Parameterized Broadcast Networks with Registers: from NP to the Frontiers of Decidability*

Lucie Guillou[1], Corto Mascle[2], and Nicolas Waldburger[3](✉)

[1] IRIF, CNRS, Université Paris Cité, Paris, France
guillou@irif.fr
[2] LaBRI, Université de Bordeaux, Bordeaux, France
corto.mascle@labri.fr
[3] IRISA, Université de Rennes, Rennes, France
nicolas.waldburger@irisa.fr

Abstract. We consider the parameterized verification of networks of agents which communicate through unreliable broadcasts. In this model, agents have local registers whose values are unordered and initially distinct and may therefore be thought of as identifiers. When an agent broadcasts a message, it appends to the message the value stored in one of its registers. Upon reception, an agent can store the received value or test it for equality against one of its own registers. We consider the coverability problem, where one asks whether a given state of the system may be reached by at least one agent. We establish that this problem is decidable, although non-primitive recursive. We contrast this with the undecidability of the closely related target problem where all agents must synchronize on a given state. On the other hand, we show that the coverability problem is NP-complete when each agent only has one register.

Keywords: Parameterized verification · Well quasi-orders · Distributed systems

1 Introduction

We consider Broadcast Networks of Register Automata (BNRA), a model for networks of agents communicating by broadcasts. These systems are composed of an arbitrary number of agents whose behavior is specified with a finite automaton. This automaton is equipped with a finite set of private registers that contain values from an infinite unordered set. Initially, registers all contain distinct values, so these values can be used as identifiers. A broadcast message is composed of a symbol from a finite alphabet along with the value of one of the sender's registers. When an agent broadcasts a message, any subset of agents may receive it; this models unreliable systems with unexpected crashes and disconnections. Upon reception, an agent may store the received value or test it for equality with one of its register values. For example, an agent can check that several received messages have the same value.

* Partly supported by ANR project PaVeDyS (ANR-23-CE48-0005).

N. Kobayashi and J. Worrell (Eds.): FoSSaCS 2024, LNCS 14575, pp. 250–270, 2024.
https://doi.org/10.1007/978-3-031-57231-9_12

This model was introduced in [10], as a natural extension of Reconfigurable Broadcast Networks [12]. In [10], the authors established that coverability is undecidable if the agents are allowed to send two values per message. They moreover claimed that, with one value per message, coverability was decidable and PSPACE-complete; however, the proof turned out to be incorrect [22]. As we will see, the complexity of that problem is in fact much higher.

In this paper we establish the decidability of the coverability problem and its completeness for the hyper-Ackermannian complexity class $\mathbf{F}_{\omega^\omega}$, showing that the problem has nonprimitive recursive complexity. The lower bound comes from lossy channel systems, which consist (in their simplest version) of a finite automaton that uses an unreliable FIFO memory from which any letter may be erased at any time [3, 8, 26]. We further establish that our model lies at the frontier of decidability by showing undecidability of the target problem (where all agents must synchronize in a given state). We contrast these results with the NP-completeness of the coverability problem if each agent has only one register.

Related work Broadcast protocols are a widely studied class of systems in which processes are represented by nodes of a graph and can send messages to their neighbors in the graph. There are many versions depending on how one models processes, the communication graph, the shape of messages... A model with a fully connected communication graph and messages ranging over a finite alphabet was presented in [13]. When working with parameterized questions over this model (*i.e.*, working with systems of arbitrary size), many basic problems are undecidable [14]; similar negative results were found for Ad Hoc Networks where the communication graph is fixed but arbitrary [12]. This lead the community to consider Reconfigurable Broadcast Networks (RBN) where a broadcast can be received by an arbitrary subset of agents [12].

Parameterized verification problems over RBN have been the subject of extensive study in recent years, concerning for instance reachability questions [5, 11], liveness [9] or alternative communication assumptions [4]; however, RBN have weak expressivity, in particular because agents are anonymous. In [10], RBN were extended to BNRA, the model studied in this article, by the addition of registers allowing processes to exchange identifiers.

Other approaches exist to define parameterized models with registers [6], such as dynamic register automata in which processes are allowed to spawn other processes with new identifiers and communicate integers values [1]. While basic problems on these models are in general undecidable, some restrictions on communications allow to obtain decidability [2, 20].

Parameterized verification problems often relate to the theory of well quasi-orders and the associated high complexities obtained from bounds on the length of sequences with no increasing pair (see for example [25]). In particular, our model is linked to data nets, a classical model connected to well-quasi-orders. Data nets are Petri nets in which tokens are labeled with natural numbers and can exchange and compare their labels using inequality tests [18]; in this model, the coverability problem is $\mathbf{F}_{\omega^{\omega^\omega}}$-complete [15]. When one restricts data nets to only equality tests, the coverability problem becomes $\mathbf{F}_{\omega^\omega}$-complete [21]. Data

nets with equality tests do not subsume BNRA. Indeed, in data nets, each process can only carry one integer at a time, and problems on models of data nets where tokens carry tuples of integers are typically undecidable [17].

Overview We start with the model definition and some preliminary results in Section 2. As our decidability proof is quite technical, we start by proving decidability of the coverability problem in a subcase called *signature protocols* in Section 3. We then rely on the intuitions built in that subcase to generalize the proof to the general case in Section 4. We also show the undecidability of the closely-related target problem. Finally, we prove the NP-completeness of the coverability problem for protocols with one register in Section 5. Due to space constraints, a lot of proofs, as well as some technical definitions, are only sketched in this version. Detailed proofs can be found in the full version, available here.

In this document, each notion is linked to its *definition* using the knowledge package. On electronic devices, clicking on words or symbols allows to access their definitions.

2 Preliminaries

2.1 Definitions of the Model

A *Broadcast Network of Register Automata* (BNRA) [10] is a model describing broadcast networks of agents with local registers. A finite transition system describes the behavior of an agent; an agent can broadcast and receive messages with integer values, store them in local registers and perform (dis)equality tests. There are arbitrarily many agents. When an agent broadcasts a message, every other agent may receive it, but does not have to do so.

Definition 1. *A protocol with r registers is a tuple $\mathcal{P} = (Q, \mathcal{M}, \Delta, q_0)$ with Q a finite set of states, $q_0 \in Q$ an initial state, $\mathcal{M}$ a finite set of message types and $\Delta \subseteq Q \times \mathsf{Op} \times Q$ a finite set of transitions, with operations* $\mathsf{Op} =$

$$\{\boldsymbol{br}(m,i), \boldsymbol{rec}(m,i,*), \boldsymbol{rec}(m,i,\downarrow), \boldsymbol{rec}(m,i,=), \boldsymbol{rec}(m,i,\neq) \mid m \in \mathcal{M}, 1 \leq i \leq r\}.$$

*Label **br** stands for broadcasts and **rec** for receptions. In a reception $\boldsymbol{rec}(m,i,\alpha)$, α is its* action*. The set of actions is* $\mathsf{Actions} := \{=, \neq, \downarrow, *\}$*, where '$=$' is an* equality test*, '$\neq$' is a* disequality test*, '$\downarrow$' is a* store action *and '$*$' is a* dummy action *with no effect. The* size *of $\mathcal{P}$ is $|\mathcal{P}| := |Q| + |\mathcal{M}| + |\Delta| + r$.*

We now define the semantics of those systems. Essentially, we have a finite set of agents with r registers each; all registers initially contain distinct values. A step consists of an agent broadcasting a message that other agents may receive.

Definition 2 (Semantics). *Let $(Q, \mathcal{M}, \Delta, q_0)$ be a protocol with r registers, and $\mathbb{A}$ a finite non-empty set of agents. A configuration over $\mathbb{A}$ is a function*

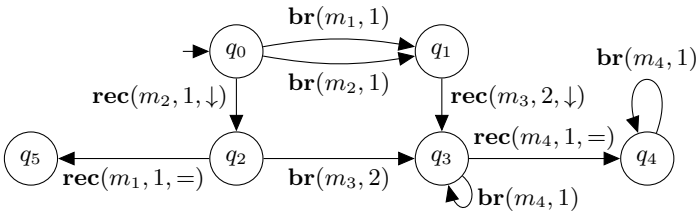

Fig. 1: Example of a protocol.

$\gamma : \mathbb{A} \to Q \times \mathbb{N}^r$ *mapping each agent to its state and its register values. We write* $\mathsf{st}(\gamma)$ *for the state component of* γ *and* $\mathsf{data}(\gamma)$ *for its register component.*

An initial configuration γ *is one where for all* $a \in \mathbb{A}$, $\mathsf{st}(\gamma)(a) = q_0$ *and* $\mathsf{data}(\gamma)(a, i) \neq \mathsf{data}(\gamma)(a', i')$ *for all* $(a, i) \neq (a', i')$.

Given a finite non-empty set of agents $\mathbb{A}$ *and two configurations* γ, γ' *over* $\mathbb{A}$, *a* step $\gamma \to \gamma'$ *is defined when there exist* $m \in \mathcal{M}$, $a_0 \in \mathbb{A}$ *and* $i \in [1, r]$ *such that* $(\mathsf{st}(\gamma)(a_0), \boldsymbol{br}(m, i), \mathsf{st}(\gamma')(a_0)) \in \Delta$, $\mathsf{data}(\gamma)(a_0) = \mathsf{data}(\gamma')(a_0)$ *and, for all* $a \neq a_0$, *either* $\gamma'(a) = \gamma(a)$ *or there exists* $(\mathsf{st}(\gamma)(a), \boldsymbol{rec}(m, j, \alpha), \mathsf{st}(\gamma')(a)) \in \Delta$ *s.t.* $\mathsf{data}(\gamma')(a, j') = \mathsf{data}(\gamma)(a, j')$ *for* $j' \neq j$ *and:*

- *if* $\alpha =$ '$*$' *then* $\mathsf{data}(\gamma')(a, j) = \mathsf{data}(\gamma)(a, j)$,
- *if* $\alpha =$ '$\downarrow$' *then* $\mathsf{data}(\gamma')(a, j) = \mathsf{data}(\gamma)(a_0, i)$,
- *if* $\alpha =$ '$=$' *then* $\mathsf{data}(\gamma')(a, j) = \mathsf{data}(\gamma)(a, j) = \mathsf{data}(\gamma)(a_0, i)$,
- *if* $\alpha =$ '$\neq$' *then* $\mathsf{data}(\gamma')(a, j) = \mathsf{data}(\gamma)(a, j) \neq \mathsf{data}(\gamma)(a_0, i)$.

A run *over* $\mathbb{A}$ *is a sequence of steps* $\rho : \gamma_0 \to \gamma_1 \to \cdots \to \gamma_k$ *with* $\gamma_0, \ldots, \gamma_k$ *configurations over* $\mathbb{A}$. *We write* $\gamma_0 \xrightarrow{*} \gamma_k$ *when there exists such a run. A run is* initial *when* γ_0 *is an initial configuration.*

Remark 3. In our model, agents may only send one value per message. Indeed, coverability is undecidable if agents can broadcast several values at once [10].

Example 4. Figure 1 shows a protocol with 2 registers. Let $\mathbb{A} = \{a_1, a_2\}$. We denote by $\langle \mathsf{st}(\gamma)(a_1), \mathsf{data}(\gamma)(a_1), \mathsf{st}(\gamma)(a_2), \mathsf{data}(\gamma)(a_2) \rangle$ a configuration γ over $\mathbb{A}$. The following sequence is an initial run:

$$\langle q_0, (1,2), q_0, (3,4) \rangle \to \langle q_1, (1,2), q_2, (1,4) \rangle \to \langle q_3, (1,4), q_3, (1,4) \rangle \\ \to \langle q_4, (1,4), q_3, (1,4) \rangle \to \langle q_4, (1,4), q_4, (1,4) \rangle$$

The broadcast messages are, in this order: $(m_2, 1)$ by a_1, $(m_3, 4)$ by a_2, $(m_4, 1)$ by a_2 and $(m_4, 1)$ by a_1. In this run, each broadcast message is received by the other agent; in general, however, this does not have to be true. □

Remark 5. From a run $\rho : \gamma_0 \xrightarrow{*} \gamma$, we can build a larger run ρ' in which, for each agent a of ρ, there are arbitrarily many extra agents in ρ' that end in the same state as a, all with distinct register values. To obtain this, ρ' make many

copies of ρ run in parallel on disjoint sets of agents. Because all these copies of ρ do not interact with one another and because all agents start with distinct values in initial configurations, the different copies of ρ have no register values in common. This property is called *copycat principle*: if state q is coverable, then for all n there exists an augmented run which puts n agents on q.

Definition 6. *The* coverability problem Cover *asks, given a protocol $\mathcal{P}$ and a state q_f, whether there is a finite non-empty set of agents $\mathbb{A}$, an initial run $\gamma_0 \xrightarrow{*} \gamma_f$ over $\mathbb{A}$ that* covers *q_f,* i.e., *there is $a \in \mathbb{A}$ such that $\mathsf{st}(\gamma_f)(a) = q_f$.*

The target problem Target *asks, given a protocol $\mathcal{P}$ and a state q_f, whether there is there is a finite non-empty set of agents $\mathbb{A}$ and an initial run $\gamma_0 \xrightarrow{*} \gamma_f$ over $\mathbb{A}$ such that, for every $a \in \mathbb{A}$, $\mathsf{st}(\gamma_f)(a) = q_f$,* i.e., *all agents end on q_f.*

Example 7. Let $\mathcal{P}$ the protocol of Figure 1. As proven in Example 4, $(\mathcal{P}, q_4)$ is a positive instance of Cover and Target. However, let $\mathcal{P}'$ the protocol obtained from $\mathcal{P}$ by removing the loop on q_4; $(\mathcal{P}', q_4)$ becomes a negative instance of Target. Indeed, there must be an agent staying on q_3 to broadcast m_4. Also, $(\mathcal{P}, q_5)$ is a negative instance of Cover: we would need to be able to have one agent on q_2 and one agent on q_0 with the same value in their first registers. However, an agent in q_0 has performed no transition so it cannot share register values with other agents. □

Remark 8. In [10], the authors consider the *query problem* where one looks for a run reaching a configuration satisfying some *queries*. In fact, this problem exponentially reduces to Cover hence our complexity result of $\mathbf{F}_{\omega^\omega}$ also holds for the query problem. In the case with one register, one can even find a polynomial-time reduction hence our NP result also holds with queries.

We finally introduce *signature BNRA*, an interesting restriction of our model where register 1 is *broadcast-only* and all other registers are *reception-only*. Said otherwise, the first register acts as a permanent identifier with which agents sign their messages. An example of such a protocol is displayed in Fig. 2. Under this restriction, a message is composed of a message type along with the identifier of the sender. This restriction is relevant for pedagogical purposes: we will see that it falls into the same complexity class as the general case but makes the decidability procedure simpler.

Definition 9 (Signature protocols). *A* signature protocol *with r registers is a protocol $\mathcal{P} = (Q, \mathcal{M}, \Delta, q_0)$ where register 1 appears only in broadcasts in Δ and registers $i \geq 2$ appear only in receptions in Δ.*

2.2 Classical Definitions

Fast-growing hierarchy For α an ordinal in Cantor normal form, we denote by $\mathscr{F}_\alpha$ the class of functions corresponding to level α in the Fast-Growing Hierarchy. We denote by $\mathbf{F}_\alpha$ the associated complexity class and use the notion of $\mathbf{F}_\alpha$-completeness. All these notions are defined in [23]. We will specifically work with

complexity class $\mathbf{F}_{\omega^\omega}$. For readers unfamiliar with these notions, $\mathbf{F}_{\omega^\omega}$-complete problems are decidable but with very high complexity (non-primitive recursive, and even much higher than the Ackermann class $\mathbf{F}_\omega$).

We highlight that our main result is the decidability of the problem. We show that the problem lies in $\mathbf{F}_{\omega^\omega}$ because it does not complicate our decidability proof significantly; also, it fits nicely into the landscape of high-complexity problems arising from well quasi-orders.

Well-quasi orders For our decidability result, we rely on the theory of well quasi-orders in the context of subword ordering. Let Σ be a finite alphabet, $w_1, w_2 \in \Sigma^*$, w_1 is a *subword* of w_2, denoted $w_1 \preceq w_2$, when w_1 can be obtained from w_2 by erasing some letters. A sequence of words $w_0, w_1, \ldots$ is *good* if there exist $i < j$ such that $w_i \preceq w_j$, and *bad* otherwise. Higman's lemma [16] states that every bad sequence of words over a finite alphabet is finite, but there is no uniform bound. In order to bound the length of all bad sequences, one must bound the growth of the sequence of words. We will use the following result, known as the Length function theorem [24]:

Theorem 10 (*Length function theorem* [24]). *Let Σ a finite alphabet and $g : \mathbb{N} \to \mathbb{N}$ a primitive recursive function. There exists a function $f \in \mathscr{F}_{\omega^{|\Sigma|-1}}$ such that, for all $n \in \mathbb{N}$, every bad sequence $w_1, w_2, \ldots$ such that $|w_i| \leq g^{(i)}(n)$ for all i has at most $f(n)$ terms (where $g^{(i)}$ denotes g applied i times).*

2.3 A Complexity Lower Bound for COVER Using LCS

Lossy channel systems (LCS) are systems where finite-state processes communicate by sending messages from a finite alphabet through lossy FIFO channels. Unlike in the non-lossy case [7], reachability of a state is decidable for lossy channel systems [3], but has non-primitive recursive complexity [26] and is in fact $\mathbf{F}_{\omega^\omega}$-complete [8]. By simulating LCS using BNRA, we obtain our $\mathbf{F}_{\omega^\omega}$ lower bound for the coverability problem:

Proposition 11. COVER *for signature BNRA is* $\mathbf{F}_{\omega^\omega}$*-hard.*

Proof sketch. Given an LCS $\mathcal{L}$, we build a signature protocol $\mathcal{P}$ with two registers. Each agent starts by receiving a foreign identifier and storing it in its second register; using equality tests, it then only accepts messages with this identifier. Each agent has at most one predecessor, so the communication graph is a forest where messages propagate from roots to leaves. Each branch simulates an execution of $\mathcal{L}$. Each agent of the branch simulates a step of the execution: it receives from its predecessor a configuration of $\mathcal{L}$, chooses the next configuration of $\mathcal{L}$ and broadcasts it, sending first the location of $\mathcal{L}$ and then, letter by letter, the content of the channel. It could be that some messages are not received, hence the lossiness. □

3 Coverability Decidability for Signature Protocols

This section and the next one are dedicated to the proof of our main result:

Theorem 12. COVER *for BNRA is decidable and* $\mathbf{F}_{\omega^\omega}$*-complete.*

For the sake of clarity, in this section, we will first focus on the case of signature BNRA. As a preliminary, we start by defining a notion of local run meant to represent the projection of a run onto a given agent.

3.1 Local runs

A *local configuration* is a pair $(q, \nu) \in Q \times \mathbb{N}^r$. An *internal step* from (q, ν) to (q', ν') with transition $\delta \in \Delta$, denoted $(q, \nu) \xrightarrow{\mathsf{int}(\delta)} (q', \nu')$, is defined when $\nu = \nu'$ and $\delta = (q, \mathbf{br}(m, i), q')$ is a broadcast. A *reception step* from (q, ν) to (q', ν') with transition $\delta \in \Delta$ and value $v \in \mathbb{N}$, denoted $(q, \nu) \xrightarrow{\mathsf{ext}(\delta, v)} (q', \nu')$, is defined when δ is of the form $(q, \mathbf{rec}(m, j, \alpha), q')$ with $\nu(j') = \nu'(j')$ for all $j' \neq j$ and:

- if $\alpha =$ '$*$' then $\nu(j) = \nu'(j)$,
- if $\alpha =$ '$=$' then $\nu(j) = \nu'(j) = v$,
- if $\alpha =$ '$\downarrow$' then $\nu'(j) = v$,
- if $\alpha =$ '$\neq$' then $\nu(j) = \nu'(j) \neq v$.

Such a reception step corresponds to receiving message (m, v); in a local run, one does not specify the origin of a received message. A *local step* $(q, \nu) \to (q', \nu')$ is either a reception step or an internal step. A *local run* u is a sequence of local steps denoted $(q_0, \nu_0) \xrightarrow{*} (q, \nu)$. Its *length* $|u|$ is its number of steps.

A value $v \in \mathbb{N}$ appearing in u is *initial* if it appears in ν_0 and *non-initial* otherwise. For $v \in \mathbb{N}$, the v-*input* $\mathsf{In}_v(u)$ (resp. v-*output* $\mathsf{Out}_v(u)$) is the sequence $m_0 \cdots m_\ell \in \mathcal{M}^*$ of message types received (resp. broadcast) with value v in u.

3.2 Unfolding Trees

We first prove decidability of COVER for signature BNRA. Note that, in signature protocols, the initial values of reception-only registers are not relevant as they can never be shared with other agents. We deduce from this idea the following informal observation:

Observation 13 *In signature BNRA, when some agent receives a message, it can compare the value of the message only with the ones of previously received messages,* i.e.*, check whether the sender is the same.*

If we want to turn a local run u of an agent a into an actual run, we must match a's receptions with broadcasts. Because of Observation 13, what matters is not the actual values of the receptions in u but which ones are equal to which. Therefore, for a value v received in u, if $m_1 \dots m_k \in \mathcal{M}^*$ are the message types received in u with value v in this order, it means that to execute u, a need another agent a' to broadcast messages types m_1 to m_k, all with the same value. We describe what an agent needs from other agents as a set of specifications which are words of $\mathcal{M}^*$.

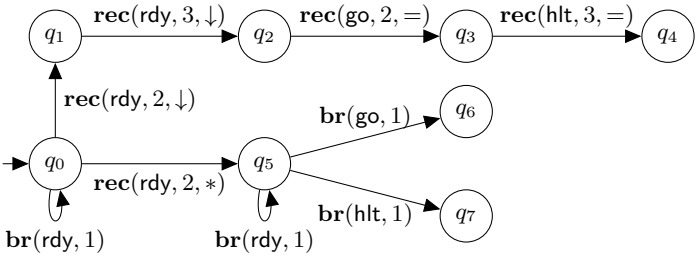

Fig. 2: Example of a signature protocol.

To represent runs, we consider unfolding trees that abstract runs by representing such specifications, dependencies between them and how they are carried out. In this tree, each node is assigned a local run and the specification that it carries out. Because of copycat arguments, we will in fact be able to duplicate agents so that each agent only accomplishes one task, hence the tree structure.

Definition 14. *An unfolding tree τ over $\mathcal{P}$ is a finite tree where nodes μ have three labels:*

- *a local run of $\mathcal{P}$, written $\mathbf{lr}(\mu)$;*
- *a value in $\mathbb{N}$, written $\mathbf{val}(\mu)$;*
- *a specification $\mathbf{spec}(\mu) \in \mathcal{M}^*$.*

Moreover, all nodes μ in τ must satisfy the three following conditions:

(i) Initial values of $\mathbf{lr}(\mu)$ are never received in $\mathbf{lr}(\mu)$,
(ii) $\mathbf{spec}(\mu) \preceq \mathsf{Out}_{\mathbf{val}(\mu)}(\mathbf{lr}(\mu))$, (recall that $\preceq$ denotes the subword relation)
(iii) For each value v received in $\mathbf{lr}(\mu)$, μ has a child μ' s.t. $\mathsf{In}_v(\mathbf{lr}(\mu)) \preceq \mathbf{spec}(\mu')$.

Lastly, given τ an unfolding tree, we define its size by $|\tau| := \sum_{\mu \in \tau} |\mu|$ where $|\mu| := |\mathbf{lr}(\mu)| + |\mathbf{spec}(\mu)|$. Note that the size of τ takes into account the size of its nodes, so that a tree τ can be stored in space polynomial in $|\tau|$ (renaming the values appearing in τ if needed).

We explain this definition. Condition (i) enforces that the local run cannot cheat by receiving its initial values. Condition (ii) expresses that $\mathbf{lr}(\mu)$ broadcasts (at least) the messages of $\mathbf{spec}(\mu)$. We can use the subword relation $\preceq$ (instead of equality) because messages do not have to be received. Condition (iii) expresses that, for each value v received in the local run $\mathbf{lr}(\mu)$, μ has a child who is able to broadcast the sequence of messages that $\mathbf{lr}(\mu)$ receives with value v.

Example 15. Figure 2 provides an example of a signature protocol. Let $\mathbb{A} = \{a_1, a_2, a_3\}$. We denote a configuration γ by $\langle \mathsf{st}(\gamma)(a_1), (\mathsf{data}(\gamma)(a_1)), \mathsf{st}(\gamma)(a_2), (\mathsf{data}(\gamma)(a_2)), \mathsf{st}(\gamma)(a_3), (\mathsf{data}(\gamma)(a_3)) \rangle$. Irrelevant register values are denoted by $\_$. Let ρ be the run over $\mathbb{A}$ of initial configuration $\langle q_0, (1, \_, \_), q_0, (2, \_, \_), q_0, (3, \_, \_) \rangle$ where the following occurs:

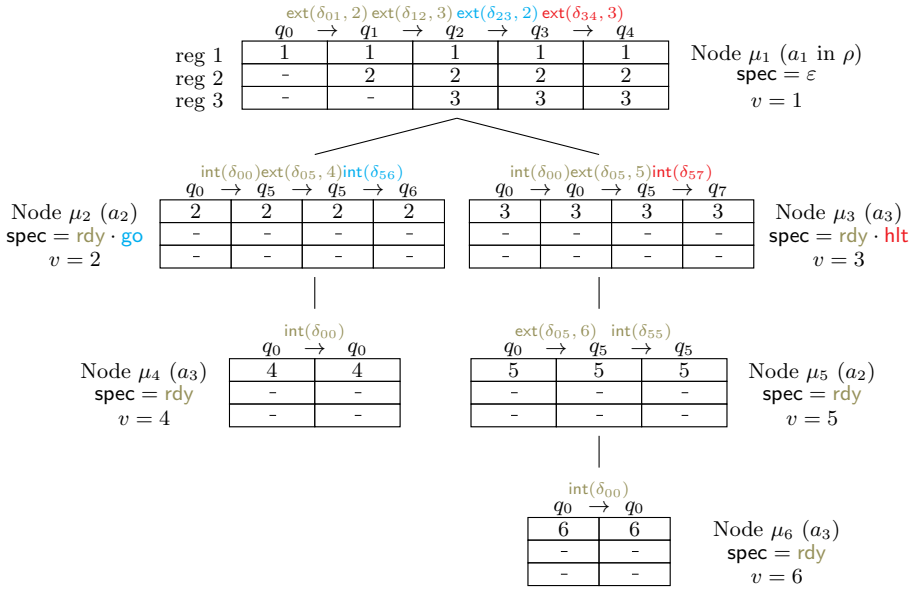

Fig. 3: Example of an unfolding tree derived from ρ. Grids correspond to local runs, a column of a grid is a local configuration. Transition δ_{ij} is the transition between state q_i and state q_j, for example $\delta_{01} = (q_0, \mathbf{rec}(\mathsf{rdy}, 2, \downarrow), q_1)$. If δ is a reception of $m \in \mathcal{M}$, $\mathsf{ext}(\delta, v)$ corresponds to receiving message (m, v); if δ is a broadcast of $m \in \mathcal{M}$, $\mathsf{int}(\delta)$ corresponds to broadcasting (m, id) where id is the value in the first register of the agent. Initial values of reception-only registers are irrevelant and written as '_'. Colors correspond to message types.

- a_2 broadcasts rdy, a_1 receives: $\langle q_1, (1,2,\_), q_0, (2,\_,\_), q_0, (3,\_,\_)\rangle$,
- a_3 broadcasts rdy, a_1 and a_2 receive: $\langle q_2, (1,2,3), q_5, (2,\_,\_), q_0, (3,\_,\_)\rangle$,
- a_2 broadcasts rdy, a_3 receives: $\langle q_2, (1,2,3), q_5, (2,\_,\_), q_5, (3,\_,\_)\rangle$,
- a_2 broadcasts go, a_1 receives: $\langle q_3, (1,2,3), q_6, (2,\_,\_), q_5, (3,\_,\_)\rangle$,
- a_3 broadcasts hlt, a_1 receives: $\langle q_4, (1,2,3), q_6, (2,\_,\_), q_7, (3,\_,\_)\rangle$.

Figure 3 provides an unfolding tree derived from ρ by applying a procedure introduced later. Because agents a_2 and a_3 broadcast to several other agents, they each correspond to several nodes of the tree.

We explain why this tree is an unfolding tree. Condition (i) is trivially satisfied. Condition (ii) holds at every node because the local run of each node exactly broadcasts the specification of the node. Condition (iii) is satisfied at μ_1: $\mathsf{In}_2(\mathbf{lr}(\mu_1)) = \mathsf{rdy} \cdot \mathsf{go} = \mathbf{spec}(\mu_2)$ and $\mathsf{In}_3(\mathbf{lr}(\mu_1)) = \mathsf{rdy} \cdot \mathsf{hlt} = \mathbf{spec}(\mu_3)$. It is also satisfied at μ_2, μ_3 and μ_5 because their local runs only receive rdy and they each have a child with specification rdy. It is trivially satisfied at μ_4 and μ_6 as their local runs have no reception. □

Lemma 16. *Given a signature protocol $\mathcal{P}$ with a state q_f, q_f is coverable in $\mathcal{P}$ if and only if there exists an unfolding tree whose root is labelled by a local run covering q_f. We call such an unfolding tree a* coverability witness.

Proof. Given a run ρ, agent a *satisfies a specification* $w \in \mathcal{M}^*$ in ρ if the sequence of message types broadcast by a admits w as subword.

Let τ be a coverability witness. We prove the following property by strong induction on the depth of μ: for every μ in τ, there exists a run ρ with an agent a whose local run in ρ is $\mathbf{lr}(\mu)$ and who satisfies specification $\mathbf{spec}(\mu)$. This is trivially true for leaves of τ because their local runs have no reception (by condition (iii)) hence are actual runs by themselves. Let μ a node of τ, $u := \mathbf{lr}(\mu)$ and $v_1, \dots, v_c$ the values received in u. These values are non-initial thanks to condition (i); applying condition (iii) gives the existence of corresponding children $\mu_1, \dots, \mu_c$ in τ. We apply the induction hypothesis on the subtrees rooted in $\mu_1, \dots, \mu_c$ to obtain runs $\rho_1, \dots, \rho_c$ satisfying the specifications of the children of μ. Up to renaming agents, we can assume the set of agents of these runs are disjoint; up to renaming values, we can assume that $v_j = \mathbf{val}(\mu_j)$ for all j and that all agents start with distinct values. We build an initial run ρ whose agents is the union of the agents of the c runs along with a fresh agent a. In ρ, we make ρ_1 to ρ_c progress in parallel and make a follow the local run u, matching each reception with value v_j in u with a broadcast in ρ_j. This is possible because, for all j, $\mathsf{In}_{v_j}(u) \preceq \mathbf{spec}(\mu_j) \preceq \mathsf{Out}_{v_j}(\rho_j)$ (by (ii)).

Conversely, we prove the following by induction on the length of ρ: for every initial run ρ, for every agent a in ρ and for every $v \in \mathbb{N}$, there exists an unfolding tree whose root has as local run the projection of ρ onto a and as specification the v-output of a in ρ. If ρ is the empty run, consider the unfolding tree with a single node whose local run and specification are empty. Suppose now that ρ has non-zero length, let a an agent in ρ, $v \in \mathbb{N}$ and let ρ_p the prefix run of ρ of length $|\rho| - 1$. Let τ_1 the unfolding tree obtained by applying the induction hypothesis to ρ_p, a and v, and consider τ_2 obtained by simply appending the last step of a in ρ to the local run at the root of τ_1. If this last step is a broadcast, we obtain an unfolding tree; if the broadcast value is v, we append the broadcast message type to the specification at the root of τ_2 and we are done. Suppose that, in the last step of ρ, a performs a reception $(q, \mathbf{rec}(m, i, \alpha), q')$ of a message (m, v'). We might need to adapt τ_2 to respect condition (iii) at the root. Let a' the agent broadcasting in the last step of ρ. Let τ_3 the unfolding tree obtained by applying the induction to ρ_p, a' and v'. Let τ_4 the unfolding tree obtained by appending the last broadcast to the local run at the root of τ_3 and the corresponding message type to the specification at the root of τ_3. Attaching τ_4 below the root of τ_2 gives an unfolding tree satisfying the desired properties. □

The unfolding tree τ of Figure 3 is built from ρ of Example 15 using the previous procedure. Observe that the unfolding tree τ is a coverability witness for q_4. However, one can find a smaller coverability witness. Indeed, in the right branch of τ, μ_5 and μ_6 have the same specification, therefore μ_5 can be deleted and replaced with μ_6. More generally, we would have also been able to shorten the tree if we had $\mathbf{spec}(\mu_5) \preceq \mathbf{spec}(\mu_6)$.

Remark 17. With the previous notion of coverability witness, the root has to cover q_f but may have an empty specification. However, we will later need the length of the specification of a node to be equal to the number of tasks that it must carry out. For this reason, we will, in the rest of this paper, consider that the roots of coverability witnesses have a specification of length 1. This can be formally achieved by introducing a new message type m_f that may only be broadcast from q_f and require that, at the root, $\mathsf{spec} = m_f$.

3.3 Bounding the Size of a Coverability Witness

In all the following, we fix a positive instance $(\mathcal{P}, q_f)$ of Cover with $r+1$ registers (*i.e.*, r registers used for reception) and a coverability witness τ of minimal size. We turn the observation above into an argument that will be useful towards bounding the length of branches of a coverability witness:

Lemma 18. *If a coverability witness τ for $(\mathcal{P}, q_f)$ of minimal size has two nodes μ, μ' with μ a strict ancestor of μ' then $\mathbf{spec}(\mu)$ cannot be a subword of $\mathbf{spec}(\mu')$.*

Proof. Otherwise, replacing the subtree rooted in μ with the one rooted in μ' would contradict minimality of τ. □

We would now like to use the Length function theorem to bound the height of τ, using the previous lemma. To do so, we need a bound on the size of a node with respect to its depth. The following lemma bounds the number of steps of a local run between two local configurations: we argue that if the local run is long enough we can replace it with a shorter one that can be executed using the same input. This will in turn bound the length of a local run of a node with respect to the size of its specification, which is the first step towards our goal.

Lemma 19. *There exists a primitive recursive function ψ so that, for every local run $u : (q, \nu) \xrightarrow{*} (q', \nu')$, there exists $u' : (q, \nu) \xrightarrow{*} (q', \nu')$ with $|u'| < \psi(|\mathcal{P}|, r)$ and for all value $v' \in \mathbb{N}$, there exists $v \in \mathbb{N}$ such that $\mathsf{In}_{v'}(u') \preceq \mathsf{In}_v(u)$.*

Proof. Let $\psi(n, 0) = n + 1$ and $\psi(n, k+1) = 2\,\psi(n,k) \cdot (|\Delta|^{2\,\psi(n,k)} + 1) + 1$ for all k. Observe that $\psi(n, k)$ is a tower of exponentials of height k, which is primitive-recursive although non-elementary. A register $i \geq 2$ is *active* in a local run u if u has some '$\downarrow$' action on register i. Let u a local run, k the number of active registers in u, $n := |\mathcal{P}|$ and $M := \psi(n, k)$. We prove by induction on the number k of active registers in u that if $|u| \geq \psi(n, k)$ then u can be shortened.

If $k = 0$, any state repetition can be removed. Suppose that $|u| > \psi(n, k+1)$ and that the set I of active registers of u is such that $|I| = k + 1$. If there exists an infix run of u of length M with only k active registers, we shorten u using the induction hypothesis. Otherwise, every sequence of M steps in u has a '$\downarrow$' on every register of I. Because $|u| > 2M\,(|\Delta|^{2M} + 1)$, u contains at least $|\Delta|^{2M} + 1$ disjoint sequences of length $2M$ and some $s \in \Delta^{2M}$ appears twice: in infix run u_1 first, then in infix run u_2. We build a shorter run u' by removing all steps between u_1 and u_2 and merging u_1 and u_2 (see Fig. 4). We need suitable values

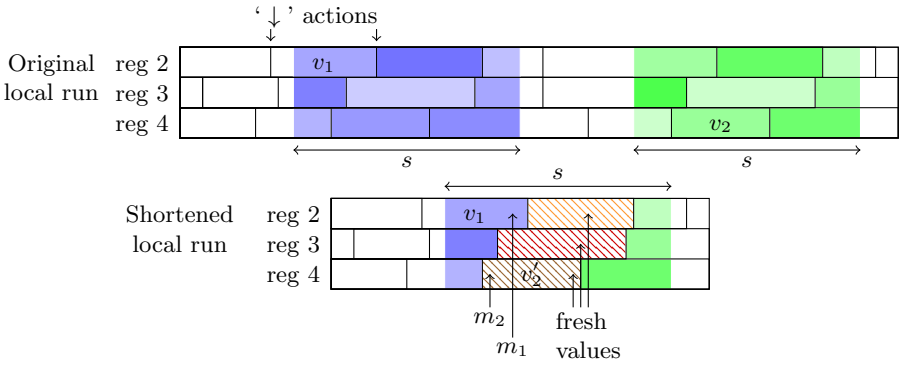

Fig. 4: Illustration of the proof of Lemma 19.

for the reception steps in s in the shortened run u'. For a given register $i \in I$, we would like to pick a ‘ ↓ ’ step on register i in s, use values from u_1 before that step and values from u_2 after that step. This would guarantee that all equality and disequality tests still pass. However, there is an issue if a value v appears in several registers in u. For example, if $v_1 = v_2 = v$ in Figure 4, we might interleave receptions of v on registers 2 and 4: if we had a $\mathsf{ext}(\mathbf{rec}(m_1, 2, =), v)$ in u_1 and a $\mathsf{ext}(\mathbf{rec}(m_2, 4, =), v)$ in u_2, we could have m_1 before m_2 in $\mathsf{In}_v(u)$ but m_1 after m_2 in $\mathsf{In}_v(u')$, so that we do not have $\mathsf{In}_v(u') \preceq \mathsf{In}_v(u)$. We solve this issue by introducing fresh values between values of u_1 and values of u_2; because $|s| = 2M$, there is a ‘ ↓ ’ for each register in I in each half of s. In the shortened run u', before the *first* ‘ ↓ ’ on register i (excluded), we use values of u_1, and after the *last* ‘ ↓ ’ on register i (included), we use values of u_2. For every value v appearing in register i between these two steps in u_1, we select a fresh value v_f (*i.e.*, a value that does not appear anywhere in the run) and consistently replace v with v_f (hatched blocks in Fig. 4). With this technique, receptions with values from u_1 and receptions with values from u_2 cannot get interleaved in u'. Therefore, for every value that appeared in u, we have $\mathsf{In}_v(u') \preceq \mathsf{In}_v(u)$. Also, for every fresh value v' there is a value v such that $\mathsf{In}_{v'}(u') \preceq \mathsf{In}_v(u)$. Moreover, u' is shorter than u; we conclude by iterating this shortening procedure. □

Using the previous lemma, we will bound the size of a node in τ with respect to its specification therefore with respect to its parent's size. By induction, we will then obtain a bound depending on the depth, and apply the Length function theorem to bound the height of the tree.

Lemma 20. *For all nodes μ, μ' in τ:*

1. $|\mathbf{lr}(\mu)| \leq \psi(|\mathcal{P}|, r)\,|\mathbf{spec}(\mu)|$,
2. *if μ is the child of μ',* $|\mathbf{spec}(\mu)| \leq \psi(|\mathcal{P}|, r)\,|\mathbf{spec}(\mu')|$.

Proof. Thanks to Remark 17, we assume that the specification at the root is of length 1. For the first item, by minimality of τ, $\mathbf{lr}(\mu)$ ends with the last broadcast

required by $\mathbf{spec}(\mu)$; we identify in $\mathbf{lr}(\mu)$ the broadcast steps witnessing $\mathbf{spec}(\mu)$ and shorten the local run between these steps using Lemma 19. We thus obtain $|\mathbf{lr}(\mu)| \leq \psi(|\mathcal{P}|, r)\,|\mathbf{spec}(\mu)|$, proving 1. For the second item, by minimality of τ, $|\mathbf{spec}(\mu)| \leq \max_{v\in\mathbb{N}} |\mathsf{ln}_v(\mathbf{lr}(\mu'))| \leq |\mathbf{lr}(\mu')| \leq \psi(|\mathcal{P}|, r)\,|\mathbf{spec}(\mu')|$. □

Proposition 21. *There exists a function f of class $\mathscr{F}_{\omega^{|\mathcal{M}|-1}}$ s.t. $|\tau| \leq f(|\mathcal{P}|)$.*

Proof. Let $n := |\mathcal{P}|$, let $r+1$ be the number of registers in $\mathcal{P}$. Thanks to Lemma 18, for all $\mu \neq \mu'$ in τ with μ ancestor of μ', $\mathbf{spec}(\mu)$ is not a subword of $\mathbf{spec}(\mu')$. Let $\mu_1, \ldots, \mu_m$ the node appearing in a branch of τ, from root to leaf. The sequence $\mathbf{spec}(\mu_1), \ldots, \mathbf{spec}(\mu_m)$ is a bad sequence. For all $i \in [1, m]$, $|\mathbf{spec}(\mu_{i+1})| \leq \psi(n, r)\,|\mathbf{spec}(\mu_i)|$ by Lemma 20. By direct induction, $|\mathbf{spec}(\mu_i)|$ is bounded by $g^{(i)}(n)$ where $g : n \mapsto n\,\psi(n, n)$ is a primitive recursive function. Let h of class $\mathscr{F}_{\omega^{|\mathcal{M}|-1}}$ the function obtained when applying the Length function theorem on g and $\mathcal{M}$; we have $m \leq h(n)$.

By immediate induction, thanks to Lemma 20.2, for every node μ at depth d, $|\mathbf{spec}(\mu)| \leq \psi(n, r)^{d+1}$ which, by Lemma 20.1 and because $d \leq h(n)$, bounds the size of every node by $h'(n) = \psi(n, n)^{h(n)+2}$. By minimality of τ, the number of children of a node is bounded by the number of values appearing in its local run hence by $h'(n)$, so the total number of nodes in τ is bounded by $h'(n)^{h(n)+1}$ and the size of τ by $f(n) := h'(n)^{h(n)+2}$. Because $\mathscr{F}_{\omega^{|\mathcal{M}|-1}}$ is closed under composition with primitive-recursive functions, f is in $\mathscr{F}_{\omega^{|\mathcal{M}|-1}}$. □

The previous argument shows that Cover for signature protocols is decidable and lies in complexity class $\mathbf{F}_{\omega^\omega}$. Because the hardness from Proposition 11 holds for signature protocols, Cover is in fact complete for this complexity class.

We now extend this method to the general case.

4 Coverability Decidability in the General Case

4.1 Generalizing Unfolding Trees

In the general case, a new phenomenon appears: an agent may broadcast a value that it did not initially have but that it has received and stored. In particular, an agent starting with value v could broadcast v then require someone else to make a broadcast with value v as well. For example, in the run described in Example 4, 1 is initially a value of a_1 that a_2 receives and rebroadcasts to a_1.

We now have two types of specifications. *Boss specifications* describe the task of broadcasting with one of its own initial values; this is the specification we had in signature protocols and, as before, it consists of a word $\mathsf{bw} \in \mathcal{M}^*$ describing a sequence of message types that should be all broadcast with the same value. *Follower specifications* describe the task of broadcasting with a non-initial value received previously. More precisely, a follower specification is a pair $(\mathsf{fw}, \mathsf{fm}) \in \mathcal{M}^* \times \mathcal{M}$ asking to broadcast a message (fm, v) under the condition of previously receiving the sequence of message types fw with value v.

A key idea is that, if an agent that had v initially receives some message (m, v), then intuitively we can isolate a subset of agents that did not have v initially but that are able to broadcast (m, v) after receiving a sequence of messages with that value. We can then copy them many times in the spirit of the copycat principle. Each copy receives the necessary sequence of messages in parallel, and they then provide us with an unbounded supply of messages (m, v). In short, if an agent broadcasts (m, v) while not having v as an initial value, then we can consider that we have an unlimited supply of messages (m, v).

Example 22. Assume that $\mathbb{A} = \{a_1, a_2, a_3\}$ and let v be initial for a_1. Consider an execution where the broadcasts with value v are: a_1 broadcasts $\mathsf{a} \cdot \mathsf{b}$, then a_2 broadcasts c, then a_1 broadcasts a^3 then a_3 broadcasts b. The follower specification of a_2's task would be of the form (w, c) where $w \preceq \mathsf{a} \cdot \mathsf{b}$: a_2 must be able to broadcast (c, v) once $\mathsf{a} \cdot \mathsf{b}$ has been broadcast with value v. By contrast, a_3's follower specification would be of the form $(w \cdot w', \mathsf{c})$ where $w \preceq \mathsf{a} \cdot \mathsf{b}$ and $w' \in \{\mathsf{a}, \mathsf{c}\}^*$ is a subword of a^3 enriched with as many c as desired, because a_2 may be cloned at will. For example, one could have $w = \mathsf{b}$ and $w' = \mathsf{c} \cdot \mathsf{a} \cdot \mathsf{c}^4 \cdot \mathsf{a} \cdot \mathsf{c}^2$. This idea is formalized in the full version of the paper with the notion of *decomposition*. Using this notion, the previous condition becomes: $w \cdot w'$ *admits decomposition* $(\mathsf{a} \cdot \mathsf{b}, \mathsf{c}, \mathsf{a}^3)$. □

In our new *unfolding trees*, a node is either a *boss node* or a *follower node*, depending on its type of specification. A *boss node* with a boss specification bw must broadcast that sequence of message types with one of its initial values. A *follower node* μ with follower specification $(\mathsf{fw}, \mathsf{fm})$ is allowed to receive sequence of messages fw with value $\mathbf{val}(\mu)$ (which must be non-initial) without it being broadcast by its children. Other conditions are similar to the ones for signature protocols: if μ is a node and $v \neq \mathbf{val}(\mu)$ a non-initial value received in its local run, μ must have a boss child broadcasting this word. Moreover, for each (m, v) received where v is an initial value of the local run, μ must have a follower child that is able to broadcast (m, v) after receiving messages sent previously with value v; the formal statement is more technical because it takes into account the observation of Example 22. The formal definition of *unfolding tree* is given in the full version.

Example 23. Figure 5 depicts the unfolding tree associated to a_1 in the run of Example 4. Follower node μ_3 can have a m_2 reception that is not matched by its children because m_2 is in $\mathsf{fw}(\mu_3)$. μ_1 broadcasts $(m_2, 1)$ before receiving $(m_4, 1)$ hence the follower specification of μ_3 witnesses broadcast of $(m_4, 1)$. □

A *coverability witness* is again an unfolding tree whose root covers q_f (or broadcasts a message m_f, see Remark 17), with the extra condition that the root is a boss node (a follower node implicitly relies on its parent's ability to broadcast).

Proposition 24. *An instance of* Cover $(\mathcal{P}, q_f)$ *is positive if and only if there exists a coverability witness for that instance.*

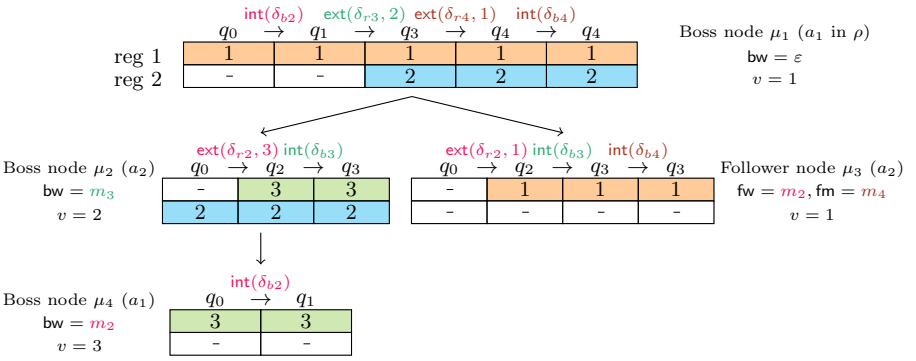

Fig. 5: Example of an unfolding tree. δ_{ri} (resp. δ_{bi}) denotes the reception (resp. broadcast) transition of message m_i in the protocol described in Fig. 1. Values that are never broadcast are omitted and written as '-'.

Proof sketch. The proof is quite similar to the one of Lemma 16, but is made more technical by the addition of follower nodes. When translating an unfolding tree to a run, if the root of the tree is a follower node μ of specification $(\mathsf{fw}, \mathsf{fm})$, then we actually obtain a *partial run*, *i.e.*, a run except that the receptions from fw are not matched by broadcasts in the run. We then combine this partial run with the run corresponding to the parent of μ and with the runs of other children of μ so that every reception is matched with a broadcast. For the translation from run to tree, we inductively construct the tree by extracting from the run the agents and values responsible for satisfying the specifications of each node and analyzing the messages they receive to determine their set of children (as in Example 22). □

Bounding the Size of the Unfolding Tree. Our aim is again to bound the size of a minimal coverability witness. In the following, we fix an instance $(\mathcal{P}, q_f)$ with r registers and a coverability witness of minimal size. We start by providing new conditions under which a branch can be shortened; for boss specifications, it is the condition of Lemma 18 but for follower specifications, the subword relation goes the opposite direction because the shorter the requirement fw, the better.

Lemma 25. *Let $\mu \neq \mu'$ be two nodes of τ such that μ is an ancestor of μ'. If one of those conditions holds, then τ can be shortened (contradicting its minimality):*

- *μ and μ' are boss nodes with boss specifications respectively bw and bw', and $\mathsf{bw} \preceq \mathsf{bw}'$;*
- *μ and μ' are follower nodes with follower specifications respectively $(\mathsf{fw}, \mathsf{fm})$ and $(\mathsf{fw}', \mathsf{fm}')$, and $\mathsf{fw}' \preceq \mathsf{fw}$ and $\mathsf{fm}' = \mathsf{fm}$.*

We can generalize Lemma 19 to bound the size of a node by the number of messages that it must broadcast times a primitive-recursive function $\psi(|\mathcal{P}|, r)$. The proof is more technical than the one of Lemma 19 but the idea is essentially

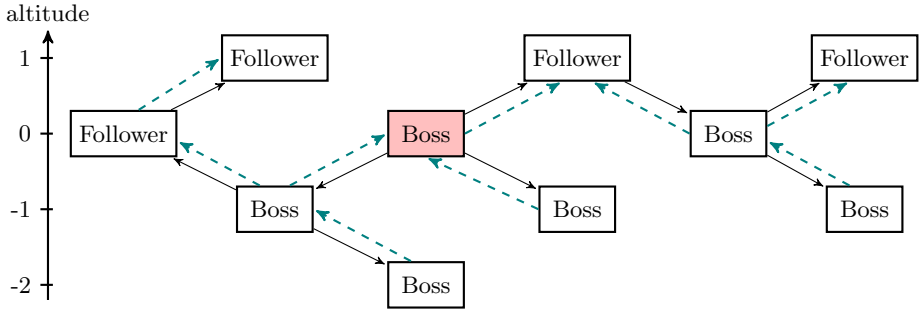

Fig. 6: Rearrangement of a tree. The root is in red, black solid arrows connect parents to children, blue dashed arrows highlight that long words of messages are sent upwards.

the same. The formal statement is given below. One can therefore bound the size of a node with respect to the size of the nodes that it must broadcast to.

Lemma 26. *There exists a primitive recursive function ψ such that, for every protocol $\mathcal{P}$ with r registers, for all local runs $u_0 : (q_0, \nu_0) \xrightarrow{*} (q, \nu)$, $u : (q, \nu) \xrightarrow{*} (q', \nu')$, $u_f : (q', \nu') \xrightarrow{*} (q_f, \nu_f)$, there exists a local run $u' : (q, \nu) \xrightarrow{*} (q', \nu')$ with $|u'| \leq \psi(|\mathcal{P}|, r)$ and for all $v' \in \mathbb{N}$:*

1. *if v' appears in u_0, u, or u_f, $\mathsf{In}_{v'}(u') \preceq \mathsf{In}_{v'}(u)$,*
2. *otherwise, there exists $v \in \mathbb{N}$, not initial in u_0, such that $\mathsf{In}_{v'}(u') \preceq \mathsf{In}_{v}(u)$.*

It is however now much harder than in the signature case to bound the size of the coverability witness. Indeed, the broadcasts no longer go only from children to parents in the unfolding tree. If μ_p is the parent of μ_c, then μ_c broadcasts to μ_p if μ_c is a boss node, but μ_p broadcasts to μ_c if μ_c is a follower node, in which case μ_c only broadcasts one message to μ_p. Therefore, we cannot in general bound $|\mu_p|$ with respect to $|\mu_c|$ nor $|\mu_c|$ with respect to $|\mu_p|$, making us unable to apply the Length function theorem immediately.

This leads us to arrange the unfolding tree so that long broadcast sequences are sent upwards, using the notion of altitude depicted in Figure 6, formally defined as follows. The *altitude* of the root is 0, the altitude of a boss node is the altitude of its parent minus one, and the altitude of a follower node is the altitude of its parent plus one. We denote the altitude of μ by $\mathbf{alt}(\mu)$. This way the nodes of maximal altitude are the ones that do not need to send long sequences of messages. We will bound the size of nodes with respect to their altitude, from the highest to the lowest, and then use the Length function theorem to bound the maximal and minimal altitudes. We present here a sketch of the proof.

Let $\mathbf{altmax} \geq 0$ (resp. $\mathbf{altmin} \leq 0$) denote the maximum (resp. minimum) altitude in τ. We first bound the size of a node with respect to the difference between its altitude and $\mathbf{altmax}$.

Lemma 27. *There is a primitive recursive function f_0 such that, for every node μ of τ, $|\mu| \leq f_0(|\mathcal{P}| + \mathbf{altmax} - \mathbf{alt}(\mu))$.*

Proof sketch. We proceed by induction on the altitude, from highest to lowest. A node of maximal altitude has at most one message to broadcast (a follower node must broadcast one message to its parent), so its size is bounded by $\psi(|\mathcal{P}|, r)$ by Lemma 26 (applying the Lemma to its local run minus its final step, *i.e.*, the step making the broadcast to its parent). Let μ be a node of τ whose neighbors of higher altitude have size bounded by K. We claim that $|\mu| \leq (\psi(|\mathcal{P}|, r) + 2)\,(|\mathcal{M}|\, r\, K + K)$, with ψ the primitive-recursive function defined in Lemma 26. The idea is similar to the one for Lemma 20. The neighbors of higher altitude are the nodes which require sequences of messages from μ. Their size bounds the number of messages that μ needs to send; we then apply Lemma 26 to bound the size of the local run of μ. We finally obtain f_0 by iteratively applying the inequality above. □

We now bound **altmax** and **altmin**:

Lemma 28. **altmax** *and* $|\mathbf{altmin}|$ *are bounded by a function of class* $\mathscr{F}_{\omega^{|\mathcal{M}|}}$.

Proof sketch. We first bound **altmax**. Consider a branch of τ that has a node at altitude **altmax**. We follow this branch from the root to a node of altitude **altmax**: for every $j \in [1, \mathbf{altmax}]$, let μ_j be the first node of the branch that has altitude j. All such nodes are necessarily follower nodes as they are above their parent. Sequence $\mu_{\mathbf{altmax}}, \dots, \mu_2, \mu_1$ is so that the ith term is at altitude $\mathbf{altmax} - i$ hence its size is bounded by $f_0(|\mathcal{P}| + i)$ (Lemma 27). With the observation of Lemma 25, we retrieve from the follower specifications of this sequence of nodes a bad sequence and we apply the Length function theorem to bound **altmax**. This yields in turn a bound on the size of the root of τ. In order to bound **altmin**, we proceed similarly, using boss nodes this time. We follow a branch from the root to a node of altitude **altmin**. The sequence of nodes that are lower than all previous ones yields a sequence of boss specifications, which is a bad sequence by Lemma 25, and whose growth can be bounded using Lemma 27 and the bound on **altmax**. We apply the Length function theorem to bound $|\mathbf{altmin}|$. □

Once we have bounded **altmax** and **altmin**, we can infer a bound on the size of all nodes (Lemma 27), and then on the length of branches: by minimality, a branch cannot have two nodes with the same specification. The bound on the size of the tree then follows from the observation that bounding the size of nodes of τ also allows to bound their number of children.

We obtain a computable bound (of the class $\mathscr{F}_{\omega^\omega}$) on the size of a minimal coverability witness if it exists. Our decidability procedure computes that bound, enumerates all trees of size below the bound and checks for each of them whether it is coverability witness. This yields the main result of this paper:

Theorem 12. Cover *for BNRA is decidable and* $\mathbf{F}_{\omega^\omega}$*-complete.*

4.2 Undecidability of the target problem

A natural next problem, after Cover, is the target problem (Target). Our Cover procedure heavily relies on the ability to add agents at no cost. For Target we need to guarantee that those agents can then reach the target state, which makes the problem harder. In fact, Target is undecidable, which indicates that our model lies at the frontier of decidability.

Proposition 29. Target *is undecidable for BNRA, even with two registers.*

Proof sketch. We simulate a Minsky machine with two counters. As in Proposition 11, each agent starts by storing some other agent's identifier, called its "predecessor". It then only accepts messages from its predecessor. As there are finitely many agents, there is a cycle in the predecessor graph.

In a cycle, we use the fact that *all* agents must reach state q_f to simulate faithfully a run of the machine: agents alternate between receptions and broadcasts so that, in the end, they have received and sent the same number of messages, implying that no message has been lost along the cycle. We then simulate the machine by having an agent (the leader) choose transitions and the other ones simulate the counter values by memorizing a counter (1 or 2) and a binary value (0 or 1). For instance, an increment of counter 1 takes the form of a message propagated in the cycle from the leader until it finds an agent simulating counter 1 and having bit 0. This agent switches to 1 and sends an acknowledgment that propagates back to the leader. □

5 Cover in 1-BNRA

In this section, we establish the NP-completeness of the restriction of Cover to BNRA with one register per agent, called 1-BNRA. Here we simply sketch the key observations that allow us to abstract runs into short witnesses, leading to an NP algorithm for the problem.

In 1-BNRA, thanks to the copycat principle, any message can be broadcast with a fresh value, therefore one can always circumvent '$\neq$' tests. In the end, our main challenge for 1-BNRA is '$=$' tests upon reception. For this reason, we look at clusters of agents that share the value in their registers.

Consider a run in which some agent a reaches some state q,; we can duplicate a many times to have an unlimited supply of agents in state q. Now assume that, at some point in the run, agent a stored a received value. Consider the last storing action performed by a: a was in a state q_1 and performed transition $(q_1, \mathbf{rec}(m, 1, \downarrow), q_2)$ upon reception of a message (m, v). Because we can assume that we have an unlimited supply of agents in q_1 thanks to the copycat principle, we can make as many agents as we want take transition $(q_1, \mathbf{rec}(m, 1, \downarrow), q_2)$ at the same time as a by receiving the same message (m, v). These new agents end up in q_2 with value v, and then follow a along every transition until they all reach q, still with value v. In summary, because a has stored a value in the run, we can have an unlimited supply of agents in state q with the same value as a.

Following those observations, we define an abstract semantics with abstract configurations of the form (S, b, K) with $S, K \subseteq Q$ and $b \in Q \cup \{\bot\}$. The first component S is a set of states that we know we can cover (hence we can assume that there are arbitrarily many agents in all these states). We start with $S = \{q_0\}$ and try to increase it. To do so, we use the two other components (the *gang*) to keep track of the set of agents sharing a value v: b (the *boss*) is the state of the agent which had that value at the start, K (the *clique*) is the set of states covered by other agents with that value. As mentioned above, we may assume that every state of K is filled with as many agents with value v as we need. We will thus define abstract steps which allow to simulate steps of the agents with the value we are following. When they cover states outside of S, we may add those to S and reset b to q_0 and K to $\emptyset$, to then start following another value. We can bound the length of relevant abstract runs, and thus use them as witnesses for our NP upper bound.

The NP lower bound follows from a reduction from 3SAT. An agent a sends a sequence of messages representing a valuation, with its identifier, to other agents who play the role of an external memory by broadcasting back the valuation. This then allows a to check the satisfaction of a 3SAT formula.

Theorem 30. *The coverability problem for 1-BNRA is* NP*-complete.*

6 Conclusion

We established the decidability (and $\mathbf{F}_{\omega^\omega}$-completeness) of the coverability problem for BNRA, as well as the NP-completeness of the problem for 1-BNRA. Concerning future work, one may want to push decidability further, for instance by enriching our protocols with inequality tests, as done in classical models such as data nets [15]. Reductions of other distributed models to this one are also being studied.

Acknowledgements. We are grateful to Arnaud Sangnier for encouraging us to work on BNRA, for the discussions about his work in [10] and for his valuable advice. We also thank Philippe Schnoebelen for the interesting discussion and Sylvain Schmitz for the exchange on complexity class $\mathbf{F}_{\omega^\omega}$ and related topics.

References

1. Abdulla, P.A., Atig, M.F., Kara, A., Rezine, O.: Verification of dynamic register automata. In: 34th International Conference on Foundation of Software Technology and Theoretical Computer Science, FSTTCS 2014. LIPIcs, vol. 29, pp. 653–665. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2014). https://doi.org/10.4230/LIPIcs.FSTTCS.2014.653
2. Abdulla, P.A., Atig, M.F., Kara, A., Rezine, O.: Verification of buffered dynamic register automata. In: Networked Systems, NETYS 2015. Lecture Notes in Computer Science, vol. 9466, pp. 15–31. Springer (2015). https://doi.org/10.1007/978-3-319-26850-7_2

3. Abdulla, P.A., Jonsson, B.: Verifying programs with unreliable channels. Information and Computation **127**(2), 91–101 (1996). https://doi.org/10.1006/inco.1996.0053
4. Balasubramanian, A.R., Bertrand, N., Markey, N.: Parameterized verification of synchronization in constrained reconfigurable broadcast networks. In: Tools and Algorithms for the Construction and Analysis of Systems, TACAS 2018. Lecture Notes in Computer Science, vol. 10806, pp. 38–54. Springer (2018). https://doi.org/10.1007/978-3-319-89963-3_3
5. Balasubramanian, A.R., Guillou, L., Weil-Kennedy, C.: Parameterized analysis of reconfigurable broadcast networks. In: Foundations of Software Science and Computation Structures, FoSSaCS 2022. Lecture Notes in Computer Science, vol. 13242, pp. 61–80. Springer (2022). https://doi.org/10.1007/978-3-030-99253-8_4
6. Bollig, B., Ryabinin, F., Sangnier, A.: Reachability in distributed memory automata. In: Annual Conference on Computer Science Logic, CSL 2021. LIPIcs, vol. 183, pp. 13:1–13:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2021). https://doi.org/10.4230/LIPIcs.CSL.2021.13
7. Brand, D., Zafiropulo, P.: On communicating finite-state machines. Journal of the ACM **30**(2), 323–342 (1983). https://doi.org/10.1145/322374.322380
8. Chambart, P., Schnoebelen, P.: The ordinal recursive complexity of lossy channel systems. In: Annual IEEE Symposium on Logic in Computer Science, LICS 2008. pp. 205–216. IEEE Computer Society (2008). https://doi.org/10.1109/LICS.2008.47
9. Chini, P., Meyer, R., Saivasan, P.: Liveness in broadcast networks. Computing **104**(10), 2203–2223 (2022). https://doi.org/10.1007/s00607-021-00986-y
10. Delzanno, G., Sangnier, A., Traverso, R.: Parameterized verification of broadcast networks of register automata. In: Reachability Problems , RP 2013. Lecture Notes in Computer Science, vol. 8169, pp. 109–121. Springer (2013). https://doi.org/10.1007/978-3-642-41036-9_11
11. Delzanno, G., Sangnier, A., Traverso, R., Zavattaro, G.: On the complexity of parameterized reachability in reconfigurable broadcast networks. In: IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2012. LIPIcs, vol. 18, pp. 289–300. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2012). https://doi.org/10.4230/LIPIcs.FSTTCS.2012.289
12. Delzanno, G., Sangnier, A., Zavattaro, G.: Parameterized verification of ad hoc networks. In: CONCUR 2010. Lecture Notes in Computer Science, vol. 6269, pp. 313–327. Springer (2010). https://doi.org/10.1007/978-3-642-15375-4_22
13. Emerson, E.A., Namjoshi, K.S.: On model checking for non-deterministic infinite-state systems. In: Annual IEEE Symposium on Logic in Computer Science, LICS 1998. pp. 70–80. IEEE Computer Society (1998). https://doi.org/10.1109/LICS.1998.705644
14. Esparza, J., Finkel, A., Mayr, R.: On the verification of broadcast protocols. In: 14th Annual IEEE Symposium on Logic in Computer Science, Trento, Italy, July 2-5, 1999. pp. 352–359. IEEE Computer Society (1999). https://doi.org/10.1109/LICS.1999.782630
15. Haddad, S., Schmitz, S., Schnoebelen, P.: The ordinal-recursive complexity of timed-arc petri nets, data nets, and other enriched nets. In: Proceedings of the 27th Annual IEEE Symposium on Logic in Computer Science, LICS 2012, Dubrovnik, Croatia, June 25-28, 2012. pp. 355–364. IEEE Computer Society (2012). https://doi.org/10.1109/LICS.2012.46

16. Higman, G.: Ordering by divisibility in abstract algebras. Proceedings of the London Mathematical Society **s3-2**(1), 326–336 (1952). https://doi.org/10.1112/plms/s3-2.1.326
17. Lasota, S.: Decidability border for petri nets with data: WQO dichotomy conjecture. In: Kordon, F., Moldt, D. (eds.) Application and Theory of Petri Nets and Concurrency - 37th International Conference, PETRI NETS 2016, Toruń, Poland, June 19-24, 2016. Proceedings. Lecture Notes in Computer Science, vol. 9698, pp. 20–36. Springer (2016). https://doi.org/10.1007/978-3-319-39086-4_3, https://doi.org/10.1007/978-3-319-39086-4_3
18. Lazic, R., Newcomb, T.C., Ouaknine, J., Roscoe, A.W., Worrell, J.: Nets with tokens which carry data. Fundam. Informaticae **88**(3), 251–274 (2008). https://doi.org/10.1007/978-3-540-73094-1_19
19. Minsky, M.L.: Computation: Finite and Infinite Machines. Prentice-Hall, Inc., USA (1967)
20. Rezine, O.: Verification of networks of communicating processes: Reachability problems and decidability issues. Ph.D. thesis, Uppsala University, Sweden (2017)
21. Rosa-Velardo, F.: Ordinal recursive complexity of unordered data nets. Information and Computation **254**, 41–58 (2017). https://doi.org/10.1016/j.ic.2017.02.002
22. Sangnier, A.: Erratum to parameterized verification of broadcast networks of register automata (2023), https://www.irif.fr/~sangnier/publications.html
23. Schmitz, S.: Complexity hierarchies beyond elementary. ACM Transactions on Computation Theory **8**(1), 3:1–3:36 (2016). https://doi.org/10.1145/2858784
24. Schmitz, S., Schnoebelen, P.: Multiply-recursive upper bounds with Higman's lemma. In: International Colloquium on Automata, Languages and Programming, ICALP 2011. Lecture Notes in Computer Science, vol. 6756, pp. 441–452. Springer (2011). https://doi.org/10.1007/978-3-642-22012-8_35
25. Schmitz, S., Schnoebelen, P.: The power of well-structured systems. In: D'Argenio, P.R., Melgratti, H.C. (eds.) CONCUR 2013 - Concurrency Theory - 24th International Conference, CONCUR 2013, Buenos Aires, Argentina, August 27-30, 2013. Proceedings. Lecture Notes in Computer Science, vol. 8052, pp. 5–24. Springer (2013). https://doi.org/10.1007/978-3-642-40184-8_2
26. Schnoebelen, P.: Verifying lossy channel systems has nonprimitive recursive complexity. Information Processing Letters **83**(5), 251–261 (2002). https://doi.org/10.1016/S0020-0190(01)00337-4