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
[2] LaBRI, Université de Bordeaux
[3] IRISA, Université de Rennes

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).

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, most proofs are postponed to the appendix.

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 $\gamma : \mathbb{A} \to Q \times \mathbb{N}^r$ 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.*

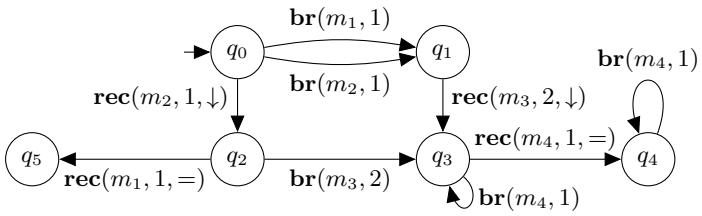


Fig. 1: Example of a protocol.

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 $w_0, w_1, \dots$ 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 $w_1, w_2, \dots$ 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. The full proof can be found in Appendix A. □

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 $\alpha =$ '$\downarrow$' then $\nu'(j) = v$,
- if $\alpha =$ '$=$' then $\nu(j) = \nu'(j) = v$,
- 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 $\mathcal{M}^*$.

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.

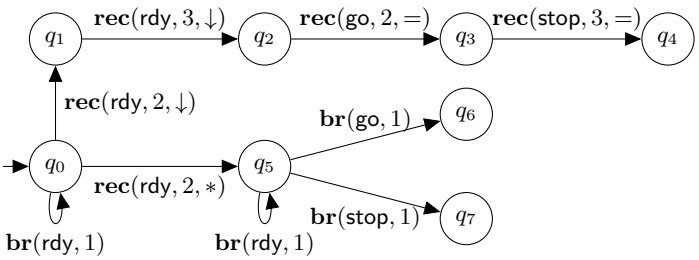


Fig. 2: Example of a signature protocol.

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:

- 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 stop, a_1 receives: $\langle q_4, (1, 2, 3), q_6, (2, \_, \_), q_7, (3, \_, \_)\rangle$.

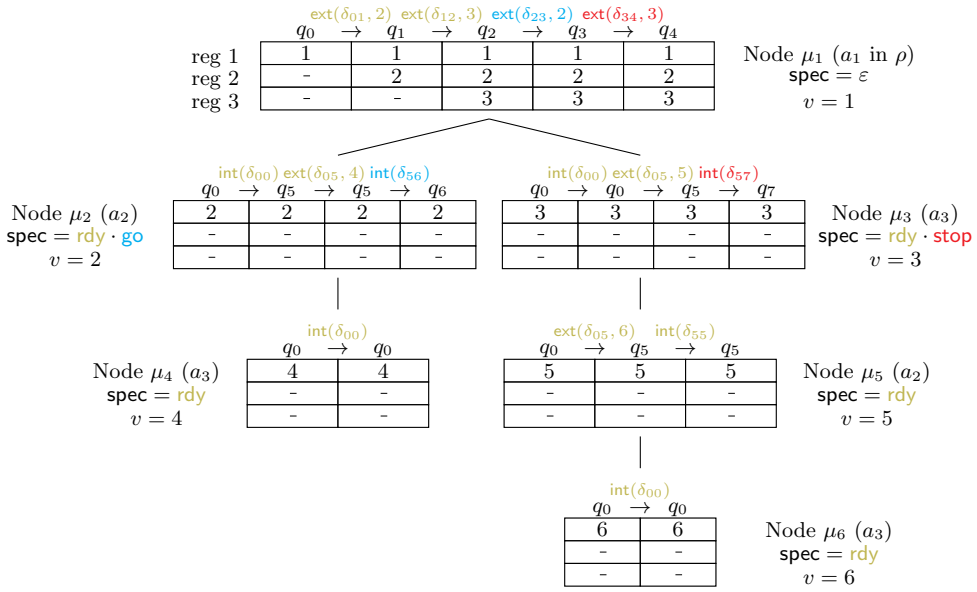


Fig. 3: Example of an unfolding tree derived from ρ. Grids correspond to local runs, a column of a grid is a local configuration. Transition δ_{ij} is the transition between state q_i and state q_j, for example $\delta_{01} = (q_0, \mathbf{rec}(\mathsf{rdy}, 2, \downarrow), q_1)$. If δ is a reception 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.

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 $\mathsf{In}_3(\mathbf{lr}(\mu_1)) = \mathsf{rdy} \cdot \mathsf{stop} = \mathbf{spec}(\mu_3)$. 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 such 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 for the reception steps in s in the shortened run u'. For a given register $i \in I$, we would like to pick a '$\downarrow$' 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

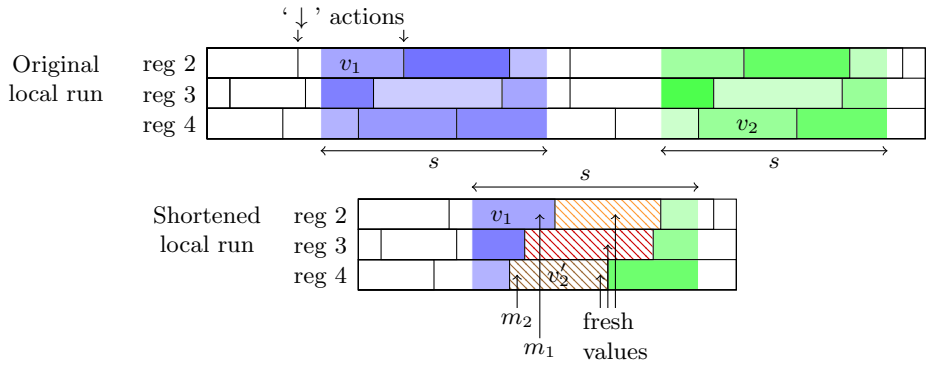


Fig. 4: Illustration of the proof of Lemma 19.

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 appendix 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 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 Appendix B.

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.*

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

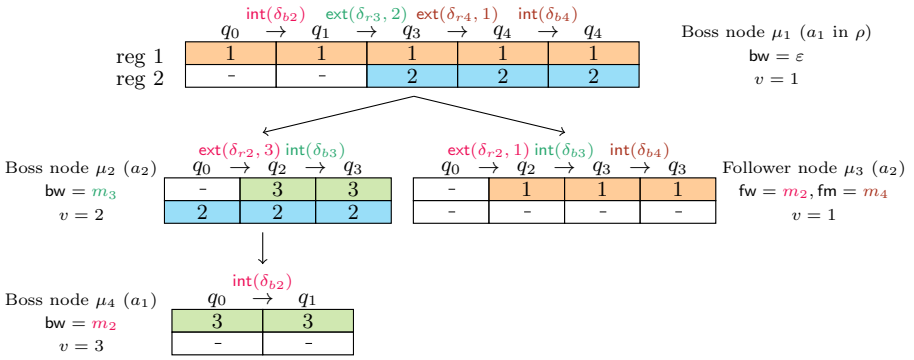


Fig. 5: Example of an unfolding tree. δ_{ri} (resp. δ_{bi}) denotes the reception (resp. broadcast) transition of message m_i in the protocol described in Fig. 1. Values that are never broadcast are omitted and written as '-'.

children of μ so that every reception is matched with a broadcast, as detailed in Lemma 35. 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). See Appendix C for the proof. □

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 and $\mathbf{bw}(\mu) \preceq \mathbf{bw}(\mu')$;*
- *μ and μ' are follower nodes, $\mathbf{fw}(\mu') \preceq \mathbf{fw}(\mu)$ and $\mathbf{fm}(\mu') = \mathbf{fm}(\mu)$.*

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 the same. The formal statement is given below, and the proof can be found in Appendix E. 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 $v' \in \mathbb{N}$:*

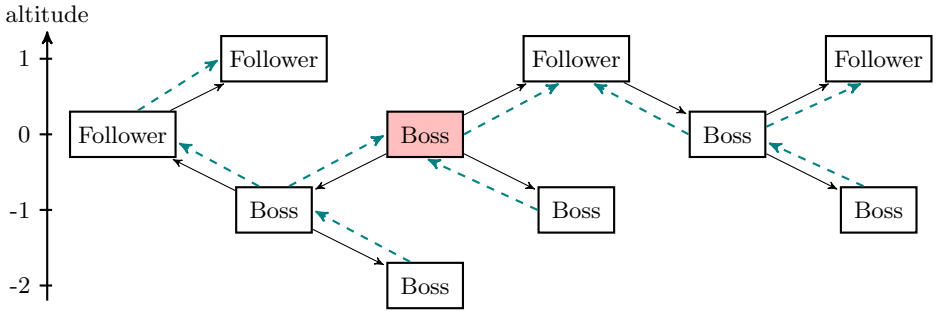


Fig. 6: Rearrangement of a tree. The root is in red, black solid arrows connect parents to children, blue dashed arrows highlight that long words of messages are sent upwards.

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; details are postponed to Appendix F.

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. Lemma 37 in the appendix details these ideas. We finally obtain f_0 by iteratively applying the inequality above. □

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

Lemma 28. **altmax** *and* |**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 (Figure 10 in the appendix). All such nodes are necessarily follower nodes as they are above their parent. Sequence $\mu_{\mathbf{altmax}}, \ldots, \mu_2, \mu_1$ 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 specifications of this sequence of nodes a bad sequence and we apply the Length function theorem to bound **altmax** (Lemma 38 in the appendix). 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 |**altmin**| (Lemma 39). □

Once we have bounded **altmax** and **altmin**, we can infer a bound on the size of all nodes (Lemma 27 in the appendix), 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 Target

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. See Appendix H for the full proof. □

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. Due to space constraints, formal proofs are not included in this section; they can be found in Appendix I. 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

A Proof of Proposition 11

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

Proof. We provide here a polynomial-time reduction from reachability for lossy channel systems with a single channel. A lossy channel system with a single channel is a finite-state machine that has the ability to buffer symbols in a lossy FIFO queue [26]. Let $\mathcal{L} := (L, \Sigma, D)$ be a lossy channel system, where L is a finite set of locations, Σ is a finite set of symbols and $D \subseteq L \times \{\mathsf{read}(x), \mathsf{write}(x) \mid x \in \Sigma\} \times L$; $\mathsf{write}(x)$ corresponds to writing x at the end of the channel and $\mathsf{read}(x)$ to reading x at the beginning the channel. A configuration of $\mathcal{L}$ is a pair in $L \times \Sigma^*$ denoting the location and the content of the channel. There exists a step from (l, w) to (l', w') using $d \in D$, denoted $(l, w) \xrightarrow{d}_{\mathcal{L}} (l', w')$, when

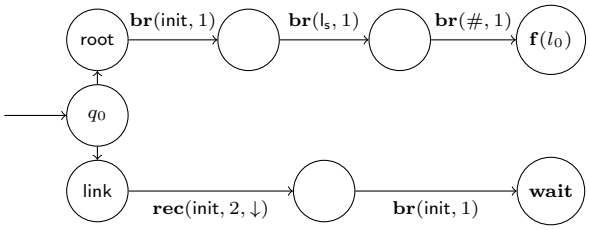
- $d = (l, \mathsf{write}(x), l')$ for some $x \in \Sigma$ and $w' \preceq w \cdot x$,
- $d = (l, \mathsf{read}(x), l')$ for some $x \in \Sigma$ and $x \cdot w' \preceq w$.

where $\preceq$ denotes the subword order, which encodes the lossiness of the channel. The existence of a step is denoted $(l, w) \to_{\mathcal{L}} (l', w')$, and its transitive closure is denoted $\xrightarrow{*}_{\mathcal{L}}$. The (location) *reachability problem* asks, given $\mathcal{L}$ and two locations $l_s, l_f \in L$, whether $(l_s, \varepsilon) \xrightarrow{*}_{\mathcal{L}} (l_f, w)$ for some w.

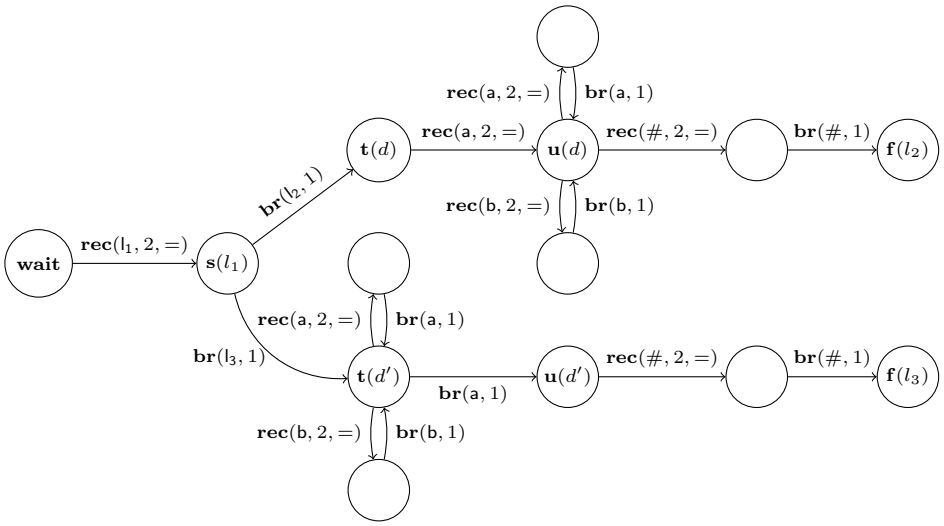
We construct a signature protocol $\mathcal{P}$ with two registers (register 1 is broadcast-only, register 2 is reception-only) and a state q_f that may be covered if and only if $(\mathcal{L}, q_i, q_f)$ is a positive instance of the reachability problem. In some initial phase, each agent may become a *link* and store some other agent's identifier (its *predecessor*); in that case, it will test further messages for equality so that only messages from the predecessor are accepted. Otherwise, it will become a *root* and will not receive any message in the future. A depiction of this initial phase can be found in Figure 7a. The predecessor graph of a run of $\mathcal{P}$, defined by the graph $(\mathbb{A}, E)$ where $(a', a) \in E$ whenever a' is the predecessor of a, will be a forest where each branch will simulate an execution of $\mathcal{L}$. A given agent a of a given branch encodes one step of the execution in the lossy channel system. A root agent is at the root of its tree and simply broadcasts the initial configuration (l_s, ε) of $\mathcal{L}$; special character # encodes the end of the broadcast of the channel's content. If a is a link agent, then it will receive from its predecessor a location of the system and (a subword of) the content of the channel. Agent a will in turn broadcast to the next agent in the branch, sending the new location of the system and the new content of the channel which a rebroadcasts on-the-fly letter by letter as it receives it. Agent a only modifies the beginning of the channel if it decides to encode a pop and the end of the channel if it decides to encode a push. Messages might get lost, which is why we cannot encode non-lossy channel systems.

Observe that this construction only guarantees that agents have at most one predecesor; it does not guarantee that all agents are in the same branch or that any agent is the predecessor of at most one agent. This is fine because the information only propagates forward in a branch, and never propagates in between branches.

From state **wait**, an agents will receive the location l from its predecessor and go to state $\mathbf{s}(l)$; from there, it will non-deterministically pick a transition of $\mathcal{L}$ and apply it. See Figure 7b for a depiction of transitions from some state $\mathbf{s}(l_1)$ (all $\mathbf{s}(l)$, $l \neq l_1$, and corresponding transitions are omitted in the figure). Finally, the objective state of our system is $q_f := \mathbf{f}(l_f)$.



(a) The initial part of $\mathcal{P}$



(b) Part of $\mathcal{P}$ encoding transitions from l_1. Here, $\mathcal{L}$ has symbols $\Sigma = \{\mathsf{a}, \mathsf{b}\}$ and there are two transitions from l_1: $d = (l_1, \mathsf{read}(a), l_2)$ and $d' = (l_1, \mathsf{write}(b), l_3)$.

Fig. 7: Depiction of the protocol $\mathcal{P}$ built in the lossy channel system reduction of Proposition 11.

We claim that $(\mathcal{P}, q_f)$ is a positive instance of Cover if and only if $(\mathcal{L}, l_f)$ is a positive instance of the reachability problem for lossy channel systems. First, suppose that there exists $w \in \Sigma^*$ such that $(l_0, \varepsilon) \xrightarrow{*}_{\mathcal{L}} (l_f, w)$. Decompose the witness into $(l_0, w_0) \to_{\mathcal{L}} (l_1, w_1) \to_{\mathcal{L}} (l_2, w_2) \cdots \to_{\mathcal{L}} (l_n, w_n)$ with $l_n = l_f$ and $w_n = w$. We build an initial run of $\mathcal{P}$ that covers q_f. It has set of agents $\mathbb{A} := \{0, \ldots, n\}$. Agent 0 becomes the root and for all $i \geq 1$, agent i becomes a link

with predecessor agent $i-1$. By induction on i, we build an execution using agents 0 to i such that agent i ends on state $\mathbf{f}(l_i)$ and the sequence of message types sent by agent i admits as subword $\mathsf{init}\cdot\mathsf{l_i}\cdot w_i\cdot\#$. For $i=0$, this condition is easily met by making agent 0 become root. We make agent $i+1$ do the following. It receives from agent i state l_i and goes to $\mathbf{s}(l_i)$. It moves to $\mathbf{t}(d)$ where $d\in D$ is such that $(l_i,w_i)\xrightarrow{d}_{\mathcal{L}}(l_{i+1},w_{i+1})$. It then follows the branch and gets to $\mathbf{f}(l_{i+1})$.

- if $d=(l_i,\mathsf{write}(x),l_{i+1})$ is a push, then $w_{i+1}=w_i'\cdot y$ where $w_i'\preceq w_i$ and $y\in\{\varepsilon,x\}$. We can make agent $i+1$ broadcast $\mathsf{init}\cdot\mathsf{l_i}\cdot w_i'\cdot x\cdot\#$ which admits as subword $\mathsf{init}\cdot\mathsf{l_{i+1}}\cdot w_{i+1}\cdot\#$.
- if $d=(l_i,\mathsf{read}(x),l_{i+1})$ is a pop then $w_i=u\cdot w_{i+1}'$ where $w_{i+1}\preceq w_{i+1}'$ and $x\preceq u$. By lossiness, agent $i+1$ only receives x from u and goes to $\mathbf{u}(d)$. We make it broadcast $\mathsf{init}\cdot\mathsf{l_{i+1}}\cdot w_{i+1}'\cdot\#$ which admits as subword $\mathsf{init}\cdot\mathsf{l_{i+1}}\cdot w_{i+1}\cdot\#$.

This concludes the induction step. When applied to $i=n$, this builds an initial run where agent n ends on $\mathbf{f}(l_n)$, which is a witness that $(\mathcal{P},q_f)$ is positive.

Suppose now that $(\mathcal{P},q_f)$ is positive. Let $\rho:\gamma_0\xrightarrow{*}\gamma_f$ where γ_f covers q_f. The predecessor graph obtained at the end of ρ is a forest; there can be no cycle as it would imply that all agents in the cycle are link, which is a contradiction by considering the agent of the cycle sending the first init message. Consider in ρ a branch of agents $a_0,\dots,a_n$ such that a_i is the predecessor of a_{i+1} for each $i\in[0,n-1]$, a_0 is the root and a_n covers q_f.

From the run ρ projected onto this branch, it is quite simple to build an execution of $\mathcal{L}$ that covers q_f. By structure of the protocol, because a_n covers q_f, every agent a_i ends on some $\mathbf{f}(l_i)$ and broadcasts a word of the form $\mathsf{init}\cdot l_i\cdot w_i\cdot\#$; this can be proven with an immediate backwards induction. It then suffices to analyse the behavior of a_{i+1} to prove that $(l_i,w_i)\to_{\mathcal{L}}(l_{i+1},w_{i+1})$. In particular, because a_0 is a root, $l_0=l_s$ and $w_0=\varepsilon$, which concludes the proof. □

B Definitions and Notations of Section 4

We will first define the notion of decomposition, which is a formalization of the observation in Example 22. A *decomposition* is a tuple $\mathtt{dec}=(w_0,m_1,\dots,m_\ell,w_\ell)$ with $w_0,\dots,w_\ell\in\mathcal{M}^*$, and $m_1,\cdots,m_\ell\in\mathcal{M}$, with $m_i\neq m_j$ for all $i\neq j$. In particular we have $\ell\leq|\mathcal{M}|$. A word $w\in\mathcal{M}^*$ *admits decomposition* $\mathtt{dec}=(w_0,m_1,\dots,m_\ell,w_\ell)$ if $w\preceq w_0'w_1'\cdots w_\ell'$ where for all j, w_j' can be obtained from w_j by adding letters from $\{m_1,\dots,m_j\}$. We denote by $\mathcal{L}^{\mathtt{dec}}$ the language of words that admit decomposition $\mathtt{dec}$.

We are now ready to formally define unfolding trees in the general case.

Definition 31. *An unfolding tree τ over $\mathcal{P}$ is a finite tree where nodes μ have three labels:*

- *a local run of $\mathcal{P}$, written $\mathbf{lr}(\mu)$, starting in the initial state with distinct register values;*
- *a value in $\mathbb{N}$, written $\mathbf{val}(\mu)$;*

– *a specification* $\mathbf{spec}(\mu)$*, which is either a word* $\mathbf{bw}(\mu) \in \mathcal{M}^*$ *(boss specification) or a pair* $(\mathbf{fw}(\mu), \mathbf{fm}(\mu)) \in \mathcal{M}^* \times \mathcal{M}$ *(follower specification). In the first case we say that the node is a boss node, otherwise it is a follower node.*

Moreover, all nodes μ *must satisfy the four following conditions:*

(i) *For each non-initial value* $v \neq \mathbf{val}(\mu)$ *of* $\mathbf{lr}(\mu)$*,* μ *has a child* μ' *which is a boss node such that* $\mathsf{In}_v(\mathbf{lr}(\mu))$ *is a subword of* $\mathbf{bw}(\mu')$*.*

(ii) *For each initial value* v *in* $\mathbf{lr}(\mu)$*, there is a decomposition* $\mathtt{dec} = (w_0, m_1, w_1, \dots, m_\ell, w_\ell)$ *s.t.:*
  – $\mathbf{lr}(\mu)$ *may be split into successive local runs* $u_0, \dots, u_\ell$ *where, for all* $i \in [1, \ell]$*,* $w_i \preceq \mathsf{Out}_v(u_i)$ *and* $\mathsf{In}_v(u_i) \in \{m_1, \dots, m_{i-1}\}^*$*,*
  – *for all* $i \in [1, \ell]$*,* μ *has a child* μ_i *which is a follower node such that* $\mathbf{fm}(\mu_i) = m_i$ *and* $\mathbf{fw}(\mu_i) \in \mathcal{L}^{\mathtt{dec}_i}$ *where* $\mathtt{dec}_i = (w_0, m_1, w_1, \dots, m_{i-1}, w_{i-1})$*.*

(iii) *If* μ *is a follower node then* $\mathbf{val}(\mu)$ *is not an initial value of* u*,* $\mathsf{In}_{\mathbf{val}(\mu)}(u) \preceq \mathbf{fw}(\mu)$ *and* $\mathsf{Out}_{\mathbf{val}(\mu)}(u)$ *contains* $\mathbf{fm}(\mu)$*.*

(iv) *If* μ *is a boss node, then* $\mathbf{val}(\mu)$ *is an initial value of* $\mathbf{lr}(\mu)$ *and the decomposition* $\mathtt{dec}$ *of (ii) for* $\mathbf{val}(\mu)$ *satisfies that* $\mathbf{bw}(\mu) \in \mathcal{L}^{\mathtt{dec}}$*.*

Lastly, as before, given τ *an unfolding tree, we define its size by* $|\tau| := \sum_{\mu \in \tau} |\mathbf{lr}(\mu)| + |\mathbf{spec}(\mu)| + 1$*.*

We now explain this definition. Let μ be a node of an unfolding tree τ and let $u := \mathbf{lr}(\mu)$. As before, u encodes the local run of a given agent, $\mathbf{spec}(\mu)$ encodes the specification that this local run carries out and $\mathbf{val}(\mu)$ encodes the value for which the specification is carried out.

Conditions (i) and (ii) state that the specifications of the children of μ are witnesses that messages received in the local run $\mathbf{lr}(\mu)$ can be broadcast by other agents. Conditions (iii) and (iv) state that μ is a witness that its specification is carried out.

As before, condition (i) expresses that, for every non-initial value v of u, μ must have a boss child witnessing that $\mathsf{In}_v(u)$ can indeed be broadcast. Because v was first stored by a reception step of u, any other (fresh) value with sequence of message types containing $\mathsf{In}_v(u)$ also works and we do not impose the value label of this child to be v.

We now explain condition (ii), which states the existence of a decomposition for each initial value, which serves as a summary of the broadcasts made with that value. Let v be an initial value of u. Consider a run where u is the local run of agent a. If another agent broadcasts with value v, it has first received and stored v. By duplicating agents, we may afterwards assume that we have an unlimited supply of messages (m, v). Therefore the crucial information is the times at which each message type is first broadcast with v by an agent other than a.

The decomposition $\mathtt{dec} = (w_0, m_1, w_1, \dots, m_\ell, w_\ell)$ should be understood as follows: In the run we are representing, a first broadcasts messages of w_0 with value v, after which other agents are able to broadcast (m_1, v). Then a broadcasts w_1 (and may receive m_1), after which other agents can broadcast (m_2, v), and

so on. To sustain that description, we need to be able to split u into $u_0, \dots, u_\ell$, with each u_i broadcasting w_i and only receiving messages $m_1, \dots, m_{i-1}$ over value v. We also need a follower child for each m_i to witness that other agents may broadcast it. For every i, the sequence of messages available with value v during u_i is $\mathsf{Out}_v(u_i)$ expanded by freely adding symbols from $\{m_1, \dots, m_{i-1}\}$. Therefore, the follower child μ_i responsible for the broadcast of (m_i, v) may first receive with value v a subword of $w'_0 \cdot w'_1 \cdots w'_{i-1}$ where, for all $j \leq i-1$, w_j is obtained from $\mathsf{Out}_v(u_i)$ by adding symbols from $\{m_1, \dots, m_{j-1}\}$, which we state as $\mathbf{fw}(\mu_i) \in \mathcal{L}^{\mathtt{dec}_i}$.

Condition (iii) directly states that a follower node μ receives word $\mathbf{fw}(\mu)$ with value $\mathbf{val}(\mu)$ and broadcasts message $(\mathbf{fm}(\mu), \mathbf{val}(\mu))$. Condition (iv) expresses that a boss node witnesses the broadcast of a sequence of messages $\mathbf{bw}(\mu)$ with a single value; whereas in the signature protocol case, in this sequence, some messages may come from auxiliary agents encoded in follower children, which is why we have the condition that $\mathbf{bw}(\mu) \in \mathcal{L}^{\mathtt{dec}}$ and not simply $\mathsf{Out}_{\mathbf{val}(\mu)}(u) \preceq \mathbf{bw}(\mu)$.

C Proof of Proposition 24

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

We make the meaning of the specification labels more concrete by defining the criteria for an initial run to satisfy a specification.

- A run ρ satisfies a boss specification bw if there exists $v \in \mathbb{N}$ such that bw is a subword of the sequence of messages sent with value v in ρ.
- A run ρ satisfies a follower specification $(\mathsf{fw}, \mathsf{fm})$ if there exist a value v and an agent a such that v is not an initial value of a, the v-input of a in ρ is a subword of fw and agent a broadcasts fm with value v at some point.

For an unfolding tree, satisfying a specification simply means that its root is labeled with that specification or a better one.

- An unfolding tree satisfies a boss specification bw if its root μ is a boss node and bw is a subword of its specification label $\mathbf{bw}(\mu)$.
- An unfolding tree satisfies a follower specification $(\mathsf{fw}, \mathsf{fm})$ if its root μ is a follower node such that $\mathsf{fm} = \mathbf{fm}(\mu)$ and $\mathbf{fw}(\mu)$ is a subword of fw.

Note that Cover reduces to the existence of an initial run satisfying a boss specification: it suffices to consider $\mathcal{M}' := \mathcal{M} \cup \{m_f\}$ with $m_f \notin \mathcal{M}$ and to add a loop on q_f broadcasting m_f. Therefore, it suffices to prove that initial runs of $\mathcal{P}$ and unfolding trees satisfy the same boss specifications. We prove the two directions in the two following sections: we construct an unfolding tree from a run in Section C.1 and a run from an unfolding tree in Section C.2.

C.1 Construction of an unfolding tree from an initial run

Lemma 32. *If there exists an initial run ρ of $\mathcal{P}$ satisfying some specification* spec *then there exists a finite unfolding tree τ over $\mathcal{P}$ satisfying* spec.

Proof. We proceed by strong induction on $\mathbb{N} \times \{\text{follower}, \text{boss}\}$ ordered by lexicographic order (follower being lower than boss). The first component is the length of the run and the second component is the type of specification.This means that, for a fixed run length, we prove our property for boss specifications then for follower specifications.

If ρ is of length 0, then its specification must be an empty boss specification and the tree with a single node labeled with an empty local run and an empty specification satisfies it.

Let ρ be an initial run, and assume that the property is true for initial runs whose length is less than the one of ρ. Write $\mathbb{A}$ the set of agents of ρ. Because ρ satisfies spec, there exist a value v and an agent a in ρ such that:

- if spec is a boss specification bw, bw is a subword of the sequence of message types sent with value v in ρ and a is the agent which has v as an initial value,
- if spec is a follower specification $(\mathsf{fw}, \mathsf{fm})$, then a is an agent whose v-input is a subword of fw and that broadcasts fm with value v.

If the last step of u does not include a broadcast with value v, then we remove it and apply the induction hypothesis. Assume that the last step of ρ involves a broadcast with value v. Let u the local run of a in ρ. We set the root μ of our tree τ to have local run u, value v and specification spec as labels, and attach subtrees to it so that it forms an unfolding tree.

We do the following for every value v' in u. If v' is non-initial in u and $v' \neq v$, because the last step of ρ is a broadcast with value v, we apply the induction hypothesis on ρ without its last step to obtain an unfolding tree τ' whose root is a boss node with boss specification $\mathsf{In}_{v'}(u)$, and we attach τ' below μ in τ. If $v' = v$ and v' is non-initial then spec is a follower specification and we do not need to add any children, as the v'-input of u is covered by the specification. Assume now that v' is initial in u. We split ρ according to the first broadcast of each message type by agents other than a; we obtain $m_1, \ldots, m_\ell \in \mathcal{M}$ distinct message types such that $\rho = \rho_0 \cdot \rho_1 \cdot \ldots \cdot \rho_\ell$ where, for all $i \in [1, l]$:

- agents from $\mathbb{A} \setminus \{a\}$ do not broadcast m_i with value v' in $\rho_0 \cdot \ldots \cdot \rho_{i-1}$,
- ρ_i starts with a broadcast of m_i with value v' by some agent $a_i \in \mathbb{A} \setminus \{a\}$.

Let w_i the sequence broadcast by a with value v' in ρ_i. This forms a decomposition $\mathtt{dec} := (w_0, m_1, w_1, \ldots, m_\ell, w_\ell)$. For all $i \in [1, \ell]$, we write $\mathtt{dec}_i$ for the decomposition $\mathtt{dec}_i = (w_0, m_1, w_1, \ldots, m_{i-1}, w_{i-1})$. For every $i \in [1, \ell - 1]$, let $\tilde{\rho}_i$ the run $\rho_0 \cdot \ldots \cdot \rho_{i-1}$ plus the first step of ρ_i (where (m_i, v') is broadcast). Let w'_i the v'-input of a_i in $\tilde{\rho}_i$: it is in $\mathcal{L}^{\mathtt{dec}_i}$ by construction of $\mathtt{dec}$. Agent a_i is a witness that $\tilde{\rho}_i$ satisfies follower specification (w'_i, m_i). If $\tilde{\rho}_i$ is smaller than ρ, we apply the induction hypothesis and obtain an unfolding tree τ_i satisfying

follower specification (w'_i, m_i). If $\tilde{\rho}_i = \rho$, then $i = \ell$, $v' = v$ (ρ ends with a broadcast with value v) and ρ ends with a broadcast (m_ℓ, v) by agent a_ℓ. Moreover, since $v' = v$, v is initial for a and spec is a boss specification bw. Because in our induction order boss specifications are above follower specifications, we can apply the induction hypothesis and obtain an unfolding tree τ_ℓ satisfying follower specification (w'_ℓ, m_ℓ). We have obtained, for every $i \in [1, \ell]$, an unfolding tree τ_i satisfying follower specification (w'_i, m_i), which we attach below μ.

Overall we have attached one boss node below μ for each non-initial value in u (except v if μ is a follower node), thanks to which (i) is satisfied at μ, and there are well-chosen follower nodes below μ for each initial value in u, thanks to which (ii) is satisfied as well. Conditions (iii) and (iv) hold because ρ satisfies spec for agent a. We have therefore built an unfolding tree τ satisfying spec. □

C.2 Construction of an initial run from an unfolding tree

Lemma 33. *If there exists an unfolding tree over $\mathcal{P}$ satisfying a boss specification $\mathsf{bw} \in \mathcal{M}^*$ then there exists an initial run ρ of $\mathcal{P}$ satisfying bw.*

We start by defining partial runs, which are runs where some receptions are not matched by broadcasts. Intuitively, this will allow us to build partial runs from unfolding trees whose root is a follower node, which implicitly rely on their parent for some broadcasts. We will therefore construct inductively partial runs from nodes of the tree; we will obtain complete runs by matching reception and broadcasts of different partial runs.

Definition 34. *Let γ, γ' two configurations.*

A partial step $\gamma \to_p \gamma'$ is defined if either $\gamma \to \gamma'$ (normal step) or there exist $m \in \mathcal{M}$, $v \in \mathbb{N}$ such that for all agent a either $\gamma(a) = \gamma'(a)$ or $\gamma(a) \xrightarrow{\mathsf{ext}(\delta, v)} \gamma'(a)$ with δ a transition receiving message type m (unmatched reception).

A partial run is a sequence of partial steps. Note that a local run can be seen as a partial run with a single agent. A partial run is initial if it starts in an initial configuration. The v-input $\mathsf{In}_v(\rho)$ of a partial run ρ is the sequence $m_0 \cdots m_k$ of message types corresponding to unmatched receptions with value v in ρ. Its v-output $\mathsf{Out}_v(\rho)$ is the sequence of message types corresponding to broadcasts with value v in ρ.

We will prove the following, more general version of Lemma 33:

Lemma 35. *For every unfolding tree τ:*

- *if τ satisfies a boss specification $\mathsf{bw} \in \mathcal{M}^*$, then there exists an initial run ρ satisfying bw,*
- *if τ satisfies a follower specification $(\mathsf{fw}, \mathsf{fm})$ then there exist an initial partial run ρ and a value v such that:*
  - *all unmatched receptions in ρ are with value v,*
  - *$\mathsf{In}_v(\rho) \preceq \mathsf{fw}$,*
  - *$\mathsf{Out}_v(\rho)$ contains fm.*

Proof. We prove it by induction on the size of the unfolding tree. Let τ be a unfolding tree, μ its root, $u := \mathbf{lr}(\mu)$ and V the set of values appearing in u. We will combine u with runs given by children of μ to construct ρ satisfying the desired property. We build ρ inductively in two steps.

Step 1: Non-initial Values For each non-initial value $v \neq \mathbf{val}(\mu)$ of u, μ has a boss child of specification w such that $\mathsf{In}_v(u) \preceq w$. By induction hypothesis, there is an initial run ρ' satisfying boss specification w. Up to renaming agents, assume that ρ and ρ' have disjoint agents. We rename values in ρ' so that w is broadcast in ρ' with value v, and ρ' has no other shared value with ρ. We use the broadcasts made by ρ' to match the unmatched receptions with value v in ρ: this gives us a new partial run ρ with no unmatched reception with value v and whose behaviour on other values of V is the same as before.

Step 2: Initial Values Let v be an initial value of u, and $\mathtt{dec} = (w_0, m_1, w_1, \dots, m_\ell, w_\ell)$ the decomposition from condition (ii). We have that, for all $j \in [1, \ell]$, μ has a follower child μ_j labelled by $\mathbf{fm}(\mu_j) = m_j$ and $\mathbf{fw}(\mu_j) \in \mathcal{L}^{\mathtt{dec}_j}$ with $\mathtt{dec}_j = (w_0, m_1, w_1, \dots, m_{j-1}, w_{j-1})$.

The behavior of the run ρ with respect to v is the one of u, as we have not added any broadcasts or receptions with v. Hence we can split ρ into $\rho_0, \dots, \rho_\ell$ with $w_i \preceq \mathsf{Out}_v(\rho_i)$ and $\mathsf{In}_v(\rho_i) \in \{m_1, \dots, m_i\}^*$ for all i.

By induction hypothesis applied to μ_j, for all j, there exists an initial partial run $\tilde{\rho}_j$ whose only unmatched receptions are on v, $\mathsf{In}_v(\tilde{\rho}_j) \preceq \mathbf{fw}(\mu_j)$ and such that $\tilde{\rho}_j$ broadcasts (m_j, v) in its last step. Again, we rename agents and values so that the sets of agents of ρ and of every $\tilde{\rho}_j$ are all disjoint and the only shared value between any of these runs is v. As $\mathbf{fw}(\mu_j) \in \mathcal{L}^{\mathtt{dec}_j}$ and $\mathsf{In}_v(\tilde{\rho}_j) \preceq \mathbf{fw}(\mu_j)$, we can split $\tilde{\rho}_j$ into $\tilde{\rho}_{j,0}, \dots, \tilde{\rho}_{j,j-1}$ so that $\mathsf{In}_v(\tilde{\rho}_{j,i}) \preceq \tilde{w}_{j,i}$ where $\tilde{w}_{j,i}$ can be obtained by adding letters from $\{m_1, \dots, m_j\}$ to w_i.

We use the following composition operation: consider ρ and one of the $\tilde{\rho}_j$. We can design a new run in which we execute both runs in parallel over disjoint sets of agents. We match each $\tilde{\rho}_{j,i}$ with ρ_i so that the broadcasts of ρ_i with value v forming w_i are received in $\tilde{\rho}_{j,i}$ and the only remaining missing broadcasts in that section of the run are of $m_1, \dots, m_i$. We obtain a run section whose v-output still contains w_i and whose v-input only contains $m_1, \dots, m_i$. This lets us get to a point where the next step in $\tilde{\rho}_j$ is a broadcast (m_j, v) and ρ has been executed up to the beginning of ρ_j. We may then use the (m_j, v) broadcast at any moment in the rest of ρ either to complete an unmatched reception or to extend the v-output of ρ.

This construction is illustrated in Figure

The resulting run ρ' can still be split into $\rho'_0 \cdots \rho'_\ell$ where $w_i \preceq \mathsf{Out}_v(\rho_i)$ and $\mathsf{In}_v(\rho_i) \in \{m_1, \dots, m_i\}^*$ for all i. Its input on all values other than v is the same as the one of ρ. This procedure can thus be iterated.

If $v = \mathbf{val}(\mu)$ then we have $\mathbf{bw}(\mu) \in \mathcal{L}^{\mathtt{dec}}$, and we need to ensure that $\mathbf{bw}(\mu)$ is broadcast in ρ with value v. Let $\mathsf{bw}_0, \dots, \mathsf{bw}_\ell$ such that $\mathbf{bw}(\mu) \preceq \mathsf{bw}_0 \cdots \mathsf{bw}_\ell$ and for all j, bw_j can be obtained by adding letters from $\{m_1, \dots, m_j\}$ in w_j. We

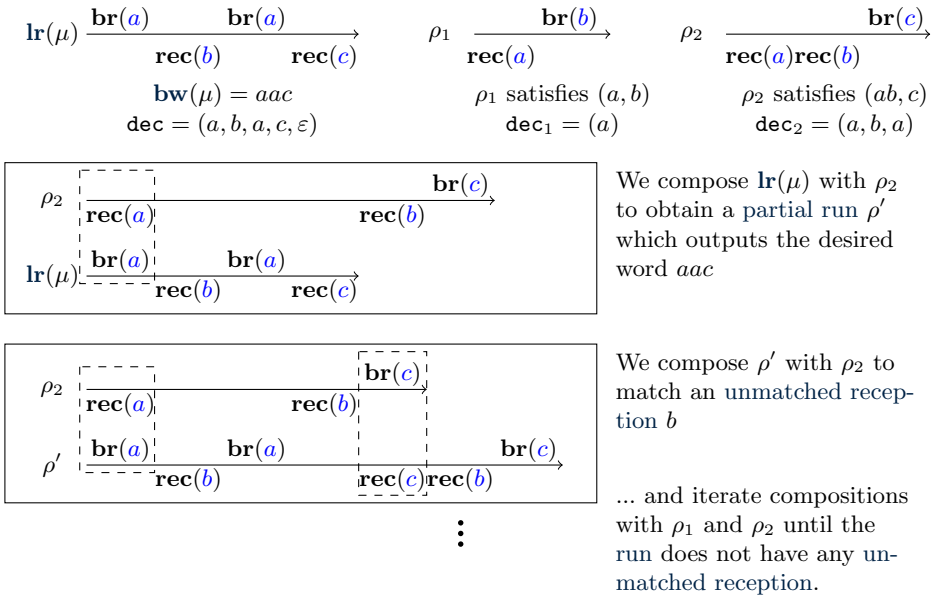


Fig. 8: An illustration of the composition operation from the proof of Lemma 33

use the composition operation to extend the output of ρ, so that $\mathsf{bw}_i \preceq \mathsf{Out}_v(\rho_i)$ for all i.

Then, in all cases, we apply the composition as many times as necessary to match the unmatched receptions: while there is an unmatched reception of some (m_j, v) we compose ρ with $\tilde{\rho}_j$ to eliminate it. We may be adding some unmatched receptions of (m_i, v) for some $i < j$. This procedure terminates as the number of unmatched receptions of $m_\ell, \dots, m_1$ decreases at each iteration for the lexicographic ordering.

In the end we obtain a run ρ with no unmatched reception on v, and such that, if $v = \mathbf{val}(\mu)$ and μ is a boss node, $\mathbf{bw}(\mu) \preceq \mathsf{Out}_v(\rho)$.

Concluding the procedure We distinguish two cases depending on the type of specification of μ.

- If μ is a boss node of boss specification bw, we apply steps 1 and 2 to every value in V, which can be done because $\mathbf{val}(\mu)$ is initial in u. We then obtain an initial run ρ with no unmatched receptions satisfying boss specification bw for value $\mathbf{val}(\mu)$.
- if μ is a follower node, we apply steps 1 and 2 to every value in $V \setminus \{\mathbf{val}(\mu)\}$ to obtain a partial run ρ with no unmatched reception on values different from $\mathbf{val}(\mu)$. Moreover, because the behavior of ρ with respect to $\mathbf{val}(\mu)$ is

the one of u, we have that $\mathsf{In}_{\mathbf{val}(\mu)}(\rho) \preceq \mathsf{fw}$ and $\mathsf{Out}_{\mathbf{val}(\mu)}(\rho)$ contains fm, concluding the proof.

□

D Proof of Lemma 25

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 and $\mathbf{bw}(\mu) \preceq \mathbf{bw}(\mu')$;*
- *μ and μ' are follower nodes, $\mathbf{fw}(\mu') \preceq \mathbf{fw}(\mu)$ and $\mathbf{fm}(\mu') = \mathbf{fm}(\mu)$.*

Proof. Let τ_μ, $\tau_{\mu'}$ be the subtrees rooted in μ, μ' respectively. Let τ' be the tree obtained by replacing τ_μ with $\tau_{\mu'}$; it is strictly smaller than τ. We prove that τ' is an unfolding tree. The only problematic node is the parent of μ if it exists, because it does not have the same children in τ and in τ'. Assume that μ is not the root of τ, and let μ_p its parent in τ'. The only problematic values are the ones that had μ as witness in conditions (i) (if μ is a boss node) and (ii) (if μ is a follower node). Let v such a value. If μ is a boss node (v is non-initial), we have $\mathsf{In}_v(\mathbf{lr}(\mu_p)) \preceq \mathbf{bw}(\mu) \preceq \mathbf{bw}(\mu')$ hence condition (i) is also satisfied. If μ is a follower node (v is initial), then, reusing the notations of condition (ii), μ was such that $\mathbf{fm}(\mu) = m_i$ and $\mathbf{fw}(\mu) \in \mathcal{L}^{\mathsf{dec}_i}$. In this case, we also have $\mathbf{fm}(\mu') = m_i$ and, because $\mathbf{fw}(\mu') \preceq \mathbf{fw}(\mu)$ and $\mathcal{L}^{\mathsf{dec}_i}$ is closed by subword, we have $\mathbf{fw}(\mu') \in \mathcal{L}^{\mathsf{dec}_i}$ and condition (ii) is satisfied. In both cases, τ' is an unfolding tree with the same root specification as τ hence a coverability witness. □

E Generalization of Lemma 19

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 $\mathsf{In}_{v'}(u') \preceq \mathsf{In}_v(u)$.*

The function ψ is actually not the same as in Lemma 19 although it is also a tower of exponentials.

We start by defining the notion of trace, which is a local run annotated with received values. A *trace* is a sequence in $(\{\mathsf{ext}(\delta, v) \mid \delta \in \Delta, v \in \mathbb{N}\} \cup \{\mathsf{int}(\delta) \mid \delta \in \Delta\})^*$. The trace of a local run u is the trace $\mathsf{tr}(u)$ corresponding to the local steps performed in u. Given a trace tr, we write $(q, \nu) \xrightarrow{\mathsf{tr}} (q', \nu')$ when there exists a local run of trace tr from (q, ν) to (q', ν').

We actually prove a more general version of the lemma. The previous lemma can be obtained simply by applying the following one with W the set of initial values of u_0 (W then contains r values) and V the set of values appearing in u_0, u or u_f.

Lemma 36. *There exists a primitive recursive function $\psi(n, r)$ such that, for every protocol $\mathcal{P}$ with r registers per agent, for every local run $u : (q, \nu) \xrightarrow{*} (q', \nu')$ in $\mathcal{P}$, for every $V \subseteq \mathbb{N}$ finite such that V contains all message values appearing in u, for every $W \subseteq V$, there exists a local run $u' : (q, \nu) \xrightarrow{*} (q', \nu')$ such that we have $\mathsf{len}(u') \leq \psi(|\mathcal{P}| - r + |W|, r)$ and:*

1. *for all $v \in V$, $\mathsf{In}_v(u') \preceq \mathsf{In}_v(u)$,*
2. *for all $v' \in \mathbb{N} \setminus V$, there exists $v \in \mathbb{N} \setminus W$ such that $\mathsf{In}_{v'}(u') \preceq \mathsf{In}_v(u)$.*

Intuitively, the set V represents values that are already used and therefore cannot be used as fresh values, and W represents values that should not be copied (*e.g.*, initial values of the run). However, stating the lemma only with W equal to the initial values in u would not allow us to apply the lemma on u an infix local run of a larger local run u', because the initial values in u would not correspond to the ones in u'. This is why we state the lemma for every $W \subseteq V$.

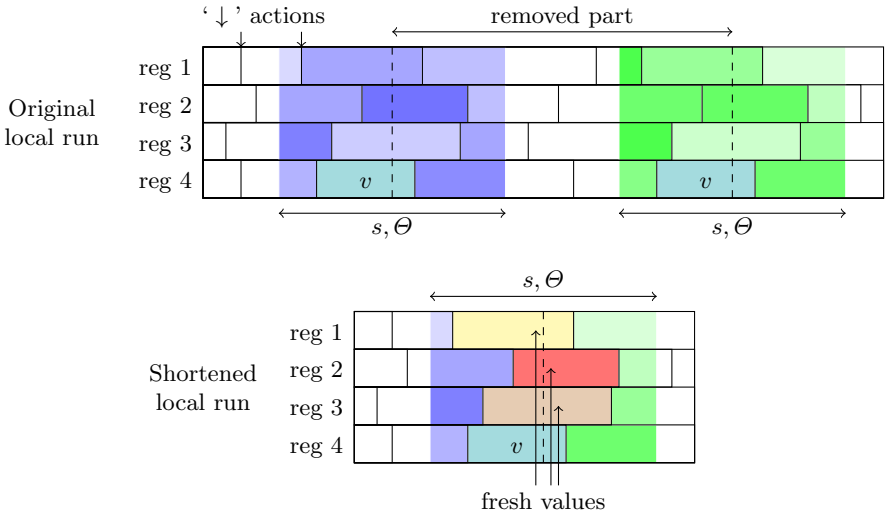


Fig. 9: Illustration of the proof of Lemma 26. $s \in \Delta^{2M}$ corresponds to the repeating sequence of transitions of length $2M$. Register 4 contains value $v \in W$. Because both sides have the same Θ, they coincide on values in W such as v; only values that are not in W are replaced by fresh values, hence v is kept in the shortened local run.

Proof. Given a local run u, register i is *active* in u if at least one ' ↓ ' step on register i is performed in u.

We define the function $\psi(n, k)$ recursively as $\psi(n, 0) = n+1$ and $\psi(n, k+1) = 2(\psi(n, k)+1)[(n+1)^{4(\psi(n,k)+1)} + 1]$. This function is clearly primitive recursive,

although non-elementary; it grows as a tower of exponentials of height k where each floor of the tower is polynomial in n.

We prove the *shortening property*:
Let W a finite set a values and $n := |\mathcal{P}| - r + |W|$. Let a local run $u : (q_i, \nu_i) \xrightarrow{*} (q_f, \nu_f)$ with k active registers with $\mathsf{len}(u) > \psi(n, k)$ and let $V \subseteq \mathbb{N}$ finite that contains every message value appearing in u such that $W \subseteq V$. We claim that u can be shortened into a local run $u' : (q_i, \nu_i) \xrightarrow{*} (q_f, \nu_f)$ with k active registers such that $\mathsf{len}(u') < \mathsf{len}(u)$ and:

- for all $v \in V$, $\mathsf{In}_v(u') \preceq \mathsf{In}_v(u)$,
- for all $v' \in \mathbb{N} \setminus V$, there exists $v \in \mathbb{N} \setminus W$ such that $\mathsf{In}_{v'}(u')$ is a subword of $\mathsf{In}_v(u)$.

We proceed by induction on the number k of active registers in the local run. If $k = 0$, register values do not change in u. As $\psi(n, 0) = n + 1 \geq |Q| + 1$, u goes through the same state twice, hence all steps in between may be removed to get u'. Then for all $v \in \mathbb{N}$, $\mathsf{In}_v(u') \preceq \mathsf{In}_v(u)$.

Suppose that the property is true for any local run with $\leq k$ active registers, and consider one $u : (q_i, \nu_i) \xrightarrow{*} (q_f, \nu_f)$ with $k + 1$ active registers such that $\mathsf{len}(u) > \psi(|\mathcal{P}| - r + |W|, k + 1)$.

First, if there exists an infix local run u_i of u of length $\psi(n, k) + 1$ with only k active registers, then it suffices to apply the induction hypothesis on u_i. Suppose now that there exists no such infix local run. Let $I \subseteq [1, r]$ the set of active registers in u, $|I| = k + 1$. Let $M := \psi(n, k)+1$, we have $\psi(n, k + 1) = 2M[((n + 1)^2)^{2M} + 1]$. In any sequence of M local steps in a row in u, there is a '$\downarrow$' transition on every register in I. We can assume that no local configuration appears twice in u (otherwise we can simply cut the steps between those two appearances to get u').

In what follows we will consider two infixes of u of length $2M$, following the same sequence of transitions and with the same values of W appearing at the same times in the same registers. Their existence is guaranteed by the length of u. As a '$\downarrow$' action is performed twice on each register in those two infixes, we will be able to reduce the run as in Fig. 9.

For every i, let δ_i the i-th transition in u, and let $\theta_i \in W \cup \{\bot\}$ be such that

$$\theta_i = \begin{cases} v \text{ if the } i\text{th step of } u \text{ is a reception step } \mathsf{ext}(\delta_i, v) \text{ with } v \in W \\ \bot \text{ otherwise} \end{cases}$$

For every $i \in [0, (n+1)^{4M}]$, we write s_i the sequence $\delta_{2M \cdot i+1}, \delta_{2M \cdot i+2}, \cdots, \delta_{2M \cdot i+2M}$ and Θ_i the sequence $\theta_{2M \cdot i}, \theta_{2M \cdot i+1}, \cdots, \theta_{2M \cdot i+2M}$. There are $|\Delta|^{2M}$ possible sequences for s_i and $[|W| + 1]^{2M}$ possible sequences for Θ_i. By the pigeonhole principle there exist two indices i_a, i_b such that the sequences s_{i_a} and s_{i_b} are equal and also the sequences Θ_{i_a} and Θ_{i_b} are equal (as $|\Delta|(|W| + 1) \leq (n + 1)^2$ because $n = |\mathcal{P}|-r+|W|$). There exist two infix local runs $u_a : (q_1, \nu_1) \xrightarrow{*} (q_2, \nu_2)$, $u_b : (q_3, \nu_3) \xrightarrow{*} (q_4, \nu_4)$ in u such that (q_2, ν_2) appears strictly before (q_3, ν_3) in u and u_a and u_b both have the same sequences of transitions, which we call s, and of receptions of values from W, which we call Θ.

Although u_a and u_b have the same sequence of transitions, their traces may differ because their reception steps may have different values. We build a trace tr such that $(q_1, \nu_1) \xrightarrow{\mathsf{tr}} (q_4, \nu_4)$ where the underlying sequence of transitions of tr is s and the receptions of values of W match Θ.

For every active register $i \in I$, let $e_i \in [1, 2M]$ denote the index of the first '$\downarrow$' on register i in s and $f_i \in [1, 2M]$ the index of the last '$\downarrow$' on register i in s. By hypothesis, because s is of length $2M$, it contains at least two '$\downarrow$' on register i, one in the first half and one in the second half, hence $e_i \leq M < M + 1 \leq f_i$.

For every $j \in [1, 2M]$, let δ_j denote the j-th transition of s. First, if δ_j is a broadcast, we define the j-th local step of tr as $\mathsf{int}(\delta_j)$. Suppose now that δ_j is a reception of the form $\mathbf{rec}(m, i, \alpha)$. The j-th local step of u_a (resp. u_b) has underlying transition δ_j hence is a reception step of the form $\mathsf{ext}(\delta_j, v_a)$ for some $v_a \in V$ (resp. $\mathsf{ext}(\delta_j, v_b)$ for some $v_b \in V$). Because V is finite and $W \subseteq V$, there exists an injective function $\phi : V \to \mathbb{N}$ such that for all $v \in W$, $\phi(v) = v$ and for all $v \notin W$, $\phi(v) \notin V$. We define the j-th local step of tr to be $\mathsf{ext}(\delta_j, v)$ where:

- if $i \notin I$ then its value stays the same throughout u, then we set $v = v_a (= v_b)$
- if $i \in I$ then
  - if $j < e_i$, $v = v_a$,
  - if $e_i \leq j < f_i$, $v = \phi(v_a)$,
  - if $f_i \leq j$, $v = v_b$.

We now claim that $(q_1, \nu_1) \xrightarrow{\mathsf{tr}} (q_4, \nu_4)$. First, for every active register i, the last '$\downarrow$' step on register i has value $\nu_4(i)$ in tr (as we are in the case $f_i \leq j$). Hence if every local step is valid then the final local configuration is (q_4, ν_4). For every $l \in [0, 2M]$, let tr_l denote the prefix of tr of length l. We prove by induction on l that tr_l is valid from (q_1, ν_1). It is trivially true for $l = 0$. Assume that we have $(q_1, \nu_1) \xrightarrow{\mathsf{tr}_l} (q, \nu)$ and let λ such that $\mathsf{tr}_l \cdot \lambda = \mathsf{tr}_{l+1}$. Let δ the underlying transition of λ. First, q is the initial state of δ because λ is valid at step $l + 1$ of u_a (and u_b). Hence if δ is a broadcast then λ is valid from (q, ν). Suppose now that λ is a reception step; let $\delta =: \mathbf{rec}(m, i, \alpha)$. Let v_a, v_b be the value of λ in u_a and u_b respectively.

Let ν_a, ν_b be the content of registers after the l-th step in u_a and u_b respectively. If $i \notin I$ then α is either '$*$' or a test, which is valid as $\nu(i) = \nu_a(i) = \nu_b(i)$ (value of register i does not change in u). Suppose now that $i \in I$, the only problematic case is the one of tests, *i.e.*, $\alpha \in \{\text{'}\neq\text{'}, \text{'}=\text{'}\}$. In this case, we prove that $v \ \alpha \ \nu(i)$. First, because the corresponding step is valid in u_a and u_b, we have $\nu_a(i) \ \alpha \ v_a$ and $\nu_b(i) \ \alpha \ v_b$. We distinguish cases depending on $l + 1$:

- $l + 1 < e_i$: $\nu(i) = \nu_a(i)$, $v = v_a$ and $\nu_a(i) \ \alpha \ v_a$.
- $e_i \leq l + 1 < f_i$: We have $v = \phi(v_a)$. Moreover, because $e_i < l + 1$, there is at least one '$\downarrow$' on register i in tr_l. Consider the last such transition in tr_l; its index j satisfies $e_i \leq j < f_i$ by definition of e_i, hence the value of the corresponding reception step in u_a is $\nu_a(i)$ and its value in tr_l is $\phi(\nu_a(i))$. One has $\nu_a(i) \ \alpha \ v_a$ therefore (by injectivity of ϕ for $\alpha = \text{'}\neq\text{'}$) $\phi(\nu_a(i)) \ \alpha \ \phi(v_a)$.
- $f_i \leq l + 1$: $\nu(i) = \nu_b(i)$ and $v = v_b$, and because the internal step is valid in u_b we have $\nu_b(i) \ \alpha \ v_b$.

This proves that λ is valid from (q, ν) which concludes the induction. We have proven that $(q_1, \nu_1) \xrightarrow{\mathsf{tr}} (q_4, \nu_4)$; moreover tr is of length $2M$ and there are at least $4M > 2M + 1$ steps between (q_1, ν_1) and (q_4, ν_4) in u. Therefore, replacing this part of u with $(q_1, \nu_1) \xrightarrow{\mathsf{tr}} (q_4, \nu_4)$ yields a shorter local run $u' : (q_i, \nu_i) \xrightarrow{*} (q_f, \nu_f)$.

It remains to prove the conditions on the v-input of u' for each v. If suffices to prove the condition for the part between (q_1, ν_1) to (q_4, ν_4), because the rest of u is left untouched.

Let (q_m, ν_m) the local configuration after M steps of tr from (q_1, ν_1); write u_1 the local run from (q_1, ν_1) to (q_m, ν_m) corresponding to the first M steps of tr in u', and u_2 the local run from (q_m, ν_m) to (q_4, ν_4) corresponding to the last M steps of tr in u'.

Let $v \in V$. We claim that $\mathsf{In}_v(u_1)$ is a subword of $\mathsf{In}_v(u_a)$ and $\mathsf{In}_v(u_2)$ is a subword of $\mathsf{In}_v(u_b)$. Indeed, in the construction of tr, the reception steps in the first M steps were those of u_a except that some values were replaced with fresh values in $\mathbb{N} \setminus V$, and similarly with u_b and the last M steps. Overall, this proves that $\mathsf{In}_v(u')$ is a subword of $\mathsf{In}_v(u)$ for every $v \in V$ and values of V satisfy condition 1.

Let $v' \in \mathbb{N} \setminus V$; v' does not appear in u. Either v' does not appear in u' in which case the desired property is true, or there exists $v \in V$ such that $v' = \phi(v)$. As ϕ is injective and $\phi(W) = W$ and $v' \notin W$, $v \notin W$. But then $\mathsf{In}_{v'}(u_1)$ is a subword of $\mathsf{In}_v(u_a)$ and $\mathsf{In}_{v'}(u_2)$ is a subword of $\mathsf{In}_v(u_b)$. Indeed, in u_1, the reception steps with value $\phi(v)$ correspond to reception steps in u_a with value v, and similarly for u_2 and u_b. This proves condition 2 for every $v' \in \mathbb{N} \setminus V$. Overall, we have proven the existence of a local run $u' : (q_i, \nu_i) \xrightarrow{*} (q_f, \nu_f)$ that satisfies conditions 1 and 2 and that is strictly shorter that u, which proves the shortening property.

We build a local run of length less that $\psi(|\mathcal{P}| - r + |W|, r)$ as follows. We start with $u^{(0)} := u$ and $V^{(0)}$ the set of values of messages appearing in $u^{(0)}$. For every k such that $\mathsf{len}(u^{(0)}) > \psi(|\mathcal{P}| - r + |W|, r)$, we apply the shortening property on $u^{(k)}$ and $V^{(k)}$ to obtain $u^{(k+1)}$ and define $V^{(k+1)}$ by $V^{(k)} \cup X$ where X is the set of values of messages in $u^{(k+1)}$, which is finite. The construction stops when $\mathsf{len}(u^{(k)}) \leq \psi(|\mathcal{P}| - r + |W|, r)$, which concludes the proof of the lemma. □

F Bounding the Size of the Minimal Coverability Witness

In this section, we prove that we can obtain a computable bound on τ, and that the problem is decidable in complexity class $\mathbf{F}_{\omega^\omega}$. All bounds provided in this section are independent from τ. We start by proving that one can bound the size of a node with respect to the size of its neighbors of higher altitude.

Lemma 37. *Let μ be a node of τ such that all neighbors of μ of higher altitude have size bounded by K. Then $|\mu| \leq (\psi(|\mathcal{P}|, r) + 2)\,(|\mathcal{M}|\, r\, K + K)$, with ψ the primitive-recursive function defined in Lemma 26.*

Proof. Let $u := \mathbf{lr}(\mu)$. For each initial value v in u, μ has at most $|\mathcal{M}|$ follower children, by definition of a decomposition. Because there are at most r initial

values, this makes at most $|\mathcal{M}|\,r$ follower children and each one of them requires at most K messages. If μ is a follower node or the root, then the number of messages it must broadcast is then bounded by $|\mathcal{M}|\,r\,K+1$. If μ is a boss node and not the root, then its parent has higher altitude hence $|\mathsf{bw}(\mu)| \leq K$ and μ must broadcast at most $|\mathcal{M}|\,r\,K+K$ messages in total. This bounds by $|\mathcal{M}|\,r\,K+K+1$ the number of messages u needs to broadcast. Lemma 26 then lets us reduce the number of steps between those messages without breaking the unfolding tree conditions: the inputs of values that were already in the run can only decrease. For the other values, let v be a value that was not in the run before, its input is a subword of the one of a previous value. Hence μ has a boss child whose specification covers the v-input of the new local run. This bounds $|u|$ by $(\psi(|\mathcal{P}|,r)+1)\,(|\mathcal{M}|\,r\,K+K)$, with ψ the function described in Lemma 26; the bound on $|\mu|$ follows. □

We now formally prove Lemma 27:

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. Let $f_0 : k \mapsto ((\psi(k,k)+2)(k^2+2))^{k+1}$. Also, let $N := |\mathcal{P}|$. First, if μ has altitude **altmax**, then it has no follower child, hence applying Lemma 37 with $K=0$ bounds $|\mu|$ by $\psi(|\mathcal{P}|,r) \leq \psi(N,N) \leq f_0(N)$. Let μ be a node with $\mathbf{alt}(\mu) < \mathbf{altmax}$, and suppose that the statement is true for altitudes greater than $\mathbf{alt}(\mu)$.

Let $d = \mathbf{altmax} - \mathbf{alt}(\mu)$. We apply Lemma 37 with $K := f_0(N+d-1)$:

$$\begin{aligned}|\mu| \leq& (\psi(N,N)+2)(N^2K+K)\\ \leq& (\psi(N+d,N+d)+2) f_0(N+d-1)((N+d)^2+2)\\ \leq& f_0(N+d)\end{aligned}$$

□

Lemma 38. *There exists a function $f_1 \in \mathscr{F}_{\omega^{|\mathcal{M}|}}$ such that* $\mathbf{altmax} \leq f_1(|\mathcal{P}|)$.

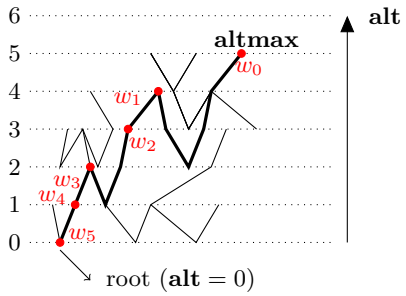
Proof. Let **altmax** be the maximal altitude of a node in τ. Consider a branch of τ reaching altitude **altmax**; for every $j \in [1, \mathbf{altmax}]$, let μ_j the first node of this branch to reach altitude j. For every $j \geq 1$, μ_j is a follower nodes as otherwise its predecessor in the branch would have altitude greater than j hence the branch must have crossed altitude j before.

We will bound **altmax** using the Length function theorem. For simplicity, we will encode the fm part using a fresh character added to our alphabet. This is why we obtain a function in $\mathscr{F}_{\omega^{|\mathcal{M}|}}$ and not $\mathscr{F}_{\omega^{|\mathcal{M}|-1}}$; in fact, one could obtain the latter bound using Theorem 5.3. of [24], but the proof would be more involved.

Let $\# \notin \mathcal{M}$ be a fresh letter. For all $i \in [1, \mathbf{altmax}]$ let $\mu'_i = \mu_{\mathbf{altmax}-i+1}$ and $w_i = \mathbf{fw}(\mu'_i) \cdot \# \cdot \mathbf{fm}(\mu'_i) \in (\mathcal{M} \cup \{\#\})^*$.

We cannot have $w_i \preceq w_j$ for $i < j$: indeed, this would imply $\mathbf{fw}(\mu'_i) \preceq \mathbf{fm}(\mu'_j)$ and $\mathbf{fm}(\mu'_i) = \mathbf{fw}(\mu'_j)$, which is a contradiction according to Lemma 25. As a

result, the sequence $(w_i)_{1\leq i\leq \mathbf{altmax}}$ is a bad sequence over the alphabet $\mathcal{M} \cup \{\#\}$. Furthermore, by Lemma 27, for all i, $|\mu_i| \leq f_0(|\mathcal{P}| + i)$ therefore $|w_i| \leq f_0^{(i)}(|\mathcal{P}|)$. Because f_0 is primitive-recursive, we can apply Theorem 10: there exists a function $f_1 \in \mathscr{F}_{\omega^{|\mathcal{M}|}}$ such that the sequence $(w_i)_{i\in[1,\mathbf{altmax}]}$ is of length at most $f_1(|\mathcal{P}|)$, hence $\mathbf{altmax} \leq f_1(|\mathcal{P}|)$. □



For all i, let $w_i = \mathbf{fw}(\mu'_i)\#\mathbf{fm}(\mu'_i)$ where μ'_i is the first follower node at altitude $\mathbf{altmax} - i + 1$.

By Lemma 27 we can bound the length of w_i by $f_0(|\mathcal{P}| + i)$, which lets us to bound the number of red dots using Lemma 25 and the Length function theorem.

Fig. 10: Illustration of the proof of Lemma 38.

The bound on the maximal altitude, along with Lemma 27, bounds the size of the root of τ by $f_0(|\mathcal{P}|+f_1(|\mathcal{P}|))$. Let $\mathbf{altmin} := \min_{\mu\in\tau} \mathbf{alt}(\mu) \leq 0$ the minimal altitude in τ. By a similar argument but with boss nodes, we can bound $|\mathbf{altmin}|$:

Lemma 39. *There exists a function $f_2 \in \mathscr{F}_{\omega^{|\mathcal{M}|}}$ such that $|\mathbf{altmin}| \leq f_2(|\mathcal{P}|)$.*

Proof. Consider a branch of τ reaching **altmin**, let $\mu_1, \ldots, \mu_{|\mathbf{altmin}|}$ be nodes of τ such that μ_i is the first node of the branch to reach altitude $-i$. All these nodes are boss nodes because a follower node has higher altitude than its parent. For all $i \in [0, \mathbf{altmin}]$ let $\mu'_i = \mu_{|\mathbf{altmin}|-i}$ and $w_i = \mathbf{bw}(\mu'_i) \in \mathcal{M}^*$. By Lemma 25, $(w_i)_{i\in[1,\mathbf{altmin}]}$ is a bad sequence. By Lemmas 27 and 38, for all i, we have $|w_i| \leq f_0(|\mathcal{P}| + f_1(|\mathcal{P}|) + i)$. Let $N := |\mathcal{P}| + f_1(|\mathcal{P}|) + 1$; we have $|w_i| \leq f_0^{(i)}(N)$.

Because f_0 is primitive recursive, we can apply Theorem 10: we obtain $h \in \mathscr{F}_{\omega^{|\mathcal{M}|-1}}$ such that $\mathbf{altmin} \leq h(N)$. Let $f_2 : n \mapsto h(n + f_1(n) + 1)$; we have $\mathbf{altmin} \leq f_2(|\mathcal{P}|)$. Moreover, f_2 is a function of $\mathscr{F}_{\omega^{|\mathcal{M}|}}$ because $f_1, h \in \mathscr{F}_{\omega^{|\mathcal{M}|}}$ and $\mathscr{F}_{\omega^{|\mathcal{M}|}}$ is closed under composition. □

With the bounds on the maximal and minimal altitudes, Lemma 27 yields a bound on the size of all nodes in the tree.

Lemma 40. *There exists $f_3 \in \mathscr{F}_{\omega^{|\mathcal{M}|}}$ such that, for all μ in τ, $|\mu| \leq f_3(|\mathcal{P}|)$.*

Proof. Using Lemmas 27, 38 and 39, we obtain that, for every node μ, the value $\mathbf{altmax}-\mathbf{alt}(\mu)$ is at most $f_1(|\mathcal{P}|)+f_2(|\mathcal{P}|)$. Applying Lemma 27 proves that, for all μ in τ, $|\mu| \leq f_0(f_1(|\mathcal{P}|)+f_2(|\mathcal{P}|)) \leq f_3(|\mathcal{P}|)$ where $f_3 : n \mapsto f_0(f_1(n)+f_2(n))$; f_3 is in $\mathscr{F}_{\omega^{|\mathcal{M}|}}$ as $\mathscr{F}_{\omega^{|\mathcal{M}|}}$ contains all primitive recursive functions and is closed under composition. □

Finally, as we now have a bound on all nodes, we can easily bound the length of branches and the number of children of all nodes to get abound on the total size of the tree.

Proposition 41. *There exists a function f of class $\mathscr{F}_{\omega^{|\mathcal{M}|}}$ s.t. $|\tau| \leq f(|\mathcal{P}|)$.*

Proof. Recall that the size of τ is defined as the sum of the sizes of its nodes.

Let $N = |\mathcal{P}|$. By Lemma 40, for each μ in τ we have $|\mu| \leq f_3(N)$, hence the local run of μ contains at most $f_3(N)$ values. By minimality of τ, each value requires at most 1 boss child and $|\mathcal{M}|$ follower children hence μ has at most $(|\mathcal{M}| + 1)f_3(N) \leq Nf_3(N)$ children. Moreover, there are less than $f_4(N) := N^{f_3(N)}$ possible specifications in τ hence each branch of τ is of length less than $f_4(N)$ (by minimality, a branch does not have twice the same specification). This bounds the total number of nodes in τ by $f_5(N) := (Nf_3(N))^{f_4(N)+1}$ hence we obtain $|\tau| \leq f(N)$ where $f : n \mapsto f_3(n)f_5(n)$. Because $\mathscr{F}_{\omega^{|\mathcal{M}|}}$ is closed under composition with primitive recursive functions, we have $f \in \mathscr{F}_{\omega^{|\mathcal{M}|}}$. □

G Proof of Theorem 12

We now prove the main result of this paper:

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

Proof. The lower bound is given by the reduction from lossy channel system reachability in Proposition 11.

For the upper bound, let $(\mathcal{P}, q_f)$ be an instance of Cover. By Propositions 24 and 41, $(\mathcal{P}, q_f)$ is positive if and only if it has a coverability witness of size bounded by $f(|\mathcal{P}|)$ where $f \in \mathscr{F}_{\omega^{|\mathcal{M}|}}$. Up to renaming agents and values, we can moreover assume that all agents and values appearing in this coverability witness are bounded by $f(|\mathcal{P}|)$, which bounds the size of the description of such a coverability witness by a polynomial in $f(|\mathcal{P}|)$. An algorithm for Cover consists in enumerating all such descriptions and accepting if one finds a coverability witness. This can all be done in time exponential in $f(|\mathcal{P}|)$, thus this algorithm terminates in time $f'(|\mathcal{P}|)$ where $f' \in \mathscr{F}_{\omega^{|\mathcal{M}|}}$. This proves that Cover lies in complexity class $\mathbf{F}_{\omega^\omega}$. □

H Proof of Proposition 29

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

Proof. We present a reduction from the halting problem for Minsky machines to Cover for signature BNRA.

A Minsky Machine with two counters is a tuple $M = (\mathrm{Loc}, \Delta, \mathtt{X}, \ell_0, \ell_f)$ where Loc is a finite set of locations, $\mathtt{X} = \{\mathtt{x}_1, \mathtt{x}_2\}$ is a set of two counters, $\Delta \subseteq \mathrm{Loc} \times \{\mathtt{x}-, \mathtt{x}+, \mathtt{x} = 0? \mid \mathtt{x} \in \mathtt{X}\} \times \mathrm{Loc}$ is a finite set of transitions, $\ell_0 \in \mathrm{Loc}$ is an initial location and $\ell_f \in \mathrm{Loc}$ is a final location. A *configuration* of a Minsky

machine is a tuple $(\ell, v_1, v_2) \in \mathrm{Loc} \times \mathbb{N} \times \mathbb{N}$ where v_1 (resp. v_2) stands for the value of the counter $\mathtt{x}_1$ (resp. $\mathtt{x}_2$). We write $(\ell, v_1, v_2) \to (\ell', v_1', v_2')$ if there is $\delta \in \Delta$ such that:

- $\delta = (\ell, \mathtt{x_i}+, \ell')$ and $v_i' = v_i + 1$, $v_{3-i} = v_{3-i}'$;
- $\delta = (\ell, \mathtt{x_i}-, \ell')$ and $v_i' = v_i - 1$, $v_{3-i} = v_{3-i}'$;
- $\delta = (\ell, \mathtt{x_i} = 0?, \ell')$ and $v_i' = v_i = 0$, $v_{3-i} = v_{3-i}'$.

An execution of the machine is a sequence $(\ell_1, v_1^{(1)}, v_2^{(1)}) \to (\ell_2, v_1^{(2)}, v_2^{(2)}) \to \ldots \to (\ell_k, v_1^{(k)}, v_2^{(k)})$. The halting problem asks whether ℓ_f is coverable. This problem is well-known to be undecidable [19].

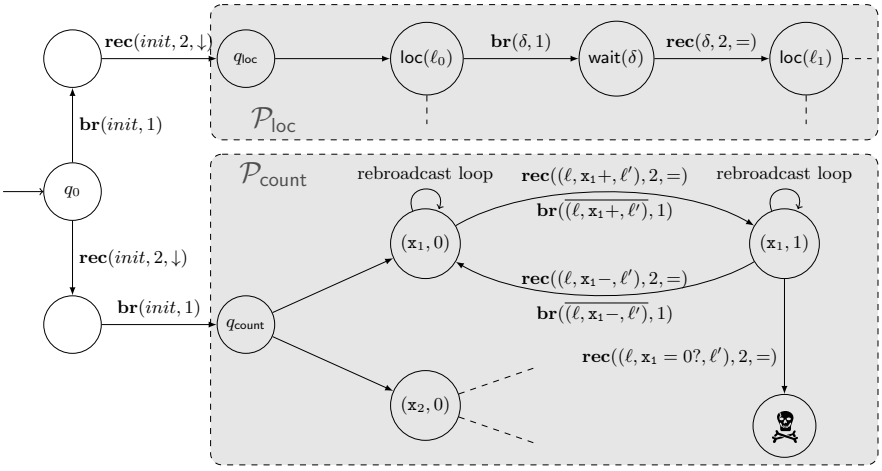


Fig. 11: Partial depiction of the protocol built in Proposition 29. Only one transition, which is (ℓ_0, δ, ℓ_1), is represented in the $\mathcal{P}_{\mathsf{loc}}$ part above; similarly, only one increment and one decrement transitions are depicted in the $\mathcal{P}_{\mathsf{count}}$ part below. The rebroadcast loops rebroadcast all transitions acting on $\mathtt{x}_2$ and all acknowledgements; the one on $(\mathtt{x}_1, 0)$ also rebroadcasts all transitions with $\mathtt{x}_1-$ and with zero-tests, and the one on $(\mathtt{x}_1, 1)$ rebroadcasts all transitions with $\mathtt{x}_1+$.

Fix a Minsky Machine $M = (\mathrm{Loc}, \Delta, \mathtt{X}, \ell_0, \ell_f)$. We build a signature protocol $\mathcal{P}$ with a state q_f such that $(\mathcal{P}, q_f)$ is a positive instance of Target if and only if ℓ_f is coverable in M. The protocol is represented in Figure 11. As in Proposition 11, in an initial phase, each agent picks a predecessor by storing its identifier and only listens to this predecessor afterwards. We call *cycle* a sequence of agents $a_1, a_2, \ldots, a_n = a_1$ where agent a_i is the predecessor of a_{i+1} for all $i < n$. As all agents have to reach the end state, they must all pick a predecessor. As there are finitely many agents in a run, a cycle will necessarily be formed in any run satisfying the Target requirement.

The rest of the construction aims at faithfully simulating the machine in a cycle: Agents in $\mathcal{P}_{\mathsf{loc}}$ sends a sequence of instructions and waits after each

one for a confirmation that it was executed. Agents in $\mathcal{P}_{\mathsf{count}}$ simulate counter values. The messages circulating in a cycle contain either a transition $\delta \in \Delta$ or an acknowledgment $\overline{\delta}$ with $\delta \in \Delta$. An agent a in $\mathcal{P}_{\mathsf{count}}$ first picks a counter $\mathtt{x}_i$ it simulates, and goes to state $(\mathtt{x}_i, 0)$. If a is in $(\mathtt{x}_i, 0)$ and receives δ corresponding to an increment of $\mathtt{x}_i$, it goes to $(\mathtt{x}_i, 1)$ and broadcast an acknowledgment $\overline{\delta}$, and conversely for decrements. If δ is a zero-test $\mathtt{x_i} = 0?$ and a is on state $(\mathtt{x}_i, 1)$ then it stops, making the whole cycle fail. Otherwise it broadcasts the acknowledgment $\overline{\delta}$. Other messages are rebroadcast as such.

An agent a in $\mathcal{P}_{\mathsf{loc}}$ starts in state $\mathsf{loc}(\ell_0)$. When in state $\mathsf{loc}(\ell)$, it picks and broadcasts a transition $\delta = (\ell, \alpha, \ell') \in \Delta$, waits for the acknowledgment $\overline{\delta}$ and goes to $\mathsf{loc}(\ell')$. In the case where δ is a zero-test, we have $\overline{\delta} = \delta$: there is no need for a distinct acknowledgment because there is no action to perform (if the test fails then no message is transmitted). When in $\mathsf{loc}(\ell_f)$, a broadcasts a special message end that propagates in the cycle and makes everyone go to q_f. When it receives itself the message *end*, it goes to q_f.

It is quite easy to see that, if ℓ_f can be covered in M, one can build a run of $\mathcal{P}$ where all agents end in q_f. Let N the highest counter value in the execution of M covering q_f. The run of $\mathcal{P}$ first puts all its agents in the same cycle; exactly one agent a_{lead} goes in $\mathcal{P}_{\mathsf{loc}}$ and $2N$ agents go in $\mathcal{P}_{\mathsf{count}}$; half of these simulate $\mathtt{x}_1$ and half $\mathtt{x}_2$, so that the largest counter value is never exceeded. It then suffices to faithfully simulate the execution of M: a_{lead} selects the corresponding sequence of transitions, their effect is always applied as we have enough agents simulating each counter. After each round the number of agents in state $(\mathtt{x}_i, 1)$ is exactly the value of $\mathtt{x}_i$ at this point in the run of the machine, hence zero-tests never cause failure. In the end a_{lead} reaches $\mathsf{loc}(\ell_f)$ and broadcasts end, allowing every agent in $\mathcal{P}_{\mathsf{count}}$ to get to q_f.

For the converse implication, suppose that we have a run ρ of $\mathcal{P}$ where all agents end in q_f. As mentioned before, there must be a cycle of agents $a_1, \ldots, a_n$ in this run. Observe that all agents alternate between broadcasts and receptions, so that to reach q_f they must all have made the same number of broadcasts and receptions. This implies that no message was lost along the cycle.

Note that there may be several agents in $\mathcal{P}_{\mathsf{loc}}$ along the cycle; however, they must all broadcast exactly the same sequence of transitions, otherwise one of them would lack an acknowledgment and would not get to q_f. Let a be the agent that first reaches $\mathsf{loc}(\ell_f)$ and a' the first agent in $\mathcal{P}_{\mathsf{loc}}$ after a in the cycle; there are only agents in $\mathcal{P}_{\mathsf{count}}$ between a and a' in the cycle, we call these agents *intermediate agents*. The intermediate agents faithfully encode the two counters and all decrements and zero-tests pass, otherwise a' would lack an acknowledgment. Therefore, the sequence of transitions of a defines an execution of M that covers ℓ_f, which concludes the proof. □

I Proofs of Section 5

In this section the register argument in receptions and broadcasts is always 1, hence we remove it. Our new set of operations is

$$Op^{\mathcal{M}} = \{\mathbf{br}(m), \mathbf{rec}(m, *), \mathbf{rec}(m, \downarrow), \mathbf{rec}(m, =), \mathbf{rec}(m, \neq) \mid m \in \mathcal{M}\}.$$

Also, given a configuration γ, we write $\mathsf{data}(\gamma)(a)$ for $\mathsf{data}(\gamma)(a, 1)$.

Given a run ρ, we write $\mathsf{ag}(\rho)$ for its set of agents, and we define the set of states appearing in it $\mathsf{cov}(\rho) := \{q \in Q \mid \exists i, \exists a, \mathsf{st}(\gamma_i)(a) = q\}$ as well as its set of values $\mathsf{val}(\rho) := \{v \in \mathbb{N} \mid \exists i, j, a, \mathsf{data}(\gamma_i)(a, j) = v\}$.

First, to simplify the proofs, we eliminate reception transitions with action '$\neq$'. This is feasible as we can execute several copies of a run in parallel (with distinct values) so that every broadcast is made in each copy with a different value. Hence if a agent receives a message, it can always receive it with a value different from its own, making disequality tests useless. We can thus replace them with receptions with '$*$'.

I.1 Removing Disequality Tests

We start by formalising the intuition that a configuration contains more agents than another one up to renaming.

Definition 42. *We define a preorder over the set of configurations as follows: $\gamma \trianglelefteq \gamma'$ if there exists an injective function $\pi : \mathsf{ag}(\gamma) \to \mathsf{ag}(\gamma')$ such that, for all $a \in \mathsf{ag}(\gamma)$, $\gamma(a) = \gamma'(\pi(a))$.*

Lemma 43. *Let $(\mathcal{P}, q_f)$ an instance of the coverability problem. This instance is positive if and only if $(\tilde{\mathcal{P}}, q_f)$ is positive, where $\tilde{\mathcal{P}}$ is equal to $\mathcal{P}$ where every disequality test '$\neq$' is replaced by dummy action '$*$'.*

Proof. First, if $(\mathcal{P}, q_f)$ is positive then so is $(\tilde{\mathcal{P}}, q_f)$, as one can easily lift any initial run in $\mathcal{P}$ to an equivalent initial run in $\tilde{\mathcal{P}}$ (transitions are less guarded in $\tilde{\mathcal{P}}$ that in $\mathcal{P}$).

Suppose now that $(\tilde{\mathcal{P}}, q_f)$ is a positive instance of the coverability problem. There exists an initial run $\tilde{\rho} : \tilde{\gamma}_0 \xrightarrow{*} \tilde{\gamma}$ in $\tilde{\mathcal{P}}$ that covers q_f. We prove by induction on the length of $\tilde{\rho}$ that there exists an initial run ρ reaching a configuration γ such that $\tilde{\gamma} \trianglelefteq \gamma$ (note that if $\tilde{\gamma}$ covers a state, then so does γ).

If $\tilde{\gamma} = \tilde{\gamma}_0$ then $\rho = \tilde{\rho}$ suffices. Suppose that $\tilde{\rho}$ has length $k \geq 1$, and that the result if true for runs of length $k-1$. Decompose $\tilde{\rho}$ into $\tilde{\rho}_{k-1} : \tilde{\gamma}_0 \xrightarrow{*} \tilde{\gamma}_{k-1}$ of length $k-1$ and a final step $\tilde{\gamma}_{k-1} \to \tilde{\gamma}_k$. By induction hypothesis, there exists $\rho_{k-1} : \gamma_0 \xrightarrow{*} \gamma_{k-1}$ such that $\tilde{\gamma}_{k-1} \trianglelefteq \gamma_{k-1}$: there exists an injective function $\pi : \tilde{\mathbb{A}} \to \mathbb{A}$ such that, for all $a \in \tilde{\mathbb{A}}$, $\tilde{\gamma}_{k-1}(a) = \gamma_{k-1}(\pi(a))$, where $\tilde{\mathbb{A}} := \mathsf{ag}(\tilde{\rho})$ and $\mathbb{A} := \mathsf{ag}(\rho)$. If $\tilde{\gamma}_{k-1} \to \tilde{\gamma}_k$ involves no reception transition from $\tilde{\mathcal{P}}$ whose corresponding transition in $\mathcal{P}$ has action '$\neq$', then we directly lift this step into a step appended at the end of ρ_{k-1} (making $\pi(a)$ take a transition whenever a

does so in $\tilde{\gamma}_{k-1} \to \tilde{\gamma}_k$). Otherwise, write $\tilde{\mathbb{A}}_{\neq}$ the subset of $\tilde{\mathbb{A}}$ corresponding to agents taking in $\tilde{\gamma}_{k-1} \to \tilde{\gamma}_k$ a reception transition from $\tilde{\mathcal{P}}$ whose corresponding transition in $\mathcal{P}$ has action '$\neq$' . Write $(q, \mathbf{br}(m), q') \in \Delta$ the broadcast transition used in this step. Using the copycat principle, we add to γ_{k-1} a fresh agent a_{new} with state q and a register value that does not appear in γ_{k-1}. We first mimic this broadcast step at the end of ρ_{k-1}, making any agent $\pi(a) \in \pi(\tilde{\mathbb{A}} \setminus \tilde{\mathbb{A}}_{\neq})$ take the transition that a takes in $\tilde{\gamma}_{k-1} \to \tilde{\gamma}_k$. We then add a new step where a_{new} broadcasts using transition $(q, \mathbf{br}(m), q')$, and every agent $\pi(a) \in \pi(\tilde{\mathbb{A}}_{\neq})$ takes the transition corresponding to the transition taken by a in $\tilde{\gamma}_{k-1} \to \tilde{\gamma}_k$. Such a transition is a reception with action '$\neq$' in $\mathcal{P}$; however, because a_{new} does not share its register value with any process from $\tilde{\mathbb{A}}$, all disequality conditions are satisfied and this step is valid. In the end, every agent $\pi(a) \in \pi(\tilde{\mathbb{A}})$ has taken the transition in $\mathcal{P}$ corresponding to the one a took in $\tilde{\mathcal{P}}$ in step $\tilde{\gamma}_{k-1} \to \tilde{\gamma}_k$, hence the configuration γ_k reached by the constructed run is such that $\tilde{\gamma}_k \trianglelefteq \gamma_k$. □

I.2 Abstraction

We now define our abstraction. We formalize the definition of a gang:

Definition 44. *Let (Q, Δ, q_0) be a protocol.*

A gang is a pair $\mathsf{G} = (\mathsf{b}, \mathsf{K}) \in (Q \cup \{\bot\}) \times 2^Q$. The element b is the boss and the set K is the clique of the gang.

Let $\rho = \gamma_0 \to \gamma_1 \to \cdots \to \gamma_k$ be a run and $v \in \mathsf{val}(\rho)$. The gang of value v in ρ, written $\mathsf{gang}_v(\rho)$, is the gang $(\mathsf{b}_v(\rho), \mathsf{K}_v(\rho))$ such that,

- *if there exists $a_0 \in \mathsf{ag}(\rho)$ such that, for every $i \in [0, k]$, $\mathsf{data}(\gamma_i)(a_0) = v$ then $\mathsf{b}_v(\rho) := \mathsf{st}(\gamma_k)(a_0)$, otherwise $\mathsf{b}_v(\rho) := \bot$,*
- *$\mathsf{K}_v(\rho) := \{q \in Q \mid \exists i \leq k, \exists a \in \mathbb{A} \setminus \{a_0\}, \gamma_i(a) = (q, v)\}$*

We define abstract runs as follows:

Definition 45. *An abstract configuration over $\mathbb{A}$ is a tuple of $2^Q \times \mathcal{G}$ where $\mathcal{G}$ designates the set of all gangs. We write $\Sigma_{\mathbb{A}}$ the set of abstract configurations over $\mathbb{A}$ and $\Sigma := \bigcup_{\mathbb{A} \subseteq \mathbb{N} \text{ finite}} \Sigma_{\mathbb{A}}$ the set of all abstract configurations.*

Given a set of states $S \subseteq Q$, a message type m and a set of operations O, we define $\overrightarrow{S^{m,A}} = \{s' \in Q \mid \exists s \in S, a \in O, (s, \boldsymbol{rec}(m, a), s') \in \Delta\}$.

Given two abstract configurations $\sigma = (S, \mathsf{b}, \mathsf{K})$ and $\sigma' = (S', \mathsf{b}', \mathsf{K}')$, there is an abstract step from σ to σ', denoted $\sigma \to \sigma'$, when $\mathsf{K}' \subseteq S'$, $\mathsf{b}' \in S' \cup \{\bot\}$ and one of the following cases is satisfied.

1. *Broadcast from clique: There exists $(q_{\boldsymbol{br}}, \boldsymbol{br}(m), q'_{\boldsymbol{br}}) \in \Delta$ such that:*
   (1) Either $\mathsf{b} = \mathsf{b}'$ or there exists $(\mathsf{b}, \boldsymbol{rec}(m, \alpha), \mathsf{b}') \in \Delta$ for some action α.
   (2) $\mathsf{K}' = \mathsf{K} \cup \{q'_{\boldsymbol{br}}\} \cup \overrightarrow{\mathsf{K}^{m,\{=,*,\downarrow\}}} \cup \overrightarrow{S^{m,\{\downarrow\}}}$.
2. *Broadcast from boss: there exists $m \in \mathcal{M}$ such that $(\mathsf{b}, \boldsymbol{br}(m), \mathsf{b}') \in \Delta$*
   (1) $\mathsf{b}, \mathsf{b}' \neq \bot$ (technically implied by the existence of $(\mathsf{b}, \boldsymbol{br}(m), \mathsf{b}')$ but written here to match other cases)

(2) $\mathsf{K}' = \mathsf{K} \cup \overrightarrow{\mathsf{K}^{m,\{=,*,\downarrow\}}} \cup \overrightarrow{S^{m,\{\downarrow\}}}$.

3. *External broadcast*: *There exists* $(q_{\boldsymbol{br}}, \boldsymbol{br}(m), q'_{\boldsymbol{br}}) \in \Delta$ *such that*
   (1) Either $\mathsf{b} = \mathsf{b}'$ *or:*
   - $\mathsf{b}' \neq \bot$ *and there exists* $(\mathsf{b}, \boldsymbol{rec}(m, *), \mathsf{b}') \in \Delta$, *or*
   - $\mathsf{b}' = \bot$ *and there exists* $(\mathsf{b}, \boldsymbol{rec}(m, \downarrow), \mathsf{b}') \in \Delta$.

   (2) $\mathsf{K}' = \mathsf{K} \cup \overrightarrow{\mathsf{K}^{m,\{*\}}}$.
4. *Gang reset*: $S' = S \cup \mathsf{K} \cup \{\mathsf{b}\}$, $\mathsf{K}' = \emptyset$ *and* $\mathsf{b}' = q_0$

Given a concrete run $\rho : \gamma_0 \xrightarrow{*} \gamma_k$, *we write* $\mathsf{abs}_v(\rho)$ *for the abstract configuration* $(S, \mathsf{gang}_v(\rho))$ *where* S *is the set of all states appearing in* ρ.

The initial abstract configuration *is* $\sigma_0 := (\{q_0\}, q_0, \emptyset)$. *As in the concrete case, an* abstract run *is a sequence* $\nu = \sigma_0, \ldots, \sigma_k$ *such that* σ_0 *is the initial configuration and, for all* i, $\sigma_i \to \sigma_{i+1}$. *We denote such a run* $\sigma_0 \xrightarrow{*} \sigma_k$. *Similarly, we denote by* $\sigma \xrightarrow{*} \sigma'$ *the existence of a sequence of steps from* σ *to* σ'.

The intuition is that we will keep track of one value at a time, while assuming that we have unlimited supplies of agents in the states we covered so far. We follow the gang of one value through the run, which allows us to discover new states. A gang reset lets us add those new states to the set of covered ones and switch to another value.

First of all we observe that if there is an abstract run covering a state then there is a short one.

Lemma 46. *For every* $\sigma \in \Sigma$ *such that* $\sigma_0 \xrightarrow{*} \sigma$, *there exists an abstract run* $\nu : \sigma_0 \xrightarrow{*} \sigma$ *of less that* $(|Q| + 2)^3$ *steps.*

Proof. Note that S may never decrease along an abstract run and that K may only decrease at gang resets. We can hence enforce in the abstract semantics that, at least every $|Q| + 2$ steps without reset, either S or K has increased. Indeed, otherwise the configuration has looped as the boss may only take $|Q|+1$ values. We may also enforce that S has strictly increased between two resets, as otherwise one may remove anything that happened between the two resets. Therefore, there are at most $|Q| - 1$ gang resets in total, and each portion of the run with no reset has at most $(|Q| + 2)(|Q| + 1)$ steps, yielding the bound. □

It remains to prove that our abstraction is sound and complete.

Completeness In this subsection we prove Lemma 48. To do so, we take a concrete run ρ in our model and any value v appearing in the reached configuration. We prove that there exists an abstract run leading to the abstract configuration $(\mathsf{cov}(\rho), \mathsf{gang}_v(\rho))$.

To construct the abstract run, we will first show that for all S such that $\mathsf{cov}(\rho) \subseteq S$, we can keep track of the set of agents carrying a value v: $(S, q_0, \emptyset) \xrightarrow{*} (S, \mathsf{b}_v(\rho), K)$ with $\mathsf{K}_v(\rho) \subseteq K$. Then, it is left to show that for all concrete run ρ and for all value v, $\sigma_0 \xrightarrow{*} (S, q_0, \emptyset)$ with $\mathsf{cov}(\rho) \subseteq S$.

Lemma 47. *For all initial runs* $\rho : \gamma_0 \xrightarrow{*} \gamma$, $S \subseteq Q$ *and* $v \in \mathsf{val}(\rho)$, *if* $\mathsf{cov}(\rho) \subseteq S$ *then there exists* K *such that* $(S, q_0, \emptyset) \xrightarrow{*} (S, \mathsf{b}_v(\rho), \mathsf{K})$ *and* $\mathsf{K}_v(\rho) \subseteq \mathsf{K}$.

Proof. Let $\mathbb{A} = \mathsf{ag}(\rho)$. As v appears in ρ, it must appear in γ_0; let a_0 be the unique agent such that $\mathsf{data}(\gamma_0)(a_0) = v$. We write $\rho : \gamma_0 \to \gamma_1 \to \ldots \to \gamma_k = \gamma$. For every $i \leq k$, let $\rho_i : \gamma_0 \xrightarrow{*} \gamma_i$ be the prefix of ρ of length i. We set $\sigma^0 = (S, q_0, \emptyset)$.

We construct by induction on i a sequence of abstract configurations $\sigma^i = (S, \mathsf{b}_i, \mathsf{K}_i)$ such that $\sigma^0 \xrightarrow{*} \sigma^i$ and $\mathsf{K}_v(\rho_i) \subseteq \mathsf{K}_i$. The statement is clear for $i = 0$.

Suppose now that $(S, q_0, \emptyset) \xrightarrow{*} \sigma^i$. If suffices to prove that $\sigma^i \to \sigma^{i+1}$. We consider the last step of ρ_{i+1}, which is referred to under the name s_{i+1} in what follows; $s_{i+1} : \gamma_i \to \gamma_{i+1}$. Let $a_{\mathbf{br}}$ the agent making the broadcast transition in s_{i+1} and $A_{\mathbf{rec}}$ the set of agents receiving this broadcast in s_{i+1}. Let $(q_{\mathbf{br}}, \mathbf{br}(m), q'_{\mathbf{br}}) \in \Delta$ denote the transition taken by $a_{\mathbf{br}}$ in s_{i+1}.

We now make the following case distinction to determine the type of the abstract step $\sigma^i \to \sigma^{i+1}$:

1. if $\mathsf{data}(\gamma_i)(a_{\mathbf{br}}) = v$ but there exists $j < i$ such that $\mathsf{data}(\gamma_j)(a_{\mathbf{br}}) \neq v$ then it is a broadcast from clique,
2. if, for all $j \leq i$, $\mathsf{data}(\gamma_j)(a_{\mathbf{br}}) = v$ then it is a broadcast from boss,
3. otherwise it is an external broadcast.

Note that $a_{\mathbf{br}}$ may not change its register value in s_{i+1} hence $\mathsf{data}(\gamma_i)(a_{\mathbf{br}}) = \mathsf{data}(\gamma_{i+1})(a_{\mathbf{br}})$.

Let a_{boss} the agent such that $\mathsf{data}(\gamma_0)(a_{\mathsf{boss}}) = v$. In case 2, $a_{\mathsf{boss}} = a_{\mathbf{br}}$; in the other two cases, $a_{\mathsf{boss}} \neq a_{\mathbf{br}}$.

It is easy to check that in all cases, the gang given by applying the abstract semantics covers $\mathsf{gang}_v(\rho_{i+1})$:

(1) In case 2, this condition is automatically satisfied. In the other two cases, we look at what a_{boss} does in s_{i+1}. If it remains idle then we have $\mathsf{b}_i = \mathsf{b}_{i+1}$. Otherwise it takes a reception transition as $a_{\mathbf{br}} \neq a_{\mathsf{boss}}$. In case 1 the condition is then satisfied. In case 3, this reception cannot have action '=' as the broadcast is from an agent with register value that is not v ($\mathsf{data}(\gamma_i)(a_{\mathbf{br}}) \neq v$ by hypothesis). For the same reason, if this reception has action '$\downarrow$' then $\mathsf{b}_{i+1} = \bot$. If this reception has action '$*$' and $\mathsf{b}_i \neq \bot$ then $\mathsf{b}_{i+1} \neq \bot$ as a_{boss} keeps value v.

(2) In all cases K_{i+1} is defined by adding to K_i all states reachable with value v from it by receiving the broadcast message m from a state of K_i with value v or from a state of S with value $v' \neq v$ (plus $q'_{\mathbf{br}}$ in case 1). As $\mathsf{K}_v(\rho_i) \subseteq K_i$, necessarily $\mathsf{K}_v(\rho_{i+1}) \subseteq K_{i+1}$

We have proven that $\sigma^i \to \sigma^{i+1}$, which concludes the induction step. Applying the result with $i = k$ proves Lemma 47. □

We may now prove completeness of the abstraction. Intuitively, we start with $S = \{q_0\}$ and we increase it in the following way: we look at the first state $q \notin S$

that is covered in ρ, we follow the gang associated with the value of an agent that covered q until the gang covers it too (using Lemma 47), then we do a gang reset to add q to S.

Lemma 48. *If ρ is an initial run and $v \in \mathsf{val}(\rho)$, then there exist S, K such that $\sigma_0 \xrightarrow{*} (S, \mathsf{b}_v(\rho), \mathsf{K})$ and $\mathsf{cov}(\rho) \subseteq S$ and $\mathsf{K}_v(\rho) \subseteq \mathsf{K}$.*

Proof. Let ρ an initial run. We construct by induction an increasing sequence of sets $C_0, \dots, C_m$ such that $C_m = \mathsf{cov}(\rho)$ and for all i, $C_i \subseteq \mathsf{cov}(\rho)$, $q_0 \in C_i$, and there exists S_i such that $\sigma_0 \xrightarrow{*} (S_i, q_0, \emptyset)$ and $C_i \subseteq S_i$. First, we set $C_0 = \{q_0\}$ and the property is verified as $\sigma_0 \xrightarrow{*} (\{q_0\}, q_0, \emptyset)$.

Now suppose we constructed $C_0, \dots, C_j$. If $C_j = \mathsf{cov}(\rho)$ we can stop. Otherwise let $\rho_p : \gamma_0 \xrightarrow{*} \gamma_p$ the longest suffix of ρ such that $\mathsf{cov}(\rho_p) \subseteq C_j$. Write s the step immediately after ρ_p in ρ. By maximality of ρ_p, $\rho_p \cdot s$ covers some state q that is not in C_j. Let a be an agent that is in q after step s, let v be its value at that point. We set $C_{i+1} = C_i \cup \{q\}$.

By induction hypothesis, there exist S_j such that $\sigma_0 \xrightarrow{*} (S_j, q_0, \emptyset)$ and $C_j \subseteq S_j$. Furthermore, as $\mathsf{cov}(\rho_p) \subseteq C_j$, by Lemma 47 there exists K_j such that $\sigma_0 \xrightarrow{*} (S_j, \mathsf{b}_v(\rho_p), \mathsf{K}_j)$ and $\mathsf{K}_v(\rho_p) \subseteq \mathsf{K}_j$.

If $q \in \mathsf{K}_j$ then applying a gang reset suffices. If not, we mimic step s with a step in the abstract semantics, as in Lemma 47 so that q is added to the clique, then apply a gang reset to reach $(S_{i+1}, q_0, \emptyset)$ with $C_{i+1} = S_i \cup \{q\} \subseteq S_{i+1}$.

This concludes our induction.

In the end, there exist S_m such that $\sigma_0 \xrightarrow{*} (S_m, q_0, \emptyset)$ and $\mathsf{cov}(\rho) \subseteq S_j$. Lemma 47 allows us to conclude the proof.¨ □

Soundness It is left to prove that our abstraction is sound, which we will do by considering an abstract run $\sigma_0 \xrightarrow{*} \sigma = (S, \mathsf{b}, \mathsf{K})$ and constructing a concrete run $\rho : \gamma_0 \xrightarrow{*} \gamma$ that follows the abstract run with an exponential amount of agents to make sure that we can send some of them through all possible transitions at each step and never run out.

Lemma 49. *For all $\sigma_0 \in \Sigma_{\mathsf{init}}$ and $\sigma = (S, b, K) \in \Sigma$ such that $\sigma_0 \xrightarrow{*} \sigma$, for all $s \in S$, there exists a reachable configuration γ covering s.*

We in fact prove the following stronger lemma, which directly implies Lemma 49.

Lemma 50. *Let $\sigma_0 \in \Sigma_{\mathsf{init}}$, and $\sigma_0 \to \sigma_1 \to \cdots \to \sigma_n$ an abstract run. For all i let $(S_i, b_i, K_i) := \sigma_i$. Let $M = |\Delta| + 1$.*

For all i, there exist a set of agents $\mathbb{A}_i$, an initial run $\rho_i : \gamma_0 \xrightarrow{*} \gamma_i$ over $\mathbb{A}_i$, agents $a_0, \cdots, a_n \in \mathbb{A}_i$ and values $v_0, \dots, v_n \in \mathbb{N}$ such that:

- *for all $s \in S_i$, there are at least M^{n-i} agents (different from a_i) in state s*
- *for all $s \in K_i$, there are at least M^{n-i} agents (different from a_i) in state s with value v_i*
- *if $b_i \neq \bot$, then a_i is in state b_i with value v_i.*

Proof. We proceed by induction on i. We set $\mathbb{A}_0 = \{1, \dots, M^n\}$, and we set $\gamma_0(a) = (q_0, a)$ for all a. Clearly γ_0 satisfies the requirements with respect to σ_0, with $a_0 = v_0 \in \mathbb{A}$.

Now assume we constructed $\gamma_0 \xrightarrow{*} \cdots \xrightarrow{*} \gamma_i$ over $\mathbb{A}_i$ satisfying the conditions of the lemma, we construct γ_{i+1} using a case distinction on the form of the transition $\sigma_i \to \sigma_{i+1}$. For each $s \in S \setminus K$ we define $\mathbb{A}_{i,s}$ as the set of agents in state s in γ_i. We have $|\mathbb{A}_{i,s}| \geq M^{n-i}$ thus we can extract $M = |\Delta| + 1$ disjoint sets of agents $(\mathbb{A}^d_{i,s})_{d \in \Delta \cup \{\varepsilon\}}$ from it, each set having M^{n-i-1} agents. Similarly, for each $s \in K$ we define $\mathbb{A}_{i,s}$ as the set of agents in state s **with value** $\mathbf{v_i}$ in γ_i. We have $|\mathbb{A}_{i,s}| \geq M^{n-i}$ thus we can extract $|\Delta| + 1$ disjoint sets of agents $(\mathbb{A}^d_{i,s})_{d \in \Delta \cup \{\varepsilon\}}$ from it each set having M^{n-i-1} agents.

Case 1: If $\sigma_i \to \sigma_{i+1}$ is a broadcast from clique $d = (q, \mathbf{br}(m), q')$ with $q \in K_i$, then we make all agents $a \in \mathbb{A}^d_{i,q}$ (which all have value v_i) execute that transition one by one. None of those broadcasts are received by any other agent, except for the last one: If $b \neq b'$ then there is a transition $(b, \mathbf{rec}(m, \alpha), b')$ and we make a_i execute it upon receiving the broadcast. We then set $a_{i+1} = a_i$. For all $k' \in K_{i+1} \setminus (K_i \cup \{q'\})$ there exists a transition $d' = (k, \mathbf{rec}(m, \alpha), k')$ such that either α is $=$ or $*$ and $k \in K_i$ or α is $\downarrow$ and $k \in S$. In both cases we make all agents of $\mathbb{A}^{d'}_{i,k}$ take that transition. We set $v_{i+1} = v_i$.

Case 2: If $\sigma_i \to \sigma_{i+1}$ is a broadcast from boss $d = (b_i, \mathbf{br}(m), b_{i+1})$, then we make a_i (which has value v_i) execute that transition, and we set $a_{i+1} = a_i$. The agents receiving that message are as follows:

For all $k' \in K_{i+1} \setminus K_i$ there exists a transition $d' = (k, \mathbf{rec}(m, \alpha), k')$ such that α is either $=$ or $*$ and $k \in K_i$ or α is $\downarrow$ and $k \in S$. In both cases we make all agents of $\mathbb{A}^{d'}_k$ take that transition. By definition of an abstract run, we must have $b_i \in S_i$. Hence we can make all agents of $\mathbb{A}^d_{i,s}$ execute d, with no agent receiving the corresponding broadcasts. We set $v_{i+1} = v_i$.

Case 3: If $\sigma_i \to \sigma_{i+1}$ is an external broadcast $d = (q, \mathbf{br}(m), q')$, then we make all agents $a \in \mathbb{A}^d_q$ execute that transition one by one. None of those broadcasts are received by any other agent, except for the last one: If $b_i \neq b_{i+1}$ then there is a transition $(b_i, \mathbf{rec}(m, \alpha), b')$ and either $b_{i+1} = b'' \neq \bot$ and $\alpha = *$ or $b_{i+1} = \bot$ and $\alpha = \downarrow$. In both cases we make a_i execute that transition, and we set $a_{i+1} = a_i$. For all $k' \in K_{i+1} \setminus K_i$ there exists a transition $d' = (k, \mathbf{rec}(m, *), k')$ with $k \in K_i$. We make all agents of $\mathbb{A}^{d'}_k$ take that transition. We set $v_{i+1} = v_i$.

Case 4: If $\sigma_i \to \sigma_{i+1}$ is a gang reset then no agent moves and we select some a_{i+1} in $\mathbb{A}_{q_0}$ and set v_{i+1} to be its value.

Throughout the case distinction we have ensured that:

- If $b_{i+1} \neq \bot$ then a_{i+1} is an agent of value v_{i+1}.
- If the step is not a gang reset, then $v_{i+1} = v_i$ and for all $k' \in K_{i+1} \setminus K_i$, there exists $d \in \Delta$ from some k to k' such that all agents of $\mathbb{A}^d_{i,k}$ take

that transition. Furthermore, if d is of the form $(k, \mathbf{rec}(m, \downarrow), k')$ then the broadcasting agent has value v_i, thus all those agents keep value $v_i = v_{i+1}$. For all $k \in K_i$, the agents of $\mathbb{A}^{\varepsilon}_{i,k}$ do not move between configurations γ_i and γ_{i+1}, hence they have state k and value v_{i+1} in γ_{i+1}.
– If the step is a gang reset, the conditions of the lemma hold trivially.

As a result, we have ensured that the conditions of the lemma were respected. This concludes our induction. □

To obtain Lemma 49, we apply Lemma 50 to an abstract run $\sigma_0 \to \cdots \sigma_n = \sigma$ from σ_0 to σ by setting $i = n$.

I.3 Conclusion

Proposition 51. *Let q_f be a state, there exists a reachable configuration covering q_f if and only if there exists a reachable abstract configuration (S, b, K) with $q_f \in S$.*

Proof. The two implications follow from Lemmas 48 and 49 respectively. □

NP-hardness. We present here a reduction from the 3SAT problem to the cover problem in 1-BNRAs.

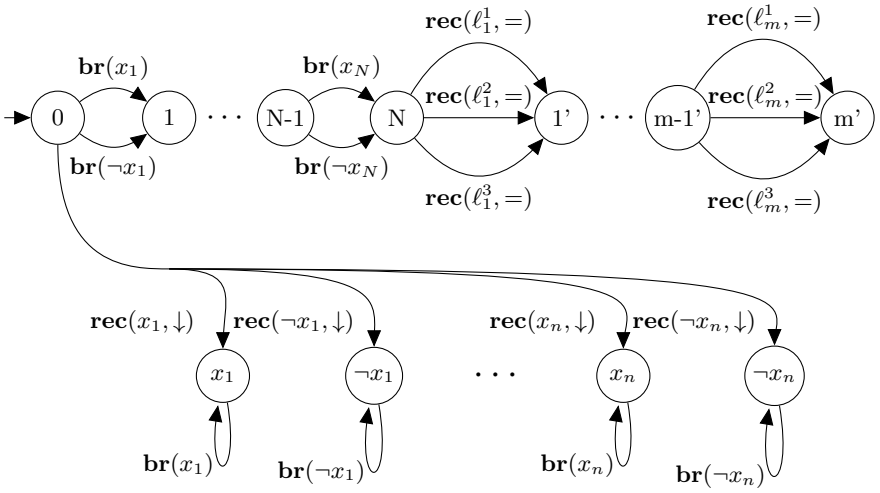


Fig. 12: The protocol used for the NP-hardness proof.

Proposition 52. *The cover problem is NP-hard.*

Proof. Let $x_1, \dots, x_n$ be variables and $\phi = \bigwedge_{j=1}^{m} C_j$ with, for all j, $C_j = \ell_j^1 \vee \ell_j^2 \vee \ell_j^3$ and $\ell_j^1, \ell_j^2, \ell_j^3 \in \{x_i, \neg x_i \mid 1 \leq i \leq n\}$.

Consider the protocol displayed in Figure 12. Our alphabet of messages is the set of literals $\{x_i, \neg x_i \mid 1 \leq i \leq n\}$. Agents may either receive a message, and repeat it forever or it may broadcast one of $x_i, \neg x_i$ for each i and then try to receive a message with one of $\ell_j^1, \ell_j^2, \ell_j^3$ for each j, with their own register value.

If ϕ is satisfied by some assignment ν, then we construct a run where an agent a broadcasts the satisfied literals while going from 0 to N and other agents receive the messages and go to the corresponding states in the lower part of the protocol while storing the register value of a. Then for each j we select some ℓ_j^p satisfied by ν. There exists i such that there is an agent in state ℓ_j^p, which broadcasts ℓ_j^p along with the initial register value of a, allowing a to go to the next state. As a result, there is a run in which an agent a reaches m'.

Now suppose there is a run ρ over some set of agents $\mathbb{A}$ such that some agent $a \in \mathbb{A}$ is in state m' in the final configuration. For each i, a has broadcast either x_i or $\neg x_i$, but not both. Let ν be the valuation assigning $\top$ to x_i if and only if a has broadcast it. For each j a has received one of $\ell_j^1, \ell_j^2, \ell_j^3$ along with its own initial register value (which we call r). For this to happen, a must have broadcast this literal before, hence it is satisfied by ν.

As a result, ν satisfies a literal of each clause of ϕ, and thus satisfies ϕ. This concludes our reduction. □

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

Proof. The lower bound is given by Proposition 52. For the upper bound, say we are given a protocol $\mathcal{P} = (Q, \mathcal{M}, \Delta, q_0, r)$ and a state q_f. By Proposition 51, there is a reachable configuration covering q_f if and only if there is an abstract run to an abstract configuration (S, b, K) with $q_f \in S$. Furthermore, by Lemma 46 if there is such an abstract run then there is one with at most $(|Q| + 2)^3$ steps. Thus we can simply guess such an abstract run and verify it in polynomial time. As a result, the cover problem is in NP. □