theorem 19 then follows.