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

Lucie Guillou
IRIF, CNRS, Université Paris Cité

Corto Mascle
LaBRI, Université de Bordeaux

Nicolas Waldburger
IRISA, Université de Rennes

Abstract

We consider the parameterized verification of arbitrarily large networks of agents which communicate by broadcasting and receiving messages. In our model, the broadcast topology is reconfigurable so that a sent message can be received by any set of agents. In addition, agents have local registers which are 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 this value for equality with 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; however, it is as hard as coverability in lossy channel systems, which is non-primitive recursive. This model lies at the frontier of decidability as other classical problems on this model are undecidable; this is in particular true for the target problem where all processes must synchronize on a given state. By contrast, we show that the coverability problem is NP-complete when each agent has only one register.

2012 ACM Subject Classification Theory of computation → Distributed computing models

Keywords and phrases Parameterized verification Well quasi-orders Distributed systems

Digital Object Identifier 10.4230/LIPIcs...1

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 arbitrarily large number of agents whose behavior is specified with a finite automaton equipped with a finite set of private registers. The registers of the agents contain values from an infinite set (represented by natural numbers). Initially, agents have distinct values in their registers: one can think of these values as identifiers. An agent may send messages made of a symbol from a finite alphabet along with the value of one of its registers. No assumption is made on the evolution of the communication graph, i.e., when an agent broadcasts a message, any subset of the other agents may receive it; this is meant to model unreliable systems with unexpected crashes and disconnections. Upon reception of a message, an agent may store the received value in one of its registers or test it for equality with one of its own values. This allows an agent, for instance, to check the origin of received messages: to do so, the broadcasting agent signs messages with its identifie, and the receiving agent stores this identifier then checks that all incoming messages have this value. This also allows an agent to obtain acknowledgement that its message was received by someone, by broadcasting with its identifier and waiting for a response with that same identifier.

This model was introduced in [13], as a natural extension of Reconfigurable Broadcast Networks [15]. That first paper claimed that the coverability problem, i.e., the problem of whether there is a run from an initial configuration to one where at least one agent is in a designated state, was decidable and even PSPACE-complete, but the proof turned out to be wrong [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. We even prove its completeness for the hyper-Ackermannian complexity class $\mathbf{F}_{\omega^\omega}$, thereby showing that the problem requires a non-primitive recursive amount of time. In fact, this problem is as hard as reachability for lossy channel systems, which are transition systems with a finite automaton that can store some letters in an unreliable FIFO memory from which any letter may be erased at any time [4, 11, 25]. We further establish that this problem lies at the frontier of decidability by showing undecidability of the target problem (where one asks whether there is a run at the end of which all agents are in a given state); we contrast these results with the NP-completeness of the coverability problem when 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. There are however many choices to make when designing a model for those systems: how individual processes are represented, whether the communication graph is fixed or can change, the type of messages they can send... A model where messages range over a finite alphabet was presented in [16], over a fully connected communication graph. It was rapidly shown that many basic parameterized problems are undecidable over that model [17]; similar negative results were found for Ad Hoc Networks where the communication graph is fixed but arbitrary [15]. This lead the community to consider Reconfigurable Broadcast Networks (RBN) where each broadcast can be received by an arbitrary subset of agents [15].

Parameterized verification problems over RBN have been the subject of extensive study in recent years [7, 8, 12, 14]. In [13], RBN were extended to BNRA, the model studied in this article, by the addition of registers allowing processes to exchange identifiers. This extension was inspired by the success of register automata, which offer a convenient formalism to express properties of words over an infinite alphabet; see [26] for a survey on the subject.

Other approaches exist to define parameterized models with registers [9], 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].

Such parameterized verification problems often relate to the theory of well quasi-orders and the associated high complexities obtained from bounds on "bad sequences" in ordered sets. In particular, our model is linked to two classical models from this field. The first one is data nets, which are Petri nets in which tokens are labeled with natural numbers and can exchange and compare their labels [19]. In general, inequality tests are allowed, but data nets with only equality tests have also been studied [21]. They do not subsume BNRA as, in data nets, each process can only carry one integer at a time (problems on models of data nets where tokens have tuples of integers as labels are typically undecidable). The second closely related model is lossy channel systems (LCS) [4]. LCS are derived from distributed models where processes communicate through pairwise channels; this model is a rich field of study in itself [5, 6]. LCS reachability is complete for the complexity class $\mathbf{F}_{\omega^\omega}$ [11, 25]; we show that the same is true for BNRA coverability and that LCS can be simulated in BNRA.

Overview We start with the model definition and some preliminary results in Section 2. We prove decidability of the coverability problem in Section 3. Finally, we prove the NP-completeness of the coverability problem with one register per process in Section 4. Due to space constraints, most proofs are postponed to the appendix.

2 Preliminaries

2.1 Definitions of the Model

A *Broadcast Network of Register Automata* (BNRA) [13] is a model describing broadcast networks of agents with local registers. A finite transition system describing the behavior of an agent; an agent can broadcast and receive messages with integer values, store them in local registers and perform equality tests. There are arbitrarily many agents. When an agent broadcasts a message, each other agent independently may or may not receive it.

▶ **Definition 1** (Protocols). *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 a set of operations*

$$\mathsf{Op} = \{\boldsymbol{br}(m,i), \boldsymbol{rec}(m,i,*), \boldsymbol{rec}(m,i,\downarrow), \boldsymbol{rec}(m,i,=), \boldsymbol{rec}(m,i,\neq) \mid m \in \mathcal{M}, 1 \leq i \leq r\}$$
$$\cup \{\boldsymbol{loc}(i,j,=), \boldsymbol{loc}(i,j,\neq) \mid 1 \leq i,j \leq r\}$$

*Label **br** stands for broadcasts, **rec** for receptions and **loc** for local tests. Given a reception $\boldsymbol{rec}(m,i,\alpha)$ or a local test $\boldsymbol{loc}(i,j,\alpha)$, 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 is either an agent performing a local action or an agent broadcasting a message received by some other agents.

▶ **Definition 2** (Semantics). *Let $(Q, \mathcal{M}, \Delta, q_0)$ be a protocol with r registers, and $\mathbb{A} \subseteq \mathbb{N}$ a finite non-empty set of agents. A configuration over a set of agents $\mathbb{A}$ is a function $\gamma : \mathbb{A} \to Q \times \mathbb{N}^r$ mapping each agent to a state and register values. We write $\mathsf{st}(\gamma)$ for the state component of γ and $\mathsf{data}(\gamma)$ for its register component. An initial configuration γ is one where for all $a \in \mathbb{A}$, $\mathsf{st}(\gamma)(a) = q_0$ and $\mathsf{data}(\gamma)(a,i) \neq \mathsf{data}(\gamma)(a',i')$ for all $(a,i) \neq (a',i')$.*

Given a finite set of agents $\mathbb{A}$ and two configurations γ, γ' over $\mathbb{A}$, a step $\gamma \to \gamma'$ is defined when one of the two following conditions is satisfied:

- *there exist $a \in \mathbb{A}$, $i,j \in [1,r]$ and a local test $(\mathsf{st}(\gamma)(a), \boldsymbol{loc}(i,j,\alpha), \mathsf{st}(\gamma')(a)) \in \Delta$ such that $\gamma(a') = \gamma'(a')$ for all $a' \neq a$, $\mathsf{data}(\gamma')(a) = \mathsf{data}(\gamma)(a)$ and*
  - *if $\alpha =$ '$=$' then $\mathsf{data}(\gamma)(a,i) = \mathsf{data}(\gamma)(a,j)$,*
  - *if $\alpha =$ '$\neq$' then $\mathsf{data}(\gamma)(a,i) \neq \mathsf{data}(\gamma)(a,j)$;*
- *there exist $m \in \mathcal{M}$, $a_0 \in \mathbb{A}$ and $i \in [1,r]$ s.t. $(\mathsf{st}(\gamma)(a_0), \boldsymbol{br}(m,i), \mathsf{st}(\gamma')(a_0)) \in \Delta$, $\mathsf{data}(\gamma)(a_0) = \mathsf{data}(\gamma')(a_0)$ 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 is a sequence of steps $\pi : \gamma_0 \to \gamma_1 \to \cdots \to \gamma_k$. We write $\gamma_0 \xrightarrow{*} \gamma_k$ when there exists such a run. A run is initial when γ_0 is an initial configuration. A run $\rho : \gamma_0 \xrightarrow{*} \gamma_f$ covers a state $q \in Q$ when there exists $a \in \mathbb{A}$ such that $\mathsf{st}(\gamma_f)(a) = q$.

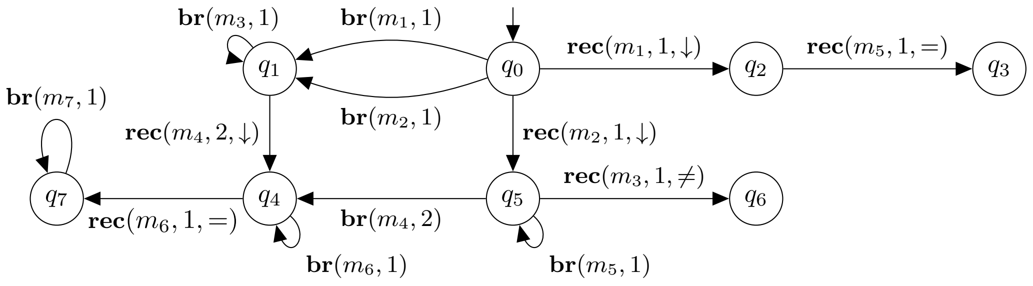


Figure 1 Example of a protocol.

▶ Remark 3. In our model, agents may only send one value per message. Indeed, [13] establishes undecidability of coverability if agents can broadcast two values at once.

▶ **Example 4.** We give an example of a protocol with 2 registers in Figure 1. Let $\mathbb{A} = \{a_1, a_2\}$. We denote a configuration γ over $\mathbb{A}$ by $\langle(\mathsf{st}(\gamma)(a_1), \mathsf{data}(\gamma)(a_1)), (\mathsf{st}(\gamma)(a_2), \mathsf{data}(\gamma)(a_2))\rangle$. The following sequence is an initial run, where x_1, y_1, x_2, y_2 are distinct natural integers:

$$\langle(q_0,(x_1,y_1)),(q_0,(x_2,y_2))\rangle \to \langle(q_1,(x_1,y_1)),(q_5,(x_1,y_2))\rangle \to$$
$$\langle(q_4,(x_1,y_2)),(q_4,(x_1,y_2))\rangle \to \langle(q_7,(x_1,y_2)),(q_4,(x_1,y_2))\rangle \to \langle(q_7,(x_1,y_2)),(q_4,(x_1,y_2))\rangle$$

The broadcast messages are, in this order: (m_2, x_1) by a_1, (m_4, y_2) by a_2, (m_6, x_1) by a_1 and (m_7, x_1) by a_1. In this run, each broadcast message is received by the other agent.

▶ Remark 5. We make the following observation: from a run $\rho : \gamma_0 \xrightarrow{*} \gamma$, we can build a run in which each agent of ρ has a clone in the same state but with different register values. To obtain this, it suffices to add new agents that mimic ρ in parallel of the original agents, with which they will not share any register values. 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 an initial run of $\mathcal{P}$ that covers q_f.*

The target reachability problem TARGET *asks, given a protocol $\mathcal{P}$ and a state q_f, whether there is an initial run $\gamma_0 \xrightarrow{*} \gamma_f$ of $\mathcal{P}$ such that $\mathsf{st}(\gamma_f)(\mathbb{A}) = \{q_f\}$, i.e all agents end on q_f.*

▶ **Example 7.** Let $\mathcal{P}$ the protocol of Example 1. As proven in Example 4, $(\mathcal{P}, q_7)$ is a positive instance of COVER. However, $(\mathcal{P}, q_7)$ is a negative instance of TARGET: there must be an agent staying on q_4 to broadcast m_6. Meanwhile, $(\mathcal{P}, q_1)$ is a positive instance of TARGET: all agents can broadcast m_1 to get to q_1. Also, $(\mathcal{P}, q_3)$ is a negative instance of COVER: we would need one agent on q_2 and one on q_5 with the same value in their first registers, hence we need a broadcast (m_1, v) and a broadcast (m_2, v) for some v. The two messages need to be sent by the agent having v as initial value, but this agent cannot send both messages.

▶ Remark 8. In [13], the authors considered "queries", which are conjunctions of formulas of the form '$q(\mathsf{z})$', '$\mathsf{reg}_j(\mathsf{z}) = \mathsf{reg}_{j'}(\mathsf{z}')$', '$\mathsf{reg}_j(\mathsf{z}) \neq \mathsf{reg}_{j'}(\mathsf{z}')$', with z, z' taken in a set of variables Var, $q \in Q$ and $j, j' \in [1, r]$. A configuration satisfies a query if we can assign an agent to each variable so that all conjuncts are satisfied. This problem reduces to COVER. Given a protocol $\mathcal{P}$ with r registers and a query ϕ with k variables, we can construct a protocol $\mathcal{P}'$ with $O(|\mathcal{P}|^k + |\phi|)$ states and kr registers that allows each agent to simulate k agents of the previous system. There is an initial run of the first BNRA satisfying ϕ if and only if there is one of the second in which one agent simulates the k agents satisfying ϕ; in order to cover q_f, this agent must check locally that ϕ is satisfied by the k agents it encodes. This reduction is exponential; note however that complexity class ($\mathbf{F}_{\omega^\omega}$) is stable by exponential reductions.

In the case of one register, we even have a polynomial-time reduction. To do so, we extend the protocol so that any agent can share its local configuration in a single broadcast (going to some sink state). In order to reach q_f, an agent must perform a sequence of transitions in which it receives the configurations of k agents and checks that they satisfy the query.

We can get rid of local equality tests at the cost of an exponential blow-up:

▶ **Proposition 9.** *There is an exponential-time reduction from* Cover *to* Cover *with no local equality tests* $\textbf{\textit{loc}}(i, j, =)$*.*

Proof sketch. From a protocol $\mathcal{P}$, we build a protocol $\mathcal{P}'$ whose registers are used to store values of multiple registers of $\mathcal{P}$, so that equality of two registers of $\mathcal{P}$ is encoded in the states of $\mathcal{P}'$. To do so, a state of $\mathcal{P}'$ stores which register of $\mathcal{P}$ is mapped to which register of $\mathcal{P}'$, hence the exponential blowup. The full proof can be found in Appendix A. ◀

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 moreover 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 problems which are decidable but with very high complexity (non-primitive recursive, and even non-multiply recursive).

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

Well-quasi orders For our decidability result, we rely on the theory of well quasi-orders in the context of subword ordering. Let Σ be a finite alphabet, $w_1, w_2 \in \Sigma^*$, w_1 is a *subword* of w_2, denoted $w_1 \preceq w_2$, when w_1 can be obtained from w_2 by erasing some letters. A sequence of words $w_0, w_1, \ldots$ is *good* if there exist $i < j$ such that $w_i \preceq w_j$, and *bad* otherwise. Higman's lemma [18] states that every bad sequence of words over a finite alphabet is finite. In order to bound the length of a bad sequence, 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 be a finite alphabet, let $g : \mathbb{N} \to \mathbb{N}$ be a primitive recursive function. There exists a function $f \in \mathscr{F}_{\omega^{|\Sigma|-1}}$ such that, for all $n \in \mathbb{N}$, every bad sequence $w_1, w_2, \ldots$ such that $|w_i| \leq g^{(i)}(n)$ for all i is of length at most $f(n)$.*

2.3 Link with 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 [10], reachability of a state is decidable for lossy channel systems [4], but has non-primitive recursive complexity [25] and is in fact $\mathbf{F}_{\omega^\omega}$-complete [11]. By simulating LCS using BNRA, we obtain our $\mathbf{F}_{\omega^\omega}$ lower bound for Cover:

▶ **Proposition 11.** *The coverability problem for BNRA is $\mathbf{F}_{\omega^\omega}$-hard.*

Proof sketch. Given an LCS $\mathcal{L}$, we build a protocol $\mathcal{P}$ with two registers. The first register is never modified and plays the role of a permanent identifier. Each agent starts by receiving a foreign identifier and storing it in its second register; it then only accepts messages with this identifier, using an equality test on every reception. This way, agents form chains where messages propagate in one direction and where each agent has at most one predecessor. Each agent of the chain simulates a step of an execution of $\mathcal{L}$: an agent 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. Some messages might get lost, hence the lossiness. The full proof can be found in Appendix B. ◀

▶ Remark 12. This reduction can be adapted to show that repeat-coverability (whether there is an infinite run covering q_f infinitely often) is undecidable for BNRA, as it is for LCS [3].

3 Cover Decidability

This section is dedicated to the proof of the main result of this paper:

▶ **Theorem 13.** *COVER for BNRA is decidable and $\mathbf{F}_{\omega^\omega}$-complete.*

Thanks to Proposition 9, we may assume that our protocols have no local equality tests (the complexity class $\mathbf{F}_{\omega^\omega}$ is stable by exponential reduction).

In order to abstract our runs, we want to understand what an agent a needs from other agents in order to cover a state q. In general, a may receive messages with several values from the same agent. However, because each message contains a single value, we will in fact be able to clone agents so that agents sending messages to a with distinct values are distinct and do not interact with each other.

We identify two types of roles that other agents must carry out, called *specifications*. First, they might need to broadcast a sequence of message types $w \in \mathcal{M}^*$ with a common value, so that a stores the value of the first such message and tests further messages for equality. We later call such a role a *boss specification*. It might also be the case that a sends some messages with a common value v that it had initially, then needs to receive a message (m, v) with that same value. To do so, some other agents should be able to broadcast (m, v) after receiving some sequence of message types w with that value v (that they did not have in their registers initially, because a did). We later call such a role a *follower specification*.

The two roles identified previously are the key to the decidability procedure. To represent runs, we consider *unfolding trees* that represent all such roles, dependencies between them and how they are carried out. The fact that we can consider trees and not general graphs is not obvious and is connected to the cloning idea mentioned above. The decidability procedure will rely on a bound of the minimum size of the unfolding tree one has to consider.

In Section 3.1, we introduce several useful notions. In Section 3.2, we define the notion of unfolding tree. In Section 3.3, we bound the size of the unfolding trees that we have to consider. In Section 3.4, we conclude by exposing our decidability procedure. In Section 3.5, we prove that the TARGET problem is, by contrast, undecidable.

3.1 Useful Definitions

We define a notion of local run, which may be seen as the projection of a run onto a given agent. In this local vision, we do not specify the origin of the received messages.

⌜ 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 $\delta = (q, \alpha, q')$

is a broadcast or a local test satisfied by ν. 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 one of the following holds:

- $\alpha =$ '$*$' and $\nu(j) = \nu'(j)$
- $\alpha =$ '$=$' and $\nu(j) = \nu'(j) = v$
- $\alpha =$ '$\downarrow$' and $\nu'(j) = v$
- $\alpha =$ '$\neq$' and $\nu(j) = \nu'(j) \neq v$.

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 $(q_0, \nu_0) \xrightarrow{*} (q, \nu)$. A value $v \in \mathbb{N}$ appearing in u is *initial* if it appears in ν_0 and *non-initial* otherwise. The *input* of u, written $\mathsf{In}(u) \in (\mathcal{M} \times \mathbb{N})^*$, is the sequence of messages of its reception steps, its *output*, written $\mathsf{Out}(u) \in (\mathcal{M} \times \mathbb{N})^*$, is the sequence of messages broadcast in u. For $v \in \mathbb{N}$, the v-*input* $\mathsf{In}_v(u)$ (resp. v-*output* $\mathsf{Out}_v(u)$) is the word $m_0 \cdots m_\ell \in \mathcal{M}^*$ such that $(m_0, v) \cdots (m_\ell, v)$ is the projection of $\mathsf{In}(u)$ (resp. $\mathsf{Out}(u)$) on $\mathcal{M} \times \{v\}$.

A *decomposition* is a tuple $\mathtt{dec} = (w_0, m_1, \ldots, m_\ell, w_\ell)$ with $w_0, \ldots, w_\ell \in \mathcal{M}^*$, 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* $\mathtt{dec} = (w_0, m_1, \ldots, m_\ell, w_\ell)$ 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 $\{m_1, \ldots, m_{j-1}\}$. We denote by $\mathcal{L}^{\mathtt{dec}}$ the language of words that admit decomposition $\mathtt{dec}$. This definition will be useful later; the intuition is that a decomposition describes the sequence of messages sent with some value v in a run. The w_i are message types sent by the agent with that value initially, and the m_i mark the times at which each message type is broadcast for the first time by another agent. This is all the information we need as if an agent manages to send some message (m, v) with a value v it did not have initially, then from this point on we can assume that we have an unlimited supply of messages (m, v), using (essentially) the copycat principle.

3.2 Unfolding Trees

An *unfolding tree* is an abstraction of a run in the form of a tree where each node is assigned a local run and a specification of its role. In this vision, the node at the root corresponds to the local run of the agent that we are interested in (*e.g.*, the agent covering q_f), and children nodes are here to provide messages that this agent needs to receive; such needs are expressed using specifications. A *boss specification* consists of a word $\mathsf{bw} \in \mathcal{M}^*$ describing a sequence of message types that should be broadcast all with the same value. A *follower specification* consists of a pair $(\mathsf{fw}, \mathsf{fm}) \in \mathcal{M}^* \times \mathcal{M}$, meaning that one must be able to broadcast fm with value v after receiving the sequence fw with value v.

We first provide the formal definition of unfolding trees. We will then explain this definition and why this notion is relevant for COVER.

▶ **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 $\mathbf{lr}(\mu)$, 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 in an unfolding tree 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 $\mathtt{dec} = (w_0, m_1, w_1, \ldots, m_\ell, w_\ell)$ 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, given τ *an unfolding tree, we define its size by* $|\tau| := \sum_{\mu \in \tau} |\mathbf{lr}(\mu)| + |\mathbf{spec}(\mu)|$. *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 now explain this definition. Let μ be a node of an unfolding tree τ and let $u := \mathbf{lr}(\mu)$. 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.

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). 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. Therefore, such an agent can be duplicated, and we may afterwards assume that we have an unlimited supply of messages (m, v). We split u into $u_0, \dots, u_\ell$ based on the first point where each type of message is received with value v. 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 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; 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 $\mathsf{Out}_{\mathbf{val}(\mu)}(u) \preceq \mathbf{bw}(\mu)$.

Our aim is to prove that we can study COVER directly on unfolding trees. We consider trees whose root is a boss node, as they suffice to witness coverability (a follower node implicitely relies on its parent's ability to broadcast some messages).

▶ **Definition 15.** *A coverability witness for* $(\mathcal{P}, q_f)$ *is an unfolding tree over* $\mathcal{P}$ *whose root* μ *is a boss node whose local run* $\mathbf{lr}(\mu)$ *covers* q_f.

▶ **Example 16.** In Figure 2 we display an unfolding tree obtained from the run of Example 4. Tables are local runs, columns are local configurations. For instance, the local run at μ_1 is

$$(q_0, (x_1, y_1)) \xrightarrow{\mathsf{int}((q_0, \mathbf{br}(m_2, 1), q_1))} (q_1, (x_1, y_1)) \xrightarrow{\mathsf{ext}((q_1, \mathbf{rec}(m_4, 2, \downarrow), q_4), y_2)} (q_4, (x_1, y_2))$$
$$\xrightarrow{\mathsf{ext}((q_4, \mathbf{rec}(m_6, 1, =), q_7), x_1)} (q_7, (x_1, y_2)) \xrightarrow{\mathsf{int}((q_7, \mathbf{br}(m_7, 1), q_7))} (q_7, (x_1, y_2))$$

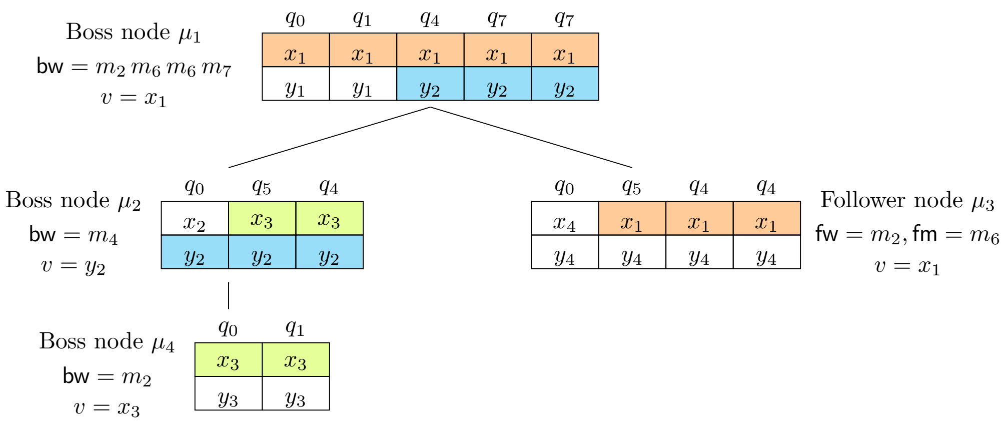


Figure 2 Example of an unfolding tree. The step labels in local runs are omitted for simplicity.

We explain why conditions (i) and (ii) are satisfied at the root μ_1 of the tree. Let $u := \mathbf{lr}(\mu_1)$ its local run. The only non-initial value in u is y_2, and $\mathsf{In}_{y_2}(u) = m_4$; condition (i) is satisfied as μ_1 has a boss child with a boss specification containing m_4. For initial value y_1, condition (ii) is satisfied as u never receives a message with value y_1. For x_1, consider the decomposition $\mathtt{dec} := (w, m_6, w')$ for $w := m_2$ and $w' := m_7$ seen as words of $\mathcal{M}^*$. Condition (ii) is satisfied thanks to μ_3 being a child of μ_1 with follower specification $(\mathbf{fm}, \mathbf{fw})$ such that $\mathbf{fm} = m_6$ and $\mathbf{fw} \in \mathcal{L}^{\mathtt{dec}_1}$ where $\mathtt{dec}_1 = (m_2)$. One can check that conditions (i) and (ii) are satisfied for the other nodes.

Condition (iii) only applies to μ_3. It is satisfied as $\mathbf{lr}(\mu_3)$ broadcasts (m_6, x_1) after receiving only (m_2, x_1) with that value. For condition (iv), it suffices to observe that μ_2 and μ_4 broadcast their bw themselves; we can consider decompositions (m_4) and (m_2) respectively. Moreover, μ_1 satisfies boss specification $\mathsf{bw} := m_1\, m_6\, m_6\, m_7$ as $\mathsf{bw} \in \mathcal{L}^{\mathtt{dec}}$ with $\mathtt{dec} := (m_2, m_6, m_7)$ as above. Therefore μ_1 is a witness that bw can be broadcast with a single value, although its local run does not broadcast m_6 itself.

The run from Example 4 involved two agents a_1 and a_2; a_1 corresponds to nodes μ_1 and μ_4 and a_2 to nodes μ_2 and μ_3. Note that, if we apply our procedure described below to build a run from τ, we would use 4 distinct agents, each playing a single role.

We now prove that Cover can be stated as the existence of an unfolding tree that is a coverability witness as defined above:

▶ **Proposition 17.** *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 translation from run to tree works by induction on the length of the run. We first define in a natural way what it means for a run to satisfy a specification. We consider a run ρ and isolate a well-chosen agent a, whose local run witnesses that the specification is satisfied. We call the induction hypothesis with the specifications expressing what a needs to receive from other agents. Each such specification is satisfied by a strict prefix of ρ (the only exception being if a satisfies a boss specification with value v and the last step of ρ is a broadcast with v by another agent; in this case, we use the induction hypothesis on ρ but with a follower specification, hence the induction is well-founded). We construct an unfolding tree by labeling the root with the specification and the local run of a, and attaching below it the subtrees obtained by induction hypothesis.

The translation from tree to run consists in an induction on the tree. A key concept is the one of partial run, which extends the notion of local run to a subset of agents: in a partial run, some receptions called *external* are not matched by a broadcast. This is meant to represent the behavior of a subtree of the unfolding tree: if the root of an unfolding tree is a follower node with specification $(\mathsf{fw}, \mathsf{fm})$ then the corresponding partial run receives an external sequence fw. The inductive step applies the induction hypothesis to the children of the root to obtain partial runs and merges them with the local run of the root by branching broadcasts to the right external receptions. See Appendix C for the full proof. ◀

3.3 Bounding the Size of the Unfolding Tree

Our aim is now to provide bounds on the size of the coverability witness. We start with two simple observations. First, for boss specifications, the longer the word broadcast, the better: if a word bw can be broadcast with a single value, then any subword of bw can also be received. For follower specifications, it goes in the opposite direction: for a fixed fm, the shorter the requirement fw, the better. The following lemma thus provides two ways of shortening an unfolding tree. Its proof can be found in Appendix D.

▶ **Lemma 18.** *Let τ be a coverability witness for $(\mathcal{P}, q_f)$. Let μ, μ' be two nodes of τ such that μ is an ancestor of μ'. If one of the conditions below holds, then there exists a coverability witness for $(\mathcal{P}, q_f)$ of size smaller than $|\tau|$:*

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

We now show that there is a computable bound on the size of the unfolding tree achieving a given specification and labeled with a protocol $\mathcal{P}$. Lemma 18 leads us towards an application of the Length function theorem; however, this theorem requires a bound on the lengths of the words. In fact, there is no reason to think that the lengths of the labels of the children of a node can be bounded with respect to the length of the label of that node.

In order to bound the size of the nodes, we use the following result, which essentially states that if there is a local run between two local configurations (q, ν) and (q', ν') then there is one of length bounded by a primitive recursive function and which does not require larger inputs than the previous one.

▶ **Lemma 19.** *There exists a primitive recursive function $\psi(n, r)$ such that, for every protocol $\mathcal{P}$ with r registers, for every local run $u_0 : (q_0, \nu_0) \xrightarrow{*} (q_f, \nu_f)$ in $\mathcal{P}$, for every section $u : (q, \nu) \xrightarrow{*} (q, \nu')$ of u_0, for every $V \subseteq \mathbb{N}$ finite such that V contains all message values appearing in u, there exists a local run $u' : (q, \nu) \xrightarrow{*} (q', \nu')$ such that we have $\mathsf{len}(u'') \leq \psi(|\mathcal{P}|, r)$ and:*

1. *for all $v' \in \mathbb{N} \setminus V$, there exists v a non-initial value of u_0 such that $\mathsf{In}_{v'}(u') \preceq \mathsf{In}_v(u)$,*
2. *for all $v \in V$, $\mathsf{In}_v(u') \preceq \mathsf{In}_v(u)$.*

Proof sketch. First, we prove that any long portion of u must change the value of every register at least once; otherwise we can shorten the run using an induction on the number of registers. We then manage to prove that, if u includes twice the same sequence of transitions of sufficient length, then we can cut off anything in the middle and glue back together the ends. While shortening the local run we may add some fresh values to it (see Figure 5 in the appendix), which is not a problem as we ensure that they are less constraining than the ones that were in the original run. For technical reasons, we want to prevent fresh values added in the proof from mimicking initial values of the agent. See Appendix E for the full proof. ◀

▶ Remark 20. The function $\psi(n,k)$ defined above is actually a tower of exponentials of height k where each floor is a polynomial in n. Perhaps surprinsingly, this bound is tight: one may need a local run of length a tower of exponentials to reach a given local configuration while being allowed to receive sequences of messages of same value from a given fixed set.

If we had in our unfolding tree only boss nodes or only follower nodes, then the previous result would allow us to apply the Length function theorem. Indeed, we can bound the size of a node with respect to the nodes to which it must send long words of messages. This means we can find a bound for a node μ that depends on μ's follower children's size and, if μ is a boss node, with respect to its parent's size. However, we cannot bound the size of an unfolding tree from the root to the leaves because of follower nodes nor from the leaves to the root because of boss nodes. We thus rearrange the tree as in Figure 3 to make it so that long sequences of messages are sent upwards. We formalize this with the notion of altitude:

▶ **Definition 21.** *Let τ an unfolding tree. We define the altitude of a node μ of τ, written* **alt**(μ)*, recursively as follows:*

- *The altitude of the root is* 0*,*
- *The altitude of a boss node is the altitude of its parent minus one,*
- *The altitude of a follower node is the altitude of its parent plus one.*

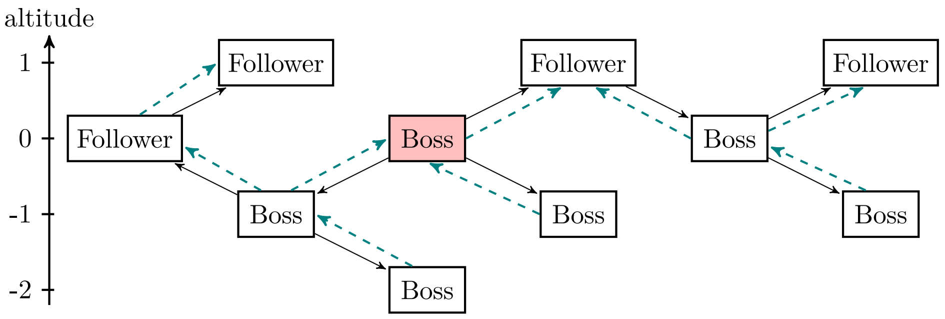


Figure 3 Rearrangement of a tree, with the root in red. Black solid arrows connect parents to children, blue dashed arrows highlight that long words of messages are sent upwards.

We now use the previous lemma to bound the label of each node μ with respect to its neighbors of higher altitude, i.e., its follower children and its parent if it is a boss node. The idea is that these nodes define the number of messages that $\mathbf{lr}(\mu)$ must output to satisfy the unfolding tree conditions. The function ψ in the statement below is the one from Lemma 19.

▶ **Lemma 22.** *Let $\mathcal{P}$ be a protocol over r registers, let τ be an unfolding tree over $\mathcal{P}$ of minimal size satisfying a boss specification* bw*, let μ be a node of τ. Let K be such that for all follower children μ_f of μ, $|\mathbf{fw}(\mu_f)| \leq K$. We have the following properties:*

1. *If μ is a boss node then*
   - *If μ is the root of τ then $\mathbf{bw}(\mu) = w$, otherwise $|\mathbf{bw}(\mu)| \leq |\mathbf{lr}(\mu')|$ with μ' its parent*
   - *In both cases $|\mathbf{lr}(\mu)| \leq \psi(|\mathcal{P}|, r)\Big[|\mathbf{bw}(\mu)| + |\mathcal{M}|rK + 1\Big]$*
2. *If μ is a follower node then $|\mathbf{fw}(\mu)| \leq |\mathbf{lr}(\mu)| \leq \psi(|\mathcal{P}|, r)\Big[1 + |\mathcal{M}|rK\Big]$*

Proof sketch. A node μ has at most $|\mathcal{M}|$ follower children for each initial value, hence at most $|\mathcal{M}|r$ in total, each one of them requires at most K messages. The node μ may have to output $\mathbf{bw}(\mu)$ extra messages to satisfy its specification if it is a boss node, or just one extra message if it is a follower node. This gives a bound on the number of messages $\mathbf{lr}(\mu)$ needs to broadcast. We mark the positions at which $\mathbf{lr}(\mu)$ sends them and use Lemma 19 to bound the length of sections of the run connecting two such broadcasts by $\psi(|\mathcal{P}|, r)$, which yields the bounds above. See Appendix F for the full proof. ◀

Thanks to the previous lemma, we bound the size of a node with respect to its altitude:

▶ **Lemma 23.** *Let $(\mathcal{P}, q_f)$ be a positive instance of* Cover*, τ a coverability witness for $(\mathcal{P}, q_f)$,* **altmax** *the maximal altitude in τ. There exists a primitive recursive function f_0 such that any node μ of τ has size bounded by $f_0(|\mathcal{P}| + \mathbf{altmax} - \mathbf{alt}(\mu))$.*

Proof sketch. Applying Lemma 22 inductively from highest to lowest altitude, we bound the sizes of the labels of all nodes at a given altitude i with respect to $\mathbf{altmax} - i$. See Appendix G for the detailed proof. ◀

▶ **Proposition 24.** *There exists a function f of class $\mathscr{F}_{\omega^{|\mathcal{M}|+1}}$ such that an instance $(\mathcal{P}, q_f)$ of* Cover *is positive if and only if it has a coverability witness τ of size bounded by $f(|\mathcal{P}|)$.*

Proof sketch. The full proof is in Appendix H. $(\mathcal{P}, q_f)$ is positive if and only if there exists a coverability witness for it thanks to Proposition 17; we consider the coverability witness τ of minimal size. In a branch of τ reaching maximal altitude, we mark the nodes that have a greater altitude than all the previous ones (see Figure 6). They are necessarily follower nodes as a boss node is below its parent. This sequence (taken from highest to lowest altitude) is so that the ith term is at altitude $\mathbf{altmax} - i$ and we can bound its size with respect to i with the previous arguments. Along with Lemma 18, we apply the Length function theorem on that sequence to bound its length hence the maximal altitude (Lemma 36).

This yields in turn a bound on the root label, as its altitude (0) has a bounded difference with the maximal one. Another application of the Length function theorem, this time with boss nodes, allows us to bound the minimal altitude of a node of this tree (Lemma 37).

Once we have bounded both the maximal and minimal altitudes, we can infer a bound on the size of all nodes using Lemma 23, and then on branches as we can shorten branches as soon as they have two nodes with the same specification. The bound on the size of the tree then follows from the observation that as nodes have bounded local runs, they only see a bounded amount of values and thus need a bounded amount of children. ◀

3.4 Decidability

In Section 3.2, we showed that unfolding trees are a sound and complete abstraction for Cover. In Section 3.3, we proved that there is 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. Details can be found in Appendix I.

▶ **Theorem 13.** *Cover for BNRA is decidable and $\mathbf{F}_{\omega^\omega}$-complete.*

3.5 Undecidability of Target

A natural next problem, after Cover, is the target reachability 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 later reach the target state, which makes the problem harder. We in fact show that Target is undecidable, which suggests that while we can obtain decidability of Cover thanks to some monotonicity properties of the problem, we cannot analyze more precisely the set of runs of such systems.

▶ **Proposition 25.** *Target is undecidable for* BNRA *with two registers.*

Proof sketch. We simulate a Minsky machine with two counters. Like for the LCS encoding, the first register is never modified and plays the role of a permanent identifier. We start with some initialisation phase where each agent stores some other agent's identifier in its second register, which will be 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 Minsky machine: we make agents alternate between receptions and broadcasts so that, in the end, they have received and sent the same number of messages, implying that no messages have 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, initiated by the leader, takes the form of a message propagated in the cycle until it finds an agent simulating counter 1 and having bit 0. This agent switches to 1 and sends an acknowledgment that eventually propagates back to the leader. See Appendix J for the full proof. ◀

4 Cover in 1-BNRA

In the section, we study the restriction of Cover to protocols with one register. We will establish that this problem is actually NP-complete, meaning that having only one register per agent makes the problem significantly easier. We shall call BNRA with one register 1-BNRA. Due to space constraints, formal proofs are not included in this section; they can be found in Appendix K. Here we intend to present the key observations that allow us to abstract runs into short witnesses, leading to an NP algorithm for the problem.

In 1-BNRA, local tests are irrelevant. Moreover, thanks to the copycat principle, any received 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 starts with value v. If the run puts some agent $a' \neq a$ in some state q with value v, then we can duplicate agent a' to have an unlimited supply of agents in state q with value v. Indeed, at some point in the run, a' was in a state q', executed a '$\downarrow$' transition, received a message with value v, stored it in its register and went to a state q''. Because we have an unlimited supply of agents in q' (thanks to the copycat principle), we also have an unlimited supply of agents in q'' with value v. We can then make all those agents copy the transitions of a', which gives us an unlimited supply of agents in state q with value v. Agent a is unique and called *boss*, agents like a' are clonable and called *followers*. For value v, the only relevant information is *boss* state, *i.e.*, the state of the agent a that had v initially (this agent cannot be cloned), and the *clique*, *i.e.*, the set of states reached by other agents with that value v.

We thus define gangs, an abstraction of configurationsof the 1-BNRA with respect to a value v. A gang is composed of a boss state and a clique. The clique may only increase throughout the abstract runs we will consider, because we assume that we always have enough copies of each agent so that we can leave many agents with value v in every state they visited. More formally, let $(Q, \mathcal{M}, \Delta, q_0)$ be a protocol. A *gang* is a pair $\mathsf{G} = (\mathsf{b}, \mathsf{K}) \in (Q \cup \{\perp\}) \times 2^Q$.

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 is the gang (b, K) such that:

- if there exists a_0 an agent which has value v in γ_0 and keeps it in all γ_i then b is its state in γ_k, otherwise $\mathsf{b} := \bot$,
- K is the set of states q such that there is an agent in q with value v in some γ_i.

In a concrete run of our system, gangs of distinct values may only interact with one another by covering states q which are needed by the other gang (in the form of a broadcast or of a '$\downarrow$' reception from q); therefore our abstraction also keeps track of the set of coverable states, which may only grow. However, it only needs to keep track of one gang at a time.

This leads us to a natural abstract semantics based on gangs. An abstract configuration consists of a set of states S (states covered so far by some agents) and a gang (b, K) (the original owner of the value v we are keeping track of and the states reached by other agents with that value). If the original owner of v stores a new value we set $b = \bot$. Abstract transitions are defined by applying transitions of the protocol while assuming that we have unlimited supplies of agents in every state of S and of agents with value v in states of K. At any time, we can apply a *gang reset*, which maintains S but reinitializes (b, K) to $(q_0, \emptyset)$ (we track a new value). We define formally this abstraction and show its soundness and completeness in Appendix K. To bound the length of relevant abstract runs, we impose that S should grow between two gang resets (otherwise they reset to the same abstract configuration) and that there may be at most $O(|Q|^2)$ abstract steps between two resets (as K can only increase and there are only $|Q| + 1$ possibilities for b). This means that if there is an abstract run covering a state, there is one of size $O(|Q|^3)$, proving the 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 which broadcast it back, playing the role of external memory, allowing a to check the satisfaction of a 3SAT formula).

These results yield the main theorem of this section:

▶ **Theorem 26.** *The coverability problem is* NP*-complete for protocols with one register.*

5 Conclusion

We have 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. One may want to enrich the transition systems of our protocols, for instance to pushdown automata. While this adds little difficulty in the general case (it suffices to extend Lemma 19 to pushdown protocols using a classical cut-hill argument), the case of one register may be trickier. Another open problem is the complexity of the target problem with one register. While we can show that this is a decidable problem, its exact complexity is unclear. Finally, one may want to extend this model with inequality tests, as in classical related models such as data nets.

Acknowledgements. We are grateful to Arnaud Sangnier for encouraging us to work on BNRA, for the discussions about his work in [13] and in general 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 Parosh Aziz Abdulla, Mohamed Faouzi Atig, Ahmet Kara, and Othmane Rezine. Verification of dynamic register automata. In *34th International Conference on Foundation of Software Technology and Theoretical Computer Science, FSTTCS 2014*, volume 29 of *LIPIcs*, pages 653–665. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2014. doi:10.4230/LIPIcs.FSTTCS.2014.653.

2 Parosh Aziz Abdulla, Mohamed Faouzi Atig, Ahmet Kara, and Othmane Rezine. Verification of buffered dynamic register automata. In *Networked Systems, NETYS 2015*, volume 9466 of *Lecture Notes in Computer Science*, pages 15–31. Springer, 2015. doi:10.1007/978-3-319-26850-7_2.

3 Parosh Aziz Abdulla and Bengt Jonsson. Undecidable verification problems for programs with unreliable channels. *Information and Computation*, 130(1):71–90, 1996. doi:10.1006/inco.1996.0083.

4 Parosh Aziz Abdulla and Bengt Jonsson. Verifying programs with unreliable channels. *Information and Computation*, 127(2):91–101, 1996. doi:10.1006/inco.1996.0053.

5 C. Aiswarya. Model checking dynamic distributed systems. In *Networked Systems, NETYS 2015*, volume 9466 of *Lecture Notes in Computer Science*, pages 48–61. Springer, 2015. doi:10.1007/978-3-319-26850-7_4.

6 C. Aiswarya. On network topologies and the decidability of reachability problem. In *Networked Systems,NETYS 2020*, volume 12129 of *Lecture Notes in Computer Science*, pages 3–10. Springer, 2020. doi:10.1007/978-3-030-67087-0_1.

7 A. R. Balasubramanian, Nathalie Bertrand, and Nicolas Markey. Parameterized verification of synchronization in constrained reconfigurable broadcast networks. In *Tools and Algorithms for the Construction and Analysis of Systems, TACAS 2018*, volume 10806 of *Lecture Notes in Computer Science*, pages 38–54. Springer, 2018. doi:10.1007/978-3-319-89963-3_3.

8 A. R. Balasubramanian, Lucie Guillou, and Chana Weil-Kennedy. Parameterized analysis of reconfigurable broadcast networks. In *Foundations of Software Science and Computation Structures, FoSSaCS 2022*, volume 13242 of *Lecture Notes in Computer Science*, pages 61–80. Springer, 2022. doi:10.1007/978-3-030-99253-8_4.

9 Benedikt Bollig, Fedor Ryabinin, and Arnaud Sangnier. Reachability in distributed memory automata. In *Annual Conference on Computer Science Logic, CSL 2021*, volume 183 of *LIPIcs*, pages 13:1–13:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.CSL.2021.13.

10 Daniel Brand and Pitro Zafiropulo. On communicating finite-state machines. *Journal of the ACM*, 30(2):323–342, 1983. doi:10.1145/322374.322380.

11 Pierre Chambart and Philippe Schnoebelen. The ordinal recursive complexity of lossy channel systems. In *Annual IEEE Symposium on Logic in Computer Science, LICS 2008*, pages 205–216. IEEE Computer Society, 2008. doi:10.1109/LICS.2008.47.

12 Peter Chini, Roland Meyer, and Prakash Saivasan. Liveness in broadcast networks. *Computing*, 104(10):2203–2223, 2022. doi:10.1007/s00607-021-00986-y.

13 Giorgio Delzanno, Arnaud Sangnier, and Riccardo Traverso. Parameterized verification of broadcast networks of register automata. In *Reachability Problems , RP 2013*, volume 8169 of *Lecture Notes in Computer Science*, pages 109–121. Springer, 2013. doi:10.1007/978-3-642-41036-9_11.

14 Giorgio Delzanno, Arnaud Sangnier, Riccardo Traverso, and Gianluigi Zavattaro. 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*, volume 18 of *LIPIcs*, pages 289–300. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2012. doi:10.4230/LIPIcs.FSTTCS.2012.289.

15 Giorgio Delzanno, Arnaud Sangnier, and Gianluigi Zavattaro. Parameterized verification of ad hoc networks. In *CONCUR 2010*, volume 6269 of *Lecture Notes in Computer Science*, pages 313–327. Springer, 2010. doi:10.1007/978-3-642-15375-4_22.

16 E. Allen Emerson and Kedar S. Namjoshi. On model checking for non-deterministic infinite-state systems. In *Annual IEEE Symposium on Logic in Computer Science, LICS 1998*, pages 70–80. IEEE Computer Society, 1998. doi:10.1109/LICS.1998.705644.

17 Javier Esparza, Alain Finkel, and Richard Mayr. On the verification of broadcast protocols. In *Symposium on Logic in Computer Science, LICS 2014*, pages 352–359. IEEE, 1999.

18 Graham Higman. Ordering by divisibility in abstract algebras. *Proceedings of the London Mathematical Society*, s3-2(1):326–336, 1952. doi:10.1112/plms/s3-2.1.326.

19 Ranko Lazic, Thomas Christopher Newcomb, Joël Ouaknine, A. W. Roscoe, and James Worrell. Nets with tokens which carry data. *Fundamenta Informaticae*, 88(3):251–274, 2008.

20 Othmane Rezine. *Verification of networks of communicating processes: Reachability problems and decidability issues*. PhD thesis, Uppsala University, Sweden, 2017. URL: https://urn.kb.se/resolve?urn=urn:nbn:se:uu:diva-334788.

21 Fernando Rosa-Velardo. Ordinal recursive complexity of unordered data nets. *Information and Computation*, 254:41–58, 2017. doi:10.1016/j.ic.2017.02.002.

22 Arnaud Sangnier. Erratum to parameterized verification of broadcast networks of register automata, 2023. https://www.irif.fr/~sangnier/publications.html.

23 Sylvain Schmitz. Complexity hierarchies beyond elementary. *ACM Transactions on Computation Theory*, 8(1):3:1–3:36, 2016. doi:10.1145/2858784.

24 Sylvain Schmitz and Philippe Schnoebelen. Multiply-recursive upper bounds with higman's lemma. In *International Colloquium on Automata, Languages and Programming, ICALP 2011*, volume 6756 of *Lecture Notes in Computer Science*, pages 441–452. Springer, 2011. doi:10.1007/978-3-642-22012-8_35.

25 Philippe Schnoebelen. Verifying lossy channel systems has nonprimitive recursive complexity. *Information Processing Letters*, 83(5):251–261, 2002. doi:10.1016/S0020-0190(01)00337-4.

26 Luc Segoufin. Automata and logics for words and trees over an infinite alphabet. In *Computer Science Logic, CSL 2006*, volume 4207 of *Lecture Notes in Computer Science*, pages 41–57. Springer, 2006. doi:10.1007/11874683_3.

A Proof of Proposition 9

▶ **Proposition 9.** *There is an exponential-time reduction from* Cover *to* Cover *with no local equality tests* $\textbf{loc}(i, j, =)$.

Proof. We construct a protocol $\mathcal{P}'$ with no local equality tests so that any initial run of $\mathcal{P}$ may be turned into an initial run of $\mathcal{P}'$ and vice-versa. The intuition is as follows. $\mathcal{P}'$ corresponds to $\mathcal{P}$, where an abstract layer is added in the treatments of the registers. In $\mathcal{P}'$, we do not directly store each register but rather equality relations between them, so that a value shared by several registers is only stored once. We dub registers of $\mathcal{P}'$ *memory slots* to distinguish them from the registers of $\mathcal{P}$. A memory slot will be used to store a value shared between registers. We will need r memory slots to do so. Moreover, we will need a mapping $\mathsf{map} : [1, r] \to [1, r]$ that tells, for each register, in which memory slot its value is stored. This mapping will be directly encoded in the set of states of $\mathcal{P}'$, which causes an exponential blow-up in the size of the protocol.

For simplicity, we allow our protocol $\mathcal{P}'$ to have dummy transitions with no effect (such transitions can be encoded using broadcasts with a dummy message type).

Let $\mathcal{P} = (Q, \mathcal{M}, \Delta, q_0, r)$. We define $\mathcal{P}' := (Q', \mathcal{M}, \Delta', q_0', r)$ as follows. The set of states is $Q' := Q \times ([1, r] \to [1, r])$, its initial state is $q_0' := (q_0, \mathsf{id})$ where $\mathsf{id}(i) = i$ for all $i \in [1, r]$.

The set of transitions Δ' is defined as follows. $\mathcal{P}'$ mimics the operations of $\mathcal{P}$ step-by-step. When $\mathcal{P}$ stores a new value, $\mathcal{P}'$ guesses whether this value is already stored in one of the memory slots (then it simply checks that the received value is the one of that memory slot with a $\textbf{rec}(m, i, =)$) or is not stored yet (then it stores it in an empty memory slot). In both cases it updates the mapping accordingly. Formally, for all $q, q' \in Q$ and $\mathsf{map}, \mathsf{map}' \in [1, r] \to [1, r]$, we have $((q, \mathsf{map}), \mathsf{op}, (q', \mathsf{map}')) \in \Delta'$ in the following cases:

- $\mathsf{op} = \textbf{br}(m, j)$, $\mathsf{map} = \mathsf{map}'$ and $(q, \textbf{br}(m, i), q') \in \Delta$ for some i such that $\mathsf{map}(i) = j$ (we broadcast the value stored in the corresponding memory slot).
- $\mathsf{op} = \textbf{rec}(m, j, \downarrow)$, there is a transition $(q, \textbf{rec}(m, i_0, \downarrow), q') \in \Delta$ with $\mathsf{map}^{-1}(j) \subseteq \{i_0\}$ and for all $i \in [1, r]$, if $i \neq i_0$ then $\mathsf{map}'(i) = \mathsf{map}(i)$ and $\mathsf{map}'(i_0) = j$ (the received value is stored into a fresh memory slot).
- $\mathsf{op} = \textbf{rec}(m, j, =)$ and either
  - there is a transition $(q, \textbf{rec}(m, i, =), q') \in \Delta$ such that $\mathsf{map}(i) = j$ and $\mathsf{map}' = \mathsf{map}$ (to test equality with register i, we test equality with the memory slot containing value of register i), or
  - there is a transition $(q, \textbf{rec}(m, i_0, \downarrow), q') \in \Delta$ such that $\mathsf{map}'(i_0) = j$ and $\mathsf{map}'(i) = \mathsf{map}(i)$ for every $i \neq i_0$ (register i_0 stores a value that is guessed as being already in a memory slot hence we simply modify the mapping).
- $\mathsf{op} = \textbf{rec}(m, j, \neq)$ and there is a transition $(q, \textbf{rec}(m, i, \neq), q') \in \Delta$ such that $\mathsf{map}(i) = j$ (to test disequality with register i, we test disequality with the memory slot containing value of register i).
- $\mathsf{op} = \textbf{loc}(j_1, j_2, \neq)$ and there is a transition $(q, \textbf{loc}(i_1, i_2, \neq), q')$ such that $\mathsf{map}(i_1) = j_1$ and $\mathsf{map}(i_2) = j_2$ (to implement a local disequality test, we do the corresponding test on memory slots; this test always fails if $j_1 = j_2$).
- $\mathsf{op} = \textbf{rec}(m, j, *)$ and there is a transition $(q, \textbf{rec}(m, i, *), q') \in \Delta$ for some $i \in [1, r]$.

Additionally, there is a dummy transition between (q, map) and (q', map') whenever there exist i_1, i_2 such that $(q, \textbf{loc}(i_1, i_2, =), q') \in \Delta$ and $\mathsf{map}(i_1) = \mathsf{map}(i_2)$ (an equality test on two registers mapped to the same memory slot automatically succeeds).

Note that we cannot in fact guarantee that values stored in the memory slots are all distinct, because, upon reception of a value, we may only perform one test and therefore cannot test that the value is different from all stored values. This is why we still need local disequality tests in $\mathcal{P}'$. In practice, when a agent in $\mathcal{P}$ performs a '$\downarrow$' reception of a broadcast with a value that is already in one of its memory slots, the corresponding agent in $\mathcal{P}'$ may non-deterministically notice that it already has this value in a memory slot (corresponds to a $\mathbf{rec}(m, j, =)$ transition) but it may also store it in a fresh memory slot (corresponds to a $\mathbf{rec}(m, j, \downarrow)$ transition). Therefore, the encoding of the content of the registers into a mapping and memory slots is not unique, even up to permutation of the memory slots. However, encodings that do not gather together registers with the same value will simply allow less transitions (it will forbid some equality tests), therefore this is not an issue.

We now prove that, for a given state $q_f \in Q$, q_f may be covered in $\mathcal{P}$ if and only if some state of $Q'_f := \{(q_f, \mathsf{map}) \mid \mathsf{map} \in [1, r] \to [1, r]\}$ may be covered in $\mathcal{P}'$. To do so, we will prove that any initial run of $\mathcal{P}$ may be turned into an initial run of $\mathcal{P}'$ and vice-versa.

Let $\mathbb{A} \subseteq \mathbb{N}$. Given a configuration γ of $\mathcal{P}$, we denote by $\mathsf{mem}(\gamma)$ the set of configurations γ' over $\mathbb{A}$ such that: for all $a \in \mathbb{A}$, by denoting $(q, \nu) := \gamma(a)$, one has $\gamma'(a) = ((q, \mathsf{map}), \nu')$ where, for every $i \in [1, r]$, $\nu'(\mathsf{map}(i)) = \nu(i)$. Moreover, we denote by $\mathsf{perf}(\gamma)$ the subset of $\mathsf{mem}(\gamma)$ containing configurations γ' that additionally satisfy that, for every agent, for every $j \neq j' \in [1, r]$, $\nu'(j) \neq \nu'(j')$ (we call such configurations *perfect configurations* of $\mathcal{P}'$).

Let $\rho : \gamma_0 \xrightarrow{*} \gamma_f$ an initial run on $\mathcal{P}$ over some set of agents $\mathbb{A}$. We prove by induction on the length of ρ that there exists an initial run $\rho' : \gamma'_0 \xrightarrow{*} \gamma'_f$ on $\mathcal{P}'$ over $\mathbb{A}$ such that $\gamma'_f \in \mathsf{perf}(\gamma_f)$. This property is trivially true for ρ of length 0. It therefore suffices to prove that given $\gamma_1 \to \gamma_2$ and $\gamma'_1 \in \mathsf{perf}(\gamma_1)$, there exists $\gamma'_2 \in \mathsf{perf}(\gamma_2)$ such that $\gamma'_1 \to \gamma'_2$. We denote by s the step $\gamma_1 \to \gamma_2$ and by s' the step $\gamma'_1 \to \gamma'_2$ that we are building. Let $a \in \mathbb{A}$ that performs a transition in s.

- If a performs a local test $\mathbf{loc}(i_1, i_2, \neq)$ in s, then it may perform the corresponding local test in s' because $\gamma'_1 \in \mathsf{mem}(\gamma_1)$ hence $\gamma'_1(a)$ maps i_1 and i_2 to memory slots of distinct values.
- If a performs a local test $\mathbf{loc}(i_1, i_2, =)$, then because $\gamma'_1 \in \mathsf{perf}(\gamma_1)$, it maps i_1 and i_2 to the same memory slot and a may perform the corresponding internal transition in $\mathcal{P}'$.
- If a performs a broadcast in s, then we make it perform the corresponding broadcast in s'. Note that, because $\gamma'_1 \in \mathsf{mem}(\gamma_1)$, the broadcast value is the same in s and s'.
- Suppose now that a performs a reception $\mathbf{rec}(m, i, \alpha)$ in s. Let v the value of the message. By the previous case, the corresponding broadcast is performed in s' with the same value v.
  - If $\alpha =$ '$*$', then we make a perform the corresponding transition in s'.
  - If $\alpha =$ '$\downarrow$', then if v appears in some memory slot j in γ'_1, we make a perform the corresponding $\mathbf{rec}(m, j, =)$ transition in s'. If v does not appear in any memory slot, then we make a take the $\mathbf{rec}(m, j, \downarrow)$ transition with j a memory slot that did not store the value of any register other that i (it necessarily exists by the pigeonhole principle). This case disjunction guarantees that $\gamma'_2(a)$ does not have twice the same value in its memory slots, hence that γ'_2 is perfect.
  - If $\alpha =$ '$\neq$', then we make a perform the corresponding transition in s' with the operation $\mathbf{rec}(m, \mathsf{map}(j), \neq)$; because $\gamma'_1 \in \mathsf{mem}(\gamma_1)$, we have $\gamma'_1(a)(\mathsf{map}(j)) = \gamma_1(a)(j) \neq v$ and the test passes.
  - If $\alpha =$ '$\neq$', then we make a perform the corresponding transition in s' with the operation $\mathbf{rec}(m, \mathsf{map}(j), =)$; because $\gamma'_1 \in \mathsf{mem}(\gamma_1)$, we have $\gamma'_1(a)(\mathsf{map}(j)) = \gamma_1(a)(j) = v$ and the test passes.

Overall, we have built $\gamma_2' \in \mathsf{perf}(\gamma_2)$ such that $\gamma_1' \to \gamma_2'$ in $\mathcal{P}'$, which concludes the induction. We have proven that for every $\rho : \gamma_0 \xrightarrow{*} \gamma_f$ there exists $\rho' : \gamma_0' \xrightarrow{*} \gamma_f'$ on $\mathcal{P}'$ such that $\gamma_f' \in \mathsf{perf}(\gamma_f)$. This proves that if q_f may be covered in $\mathcal{P}$, then Q_f' may be covered in $\mathcal{P}'$.

Conversely, let $\rho' : \gamma_0' \xrightarrow{*} \gamma_f'$ an initial run of $\mathcal{P}$ over some set of agents $\mathbb{A}$. We prove that there exists an initial run $\rho : \gamma_0 \xrightarrow{*} \gamma_f$ of $\mathcal{P}$ such that $\gamma_f' \in \mathsf{mem}(\gamma_f)$. The proof is by induction on the length of ρ'.

For ρ' of length 0, the property is trivially satisfied. For the heredity, it suffices to prove that if we are given $\gamma_1' \to \gamma_2'$ and γ_1 such that $\gamma_1' \in \mathsf{mem}(\gamma_1)$, then there exists γ_2 such that $\gamma_2' \in \mathsf{mem}(\gamma_2)$ and $\gamma_1 \to \gamma_2$. The proof is similar to the previous one but simpler. Let a an agent. We pick $\delta \in \Delta$ one of the transitions in $\mathcal{P}$ that, when building $\mathcal{P}'$, caused the transition taken by a in $\gamma_1' \to \gamma_2'$ to be added to Δ'. A simple case disjunction allows to prove that a may perform δ from γ_1 (under the assumption that the broadcast value is the same, which can be easily guaranteed by looking at the broadcasting agent). Overall we obtain a step $\gamma_1 \to \gamma_2$ with $\gamma_2' \in \mathsf{mem}(\gamma_2)$ by construction. This concludes the induction. We have proven that if Q_f' can be covered in $\mathcal{P}'$ then q_f can be covered in $\mathcal{P}$, hence the two properties are equivalent. Finally, observe that the COVER problem with a subset $Q_f' \subseteq Q'$ as objective can be easily reduced to COVER with a single state as objective (using dummy transitions with no effect) concluding the reduction. ◀

B Proof of Proposition 11

▶ **Proposition 11.** *The coverability problem for BNRA is $\mathbf{F}_{\omega^\omega}$-hard.*

Proof. It is sufficient to prove that it is as hard as reachability for lossy channel systems with a single channel, which corresponds to a single finite-state machine that has the ability to buffer symbols in a lossy FIFO queue [25]. We present here a polynomial-time reduction. Let $\mathcal{L} := (L, \Sigma)$ be a lossy channel system, where L is a finite set of locations, Σ is a finite set of symbols and $D \subseteq L \times \Sigma^* \times \{!, ?\} \times L$; '!' corresponds to a push (writing at the end of the channel) and '?' to a pop (reading at the beginning the channel). A configuration of $\mathcal{L}$ is a pair of $L \times \Sigma^*$ denoting the location of the system and the content of the channel. There exists a step from (l, w) to (l', w') using transition $d \in D$, denoted $(l, w) \xrightarrow{d}_{\mathcal{L}} (l', w')$, when

- $d = (l, u, !, l')$ for some $u \in \Sigma^*$ and $w' \preceq w \cdot u$ (a *push*),
- $d = (l, u, ?, l')$ for some $u \in \Sigma^*$ and $u \cdot w' \preceq w$ (a *pop*)

where $\preceq$ denotes the subword order. This subword order encodes the lossiness of the channel (a non-lossy channel would have equalities instead); intuitively, it expresses the fact that letters in the channel may get lost.

The existence of such a transition for some $d \in \Delta$ is denoted $(l, w) \to_{\mathcal{L}} (l', w')$, and its transitive closure is denoted $\xrightarrow{*}_{\mathcal{L}}$. The (control-state) *reachability problem* asks, given $\mathcal{L}$ and two locations $l_i, l_f \in L$, whether there exists an execution $(l_i, \varepsilon) \xrightarrow{*}_{\mathcal{L}} (l_f, w)$ for some w.

We aim at constructing a protocol $\mathcal{P}$ with two registers with a distinguished state q_f such that q_f may be covered if and only if $(\mathcal{L}, q_i, q_f)$ is a positive instance of the reachability problem. The intuition of $\mathcal{P}$ is the following. Agents will organize in chains so that each agent of a chain knows the identifier of its predecessor in the chain, hence is able to check that the messages it received come from it. Each agent of the chain is meant to encode a step of the execution in the lossy channel system. An agent of the chain will only listen to their predecessor in the chain, from which they will obtain a location of the system and (a subword of) the content of the channel. As it receives those, it will broadcast the new location of the system and the new content of the channel to the next agent of the chain.

Note that the content of the channel might become big in a lossy channel system, therefore agents will not store it but rather rebroadcast it on-the-fly letter by letter. An agent only applies a small modification at the beginning of the channel if it decides to encode a pop transition and at the end of the channel if it decides to encode a push transition. Messages might get lost, which is why we are able to encode lossy channel systems but not non-lossy ones.

In some initial phase, agents decide whether they are *root* (at the beginning of their chain) or *link*. A root agent receives no message; it simply broadcasts its identifiers and the initial configuration of $\mathcal{L}$, (l_0, ε). To encode this option, in $\mathcal{P}$, from the initial state q_0 one has the possibility to move to a part where the only sequence of transitions possible is $\mathbf{br}(\mathsf{init}, 1), \mathbf{br}(\mathsf{q_0}, 1), \mathbf{br}(\#, 1)$ which gets to $\mathbf{end}(l_0) \in Q$. The symbol $\# \in \Sigma$ is the final symbol meaning that the channel was fully broadcast.

A link agent first receives a broadcast with an identifier which it stores as the one of its predecessor. It then broadcasts its own identifier. This construction guarantees that link agents have exactly one predecessor. It does not guarantee, however, than all agents are in the same chain or that any agent is the predecessor of at most one agent. Concretely, there is a sequence of two transitions from q_0 labeled by $\mathbf{rec}(\mathsf{init}, 2, \downarrow), \mathbf{br}(\mathsf{init}, 1)$ that gets to $\mathbf{wait} \in Q$.

From $\mathbf{wait}$, there is, for every $l \in L$, a transition labeled by $\mathbf{rec}(\mathsf{l}, 2, =)$ that goes to state $\mathbf{start}(l) \in Q$. For every transition $d = (l, \mathsf{op}, l') \in D$ in $\mathcal{L}$ (*i.e.*, every transition of D whose source is location l), there is a transition in $\mathcal{P}$ labeled by $\mathbf{br}(\mathsf{l'}, 1)$ that goes from $\mathbf{start}(l)$ to $\mathbf{tr}_1(d) \in Q$. Transitions from $\mathbf{tr}_1(d)$ in $\mathcal{P}$ depend on d:

- If $d = (l, u, ?, l')$ is a pop then $\mathcal{P}$ has a sequence of transitions from $\mathbf{tr}_1(d)$ to $\mathbf{tr}_2(d) \in Q$ labeled by $\mathbf{rec}(\mathsf{u_1}, 2, =), \mathbf{rec}(\mathsf{u_2}, 2, =), \ldots, \mathbf{rec}(\mathsf{u_k}, 2, =)$ where $u = u_1 u_2 \cdots u_k$. Moreover, there is, for every $x \in \Sigma$, a loop on $\mathbf{tr}_2(d)$ labeled with the sequence of actions $\mathbf{rec}(\mathsf{x}, 2, =), \mathbf{br}(\mathsf{x}, 1)$. There is also a sequence of transitions from $\mathbf{tr}_2(d)$ to $\mathbf{end}(l')$ labeled by $\mathbf{rec}(\#, 2, =), \mathbf{br}(\#, 1)$.
- If $d = (l, u, !, l')$ is a push then there is, for every $x \in \Sigma$, a loop on $\mathbf{tr}_1(d)$ labeled with sequence of actions $\mathbf{rec}(\mathsf{x}, 2, =), \mathbf{br}(\mathsf{x}, 1)$. There also is a sequence of transitions from $\mathbf{tr}_1(d)$ to $\mathbf{tr}_2(d)$ labeled by $\mathbf{br}(\mathsf{u_1}, 1), \mathbf{br}(\mathsf{u_2}, 1), \ldots, \mathbf{br}(\mathsf{u_k}, 1)$ where $u = u_1 \cdot u_2 \cdots u_k$. From $\mathbf{tr}_2(d)$, there is a sequence of transitions going to $\mathbf{end}(l')$ labeled by $\mathbf{rec}(\#, 2, =), \mathbf{br}(\#, 1)$.

Finally, the objective state of our system is $q_f := \mathbf{end}(l_f)$.

Note that, in this protocol, register 1 is broadcast-only, therefore acting as a signature for messages, and register 2 is reception-only, therefore used to check the signature of messages.

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 $(l_0, w_0) \rightarrow_{\mathcal{L}} (l_1, w_1) \rightarrow_{\mathcal{L}} (l_2, w_2) \cdots \rightarrow_{\mathcal{L}} (l_n, w_n)$ with $l_n = l_f$ and $w_n = w$. We build an initial run of $\mathcal{P}$ that covers q_f as follows. 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 $\mathbf{end}(d_i)$ 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 met as agent 0 becomes root. When the construction has been done up until agent i, we make agent $i + 1$ do the following. It first places itself after agent i in the chain and goes to $\mathbf{wait}$. It then received from agent i state l_i and goes to $\mathbf{start}(l_i)$. It then moves to

$\mathbf{tr}_1(d)$ where $d = (l_i, \mathsf{op}, l_{i+1})$ is the transition of step $(l_i, w_i) \to_{\mathcal{L}} (l_{i+1}, w_{i+1})$. Doing so, it broadcasts $\mathsf{l_{i+1}}$. Its behavior then depends on d.

- if $d = (l_i, u, !, l_{i+1})$ is a push then $w_{i+1} \preceq w_i \cdot u$; write $w_{i+1} = w'_i \cdot u'$ where $w'_i \preceq w_i$ and $u' \preceq u$. Agent $i+1$ receives w'_i in full and rebroadcasts it while looping on $\mathbf{tr}_1(d)$. It then broadcasts u to get to $\mathbf{tr}_2(d)$. It finally receives $\#$ from agent i and rebroadcasts it, going to state $\mathbf{end}(l_{i+1})$. Overall the word broadcast by agent $i+1$ is $\mathsf{init} \cdot \mathsf{l_i} \cdot w'_i \cdot u \cdot \#$ which admits as subword $\mathsf{init} \cdot \mathsf{l_i} \cdot w_{i+1} \cdot \#$.
- if $d = (l_i, u, ?, l_{i+1})$ is a pop then $u \cdot w_{i+1} \preceq w_i$; write $w_i = u' \cdot w'_{i+1}$ where $w_{i+1} \preceq w'_{i+1}$ and $u \preceq u'$. Overall, agent $i+1$ receives from agent i a sequence of message types $\mathsf{init} \cdot \mathsf{l_i} \cdot u \cdot w'_{i+1} \cdot \#$ (some messages may get lost). From $\mathbf{tr}_1(d)$, agent $i+1$ receives u and goes to $\mathbf{tr}_2(d)$. It then receives w'_{i+1} in full and rebroadcasts it while looping on $\mathbf{tr}_2(d)$. It finally received $\#$ from agent i and rebroadcasts it, going to state $\mathbf{end}(l_{i+1})$. Overall the word broadcast by agent $i+1$ is $\mathsf{init} \cdot \mathsf{l_i} \cdot w'_{i+1} \cdot \#$ which admits as subword $\mathsf{init} \cdot \mathsf{l_i} \cdot w_{i+1} \cdot \#$.

This concludes the induction step. When applied to $i = n$, this builds an initial run where agent n ends on $\mathbf{end}(l_n)$, 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. Write $\mathbb{A}$ the set of agents in ρ. Observe that, in $\mathcal{P}$, one may never change the value of register 1. Moreover, any agent either keeps its original value in register 2 or changes it exactly once to take the value of some other agent's register 1. Therefore, we define a function $\mathbf{pred} : \mathbb{A} \to \mathbb{A} \cup \{\perp\}$ that associates to every agent a the agent $\mathbf{pred}(a)$ whose initial value in register 1 eventually becomes the value in register 2 of a (and $\perp$ if such an agent does not exist). Consider a_f the agent that first covers q_f in γ_f. Let $a_1, \ldots, a_n$ the sequence of agents such that $a_f = a_n$, $\mathbf{pred}(a_{i+1}) = a_i$ for all $i \in [0, n-1]$ and $\mathbf{pred}(a_0) = \perp$. This last agent exists as otherwise there would be a cycle, as the set of agents is finite. This cannot happen as agents receive their predecessor's identifier before they send their own.

By structure of the protocol, in order to broadcast a sequence $\mathsf{init} \cdot \mathsf{l_i} \cdot w_i \cdot \#$, an agent needs to receive a sequence $\mathsf{init} \cdot \mathsf{l_i}' \cdot w'_i \cdot \#$ from its predecessor with $(l'_i, w'_i) \to_{\mathcal{L}} (l_i, w_i)$. Hence its predecessor needs to broadcast $\mathsf{init} \cdot \mathsf{l_{i-1}} \cdot w_{i-1} \cdot \#$ with $\mathsf{l_{i-1}} = \mathsf{l_i}'$ and $w_{i-1} \preceq w'_i$, thus $(l_{i-1}, w_{i-1}) \to_{\mathcal{L}} (l_i, w_i)$. Because a_n covers $\mathbf{end}(l_f)$, it must receive a sequence of the form $\mathsf{init} \cdot \mathsf{l_f} \cdot w_n \cdot \#$, hence by induction we obtain a sequence of configurations $(l_0, w_0) \to_{\mathcal{L}} (l_1, w_1) \to_{\mathcal{L}} (l_2, w_2) \cdots \to_{\mathcal{L}} (l_n, w_n)$ with $l_n = l_f$ and $w_0 = \varepsilon$ (as a_0 broadcasts $initl_0\#$). We have proven that $(l_0, \varepsilon) \xrightarrow{*}_{\mathcal{L}} (l_f, w_n)$ and the instance of $\mathcal{L}$ is positive. ◀

C Proof of Proposition 17

▶ **Proposition 17.** *An instance of* Cover $(\mathcal{P}, q_f)$ *is positive if and only if there exists a coverability witness for that instance.*

We start with a few definitions. First, we define the local run of an agent in a run, which corresponds to the intuitive idea that a local run is used to describe the point of view of a single agent in a run. Given a run $\rho : \gamma_0 \to \gamma_1 \to \cdots \to \gamma_n$ and a agent a, the local run u of a in ρ is defined by induction. The local run of a in an empty run γ_0 is the empty local run $\gamma_0(a)$. Let u_i the local run of a in $\gamma_0 \xrightarrow{*} \gamma_i$ (prefix of length i of ρ). We define u_{i+1} as follows:

- if in $\gamma_i \to \gamma_{i+1}$, a performs a local test or a broadcast with a transition δ then u_{i+1} is u_i to which a step labeled by $\xrightarrow{\mathsf{int}(\delta)}$ is appended,

- if in $\gamma_i \to \gamma_{i+1}$, a performs a reception with a transition δ then, by letting v the value of the message received, u_{i+1} is u_i to which a step labeled by $\xrightarrow{\mathsf{ext}(\delta,v)}$ is appended,
- if $\gamma_i \to \gamma_{i+1}$ does not involve a, then $u_{i+1} := u_i$.

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 one can modify the system so that the Cover problem becomes the problem of the existence of an initial run satisfying a boss specification; to do so, it suffices to add a broadcast loop on q_f that broadcasts a special message type. Therefore, it suffices to prove that initial runs of $\mathcal{P}$ and unfolding trees satisfy the same boss specifications. To do so, we prove the two following implications:

▶ **Lemma 27.** *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*.*

▶ **Lemma 28.** *If there exists an unfolding tree over $\mathcal{P}$ satisfying a boss specification* $\mathsf{bw} \in \mathcal{M}^*$ *then there exists a finite initial run ρ of $\mathcal{P}$ satisfying* bw*.*

C.1 Proof of Lemma 27

▶ **Lemma 27.** *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 the lexicographic order on $\mathbb{N} \times \{\text{follower}, \text{boss}\}$ with the length of the run as first component and the type of specification as second component, “boss” being considered higher than “follower” (so that, for a fixed run length, we prove it for boss specifications then for follower specifications).

If $\mathsf{spec} = \varepsilon$ is an empty boss specification then the tree with a single node labeled with an empty local run and an empty specification satisfies it. This covers in particular the case of a run of length 0, as they cannot satisfy any other specification. 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 $\mathsf{spec} = \mathsf{bw}$ is a boss specification, 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.

Observe that, if the last step of u does not include a broadcast with value v, then the run obtained by deleting this last step also satisfies the specification so we directly call the induction hypothesis on this new run. From now on, we will 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.

The construction is as follows. For every non-initial value $v' \neq v$ of a in ρ, we do the following. Let w the sequence of message types sent with value v' in ρ by agents in $\mathbb{A} \setminus \{a\}$; $\mathsf{In}_{v'}(u)$ is a subword of w. 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.

For every v' initial value of a in ρ, we do the following. Let $\mathsf{ext} \in \mathcal{M}^*$ the sequence of message types sent in ρ with value v' by agents other that a. Let $m_1, \ldots, m_\ell \in \mathcal{M}$ distinct message types so that m_i is the i-th message type to appear in ext; in particular, we have $\ell \leq |\mathcal{M}|$. Let a_i the first agent other than a to broadcast m_i with value v' in ρ. For every $i \in [0, \ell]$, let u_i be the local run executed by a between the first broadcast of m_i (included) and the first broadcast of m_{i+1} (excluded) by agents of $\mathbb{A} \setminus \{a\}$ (before m_1 for $i = 0$, after m_ℓ for $i = \ell$) and let w_i the sequence of message types broadcast by a in u_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 $\mathtt{dec}_i = (w_0, m_1, w_1, \ldots, m_{i-1}, w_{i-1})$. Let ρ_i be the prefix of ρ up until the first broadcast of m_i with value v' (included), and w'_i the v'-input of a_i in ρ_i. Observe that the sequence of message types broadcast with value v' in ρ_i is in $\mathcal{L}^{\mathtt{dec}_i}$ by construction of $\mathtt{dec}$, hence in particular so is w'_i. Agent a_i is a witness that ρ_i satisfies follower specification (w'_i, m_i). If ρ_i is strictly shorter than ρ, then by applying the induction we obtain an unfolding tree τ_i satisfying follower specification (w'_i, m_i). If $\rho_i = \rho$, then ρ ends with a broadcast (m_ℓ, v') by agent a_ℓ; because the last step of ρ is a broadcast with value v, this implies $v = v'$ therefore v is initial for a, which may happen only when spec is a boss specification. In that case, because in our induction order boss specifications are above follower specifications, we can also apply the induction hypothesis and obtain an unfolding tree τ_l satisfying follower specification (w'_ℓ, m_ℓ). Either way, 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.

We claim that the tree constructed satisfies conditions (i) to (iv) thus is an unfolding tree. It suffices to check the conditions at the root μ, as all we did is attach unfolding trees below that root.

- Condition (i) is satisfied by construction of the trees τ'.
- For condition (ii), let v' an initial value of a in u. We reuse the notations from above. u may be split into successive local runs $u_0, \ldots, u_\ell$. Moreover, $w_i = \mathsf{Out}_{v'}(u_i)$ by definition and $\mathsf{In}_v(u_i) \in \{m_1, \ldots, m_{i-1}\}$ because u_i is before the first broadcast of m_i by some agent other that a. Also, for every i, τ_i satisfies follower specification (w'_i, m_i) and w'_i is the v'-input of a_i in ρ_i hence $w'_i \in \mathcal{L}^{\mathtt{dec}_i}$.
- Condition (iii), assuming that spec is a follower specification $(\mathsf{fw}, \mathsf{fm})$, is immediate as we selected a to be a witness that ρ satisfies $(\mathsf{fw}, \mathsf{fm})$ with value v.
- For condition (iv), assume that spec is a boss specification bw. We selected a so that v is an initial value of a. Let w the sequence of messages broadcast with value v in ρ (by all agents). Because ρ satisfies bw, $\mathsf{bw} \preceq w$. Moreover, $w \in \mathcal{L}^{\mathtt{dec}}$ by construction of $\mathtt{dec}$, therefore $w \in \mathcal{L}^{\mathtt{dec}}$.

We have therefore built an unfolding tree τ satisfying spec. ◀

C.2 Proof of Lemma 28

▶ **Lemma 28.** *If there exists an unfolding tree over $\mathcal{P}$ satisfying a boss specification* $\mathsf{bw} \in \mathcal{M}^*$ *then there exists a finite initial run ρ of $\mathcal{P}$ satisfying* bw.

We start with some useful definitions. Indeed, in the rest of this paper, we were using the notion of local run, that corresponds to the behaviour of a given agent in a run. For the proof of Lemma C.2, we need the more general notion of partial run, which corresponds to the projection of a run onto a subset of agents. Note that a partial run may receive message from outside, and therefore one cannot automatically turn a partial run into a run. Intuitively, we will make partial runs correspond with subtrees of an unfolding tree (by *subtree*, we mean the tree composed of a node of the original tree and every node below it).

We define a new semantics over the set of configurations, in which some messages may be received from the outside (without any condition as to where those messages come from). We denote the steps of this new semantics with $\rightarrow_p$.

▶ **Definition 29.** *Let γ, γ' two configurations.*

An internal test from γ to γ', denoted $\gamma \xrightarrow{\textbf{\textit{loc}}}_p \gamma'$, is defined when there is a step $\gamma \rightarrow \gamma'$ in which an agent does a local test.

An internal message from γ to γ', denoted $\gamma \xrightarrow{\textbf{\textit{br}},m,v}_p \gamma'$, is defined when there is a step $\gamma \rightarrow \gamma'$ in which an agent broadcasts a message (m, v).

An external message from γ to γ', denoted $\gamma \xrightarrow{\textbf{\textit{rec}},m,v}_p \gamma'$ is defined when for all agents a, either there is a local step $\gamma(a) \xrightarrow{\mathsf{ext}(m,v)} \gamma'(a)$ or $\gamma(a) = \gamma'(a)$.

A partial step $\gamma \rightarrow_p \gamma'$ is defined if either $\gamma \xrightarrow{\textbf{\textit{loc}}}_p \gamma'$ or $\gamma \xrightarrow{\textbf{\textit{br}},m,v}_p \gamma'$ or $\gamma \xrightarrow{\textbf{\textit{rec}},m,v}_p \gamma'$ for some $m \in \mathcal{M}$, $v \in \mathbb{N}$. A partial run is a sequence of partial steps $\gamma_0 \rightarrow_p \gamma_1 \rightarrow_p \cdots \rightarrow_p \gamma_k$. We write $\gamma \xrightarrow{\rho}_p \gamma'$ the existence of such a partial run ρ from γ to γ'. Note that a local run can be seen as a partial run with a single agent.

We define the v-projection $\rho|_v$ of ρ, which is the word $(b_0, x_0) \cdots (b_\ell, x_\ell) \in (\mathcal{M} \times \{in, out\})^*$ obtained from ρ by mapping every external message $\gamma \xrightarrow{\textbf{\textit{rec}},m,v}_p \gamma'$ to (m, in), every internal message $\gamma \xrightarrow{\textbf{\textit{br}},m,v}_p \gamma'$ to (m, out), and everything else to ε.

The v-input $\mathsf{In}_v(\rho)$ (resp. v-output $\mathsf{Out}_v(\rho)$) of a partial run ρ is the sequence $m_0 \cdots m_k$ such that the projection on $\mathcal{M} \times \{in\}$ (resp. $\mathcal{M} \times \{out\}$) of $\rho|_v$ is $(m_0, in) \cdots (m_k, in)$ (resp. $(m_0, out) \cdots (m_k, out)$).

A partial run is initial if it starts in an initial configuration.

Given a partial run ρ and a value $v \in \mathbb{N}$ appearing in ρ, v is initial in ρ if it is initial for some agent in ρ. Note that this is not always true for partial runs, unlike for runs, because it could be that v was received in ρ through an external message.

For all configurations γ over $\mathbb{A}$ and γ' over $\mathbb{A}'$ such that $\mathbb{A} \cap \mathbb{A}' = \emptyset$, we write $\gamma \sqcup \gamma'$ for the configuration over $\mathbb{A} \cup \mathbb{A}'$ such that $\gamma \sqcup \gamma'(a)$ is $\gamma(a)$ if $a \in \mathbb{A}$ and $\gamma'(a)$ if $a \in \mathbb{A}'$.

The following lemma explains that, when an initial partial run ρ needs to receive a given word on a non-initial value v and an initial run ρ' is able to provide this word for one of its initial values v', then we can make a new initial partial run that no longer needs to receive anything on this value by "plugging" the output of ρ' on v' and the input of ρ on v. All we need to do is execute those runs in parallel over disjoint sets of agents, after renaming values so that the v'-output of ρ' is now on v (which does not induce any conflict as v does not appear as an initial value in ρ). We execute ρ, and every time it requires a message (m, v), we run ρ' up until its next broadcast of (m, v) and use it to complete ρ.

The conditions at the end of the lemma express the fact that this operation does not affect the behaviour of ρ on the values it used before, and that if new values are introduced (the ones appearing in ρ'), they do not require any extra message from outside. Intuitively, the resulting run $\tilde{\rho}$ requires less input than ρ but achieves the same things. This lemma will be used to complete the local run of the root of an unfolding tree over its non-initial values using its boss children.

▶ **Lemma 30.** *Let ρ be an initial partial run, ρ' an initial run. If there exist values $v, v' \in \mathbb{N}$ such that $\mathsf{In}_v(\rho) \preceq \mathsf{Out}_{v'}(\rho')$ and v' is an initial value in ρ' but v is non-initial in ρ, then there exists a partial run $\tilde{\rho}$ such that*

- $\mathsf{In}_v(\tilde{\rho}) = \varepsilon$
- *for all $v'' \neq v$ such that $\rho|_{v''} \neq \varepsilon$, $\tilde{\rho}|_{v''} = \rho|_{v''}$*
- *for all $v'' \neq v$ such that $\rho|_{v''} = \varepsilon$, $\mathsf{In}_{v''}(\tilde{\rho}) = \varepsilon$.*

Proof. Let $\mathbb{A}, \mathbb{A}'$ be the sets of agents of ρ and ρ', respectively. We can rename agents so that those two sets are disjoint. Let $m_1 \cdots m_k = \mathsf{In}_v(\rho)$, ρ can be split into

$$\rho = \gamma_0 \xrightarrow{\rho_0}_p \overline{\gamma}_0 \xrightarrow{\mathsf{ext}(m_1,v)} \gamma_1 \xrightarrow{\rho_1}_p \cdots \xrightarrow{\mathsf{ext}(m_k,v)} \gamma_k \xrightarrow{\rho_k}_p \overline{\gamma}_k$$

so that for all $j \in [0, k]$, $\mathsf{In}_v(\rho_j) = \varepsilon$.

Similarly, as $m_1 \cdots m_k \preceq \mathsf{Out}_v(\rho')$, ρ' can be split into

$$\rho' = \gamma'_0 \xrightarrow{\rho'_0} \overline{\gamma}'_0 \xrightarrow{m_1,v} \gamma'_1 \xrightarrow{\rho'_1} \cdots \xrightarrow{m_k,v'} \gamma'_k \xrightarrow{\rho'_k} \overline{\gamma}'_k$$

For all configurations γ over $\mathbb{A}$ and γ' over $\mathbb{A}'$, we write $\gamma \sqcup \gamma'$ for the configuration over $\mathbb{A} \cup \mathbb{A}'$ such that $(\gamma \sqcup \gamma')(a)$ is $\gamma(a)$ if $a \in \mathbb{A}$ and $\gamma'(a)$ if $a \in \mathbb{A}'$.

We apply a renaming to the values of ρ' so that v' is mapped to v and all other values are mapped to distinct values that do not appear in ρ. Hence the only value appearing in both runs is v, and it is initial only in ρ'. As a consequence, the configuration $\gamma_0 \sqcup \gamma'_0$ is an initial configuration.

We then construct the desired run $\tilde{\rho}$ over $\mathbb{A} \cup \mathbb{A}'$ by matching the external messages in ρ with the broadcasts in ρ' and executing the rest of the run in parallel. Recall that $\gamma \sqcup \gamma'$ is the configuration obtained when putting γ and γ' side by side, which is possible when they refer to disjoint sets of agents.

We have $\tilde{\rho} = \gamma_0 \sqcup \gamma'_0 \xrightarrow{\rho_0}_p \overline{\gamma}_0 \sqcup \gamma'_0 \xrightarrow{\rho'_0}_p \overline{\gamma}_0 \sqcup \overline{\gamma}'_0 \xrightarrow{m_1,v}_p \gamma_1 \sqcup \gamma'_1 \xrightarrow{\rho_1}_p \cdots \xrightarrow{m_k,v}_p \gamma_k \sqcup \gamma'_k \xrightarrow{\rho_k}_p \overline{\gamma}_k \sqcup \gamma'_k \xrightarrow{\rho'_k}_p \overline{\gamma}_k \sqcup \overline{\gamma}'_k$.

We indeed have $\overline{\gamma}_{j-1} \sqcup \overline{\gamma}'_{j-1} \xrightarrow{m_j,v}_p \gamma_j \sqcup \gamma'_j$ as we can make all agents in $\mathbb{A}$ that receive an external messages with message m_j and value v receive the broadcast made in $\mathbb{A}'$ instead.

Hence we obtain a run with

- $\mathsf{In}_v(\tilde{\rho}) = \varepsilon$
- for all $v'' \neq v$, if $\rho|_{v''} \neq \varepsilon$ then v'' does not appear in ρ' after the renaming hence $\tilde{\rho}|_{v''} = \rho|_{v''}$
- for all $v'' \neq v$, if $\rho|_{v''} = \varepsilon$ then $\mathsf{In}_{v''}(\rho) = \varepsilon$ and $\mathsf{In}_{v''}(\rho') = \varepsilon$ as ρ' is a run and not a partial run, hence $\mathsf{In}_{v''}(\tilde{\rho}) = \varepsilon$.

This concludes our proof. ◀

The following proofs are more technical, we now want to be able to merge an initial partial run broadcasting one of its initial values v and initial partial runs that play the roles of follower nodes, i.e., receive a sequence of messages with value v (which is not initial for them) and then broadcast in turn a message with value v. The following lemma will make a step in this direction.

Its statement can be understood as follows: Say we have an initial partial run ρ which requires some input over one of its initial values v. More precisely, we have a decomposition $\mathtt{dec} = (w_0, m_1, \ldots, w_\ell)$ such that ρ can be split into parts that each output w_i while only requiring some message types in $\{m_J \mid j < i\}$ as input over v. On the other hand, we have for each i an initial partial run ρ'_i which can output m_i with some non-initial value v_j while only requiring an input over v_j that forms a word matching decomposition $\mathtt{dec}_i = (w_0, m_1, \ldots, w_{i-1})$. Then we can obtain an initial partial run $\tilde{\rho}$ that can be split the same way as ρ but no longer requires input over v, has the same v'-projection as ρ on values v' appearing in ρ, and does not require any input on other values.

Intuitively, this is done by renaming values in those runs in a suitable way so that the ρ'_i now have an output over value v. We take the last message type m_ℓ and use (many copies of) ρ'_ℓ to eliminate the (m_ℓ, v) inputs in the last part of ρ, while using the output of ρ to complete some of the input of ρ'. The remaining input then only consists of messages m_i with $i < \ell$, which allows us to conclude by induction.

▶ **Lemma 31.** *Let* $\mathtt{dec} = (w_0, m_1, \ldots, w_\ell)$ *be a decomposition,* $\gamma_0, \ldots, \gamma_{\ell+1}$ *configurations (with* γ_0 *an initial configuration) and* v *a value appearing in* γ_0 *such that for all* $j \in [0, \ell]$ *there exists an partial run* ρ_j *from* γ_j *to* γ_{j+1} *with* $w_j \preceq \mathsf{Out}_v(\rho_j)$ *and* $\mathsf{In}_v(\rho_j) \in \{m_1, \ldots, m_{j-1}\}^*$.

Suppose that for all $j \in [1, \ell]$ *there exist an initial partial run* ρ'_j *and a value* v_j *such that* $\mathsf{In}_{v_j}(\rho'_j) \in \mathcal{L}^{\mathtt{dec}_j}$ *where* $\mathtt{dec}_j = (w_0, m_1, \ldots, w_{j-1})$, $\mathsf{In}_{v'}(\rho'_j) = \varepsilon$ *for all* $v' \neq v_j$ *and the last step of* ρ'_j *is an internal message* $\xrightarrow{\mathbf{br}, m, v_j}_p$.

Then, there exist configurations $\tilde{\gamma}_0, \ldots, \tilde{\gamma}_{\ell+1}$, *with* $\tilde{\gamma}_0$ *an initial configuration, such that for all* $j \in [0, \ell]$, *there is a partial run* $\tilde{\rho}_j$ *from* $\tilde{\gamma}_j$ *to* $\tilde{\gamma}_{j+1}$ *and*

- $\mathsf{Out}_v(\rho_j) \preceq \mathsf{Out}_v(\tilde{\rho}_j)$,
- $\mathsf{In}_v(\tilde{\rho}_j) = \varepsilon$,
- *for all* $v' \neq v$, *if* $\rho_j|_{v'} \neq \varepsilon$ *then* $\tilde{\rho}_j|_{v'} = \rho_j|_{v'}$
- *for all* $v' \neq v$, *if* $\rho_j|_{v'} = \varepsilon$ *then* $\mathsf{In}_{v'}(\tilde{\rho}_j) = \varepsilon$

Proof. We prove the property by strong induction on $(|\mathsf{In}_v(\rho)|_{m_\ell}, \ldots, |\mathsf{In}_v(\rho)|_{m_1})$, with the lexicographic ordering, ρ being the initial partial run obtained by concatenating $\rho_1, \ldots, \rho_\ell$.

If $\mathsf{In}_v(\rho) = \varepsilon$ then we can set $\tilde{\rho} = \rho$ to obtain the result.

If $\mathsf{In}_v(\rho) \neq \varepsilon$, then let j be such that $\mathsf{In}_v(\rho_j) \neq \varepsilon$, then we can decompose ρ_j as $\rho_j = \gamma_j \xrightarrow{\rho_j^-}_p \gamma_j^- \xrightarrow{\mathbf{rec}, m_{j'}, v}_p \gamma_j^+ \xrightarrow{\rho_j^+}_p \gamma_{j+1}$ with $j' \leq j$.

By hypothesis, there exist a partial run $\rho'_{j'}$ and a value $v_{j'}$ such that $\mathsf{In}_{v_{j'}}(\rho'_{j'}) \in \mathcal{L}^{\mathtt{dec}_{j'}}$, $\mathsf{In}_{v'}(\rho'_{j'}) = \varepsilon$ for all $v' \neq v_{j'}$ and the last step of $\rho'_{j'}$ is an internal message $\xrightarrow{\mathbf{br}, m, v_{j'}}_p$.

Hence $\rho'_{j'}$ can be decomposed as $\gamma_{j',0} \xrightarrow{\rho'_{j',1}}_p \gamma_{j',1} \cdots \xrightarrow{\rho'_{j',j'-1}}_p \gamma_{j',j'} \xrightarrow{\mathbf{br}, m_{j'}, v}_p \gamma'_{j'}$ where, for all $i \in [1, j'-1]$, the projection of $\mathsf{In}_{v_{j'}}(\rho'_{j',i})$ on $\mathcal{M} \setminus \{m_1, \ldots, m_{i-1}\}$ is a subword of w_i.

We then proceed in a similar way as in Lemma 30. We can rename agents so that ρ and ρ'_j are on disjoint sets of agents, and apply a renaming of values in ρ'_j so that $v_{j'}$ is mapped to v and all other values are mapped to values that do not appear in ρ. In particular the γ_0 and $\gamma_{j',0}$ have no common values, hence $\gamma_0 \sqcup \gamma_{j',0}$ is an initial configuration. We then construct a sequence of runs by executing in parallel $\rho'_{j',i}$ and ρ_i for all $i \in [1, \ell]$. As $w_i \preceq \mathsf{Out}_v(\rho_i)$, we can match the internal messages of ρ_j with some external messages of $\rho'_{j',i}$ so that the remaining external messages with value v in $\rho'_{j',i}$ are all in $\{m_1, \ldots, m'_{j'}\}$. We obtain, for each $i \in [1, j'-1]$, a run ρ''_i from $\gamma_{i-1} \sqcup \gamma_{j',i-1}$ to $\gamma_i \sqcup \gamma_{j',i}$ with $\mathsf{In}_v(\rho''_i) \in \{m_1, \ldots, m_{i-1}\}^*$.

For each $i \in [j'+1, j-1]$, let ρ''_i be the run that goes from $\gamma_{i-1} \sqcup \gamma_{j',j'}$ to $\gamma_i \sqcup \gamma_{j',j'}$ by executing ρ_i on the first part and staying idle on the second one.

Let ρ''_j be the run from $\gamma_{j-1} \sqcup \gamma_{j',j'}$ to $\gamma_j \sqcup \gamma'_{j'}$ by executing ρ^-_j on the first part, then matching the $\gamma^-_j \xrightarrow{\mathbf{rec}, m_{j'}, v}_p \gamma^+_j$ step in ρ with the last step of $\rho'_{j'}$ to obtain a step $\gamma^-_j \sqcup \gamma_{j',j'} \xrightarrow{\mathbf{br}, m_{j'}, v}_p \gamma^+_j \sqcup \gamma'_{j'}$, and then executing ρ^+_j on the first part.

For each $i \in [j+1, \ell]$, let ρ''_i be the run that goes from $\gamma_{i-1} \sqcup \gamma'_{j'}$ to $\gamma_i \sqcup \gamma'_{j'}$ by executing ρ_i on the first part and staying idle on the second one.

Note that for all $i \in [0, \ell]$, for all $v' \neq v$, if $\rho_i|_{v'} \neq \varepsilon$ then $\rho''_i|_{v'} = \rho_i|_{v'}$ (as we renamed values so that ρ and ρ'_j use disjoint sets of values apart from v) and if $\rho_i|_{v'} = \varepsilon$ then $\mathsf{In}_{v'}(\rho''_i) = \mathsf{In}_{v'}(\rho'_{j',i}) = \varepsilon$. Let ρ'' be the concatenation of the ρ''_i. We can apply the induction hypothesis to the runs ρ''_i constructed above, as $|\mathsf{In}_v(\rho'')|_{m_{j'}} < |\mathsf{In}_v(\rho)|_{m_{j'}}$ and $|\mathsf{In}_v(\rho'')|_{m_{j''}} = |\mathsf{In}_v(\rho)|_{m_{j''}}$ for all $j'' > j'$.

We obtain a family of runs $\tilde{\rho}_0, \ldots, \tilde{\rho}_\ell$ such that, for all i, $\mathsf{Out}_v(\rho''_i) \preceq \mathsf{Out}_v(\tilde{\rho}_i)$, $\mathsf{In}_v(\tilde{\rho}_i) = \varepsilon$, and for all $v' \neq v$, if $\rho''_i|_{v'} \neq \varepsilon$ then $\rho''_i|_{v'} = \tilde{\rho}_i|_{v'}$ and if $\rho''_i|_{v'} = \varepsilon$ then $\mathsf{In}_{v'}(\tilde{\rho}_i) = \varepsilon$.

As a consequence, we have, for all $i \in [0, \ell]$:

- $\mathsf{Out}_v(\rho_i) \preceq \mathsf{Out}_v(\rho''_i) \preceq \mathsf{Out}_v(\tilde{\rho}_i)$
- $\mathsf{In}_v(\tilde{\rho}_i) = \varepsilon$
- for all $v' \neq v$, if $\rho_i|_{v'} \neq \varepsilon$ then $\rho''_i|_{v'} = \rho_i|_{v'} \neq \varepsilon$ and thus $\rho_i|_{v'} = \rho''_i|_{v'} = \tilde{\rho}_i|_{v'}$
- for all $v' \neq v$, if $\rho_i|_{v'} = \varepsilon$ then $\mathsf{In}_{v'}(\rho''_i) = \varepsilon$ and either $\rho''_i|_{v'} = \tilde{\rho}_i|_{v'}$, hence in particular $\mathsf{In}_{v'}(\tilde{\rho}_i) = \mathsf{In}_{v'}(\rho''_i) = \varepsilon$, or $\mathsf{In}_{v'}(\tilde{\rho}_i) = \varepsilon$.

All the conditions are satisfied, the lemma is proven. ◀

However, the previous lemma will not suffice, as if we have a boss node with a specification w and an initial value v, it may not suffice to complete the associated local run u so that it does not require input over v: some extra messages may be needed to obtain w as the output of u may not suffice. This can be done by cloning the runs ρ'_i at will, using them to produce the missing messages in w, and then using the previous lemma to complete the resulting partial run so that it does not require input over v. The following lemma uses this idea to provide a convenient extension of Lemma 31

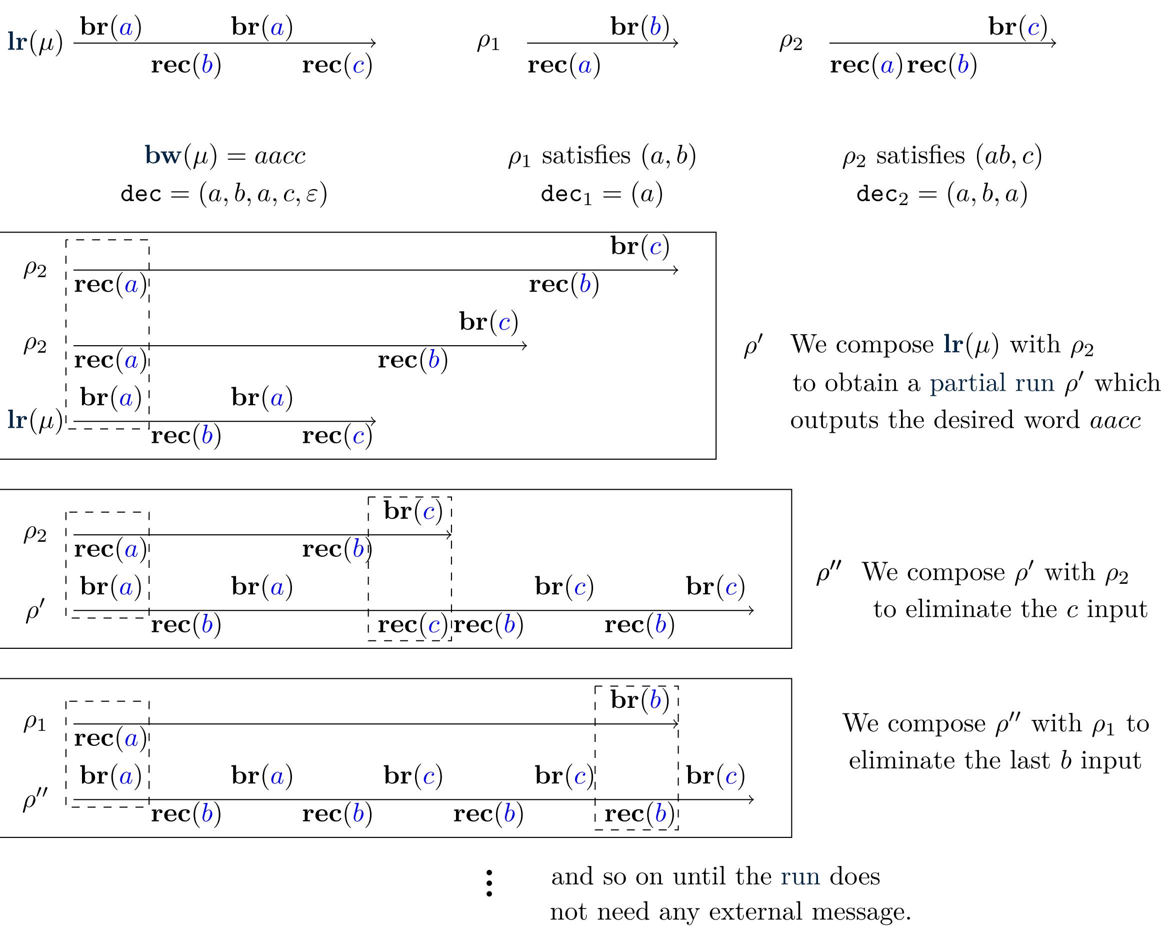
▶ **Lemma 32.** *Let $\mathtt{dec} = (w_0, m_1, \ldots, w_\ell)$ be a decomposition, $w'_0, \ldots, w'_\ell \in \mathcal{M}^*$ such that for all $i \in [0, \ell]$, the projection of w'_i on $\mathcal{M} \setminus \{m_1, \ldots, m_{i-1}\}$ is a subword of w_i.*

Let $\gamma_0, \ldots, \gamma_{\ell+1}$ be configurations, with γ_0 an initial configuration and v a value such that for all $j \in [1, \ell+1]$ there exists a partial run ρ_j from γ_{j-1} to γ_j with $w_j \preceq \mathsf{Out}_v(\rho_j)$ and $\mathsf{In}_v(\rho_j) \in \{m_1, \ldots, m_{j-1}\}^*$.

Suppose that for all $j \in [1, \ell]$ there exist an initial partial run ρ'_j and a value v_j such that $\mathsf{In}_{v_j}(\rho'_j) \in \mathcal{L}^{\mathtt{dec}_j}$ where $\mathtt{dec}_j = (w_0, m_1, \ldots, w_{j-1})$, $\mathsf{In}_{v'}(\rho'_j) = \varepsilon$ for all $v' \neq v_j$ and the last step of ρ'_j is an internal message $\xrightarrow{\mathbf{br}, m, v_j}_p$.

Then, there exist configurations $\tilde{\gamma}_0, \ldots, \tilde{\gamma}_{\ell+1}$, with $\tilde{\gamma}_0$ an initial configuration such that for all $j \in [0, \ell]$, there is a partial run $\tilde{\rho}_j$ from $\tilde{\gamma}_j$ to $\tilde{\gamma}_{j+1}$ and

- *$w'_j \preceq \mathsf{Out}_v(\tilde{\rho}_j)$,*
- *$\mathsf{In}_v(\tilde{\rho}_j) = \varepsilon$,*
- *for all $v' \neq v$, if $\rho_j|_{v'} \neq \varepsilon$ then $\tilde{\rho}_j|_{v'} = \rho_j|_{v'}$*



$\mathbf{lr}(\mu)$
$\mathbf{br}(a)$ $\mathbf{br}(a)$
$\mathbf{rec}(b)$ $\mathbf{rec}(c)$
ρ_1 $\mathbf{br}(b)$ $\mathbf{rec}(a)$
ρ_2 $\mathbf{br}(c)$ $\mathbf{rec}(a)\mathbf{rec}(b)$
$\mathbf{bw}(\mu) = aacc$
$\mathtt{dec} = (a, b, a, c, \varepsilon)$
ρ_1 satisfies (a, b)
$\mathtt{dec}_1 = (a)$
ρ_2 satisfies (ab, c)
$\mathtt{dec}_2 = (a, b, a)$
ρ' We compose $\mathbf{lr}(\mu)$ with ρ_2 to obtain a partial run ρ' which outputs the desired word $aacc$
ρ'' We compose ρ' with ρ_2 to eliminate the c input
We compose ρ'' with ρ_1 to eliminate the last b input
and so on until the run does not need any external message.

Figure 4 Illustration of Lemma 32 (we only show messages over a single value).

- *for all $v' \neq v$, if $\rho_j|_{v'} = \varepsilon$ then $\mathsf{In}_{v'}(\tilde{\rho}_j) = \varepsilon$*

Proof. Let $w' = w'_0 \cdots w'_\ell$. We proceed by strong induction on $|w'|$.

If for all j we have $w'_j \preceq w_j$ (and thus $w'_j \preceq \mathsf{Out}_v(\rho_j)$), the result is a direct consequence of Lemma 31.

Suppose it is not the case. Then there exist j, j' such that $j' \leq j$ and $m_{j'}$ appears in w'_j. Let $w'_j = w_j^- m_{j'} w_j^+$ and let w''_j be the word $w_j^- w_j^+$.

Then, there exist configurations $\widehat{\gamma}_0, \ldots, \widehat{\gamma}_{\ell+1}$, with $\widehat{\gamma}_0$ an initial configuration such that for all $i \in [0, \ell]$, there is a partial run $\widehat{\rho}_i$ from $\widehat{\gamma}_i$ to $\widehat{\gamma}_{i+1}$ and

- $w'_i \preceq \mathsf{Out}_v(\widehat{\rho}_i)$ if $i \neq j$,
- $w''_j \preceq \mathsf{Out}_v(\widehat{\rho}_j)$
- $\mathsf{In}_v(\widehat{\rho}_i) = \varepsilon$,
- for all $v' \neq v$, if $\rho_i|_{v'} \neq \varepsilon$ then $\widehat{\rho}_i|_{v'} = \rho_i|_{v'}$
- for all $v' \neq v$, if $\rho_j|_{v'} = \varepsilon$ then $\mathsf{In}_{v'}(\widehat{\rho}_i) = \varepsilon$

From these runs we infer a sequence of partial runs that output $w'_0, \ldots, w'_\ell$ with value v but have a non-empty v-input. We then use Lemma 31 to obtain the desired runs with an empty v-input.

We proceed in a similar manner as in Lemma 31. There exists a partial run $\rho'_{j'}$ and a value $v_{j'}$ such that $\mathsf{In}_{v_{j'}}(\rho'_{j'}) \in \mathcal{L}^{\mathsf{dec}_{j'}}$, $\mathsf{In}_{v'}(\rho'_{j'}) = \varepsilon$ for all $v' \neq v_{j'}$ and the last step of $\rho'_{j'}$ is an internal message $\xrightarrow{\mathbf{br}, m, v_{j'}}_p$.

Hence $\rho'_{j'}$ can be decomposed as $\gamma_{j',0} \xrightarrow{\rho'_{j',1}}_p \gamma_{j',1} \cdots \xrightarrow{\rho'_{j',j'-1}}_p \gamma_{j',j'} \xrightarrow{\mathbf{br}, m_{j'}, v}_p \gamma'_{j'}$ where, for all $i \in [1, j'-1]$, the projection of $\mathsf{In}_{v_{j'}}(\rho'_{j',i})$ on $\mathcal{M} \setminus \{m_1, \ldots, m_{i-1}\}$ is a subword of w_i.

We can rename agents so that the $\widehat{\rho}_i$ and $\rho'_{j',i}$ are on disjoint sets of agents, and apply a renaming of values in $\rho'_{j'}$ so that $v_{j'}$ is mapped to v and all other values are mapped to values that do not appear in the ρ_i. In particular $\widehat{\gamma}_0$ and $\gamma_{j',0}$ do not share any values, thus $\widehat{\gamma}_0 \sqcup \gamma_{j',0}$ is an initial configuration. We then construct a sequence of runs by executing in parallel $\rho'_{j',i}$ and $\widehat{\rho}_i$ for all $i \in [1, \ell]$. As $w_i \preceq \mathsf{Out}_v(\widehat{\rho}_i)$, we can match the internal messages of $\widehat{\rho}_i$ with some external messages of $\rho'_{j',i}$ so that the remaining external messages with value v in $\rho'_{j',i}$ are all in $\{m_1, \ldots, m'_{j'}\}$. We obtain, for each $i \in [1, j'-1]$, a run ρ''_i from $\widehat{\gamma}_{i-1} \sqcup \gamma_{j',i-1}$ to $\widehat{\gamma}_i \sqcup \gamma_{j',i}$ with $\mathsf{In}_v(\rho''_i) \in \{m_1, \ldots, m_{i-1}\}^*$.

For each $i \in [j'+1, j-1]$, let ρ''_i be the run that goes from $\widehat{\gamma}_{i-1} \sqcup \gamma_{j',j'}$ to $\widehat{\gamma}_i \sqcup \gamma_{j',j'}$ by executing $\widehat{\rho}_i$ on the first part and staying idle on the second one.

We can split $\widehat{\rho}$ into $\widehat{\gamma}_j \xrightarrow{\widehat{\rho}^-}_p \widehat{\gamma}^- \xrightarrow{\widehat{\rho}^+}_p \widehat{\gamma}_{j+1}$ Let ρ''_j be the run from $\widehat{\gamma}_{j-1} \sqcup \gamma_{j',j'}$ to $\widehat{\gamma}_j \sqcup \gamma'_{j'}$ obtained by executing $\widehat{\rho_j^-}$ on the first part, then executing the last step of $\rho'_{j'}$ to broadcast $m_{j'}$ with value v and then executing ρ_j^+ on the first part.

For each $i \in [j+1, \ell]$, let ρ''_i be the run that goes from $\widehat{\gamma}_{i-1} \sqcup \gamma'_{j'}$ to $\widehat{\gamma}_i \sqcup \gamma'_{j'}$ by executing ρ_i on the first part and staying idle on the second one.

Note that for all $i \in [0, \ell]$, for all $v' \neq v$, if $\rho_i|_{v'} \neq \varepsilon$ then $\rho''_i|_{v'} = \rho_i|_{v'}$ (as we renamed values so that ρ and ρ'_j use disjoint sets of values apart from v) and if $\rho_i|_{v'} = \varepsilon$ then $\mathsf{In}_{v'}(\rho''_i) = \mathsf{In}_{v'}(\rho'_{j',i}) = \varepsilon$.

Note that for all i we have $w'_i \preceq \mathsf{Out}_v(\widehat{\rho}_i)$ We can apply Lemma 31 to the $\widehat{\rho}_i$ and the w'_i.

We obtain a sequence of runs $\tilde{\rho}_0, \ldots, \tilde{\rho}_\ell$ such that, for all $i \in [0, \ell]$:

- $\mathsf{Out}_v(\widehat{\rho}_i) \preceq \mathsf{Out}_v(\tilde{\rho}_i)$
- $\mathsf{In}_v(\tilde{\rho}_i) = \varepsilon$
- for all $v' \neq v$, if $\widehat{\rho}_i|_{v'} \neq \varepsilon$ then $\widehat{\rho}_i|_{v'} = \tilde{\rho}_i|_{v'}$
- for all $v' \neq v$, if $\widehat{\rho}_i|_{v'} = \varepsilon$ then $\mathsf{In}_{v'}(\tilde{\rho}_i) = \varepsilon$.

As a result, we have the following properties for all i:

- $w'_i \preceq \mathsf{Out}_v(\widehat{\rho}_i) \preceq \mathsf{Out}_v(\tilde{\rho}_i)$
- $\mathsf{In}_v(\tilde{\rho}_i) = \varepsilon$
- for all $v' \neq v$, if $\rho_i|_{v'} \neq \varepsilon$ then $\rho_i|_{v'} = \widehat{\rho}_i|_{v'} = \tilde{\rho}_i|_{v'}$
- for all $v' \neq v$, if $\rho_i|_{v'} = \varepsilon$ then either $\widehat{\rho}_i|_{v'} = \varepsilon$ and thus $\mathsf{In}_{v'}(\tilde{\rho}_i) = \varepsilon$, or $\widehat{\rho}_i|_{v'} \neq \varepsilon$ and then $\tilde{\rho}_i|_{v'} = \widehat{\rho}_i|_{v'}$, thus in particular $\mathsf{In}_{v'}(\tilde{\rho}_i) = \mathsf{In}_{v'}(\widehat{\rho}_i) = \varepsilon$.

Hence the $\tilde{\rho}_i$ satisfy all the required conditions. ◀

We are now ready to prove Lemma 28.

▶ **Lemma 28.** *If there exists an unfolding tree over $\mathcal{P}$ satisfying a boss specification* $\mathsf{bw} \in \mathcal{M}^*$ *then there exists a finite initial run ρ of $\mathcal{P}$ satisfying* bw.

We actually prove a stronger statement by induction on the unfolding tree. The induction property is as follows. For every unfolding tree τ:

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

Let τ be a unfolding tree, let μ be its root.

We see $u := \mathbf{lr}(\mu)$ as an partial run with a single agent. Recall that by definition of an unfolding tree, u starts with distinct values in all its registers, hence it is an initial partial run. We are going to compose u with some runs given by the children of μ to construct a run satisfying the properties above. Let V be the set of values appearing in u and V_{init} be the set of initial values of u.

C.2.1 Step 1: Non-initial Values

We show the following statement by induction on $|V'|$ for $V' \subseteq V \setminus (V_{init} \cup \{\mathbf{val}(\mu)\})$. It expresses that we can complete u into an initial partial run that does not require any input over its non-initial values. This is done by induction, using Lemma 30 and the initial runs obtained via the boss children to eliminate the input of u over each of its non-initial values without adding some input requirements on other values or modifying its behaviour over values that were previously there. Here is the statement:

For all $V' \subseteq V \setminus (V_{init} \cup \{\mathbf{val}(\mu)\})$, there exists an initial partial run ρ such that for all $v \in \mathbb{N}$:

- If $v \in V \setminus V'$ then $\rho|_v = u|_v$
- If $v \in V' \cup \mathbb{N} \setminus V$ then $\mathsf{In}_v(\rho) = \varepsilon$

If $V' = \emptyset$ then this is clear as we can simply take $\rho := u$.

Now suppose that there exists $v \in V \setminus (V_{init} \cup \{\mathbf{val}(\mu)\} \cup V')$. Let $V'' = V' \cup \{v\}$. By induction hypothesis (on $|V'|$) there exists an initial partial run ρ' such that for all $v' \in \mathbb{N}$:

- If $v' \in V \setminus V'$ then $\rho|_{v'} = u|_{v'}$
- If $v' \in (\mathbb{N} \setminus V) \cup V'$ then $\mathsf{In}_{v'}(\rho) = \varepsilon$

As $v \neq \mathbf{val}(\mu)$, by condition (i) μ has a child μ' such that $\mathsf{In}_v(\mathbf{lr}(\mu)) \preceq \mathbf{bw}(\mu')$. By induction hypothesis (on the unfolding tree), as the subtree rooted in μ' satisfies the boss specification $\mathsf{In}_v(\mu)$, there exists an initial run ρ_v satisfying $\mathsf{In}_v(\mathbf{lr}(\mu))$.

Let $v' \in \mathbb{N}$ be such that $\mathsf{In}_v(\mathbf{lr}(\mu)) \preceq \mathsf{Out}_{v'}(\rho_v)$.

We apply Lemma 30 to obtain an initial run $\tilde{\rho}$ such that

- $\mathsf{In}_v(\tilde{\rho}) = \varepsilon$
- for all $v'' \neq v$, if $\rho|_{v''} \neq \varepsilon$ then $\tilde{\rho}|_{v''} = \rho|_{v''}$
- for all $v'' \neq v$, if $\rho|_{v''} = \varepsilon$ then $\mathsf{In}_{v''}(\tilde{\rho}) = \varepsilon$.

As a result, $\tilde{\rho}$ is an initial partial run satisfying the three requirements for V''. This concludes our induction.

In particular, with $V' = V \setminus (V_{init} \cup \{\mathbf{val}(\mu)\})$, we obtain that there exists an initial run ρ such that, for all $v \in \mathbb{N}$:

- $\mathsf{Out}_v(u) \preceq \mathsf{Out}_v(\rho)$
- If $v \in (V_{init} \cup \{\mathbf{val}(\mu)\})$ then $\rho|_v = u|_v$
- If $v \notin (V_{init} \cup \{\mathbf{val}(\mu)\})$ then $\mathsf{In}_v(\rho) = \varepsilon$

C.2.2 Step 2: Initial Values

We proceed in the same way as in the previous part: the goal is now to use the partial runs yielded by the follower children to eliminate, one by one, using Lemma 31, the input over each non-initial value of u, except maybe for $\mathbf{val}(\mu)$, which requires a special case.

We show the following statement by induction on $|V'|$ for V' such that $V \setminus (V_{init} \cup \{\mathbf{val}(\mu)\}) \subseteq V' \subseteq V \setminus \{\mathbf{val}(\mu)\}$.

For all V' such that $V \setminus (V_{init} \cup \{\mathbf{val}(\mu)\}) \subseteq V' \subseteq V \setminus \{\mathbf{val}(\mu)\}$, there exists an initial partial run ρ' such that for all $v \in \mathbb{N}$:

- If $v \in V \setminus V'$ then $\rho'|_v = u|_v$
- If $v \in V' \cup (\mathbb{N} \setminus V)$ then $\mathsf{In}_v(\rho') = \varepsilon$

If $V' = V \setminus (V_{init} \cup \{\mathbf{val}(\mu)\})$ then this is clear as we can simply take the run ρ constructed in Section C.2.1.

Now suppose there exists $v \in V \setminus (\{\mathbf{val}(\mu)\} \cup V')$. Let $V'' = V' \cup \{v\}$. By induction hypothesis (on $|V'|$) there exists an initial partial run ρ' such that for all $v' \in \mathbb{N}$:

- If $v' \in V \setminus V'$ then $\rho|_{v'} = u|_{v'}$
- If $v' \in (\mathbb{N} \setminus V) \cup V'$ then $\mathsf{In}_{v'}(\rho) = \varepsilon$

As $v \neq \mathbf{val}(\mu)$, by condition (ii), there exists a decomposition $\mathtt{dec} = (w_0, m_1, \ldots, m_\ell, w_\ell)$ such that u can be split into successive local runs $u_0, \cdots, u_\ell$ so that for all $i \in [1, \ell]$, $w_i \preceq \mathsf{Out}_v(u_i)$, $\mathsf{In}_v(u_i) \in \{m_0, \ldots, m_{i-1}\}^*$ and μ has a child μ_i such that $\mathbf{fm}(\mu_i) = m_i$ and $\mathbf{fw}(\mu_i) \in \mathcal{L}^{\mathtt{dec}_i}$ with $\mathtt{dec}_i = (w_0, m_1, \ldots, w_{i-1})$.

By induction hypothesis (on the unfolding tree), for each $i \in [1, \ell]$ there exist an initial partial run ρ'_i and a value v'_i such that $\mathsf{In}_{v''}(\rho'_i) = \varepsilon$ for all $v'' \neq v'_i$, $\mathsf{In}_{v'_i}(\rho'_i) \preceq \mathbf{fw}(\mu_i)$ and $\mathsf{Out}_{v'_i}(\rho'_i)$ contains $\mathbf{fm}(\mu_i)$.

For each i we consider the shortest prefix ρ''_i of ρ'_i ending with an internal message $\xrightarrow{\mathbf{br}, \mathbf{fm}(\mu_i), v'_i}_p$, we have that $\mathsf{In}_{v'_i}(\rho''_i) \preceq \mathsf{In}_{v'_i}(\rho'_i) \preceq \mathbf{fw}(\mu_i)$.

Furthermore, as u can be split into $u_0, \cdots, u_\ell$ so that for all $i \in [1, \ell]$, $w_i \preceq \mathsf{Out}_v(u_i)$ and $\mathsf{In}_v(u_i) \in \{m_0, \ldots, m_{i-1}\}^*$, we can do the same for ρ as $v \in V_{init}$ and thus by definition of ρ we have $\rho|_v = u|_v$. Hence we obtain a sequence of runs $\rho_0, \ldots, \rho_\ell$ such that for all $i \in [1, \ell]$, $w_i \preceq \mathsf{Out}_v(\rho_i)$ and $\mathsf{In}_v(\rho_i) \in \{m_0, \ldots, m_{i-1}\}^*$.

We apply Lemma 31 to obtain a sequence of runs $\tilde{\rho}_0, \ldots, \tilde{\rho}_\ell$ such that, for all i,

- $\mathsf{In}_v(\tilde{\rho}_i) = \varepsilon$
- for all $v' \neq v$, if $\rho_i|_v \neq \varepsilon$ then $\tilde{\rho}|_{v'} = \rho|_{v'}$

- for all $v' \neq v$, if $\rho|_v = \varepsilon$ then $\mathsf{In}_{v''}(\tilde{\rho}) = \varepsilon$.

As a result, $\tilde{\rho}$ is an initial partial run satisfying the three requirements for V''. This concludes our induction.

In particular, with $V' = V \setminus \{\mathbf{val}(\mu)\}$, we obtain that there exists an initial partial run $\widehat{\rho}$ such that for all $v \in \mathbb{N}$:

- If $v = \mathbf{val}(\mu)$ then $\widehat{\rho}|_v = \rho|_v = u|_v$
- If $v \neq \mathbf{val}(\mu)$ then $\mathsf{In}_v(\widehat{\rho}) = \mathsf{In}_v(\rho) = \varepsilon$

C.2.3 Step 3: $\mathbf{val}(\mu)$

In this final part, we use Lemma 32 to complete the partial run previously obtained so that it has enough output on $\mathbf{val}(\mu)$ to satisfy the specification of μ. In fact, if μ is a follower node, the partial run constructed so far suffices. If μ is a boss node, we need to apply Lemma 32.

In Section C.2.2 we constructed an initial partial run $\widehat{\rho}$ such that for all $v \in \mathbb{N}$:

- If $v = \mathbf{val}(\mu)$ then $\widehat{\rho}|_v = u|_v$
- If $v \neq \mathbf{val}(\mu)$ then $\mathsf{In}_v(\widehat{\rho}) = \varepsilon$

We have two cases: either μ is a boss node or a follower node.

If μ is a boss node then there exists a decomposition $\mathtt{dec} = (w_0, m_1, \ldots, w_\ell)$ such that $\mathbf{bw}(\mu) \in \mathcal{L}^{\mathtt{dec}}$ and u can be split into $u_0, \cdots, u_\ell$ so that for all $i \in [1, \ell]$, $w_i \preceq \mathsf{Out}_v(u_i)$ and $\mathsf{In}_v(u_i) \in \{m_0, \ldots, m_{i-1}\}^*$ and μ has a child μ_i such that $\mathbf{fm}(\mu_i) = m_i$ and $\mathbf{fw}(\mu_i) \in \mathcal{L}^{\mathtt{dec}_i}$, with $\mathtt{dec}_i = (w_0, m_1, \ldots, w_{i-1})$.

As $\mathbf{bw}(\mu) \in \mathcal{L}^{\mathtt{dec}}$, we have $\mathbf{bw}(\mu) = w'_0 \cdots w'_\ell$ where for all i the projection of w'_i on $\mathcal{M} \setminus \{m_1, \ldots, m_{i-1}\}$ is a subword of w_i.

We repeat the arguments from Section C.2.2, but we use Lemma 32 instead of Lemma 31 to obtain an initial run $\tilde{\rho}$ such that,

- $\mathsf{In}_v(\tilde{\rho}) = \varepsilon$ for all $v \in \mathbb{N}$
- $\mathbf{bw}(\mu) \preceq \mathsf{Out}_{\mathbf{val}(\mu)}(\tilde{\rho})$

If τ satisfies a boss specification w then $w \preceq \mathbf{bw}(\mu)$ and thus $w \preceq \mathsf{Out}_{\mathbf{val}(\mu)}(\tilde{\rho})$. As a result, $\tilde{\rho}$ satisfies w as well. If μ is a follower node then the run $\widehat{\rho}$ previously constructed is such that

- $\mathsf{In}_v(\tilde{\rho}) = \varepsilon$ for all $v \neq \mathbf{val}(\mu)$
- $\mathsf{In}_{\mathbf{val}(\mu)}(\widehat{\rho}) = \mathsf{Out}_{\mathbf{val}(\mu)}(\mathbf{lr}(\mu)) \preceq \mathbf{fw}(\mu)$
- $\mathsf{Out}_{\mathbf{val}(\mu)}(\widehat{\rho}) = \mathsf{Out}_{\mathbf{val}(\mu)}(\mathbf{lr}(\mu))$ contains $\mathbf{fm}(\mu)$

Hence $\widehat{\rho}$ satisfies the required properties. This concludes our induction.

D Proof of Lemma 18

▶ **Lemma 18.** *Let τ be a coverability witness for $(\mathcal{P}, q_f)$. Let μ, μ' be two nodes of τ such that μ is an ancestor of μ'. If one of the conditions below holds, then there exists a coverability witness for $(\mathcal{P}, q_f)$ of size smaller than $|\tau|$:*

- *μ and μ' are boss nodes and $\mathbf{bw}(\mu) \preceq \mathbf{bw}(\mu')$; or*
- *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 $\tau_{\mu'}$. The size of τ' is smaller than the one of τ, as $\tau_{\mu'}$ is a strict subtree of τ_μ.

If μ is the root of τ, then τ' is a unfolding tree because it is the subtree of τ rooted in μ'. It satisfies specification $\mathbf{spec}(\mu')$ hence also $\mathbf{spec}(\mu)$.

If μ is not the root of τ then let μ'' be the parent of μ. We have to check that τ' is a unfolding tree. All nodes other than μ'' have the same children as before, thus the conditions of unfolding trees are still respected for them. As for μ'', the only problematic cases are values v which had μ as witness in conditions (i) (if μ is a boss node) and (ii) (if μ is a follower node).

Let v a value for which μ'' relied on μ for one of the two conditions. If μ is a boss node (v is non-initial), we had $\mathsf{In}_v(\mathbf{lr}(\mu'')) \preceq \mathbf{bw}(\mu) \preceq \mathbf{bw}(\mu')$ 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.

Note that condition (iii) is satisfied at μ'' in τ' because its local run and specification did not change from τ to τ', and that (iv) is satisfied at μ'' in τ' because its local run, specification and the corresponding decomposition did not change from τ to τ'. As a result, in both cases τ' is a unfolding tree smaller than τ that satisfies spec, which concludes the proof. ◀

E Proof of Lemma 19

We start by defining the notion of trace. 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 $(q, \nu) \xrightarrow{\mathsf{tr}} (q', \nu')$ to express that there exists a local run of trace tr from (q, ν) to (q', ν').

We prove the following result:

▶ **Lemma 19.** *There exists a primitive recursive function $\psi(n, r)$ such that, for every protocol $\mathcal{P}$ with r registers, for every local run $u_0 : (q_0, \nu_0) \xrightarrow{*} (q_f, \nu_f)$ in $\mathcal{P}$, for every section $u : (q, \nu) \xrightarrow{*} (q, \nu')$ of u_0, for every $V \subseteq \mathbb{N}$ finite such that V contains all message values appearing in u, there exists a local run $u' : (q, \nu) \xrightarrow{*} (q', \nu')$ such that we have $\mathsf{len}(u'') \leq \psi(|\mathcal{P}|, r)$ and:*

1. *for all $v' \in \mathbb{N} \setminus V$, there exists v a non-initial value of u_0 such that $\mathsf{In}_{v'}(u') \preceq \mathsf{In}_v(u)$,*
2. *for all $v \in V$, $\mathsf{In}_v(u') \preceq \mathsf{In}_v(u)$.*

We actually prove a slightly more general version of the lemma, to reduce the notations in the proof. The previous lemma can be obtained simply by applying the following one with W the set of initial values of u_0 (and noticing that W then contains r values).

▶ **Lemma 33.** *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 $v' \in \mathbb{N} \setminus V$, there exists $v \in \mathbb{N} \setminus W$ such that $\mathsf{In}_{v'}(u') \preceq \mathsf{In}_v(u)$,*
2. *for all $v \in V$, $\mathsf{In}_v(u') \preceq \mathsf{In}_v(u)$.*

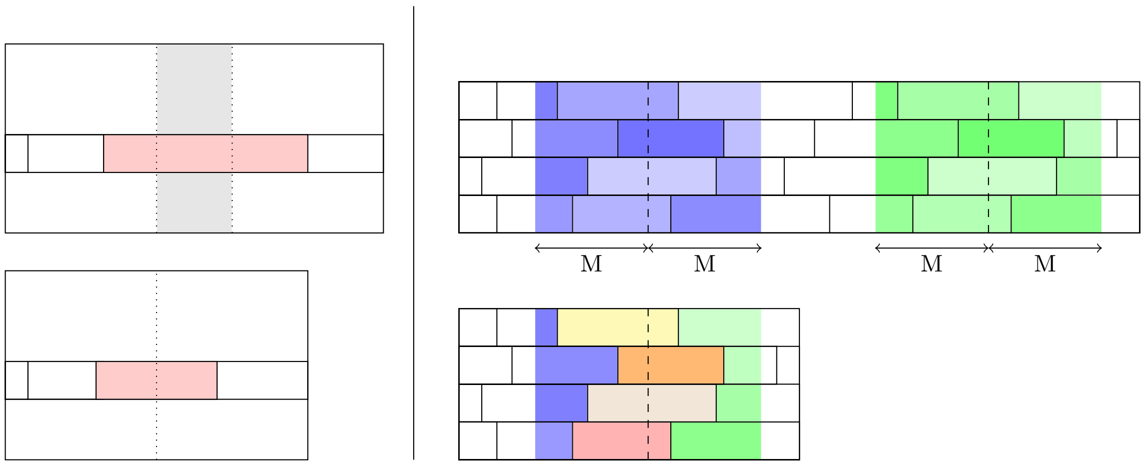


Figure 5 Illustration of the proof of Lemma 19. We represent local runs as above, where lines correspond to registers, and vertical separations are times at which a $\downarrow$ operation is performed on that register. If one register i keeps the same value for a long enough time (on the left), we apply the induction hypothesis to shorten the projection of the run on the other registers. As the value of i does not change, the resulting run is still valid. If all registers change values often, say every M steps (on the right), then if the run is long enough we can find two identical sequences of transitions during which all values are renewed twice. We can then obtain a shorter run by glueing them together as in the picture. The colored rectangles in the shortened runs correspond to fresh values that we introduce to make sure that all local disequality tests succeed.

Intuitively, the set V represents values that are already used somewhere and therefore cannot be used as fresh values, and W represents values that other values should not copy because their behavior is special (this will later correspond to initial values of the run). However, because this lemma is meant to be applied to several parts of a run, we have to state it for any value for V and W.

Proof. Given a local run u, register i is *active* in u if at least one '$\downarrow$' 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 a local run $u : (q_i,\nu_i) \xrightarrow{*} (q_f,\nu_f)$ with k active registers such that $\mathsf{len}(u) > \psi(n,k)$ and let $V \subseteq \mathbb{N}$ finite that contains every message value appearing in u. 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 $v' \in \mathbb{N}$, there exists $v \in \mathbb{N}$ such that $\mathsf{In}_{v'}(u')$ is a subword of $\mathsf{In}_v(u)$,
- for all $v \in V$, $\mathsf{In}_v(u')$ 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.

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

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 (and V, W).

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. Suppose that $\mathsf{len}(u) > \psi(n,k+1)$. Additionally, suppose that no local configuration appears twice in u (otherwise u may easily be shortened).

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 $i \in [0,(n+1)^{2M}]$, 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 matches 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 or a local test, 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).

For every register i, let $\nu_a(i), \nu_b(i)$ the content of register i after the l-th step in u_a and u_b respectively.

λ is an reception step

If λ is an reception step of the form $\mathsf{ext}(\delta, v)$, then δ has action $\mathbf{rec}(m, i, \alpha)$. Let v_a, v_b the value of the corresponding reception step in u_a and u_b respectively. If $i \notin I$ then α is either '$*$' or a test, which is valid as $\nu(i) = \nu_a(i) = \nu_b(i)$.

The only problematic cases are when $i \in I$ and either $\alpha =$ '$\neq$' or $\alpha =$ '$=$'.

In this case, we prove that $v\ \alpha\ \nu(i)$ with $\alpha \in \{$'$=$', '$\neq$'$\}$. 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 the value of $l+1$:

- $l+1 < e_i$: $\nu(i) = \nu_a(i)$, $v = v_a$ 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 $\alpha =$ '$\neq$') $\phi(\nu_a(i))\ \alpha\ \phi(v_a)$.
- $f_i \leq l+1$: $\nu(i) = \nu_b(i)$ and $v = v_b$, and because the internal step is valid in u_b we have $\nu_b(i)\ \alpha\ v_b$.

λ is an internal step

The only problematic case is when $\lambda =: \mathbf{loc}(i, i', \neq)$ with $i, i' \in [1, r]$ (we have no local equality tests thanks to Proposition 9). We have to prove that $\nu(i) \neq \nu(i')$. Because the local test is satisfied in u_a and u_b, we have $\nu_a(i) \neq \nu_a(i')$ and $\nu_b(i) \neq \nu_b(i')$.

If $i \notin I$ and $i' \notin I$ (none are active registers) then $\nu(i) = \nu_a(i) \neq \nu_a(i') = \nu(i')$.

If $i \in I$ and $i' \notin I$, then $\nu(i') = \nu_a(i') = \nu_b(i')$ and:

- if $l+1 < e_i$ then $\nu(i) = \nu_a(i) \neq \nu_a(i')$,
- if $e_i \leq l+1 < f_i$ then $\nu(i) = \phi(\nu_a(i))$ and either
  - $\nu_a(i) \in W$ hence $\nu(i) = \phi(\nu_a(i)) = \nu_a(i) \neq \nu_a(i')$
  - $\nu(i') \notin W$ therefore $\phi(\nu_a(i)) \notin V$ and, as $\nu(i') = \nu_a(i') \in V$, $\nu(i) \neq \nu(i')$
- if $f_i \leq l+1$ then $\nu(i) = \nu_b(i) \neq \nu_b(i')$.

We treat the case $i \notin I$ and $i' \in I$ symmetrically.

If $i \in I$ and $i' \in I$, recall that $e_i \leq M < f_{i'}$ and $e_{i'} \leq M < f_i$ thus $e_i < f_{i'}$ and $e_{i'} < f_i$. We again make a case disjunction on the value of $l+1$:

- $l+1 < e_i$ and $l+1 < e_{i'}$: $\nu(i) = \nu_a(i)$, $\nu(i') = \nu_a(i')$ and $\nu_a(i) \neq \nu_a(i')$.
- $e_i \leq l+1 < f_i$, $l+1 < e_{i'}$: $\nu(i) = \phi(\nu_a(i))$ and either
  - $\nu_a(i) \in W$, hence $\nu(i) = \phi(\nu_a(i)) = \nu_a(i) \neq \nu_a(i') = \nu(i')$, or
  - $\nu_a(i) \notin W$, hence $\phi(\nu_a(i)) \notin V$, and $\nu(i') = \nu_a(i') \in V$ therefore $\nu(i') \neq \nu(i)$.
- $e_{i'} \leq l+1 < f_{i'}$, $l+1 < e_i$: symmetric to the previous case.
- $e_i \leq l+1 < f_i$, $e_{i'} \leq l+1 < f_{i'}$: $\nu(i) = \phi(\nu_a(i))$, $\nu(i') = \phi(\nu_a(i'))$; however $\nu_a(i) \neq \nu_a(i')$ and ϕ is injective hence $\nu(i') \neq \nu(i)$.
- $e_i \leq l+1 < f_i$, $f_{i'} \leq l+1$: $\nu(i') = \nu_b(i') \in V$, $\nu(i) = \phi(\nu_a(i))$ and either
  - $\nu_a(i) \in W$, in which case $\nu_a(i) = \nu_b(i)$ because $\theta_{i_a} = \theta_{i_b}$ (u_a and u_b coincide on values in W) hence $\nu(i) = \phi(\nu_a(i)) = \phi(\nu_b(i)) = \nu_b(i) \neq \nu_b(i') = \nu(i')$, or
  - $\nu_a(i) \notin W$, hence $\phi(\nu_a(i)) \notin V$, and $\nu(i') = \nu_b(i') \in V$ therefore $\nu(i') \neq \nu(i)$.
- $f_i \leq l+1$, $e_{i'} \leq l+1 < f_{i'}$: symmetric to the previous case.

- $f_i \leq l+1$, $f_{i'} \leq l+1$: $\nu(i) = \nu_b(i)$, $\nu(i) = \nu_b(i)$ and $\nu_b(i) \neq \nu_b(i')$.

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 $2M+1$ 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 local run $u' : (q_i, \nu_i) \xrightarrow{*} (q_f, \nu_f)$ with strictly less steps that u.

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'.

We now prove the desired properties on u_1.

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 conditions 2.

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 1 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 $V^{(k)} \cup W$ where W 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 Proof of Lemma 22

Our aim is now to bound the size of a node with respect to its follower children (and its father if it is a boss node). Those are the nodes representing agents that may require a large input from that node. We start with a technical lemma explaining that the decompositions witnessing the satisfaction of conditions (ii) and (iii) of unfolding trees do not need to be larger than the sum of the sizes of the corresponding follower children¨.

▶ **Lemma 34.** *Let* $\mathsf{dec} = (w_0, m_1, \ldots, w_\ell)$ *be a decomposition, and for all* $i \in [0, \ell]$ *let* $\mathsf{dec}_i = (w_0, m_1, \ldots, w_i)$*. Let* $w'_0, \ldots, w'_\ell$ *be such that* $w'_i \in \mathcal{L}^{\mathsf{dec_i}}$ *for all* i*.*

Then there exists $\widehat{\mathsf{dec}} = (\widehat{w}_0, m_1, \ldots, \widehat{w}_\ell)$ *such that, for all* i*,* $\widehat{w}_i \preceq w_i$ *and* $w'_i \in \mathcal{L}^{\widehat{\mathsf{dec_i}}}$ *(where* $\widehat{\mathsf{dec}}_i = (\widehat{w}_0, m_1, \ldots, \widehat{w}_\ell)$*), and* $\sum_{i=0}^{\ell} |\widehat{w}_i| \leq \sum_{i=0}^{\ell} |w'_i|$*.*

Note that in the lemma statement, w'_i could be small regarding the decomposition dec_i, the lemma states that we can consider decompositions smaller than the words $w'_1 \ldots w'_\ell$.

Proof. In the following proof, for $M \subseteq \mathcal{M}$ a set of message types and $w \in \mathcal{M}^*$, we denote the word projection of w on M by $\pi_M(w)$.

For all i, as $w'_i \in \mathcal{L}^{\mathsf{dec_i}}$, we can split w'_i into $v_{i,0} \cdots v_{i,i}$ where, for all j, $\pi_{\mathcal{M}\setminus\{m_1,\dots,m_j\}}(v_{i,j})$ is a subword of w_j.

For all $j \in [0,\ell]$, for all $i \geq j$, $\pi_{\mathcal{M}\setminus\{m_1,\dots,m_j\}}(v_{i,j})$ is a subword of w_j. Hence we can find a subword $\widehat{w}_j$ of w_j of length at most $\sum_{i=0}^{j} |\pi_{\mathcal{M}\setminus\{m_1,\dots,m_j\}}(v_{i,j})|$ such that $\pi_{\mathcal{M}\setminus\{m_1,\dots,m_j\}}(v_{i,j})$ is a subword of $\widehat{w}_j$ for all i, by considering for each i a set of positions in w_j that forms $\pi_{\mathcal{M}\setminus\{m_1,\dots,m_j\}}(v_{i,j})$, and then setting $\widehat{w}_j$ as the word formed by the union of those sets of positions.

We then define $\widehat{\mathsf{dec}} = (\widehat{w}_0, m_1, \dots, \widehat{w}_\ell)$ and for all i, $\widehat{\mathsf{dec}}_i = (\widehat{w}_0, m_1, \dots, \widehat{w}_\ell)$.

We obtain that for all i, $w'_i = v_{i,0} \cdots v_{i,i}$ with, for all $j \in [0,i]$, $\pi_{\mathcal{M}\setminus\{m_1,\dots,m_j\}}(v_{i,j}) \preceq \widehat{w}_j$. Hence we have $w'_i \in \mathcal{L}^{\mathsf{dec_i}}$.

Furthermore we have $\sum_{i=0}^{\ell} |\widehat{w}_i| \leq \sum_{i=0}^{\ell} \sum_{j=0}^{i} |v_{i,j}| = \sum_{i=0}^{\ell} |w'_i|$. ◀

▶ **Lemma 22.** *Let $\mathcal{P}$ be a protocol over r registers, let τ be an unfolding tree over $\mathcal{P}$ of minimal size satisfying a boss specification* bw*, let μ be a node of τ. Let K be such that for all follower children μ_f of μ, $|\mathbf{fw}(\mu_f)| \leq K$. We have the following properties:*

1. *If μ is a boss node then*
   - *If μ is the root of τ then $\mathbf{bw}(\mu) = w$, otherwise $|\mathbf{bw}(\mu)| \leq |\mathbf{lr}(\mu')|$ with μ' its parent*
   - *In both cases $|\mathbf{lr}(\mu)| \leq \psi(|\mathcal{P}|, r)\Big[|\mathbf{bw}(\mu)| + |\mathcal{M}|rK + 1\Big]$*
2. *If μ is a follower node then $|\mathbf{fw}(\mu)| \leq |\mathbf{lr}(\mu)| \leq \psi(|\mathcal{P}|, r)\Big[1 + |\mathcal{M}|rK\Big]$*

Proof. Let μ be a node of τ. Let W be the set of initial values being broadcast or received in $\mathbf{lr}(\mu)$. We have $|W| \leq r$ as there is at most one such value for each register.

Let μ be a boss node. If μ is the root, then as τ satisfies bw we have $\mathsf{bw} \preceq \mathbf{bw}(\mu)$. If $\mathsf{bw} \neq \mathbf{bw}(\mu)$, we can replace $\mathbf{bw}(\mu)$ with bw: As $\mathsf{bw} \preceq \mathbf{bw}(\mu)$, for all decomposition $\mathtt{dec}$, if $\mathbf{bw}(\mu) \in \mathcal{L}^{\mathsf{dec}}$ then $\mathsf{bw} \in \mathcal{L}^{\mathsf{dec}}$, hence condition (iv) is still satisfied, and the other conditions of a unfolding tree are unaffected by this change. Furthermore the resulting unfolding tree still satisfies bw. This contradicts the minimality of τ, hence we have $\mathbf{bw}(\mu) = \mathsf{bw}$.

If μ is not the root, then let μ' be its parent. Let V' be the set of non-initial values v' of μ' such that $\mathsf{In}_{v'}(\mathbf{lr}(\mu')) \preceq \mathbf{bw}(\mu)$.

We can construct a subword bw' of $\mathbf{bw}(\mu)$ of length at most $\sum_{v' \in V'} |\mathsf{In}_{v'}(\mathbf{lr}(\mu'))|$ such that all $\mathsf{In}_{v'}(\mathbf{lr}(\mu'))$ are subwords of bw'. It suffices to select for each v' a set of positions in $\mathbf{bw}(\mu)$ forming $\mathsf{In}_{v'}(\mathbf{lr}(\mu'))$, and then taking the word formed by the union of those sets of positions. Observe that we necessarily have $\sum_{v' \in V'} |\mathsf{In}_{v'}(\mathbf{lr}(\mu'))| \leq |\mathbf{lr}(\mu')|$, thus $|\mathsf{bw}'| \leq |\mathbf{lr}(\mu')|$.

Like in the previous case, as bw' is a subword of $\mathbf{bw}(\mu)$, for all decomposition $\mathtt{dec}$, if $\mathbf{bw}(\mu) \in \mathcal{L}^{\mathsf{dec}}$ then $\mathsf{bw}' \in \mathcal{L}^{\mathsf{dec}}$. As a result, condition (iv) is still satisfied after switching the label $\mathbf{bw}(\mu)$ to bw'. Furthermore, by definition of V' and bw', for all non-initial value v' in $\mathbf{lr}(\mu')$, if $\mathsf{In}_{v'}(\mathbf{lr}(\mu')) \preceq \mathbf{bw}(\mu)$ then $\mathsf{In}_{v'}(\mathbf{lr}(\mu')) \preceq \mathsf{bw}'$, hence condition (i) is still satisfied as well. Other conditions are unaffected by this change.

Hence if $|\mathbf{bw}(\mu)| > |\mathbf{lr}(\mu')|$ we can obtain a unfolding tree smaller than τ and satisfying w, contradicting the minimality of τ. As a result, $|\mathbf{bw}(\mu)| \leq |\mathbf{lr}(\mu')|$.

We have shown the first point of property 1. For the second one, we isolate the broadcasts of $\mathbf{lr}(\mu)$ that are necessary to satisfy the unfolding tree conditions and then bound the length of $\mathbf{lr}(\mu)$ using Lemma 19.

Let $u = \mathbf{lr}(\mu)$, let v be an initial value of u. If $v = \mathbf{val}(\mu)$ then by condition (iv) there exist a decomposition $\mathtt{dec}_v = (w_{v,0}, m_{v,1}, \dots, w_{v,\ell_v})$ and follower children $\mu_{v,1}, \dots, \mu_{v,\ell_v}$ such that

- $\mathbf{bw}(\mu) \in \mathcal{L}^{\mathsf{dec_v}}$

- u can be split into $u_{v,0}, \dots, u_{v,\ell_v}$ so that for all i, $w_{v,i} \preceq \mathsf{Out}_v(u_{v,i})$ and $\mathsf{In}_v(u_{v,i}) \in \{m_{v,1}, \dots, m_{v,i-1}\}^*$
- for all i, $\mathbf{fw}(\mu_{v,i}) \in \mathcal{L}^{\mathtt{dec_{v,i}}}$, with $\mathtt{dec}_{v,i} = (w_{v,0}, m_{v,1}, \dots, w_{v,i-1})$, and $\mathbf{fm}(\mu_{v,i}) = m_{v,i}$.

Similarly, if $v \neq \mathbf{val}(\mu)$, we set $\mathtt{dec}_v = (w_{v,0}, m_{v,1}, \dots, w_{v,\ell_v})$, follower children $\mu_{v,1}, \dots, \mu_{v,\ell_v}$ and $u_{v,0}, \dots, u_{v,\ell_v}$ such that condition (ii) is satisfied.

By Lemma 34, we can assume that $\sum_{i=0}^{\ell_v} |w_{v,i}| \leq \sum_{i=0}^{\ell_v} |\mathbf{fw}(\mu_{v,i})|$ if $v \neq \mathbf{val}(\mu)$, and $\sum_{i=0}^{\ell_v} |w_{v,i}| \leq |\mathbf{bw}(\mu)| + \sum_{i=0}^{\ell_v} |\mathbf{fw}(\mu_{v,i})|$ if $v = \mathbf{val}(\mu)$.

For all $v \in W$, we define J_v as a set of positions in u corresponding to the broadcasts of the letters of all the $w_{v,i}$. Formally, let $w_v = w_{v,0} \cdots w_{v,\ell_v}$, J_v is a set of indices $j_{v,1} < j_{v,2} < \dots < j_{v,N_v}$ such that:

- $N_v = |w_v|$
- for all $a \in [1, N_v]$, the $j_{v,a}$th step of u is a broadcast of the ath letter of w_v with value v.
- for all index j', if the j'th step in u is a reception step of $m_{v,i}$ then $w_{v,0} \cdots w_{v,i-1}$ has already been broadcast, i.e., $w_{v,0} \cdots w_{v,i-1}$ is a prefix of the sequence of message types broadcast at positions in $J_v \cap [1, j']$.

We have, by definition, $|J_v| \leq \sum_{i=0}^{\ell_v} |w_{v,i}|$ for all v. Let J be the union of all J_v, we have $|J| \leq \sum_{v \in W} \sum_{i=0}^{\ell_v} |w_{v,i}| \leq |\mathbf{bw}(\mu)| + \sum_{v \in W} \sum_{i=0}^{\ell_v} |\mathbf{fw}(\mu_{v,i})| \leq |\mathbf{bw}(\mu)| + |\mathcal{M}|rK$. The last inequality is obtained by observing that u has at most r initial values (one per register), $\ell_v \leq |M|$ for all v by definition of a decomposition (as all $(m_{v,i})_{1 \leq i \leq \ell_v}$ are distinct), and $|\mathbf{fw}(\mu_{v,i})| \leq K$ for all v, i by definition of K.

We now use Lemma 19 repeatedly. Let $j_1 < \dots < j_N$ be the indices of J, in increasing order. For all $a \in [1, N]$, let (q_a, ν_a) (resp. $(q'_a, \nu'_a))$) be the local configuration right after (resp. before) the j_ath step of u, and let u_a be the section of u from (q_{a-1}, ν_{a-1}) (with (q_0, ν_0) the initial configuration of u) to (q'_a, ν'_a).

For each a, we are going to define a local run $u'_a : (q_{a-1}, \nu_{a-1}) \xrightarrow{*} (q'_a, \nu'_a)$ and a finite set of values V_a. We do this in increasing order: let V_0 be the set of values appearing in u and for each a, let u'_a be the local run obtained by applying Lemma 19 to u_a with $V = V_{a-1}$. Let V_a be V_{a-1} to which we added all values appearing in u'_a.

We obtain, for each a, a local run $u'_a : (q_{a-1}, \nu_{a-1}) \xrightarrow{*} (q'_a, \nu'_a)$ of length at most $\psi(|\mathcal{P}| - r + r, r) = \psi(|\mathcal{P}|, r)$ such that

1. for all $v' \in \mathbb{N} \setminus V_{a-1}$, there exists $v \in \mathbb{N} \setminus W$ such that $\mathsf{In}_{v'}(u'_a)$ is a subword of $\mathsf{In}_v(u_a)$,
2. for all $v \in V_{a-1}$, $\mathsf{In}_v(u'_a)$ is a subword of $\mathsf{In}_v(u_a)$.

Let u' be the local run obtained by replacing all u_a with u'_a. Using the bounds on $|J|$ and on $|u'_a|$ for each a, we obtain that $|u'| \leq (\psi(|\mathcal{P}|, r) + 1)[|\mathbf{bw}(\mu)| + |\mathcal{M}|rK]$.

We replace the local run label of μ with u'.

It is now left to prove that the resulting tree is indeed an unfolding tree. For all $v \in W$ of u' (which are the same as for u), as $v \in V_0$ and as (V_a) is an increasing sequence, $v \in V_a$ for all a. As a consequence, $\mathsf{In}_v(u'_a)$ is a subword of $\mathsf{In}_v(u_a)$ for all a. As J contains J_v, we can conclude that u' matches decomposition $\mathtt{dec}_v$ in the same way as u, hence conditions (ii) and (iv) are still satisfied.

For all non-initial value v' of u', if v' appears in u then the $\mathsf{In}_{v'}(u'_a)$ is a subword of $\mathsf{In}_{v'}(u_a)$ for all a, hence $\mathsf{In}_{v'}(u')$ is a subword of $\mathsf{In}_{v'}(u)$. Hence we can use the same boss child of μ to witness the satisfaction of condition (i) for v.

If v' does not appear in u then let a be the first index such that v' appears in u'_a. Then we have $v' \in V_b$ for all $b \geq a$, hence $\mathsf{In}_{v'}(u'_b) = \varepsilon$ for all $b \neq a$. Furthermore, there exists v such that $\mathsf{In}_{v'}(u'_a) \preceq \mathsf{In}_v(u_a)$, hence $\mathsf{In}_{v'}(u') \preceq \mathsf{In}_v(u)$. Hence we can use the boss child of

μ previously witnessing the satisfaction of condition (i) for v to witness the satisfaction of condition (i) for v'.

As a consequence, all conditions of an unfolding tree are still satisfied. By minimality of τ, we must have $|u| \leq |u'| \leq \psi(|\mathcal{P}|, r)[|\mathbf{bw}(\mu)| + |\mathcal{M}|rK + 1]$. We have shown the first point of the lemma.

Now let μ be a follower node, by condition (iii) we have $\mathsf{In}_{\mathbf{val}(\mu)}(\mathbf{lr}(\mu)) = \mathbf{fw}(\mu)$ and thus $|\mathbf{fw}(\mu)| \leq |\mathbf{lr}(\mu)|$.

The proof of the second point is analogous to the one for boss nodes: we identify the necessary broadcasts in the local run and reduce the local runs in-between them using Lemma 19. The only difference is that instead of broadcasting the letters of its boss label, μ only has to broadcast the message $(\mathbf{fm}(\mu), \mathbf{val}(\mu))$ in addition to the messages required by its follower children, hence the $|\mathbf{bw}(\mu)|$ term is replaced by 1 in the inequality. ◀

G Proof of Lemma 23

▶ **Lemma 35.** *Let $(\mathcal{P}, q_f)$ be a positive instance of* Cover*, τ a coverability witness for $(\mathcal{P}, q_f)$,* $\mathbf{altmax}$ *the maximal altitude in τ. There exists a primitive recursive function f_0 such that any node μ of τ has size bounded by $f_0(|\mathcal{P}| + \mathbf{altmax} - \mathbf{alt}(\mu))$.*

We actually prove the following, stronger statement:

▶ **Lemma 35.** *Let $\mathcal{P}$ be a protocol, let μ be a node of a unfolding tree of minimal size labeled by $\mathcal{P}$ satisfying a boss specification w. Let* $\mathbf{altmax}$ *be the maximal altitude in τ, let $N = |w| + |\mathcal{P}| + 1$ and let $f'_0 : n \in \mathbb{N} \mapsto ((n+2)^2\psi(n,n))^{n+1}$, with ψ the function defined in Lemma 19.*

- *If μ is a boss node then $|\mathbf{bw}(\mu)| \leq f'_0(|\mathcal{P}| + \mathbf{altmax} - \mathbf{alt}(\mu))$ and $|\mathbf{lr}(\mu)| \leq f'_0(|\mathcal{P}| + \mathbf{altmax} - \mathbf{alt}(\mu) + 1)$.*
- *If μ is a follower node then $|\mathbf{fw}(\mu)| \leq |\mathbf{lr}(\mu)| \leq f'_0(|\mathcal{P}| + \mathbf{altmax} - \mathbf{alt}(\mu) + 1)$.*

Proof. The proof is by strong induction on $\mathbf{altmax} - \mathbf{alt}(\mu)$.

First, let μ such that $\mathbf{alt}(\mu) = \mathbf{altmax}$. This implies that μ has no follower node (hence we can set $K = 0$ in Lemma 22). If μ is the root of the tree, then it is a boss node; we can impose that its boss specification is of length 1 (coverability of q_f) and applying Lemma 22 proves that $|\mathbf{lr}(\mu)| \leq \psi(|\mathcal{P}|, r) \leq f'_0(|\mathcal{P}|)$ (ψ defined in Lemma 19 is of course monotone for both arguments). If μ is not the root of the tree, then it is necessarily a follower node (a boss node with a parent would not have maximum altitude), and the application of Lemma 22 again yields that $|\mathbf{fw}(\mu)| \leq |\mathbf{lr}(\mu)| \leq \psi(|\mathcal{P}|, r) \leq f'_0(|\mathcal{P}|)$.

Let μ be a node, and suppose that the statement is true for all nodes of altitude greater than $\mathbf{alt}(\mu)$ and that $\mathbf{alt}(\mu) < \mathbf{altmax}$. Let $\mu_1, \ldots, \mu_k$ be its follower children, and K the maximum of all $|\mathbf{fw}(\mu_i)|$. By induction hypothesis for all i we have $|\mathbf{fw}(\mu_i)| \leq f'_0(|\mathcal{P}| + \mathbf{altmax} - \mathbf{alt}(\mu))$ because $\mathbf{alt}(\mu_i) = \mathbf{alt}(\mu) + 1$, and thus $K \leq f'_0(|\mathcal{P}| + \mathbf{altmax} - \mathbf{alt}(\mu))$. If μ is not the root of τ then let μ' be its parent.

Suppose first that μ is a boss node. If it is at the root then we can assume that $|\mathbf{bw}(\mu)| \leq 1$. If not, μ' is above μ therefore we have $|\mathbf{bw}(\mu)| \leq |\mathbf{lr}(\mu')| \leq f'_0(|\mathcal{P}| + \mathbf{altmax} - \mathbf{alt}(\mu))$ by Lemma 22 and induction hypothesis. In both cases we have $|\mathbf{bw}(\mu)| \leq f'_0(|\mathcal{P}| + \mathbf{altmax} - \mathbf{alt}(\mu))$. Again by Lemma 22, we obtain

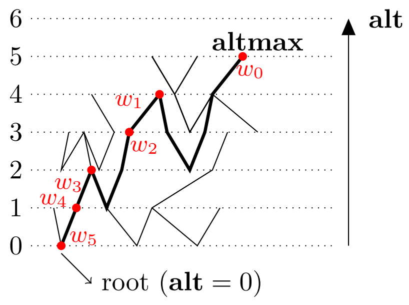
$$|\mathbf{lr}(\mu)| \leq \psi(|\mathcal{P}|, r)\Big[f'_0(|\mathcal{P}| + \mathbf{altmax} - \mathbf{alt}(\mu)) + |\mathcal{M}|rK + 1\Big].$$

As a result,

$$\begin{aligned}|\mathbf{lr}(\mu)| &\leq \psi(|\mathcal{P}|, r) f'_0(|\mathcal{P}| + \mathbf{altmax} - \mathbf{alt}(\mu))(|\mathcal{M}|r + 2)\\ &\leq (|\mathcal{P}| + 2)^2 \psi(|\mathcal{P}|, |\mathcal{P}|) f'_0(|\mathcal{P}| + \mathbf{altmax} - \mathbf{alt}(\mu))\\ &\leq f'_0(|\mathcal{P}| + \mathbf{altmax} - \mathbf{alt}(\mu) + 1)\end{aligned}$$

Suppose now that μ is a follower node. By Lemma 22, we have $|\mathbf{fw}(\mu)| \leq |\mathbf{lr}(\mu)| \leq \psi(|\mathcal{P}|, r)\left[1 + |\mathcal{M}|rK\right]$. By the same argument as above, this proves that $|\mathbf{lr}(\mu)| \leq f'_0(|\mathcal{P}| + \mathbf{altmax} - \mathbf{alt}(\mu) + 1)$. ◀

H Proof of Proposition 24



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

Lemma 23 bounds the length of w_i by $f_0(i) + 2$, which allows us to bound the number of red dots using Lemma 18 and the Length function theorem.

Figure 6 Illustration of the proof of Lemma 36

.

▶ **Lemma 36.** *There exists a function f_1 of the class $\mathscr{F}_{\omega^{|\mathcal{M}|}}$ such that for all protocol $\mathcal{P}$, for all unfolding tree τ of minimal size labeled by $\mathcal{P}$ satisfying a boss specification w, the maximal altitude of a node of τ is bounded by $f_1(|\mathcal{P}| + |w|)$.*

Proof. Let τ be a unfolding tree of minimal size labeled by $\mathcal{P}$ satisfying a boss specification w. Let **altmax** be the maximal altitude of a node in τ. Let μ_0 the root of the tree, and $\mu_0, \ldots, \mu_m$ a branch of the tree going from the root to a node μ_m of altitude **altmax** (for all i, μ_i is a child of μ_{i-1}). For all $a \in [1, \mathbf{altmax}]$, let j_a be the minimal index such that $\mathbf{alt}(\mu_{j_a}) = a$. This index exists as $\mathbf{alt}(\mu_0) = 0$, $\mathbf{alt}(\mu_m) = \mathbf{altmax}$ and for all i, $\mathbf{alt}(\mu_i) \leq \mathbf{alt}(\mu_{i-1}) + 1$. Furthermore, for every a, for every $j < j_a$, $\mathbf{alt}(\mu_j) < a$. As a consequence, we have $j_1 < j_2 < \ldots < j_{\mathbf{altmax}}$. Moreover, μ_{j_a} is necessarily a follower node as otherwise we would have $\mathbf{alt}(\mu_{j_a - 1}) = a + 1$ which contradicts minimality of j_a.

We will bound **altmax** using Theorem 10. 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, and let $N = |w| + |\mathcal{P}| + 1$. For all $i \in [1, \mathbf{altmax}]$ let $\mu'_i = \mu_{j_{\mathbf{altmax}-i+1}}$ and $w_i = \mathbf{fw}(\mu'_i)\#\mathbf{fm}(\mu'_i) \in (\mathcal{M} \cup \{\#\})^*$.

We claim that we cannot have $w_{i_1} \preceq w_{i_2}$ for any $i_1 < i_2$. Suppose by contradiction that there exist such i_1 and i_2 with $i_1 < i_2$. Then we would have $\mathbf{fw}(\mu'_{i_1}) \preceq \mathbf{fm}(\mu'_{i_2})$ and $\mathbf{fm}(\mu'_{i_1}) = \mathbf{fw}(\mu'_{i_2})$. Therefore, as μ_{i_1} is a descendant of μ_{i_2}, by second case of Lemma 18 there would exist a unfolding tree of smaller size than τ satisfying w, contradicting the minimality of τ.

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 23, for all i, as $\mathbf{alt}(\mu'_i) = \mathbf{altmax} - i + 1$, we have $|\mathbf{fw}(\mu_{j_a})| \leq f_0(|\mathcal{P}| + i)$ therefore $|w_i| \leq f_0^{(i)}(|\mathcal{P}|)$ (with $f_0^{(i)}$ denoting f_0 applied i times).

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 $f'_1(N)$. This prove that $\mathbf{altmax} \leq f'_1(N) = f'_1(|\mathcal{P}|+|w|+1)$. Letting $f_1 : n \mapsto f'_1(n+1)$ concludes the proof. ◀

The bound on the maximal altitude, along with Lemma 23, gives us a bound on the size of the root. By using a similar argument as for the maximal altitude, we consider a branch reaching minimal altitude to obtain a bad sequence of boss nodes and apply the Length function theorem again to bound the minimal altitude.

▶ **Lemma 37.** *There exists a function f_2 of the class $\mathscr{F}_{\omega^{|\mathcal{M}|+1}}$ such that for all protocol $\mathcal{P}$, for all unfolding tree τ of minimal size labeled by $\mathcal{P}$ satisfying a boss specification w, the absolute value of the minimal altitude of a node of τ is bounded by $f_2(|\mathcal{P}| + |w|)$.*

Proof. We proceed similarly as in the proof of Lemma 36.

Let τ be a unfolding tree of minimal size labeled by $\mathcal{P}$ satisfying a boss specification w. Let $\mathbf{altmin}$ be the minimal altitude of a node in τ. Let $\mu_0, \ldots, \mu_m$ be nodes of τ such that μ_0 is its root, $\mathbf{alt}(\mu_m) = \mathbf{altmin}$, and for all $i \in [1, m]$, μ_i is a child of μ_{i-1}. For all $a \in [\mathbf{altmin}, 0]$, let j_a be the minimal index such that $\mathbf{alt}(\mu_{j_a}) = a$. This index exists as $\mathbf{alt}(\mu_0) = 0$, $\mathbf{alt}(\mu_m) = \mathbf{altmin}$ and for all i, $\mathbf{alt}(\mu_i) \geq \mathbf{alt}(\mu_{i-1}) - 1$. Furthermore as j_a is the minimal such index, we cannot have $\mathbf{alt}(\mu_j) \leq a$ for any $j < j_a$ as otherwise there would be a j' such that $j' \leq j < j_a$ with $\mathbf{alt}(\mu_{j'}) = a$.

As a consequence, we have $j_0 < j_1 < j_2 < \ldots < j_{|\mathbf{altmin}|}$. Moreover, μ_{j_a} is necessarily a boss node for all $a < 0$ as otherwise we would have $\mathbf{alt}(\mu_{j_a - 1}) = a - 1 \leq a$, and μ_{j_0} is the root and therefore also a boss node as τ satisfies w.

Let $N := |w| + |\mathcal{P}| + 1$. For all $i \in [0, |\mathbf{altmin}|]$ let $\mu'_i = \mu_{j_{-i}}$ and $w_i = \mathbf{bw}(\mu'_i) \in \mathcal{M}^*$.

We claim that we cannot have $w_{i_1} \preceq w_{i_2}$ for any $i_1 < i_2$. Suppose by contradiction that there exist such i_1 and i_2 with $i_1 < i_2$. Then we would have $\mathbf{bw}(\mu'_{i_1}) \preceq \mathbf{bw}(\mu'_{i_2})$, hence, as μ_{i_2} is a descendant of μ_{i_1}, by first case of Lemma 18 there would exist a unfolding tree of smaller size than τ satisfying w, contradicting the minimality of τ.

As a result, the sequence $(w_i)_{1 \leq i \leq \mathbf{altmax}}$ is a bad sequence for the subword order over the alphabet $\mathcal{M} \cup \{\#\}$. By Lemma 23, for all μ'_i we have $|\mathbf{bw}(\mu'_i)| \leq f_0(|\mathcal{P}| + \mathbf{altmax} - \mathbf{alt}(\mu'_i) + 1) = f_0(|\mathcal{P}| + \mathbf{altmax} + i + 1)$. By Lemma 36, we have $\mathbf{altmax} \leq f_1(N)$ and thus $|\mathbf{bw}(\mu'_i)| \leq f_0(N + f_1(N) + i)$. Let $g : k \geq 1 \mapsto f_0(k)$; clearly, for all i, $|w_i| \leq g^{(i)}(M)$ where $M := f_0(N + f_1(N) + 1)$ (obviously, $f_0(k) > k + 1$) and g is primitive-recursive.

We apply Theorem 10 to obtain $h \in \mathscr{F}_{\omega^{|\mathcal{M}|}}$ such that $(w_i)1 \leq i \leq \mathbf{altmax}$ cannot have more than $h(M)$ elements, thus $|\mathbf{altmin}| \leq h(M) + 1 = f_4(f_0(N + f_1(N) + 1))$. By defining $f_2 : n \mapsto f_4(f_0(n + f_1(n+1) + 2))$, this means that $|\mathbf{altmin}| \leq f_2(|\mathcal{P}| + |w|)$. Moreover, f_2 is a function of $\mathscr{F}_{\omega^{|\mathcal{M}|+1}}$ as f_0, f_1 and h are in $\mathscr{F}_{\omega^{|\mathcal{M}|}} \subseteq \mathscr{F}_{\omega^{|\mathcal{M}|+1}}$ and $\mathscr{F}_{\omega^{|\mathcal{M}|+1}}$ is closed by composition with functions of $\mathscr{F}_{\omega^{|\mathcal{M}|}}$. ◀

With the bounds on the maximal and minimal altitudes, Lemma 23 yields a bound on the size of all nodes in the tree.

▶ **Lemma 38.** *There exists a function f_3 of the class $\mathscr{F}_{\omega^{|\mathcal{M}|+1}}$ such that for all protocol $\mathcal{P}$, for all unfolding tree τ of minimal size labeled by $\mathcal{P}$ satisfying a boss specification w, for all node μ of τ,*

- $|\mathbf{lr}(\mu)| \leq f_3(|\mathcal{P}| + |w|)$

- *If μ is a boss node then $|\mathbf{bw}(\mu)| \leq f_3(|\mathcal{P}| + |w|)$.*
- *If μ is a follower node then $|\mathbf{fw}(\mu)| \leq f_3(|\mathcal{P}| + |w|)$.*

Proof. Using Lemmas 23, 36 and 37, we obtain that, for every node μ, the value $\mathbf{altmax} - \mathbf{alt}(\mu)$ is at most $f_1(|\mathcal{P}| + |w|) + f_2(|\mathcal{P}| + |w|)$. We then apply Lemma 23 to bounds on $\mathbf{lr}(\mu)$ and $\mathbf{spec}(\mu)$ by $f_0(f_1(|\mathcal{P}| + |w|) + f_2(|\mathcal{P}| + |w|) + 1) \leq f_3(|\mathcal{P}| + |w|)$ where $f_3 : n \mapsto f_0(f_1(n) + f_2(n) + 1)$; f_3 is in $\mathscr{F}_{\omega^{|\mathcal{M}|+1}}$ as $\mathscr{F}_{\omega^{|\mathcal{M}|+1}}$ is closed by composition with elementary functions. ◀

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 24.** *There exists a function f of class $\mathscr{F}_{\omega^{|\mathcal{M}|+1}}$ such that an instance $(\mathcal{P}, q_f)$ of* Cover *is positive if and only if it has a coverability witness τ of size bounded by $f(|\mathcal{P}|)$.*

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

Thanks to Proposition 17, $(\mathcal{P}, q_f)$ is positive if and only if there exists a coverability witness for $(\mathcal{P}, q_f)$. Let τ be a coverability witness of $(\mathcal{P}, q_f)$, *i.e.*, its root is a boss node whose local run covers q_f. Let $N = |\mathcal{P}| + 1$. Observe that one can encode the condition that this local run covers q_f in the boss specification of the protocol (adding a special broadcast message only broadcast from q_f); in the following we will forget about the condition that the local run at the root covers q_f.

We first bound the height of τ (*i.e.*, the difference between its maximum and minimum altitude), then its arity (the maximal number of children of a node), which allows us to obtain a bound on the number of nodes. The sizes of the nodes will finally be bounded using the height.

By Lemma 38, for all boss node μ in τ we have $|\mathbf{bw}(\mu)| \leq f_3(N)$. If a branch contains strictly more than $\sum_{i=0}^{f_3(N)} |\mathcal{M}|^i$ follower nodes, then there are two nodes in it with the same boss label, hence τ can be reduced and still satisfy w, contradicting the minimality of τ.

Similarly, by Lemma 38, for all follower node μ in τ we have $|\mathbf{fw}(\mu)| \leq f_3(N)$. If a branch contains strictly more than $|\mathcal{M}| \sum_{i=0}^{f_3(N)} |\mathcal{M}|^i$ follower nodes, then there are two nodes in it with the same follower label, hence τ can be reduced and still satisfy w, contradicting the minimality of τ.

As a result, each branch of τ contains at most $(|\mathcal{M}| + 1) \sum_{i=0}^{f_3(N)} |\mathcal{M}|^i$ nodes. We simplify this expression: let $H : n \mapsto n(f_3(n) + 1)n^{f_3(n)+1}$, each branch of τ has less than $H(N)$ nodes.

Furthermore, by Lemma 22, each node has at most $|\mathcal{M}|r$ follower children. Moreover, if a node μ has more than $f_3(N)$ boss children, then one of them can be removed as at most $f_3(N)$ non-initial values appear in $\mathbf{lr}(\mu)$; this would contradict the minimality of τ. As a consequence, each node in τ has at most $f_3(N) + |\mathcal{M}|(r + 1)$ children.

By combining the bounds on the length of branches and the arity of nodes, we obtain that τ has at most $\sum_{h=0}^{H}(f_3(N) + |\mathcal{M}|r)^h$ nodes. Let $G : n \mapsto (H(n) + 1)(f_3(n) + n^2)^{H(n)}$. The total number of nodes in τ is at most $G(N)$.

Finally, we obtain a bound on the size of τ by taking the product of the bounds on the number of nodes and on the size of each node label:

$$|\tau| \leq G(N)(2f_3(N) + 1)$$

Let $f : n \mapsto G(n + 1)(2f_3(n + 1) + 1)$; using the fact that $n \leq f_3(n)$, we can bound f_4 by the composition of an elementary fonction with f_3, hence as $f_3 \in \mathscr{F}_{\omega^{\mathcal{M}}}$ and $\mathscr{F}_{\omega^{\mathcal{M}}}$ is closed by composition with elementary functions, we have that $f_4 \in \mathscr{F}_{\omega^{\mathcal{M}}}$. Overall, we have shown that the size of a minimal coverability witness for $(\mathcal{P}, q_f)$ is bounded by $f(|\mathcal{P}|)$ with $f \in \mathscr{F}_{\omega^{|\mathcal{M}|}}$. ◀

I Proof of Theorem 13

▶ **Theorem 13.** *Cover for BNRA is decidable and* $\mathbf{F}_{\omega^\omega}$*-complete.*

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

For the upper bound, let $\mathcal{P}$ be a protocol with $r \geq 1$ registers over messages $\mathcal{M}$ and q one of its states. We add a new message m to $\mathcal{M}$ and a new transition $\mathbf{br}(m, 1)$ from q to itself broadcasting m. Clearly the new protocol $\mathcal{P}'$ satisfies the boss specification m if and only if q is coverable in $\mathcal{P}$.

By Lemmas 27 and 28, there is an initial run of $\mathcal{P}'$ satisfying m if and only if there is a unfolding tree satisfying m.

By Lemma 24, there is such a unfolding tree if and only if there is one of size at most $f(|\mathcal{P}|)$, where f is a function of the class $\mathscr{F}_{\omega^{|\mathcal{M}|}}$.

The last problem we have is that the size of an unfolding tree does not take into account the values used in it. This can be solved by noticing that the conditions in the definition of a unfolding tree and the condition to satisfy a specification still hold after applying an injective renaming to the values.

Suppose there exists τ a unfolding tree satisfying m, let V be the set of values appearing in that tree. In a node μ the values appearing are $\mathbf{val}(\mu)$, the initial values of $\mathbf{lr}(\mu)$ (there are r such values), the non-initial values (at most $|\mathbf{lr}(\mu)|$ as they all need to be received). Hence there are at most $|\mathbf{lr}(\mu)| + r + 1$ different values appearing in each node. In total there are thus at most $\sum_{\mu \in \tau} |\mathbf{lr}(\mu)| + r + 1$ different values in τ, which is less than $(r+2)|\tau|$.

We can apply an injective renaming $V \to [1, (r+2)|\tau|]$ to the values to obtain another unfolding tree satisfying m and whose values do not exceed $(r+2)|\tau|$. As a result, there exists a unfolding tree satisfying m if and only if there is one of size at most $f(|\mathcal{P}|)$ in which values do not exceed $f(|\mathcal{P}|)$. A description of such a tree requires a space that is polynomial in its size.

We can therefore bound the size of the description of such a unfolding tree by a function of $\mathscr{F}_{\omega^{|\mathcal{M}|}}$. An algorithm for Cover thus consists in enumerating all trees labeled with local runs of $\mathcal{P}$ of size at most $f(|\mathcal{P}|)$ with values not exceeding $f(|\mathcal{P}|)$, and accepting if and only if one of them satisfies the conditions to be a unfolding tree satisfying m. This can all be done in exponential time in $f(|\mathcal{P}|)$, thus this algorithm terminates in time $f'(|\mathcal{P}|)$ where f' is a function of $\mathscr{F}_{\omega^{|\mathcal{M}|}}$.

Cover for BNRA therefore lies in the complexity class $\mathbf{F}_{\omega^\omega}$. ◀

J Proof of Proposition 25

▶ **Proposition 25.** *Target is undecidable for* BNRA *with two registers.*

Proof. We present a reduction from the halting problem for Minsky machines to the target reachability problem for BNRA.

A Minsky Machine 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, $\Delta \subseteq \mathrm{Loc} \times (\{\mathtt{x}-\}_{\mathtt{x}\in\mathtt{X}} \cup \{\mathtt{x}+\}_{\mathtt{x}\in\mathtt{X}} \cup \{\mathtt{x} = 0\}_{\mathtt{x}\in\mathtt{X}}) \times \mathrm{Loc}$ 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. $\mathtt{x}_2$). For two configurations (ℓ, v_1, v_2) and (ℓ', v'_1, v'_2), we write $(\ell, v_1, v_2) \to (\ell', v'_1, v'_2)$ if one of the following condition holds for some $\delta \in \Delta$:

- $\delta = (\ell, \mathtt{x_i}+, \ell')$ 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 $(\ell_1, v_1^1, v_2^1) \to (\ell_2, v_1^2, v_2^2) \to \ldots \to (\ell_k, v_1^k, v_2^k)$. We say that an execution is initial if it starts with the initial configuration $(l_0, 0, 0)$. The halting problem asks whether there exists an initial execution ending with a configuration (ℓ_f, v_1, v_2) where v_1, v_2 are any values in $\mathbb{N}$. This problem is well-known to be undecidable.

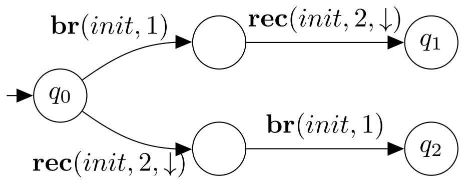


Figure 7 Initialisation for the Target encoding

Fix a Minsky Machine $M = (\mathrm{Loc}, \Delta, \mathtt{X}, \ell_0, \ell_f)$. The simulation goes as follows:

- each agent will have two registers: one with its identity, which it will broadcast, and another in which it will store the value of a *predecessor* process. To do so, in an initialization phase, each process does one of the following things: (i) it starts by broadcasting its first register value and then waits to receive a value and stores it in its second register, (ii) it waits first for a value to store in its second register and then broadcasts its first register value. This initialization part will appear in the protocol as displayed in Figure 7. Note that, after the initialization phase (once all processes reached q_1 and q_2, which eventually happens as otherwise they would not all reach q_f), there is necessarily a *cycle of knowledge* in the reached configuration. A cycle of knowledge in a configuration γ is a subset of agents $\{a_1, \ldots a_k\}$ such that for all $1 \leq i < j \leq k$, $\mathsf{data}(\gamma)(a_i)(1) \neq \mathsf{data}(\gamma)(a_j)(1)$, for all $2 \leq i \leq k$, $\mathsf{data}(\gamma)(a_i)(2) = \mathsf{data}(\gamma)(a_{i-1})(1)$ and $\mathsf{data}(\gamma)(a_1)(2) = \mathsf{data}(\gamma)(a_k)(1)$ (see Figure 8). Indeed, consider the function which associates to each agent a, an agent a' such that $\mathsf{data}(\gamma)(a)(2) = \mathsf{data}(\gamma)(a')(1)$. This function is total as γ is such that all agents reached states q_1 or q_2. By the pigeonhole principle, there exists a cycle of knowledge.
  From then on, an agent will broadcast only its first register value (along with some messages) and shall only receive messages sent with its second register value. In other words, every reception transition after the initialisation phase is of the form $\mathbf{rec}(m, 2, =)$ for some $m \in \mathcal{M}$ and every broadcast transition after the initialisation phase is of the form $\mathbf{br}(m, 1)$ for some $m \in \mathcal{M}$;
- The agents which made the first choice (i) during the initialization phase will simulate a machine execution. We shall name this type of agents *leader* agents. The other agents will simulate counter values, each agent chooses one of the two counters $\mathtt{x}_i$ to simulate, and will from then on travel between two states $(\mathtt{x}_i, 0)$ and $(\mathtt{x}_i, 1)$.
- After initialization, leader agents are in location ℓ_0 and other agents are all on one of the states $(\mathtt{x}_i, 0)$.

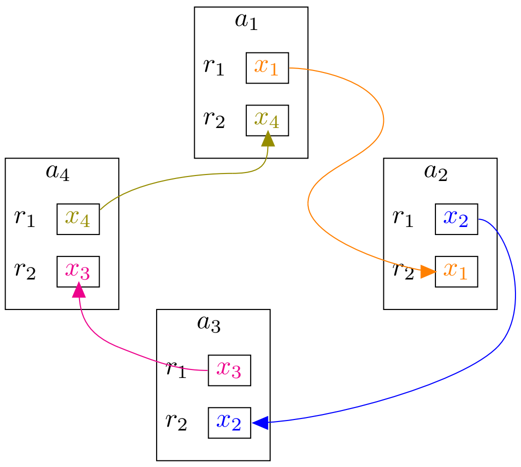


Figure 8 Illustration of a cycle of knowledge of 4 agents. Each box represents an agent along with its register values.

A leader simulates the Minsky machine by broadcasting, with its identifier, a sequence of transitions $\delta_0, \dots, \delta_n$ that match a path in the machine $\ell_0 \xrightarrow{\delta_0} \ell_1 \cdots \xrightarrow{\delta_n} \ell_n$.
After each broadcast of a δ_i, it waits for a message from its predecessor, either δ_i itself if it is a test or $\overline{\delta_i}$ if it is an increment or decrement ($\overline{\delta_i}$ is an acknowledgement that δ_i was executed). Once it reaches a final location, it broadcasts a message **end**, waits to receive the same message from its predecessor, and then enters a special state q_f and stops.
When a non-leader agent receives a message δ from its predecessor, if it is an operation it can apply then it does so and broadcasts an acknowledgement $\overline{\delta}$ (for instance if it is an increment $\mathtt{inc_1}$ and it is in state $(\mathtt{x}_1, 0)$ it goes to $(\mathtt{x}_1, 1)$). If it is a test $\mathtt{x_i} = 0$ and it is in state $(\mathtt{x}_i, 1)$ then it stops and broadcasts nothing. If it is **end** then it forwards it, goes to state q_f and stops. In all other cases it simply forwards the message with its identifier.

If the Minsky machine has an accepting run, then we can reach a configuration with all agents in q_f: let M be the maximal values of a counter in the accepting run, it suffices to make agents form a single cycle of knowledge with $2M + 1$ agents with one leader and M non-leaders for each counter. One step of the run is simulated by making the leader choose and broadcast the corresponding transition, then making all members of the cycle receive the message from their predecessor and send one to their successor according to the rules described above. It is easy to show that after the leader receives its nth acknowledgement, the number of agents in state $(\mathtt{x}_i, 1)$ is exactly the value of the counter $\mathtt{x}_i$ after the nth step of the Minsky machine execution. Therefore, as all 0-tests are successful in the run, when the leader broadcasts one over counter $\mathtt{x}_i$, all agents of that counter are in state $(\mathtt{x}_i, 0)$, hence the message is passed around the cycle and comes back to the leader. We made the cycle large enough so that before any increment of $\mathtt{x}_i$, there is an agent in state $(\mathtt{x}_i, 0)$ which can apply it and send the acknowledgement.

Eventually, the leader reaches ℓ_f, sends **end**, which goes around the cycle and comes back to the leader, leaving all agents in state q_f.

For the other direction, suppose there is a run of the BNRA with at least one agent, and such that at the end all agents are in q_f. As each agent must have executed the initialization phase to reach q_f, they all have a predecessor. As there are finitely many agents, there is necessarily a cycle of knowledge that is formed. We focus on one cycle of knowledge (any of them is suitable to simulate the machine). Let $a_1, \cdots, a_k$ be the agents of that cycle, a_i

being the predecessor of a_{i+1} for all $i \in [1, k-1]$ and a_k being the predecessor of a_1. Note that there is at least one leader in the cycle of knowledge, otherwise, no agent sent its value first and the cycle cannot be created. Hence we can assume a_1 to be a leader.

An important observation is that in order to reach q_f, each agent must send and receive the same number of messages. Furthermore, each agent only receives messages from their predecessor. This means that if we write, for all i, s_i and r_i for the number of messages sent and received by a_i during the run, we have $r_1 = s_1 \geq r_2 = s_2 \geq \cdots \geq r_k = s_k \geq r_1$. Hence all s_i and r_i are equal, and in particular every agent of the cycle receives all broadcasts from its predecessor.

In order to reach q_f, each leader must receive a sequence of messages matching the acknowledgements of the messages it sent. As all agents of the cycle send and receive the same number of messages, all leaders must follow the same sequence of transitions and receive all the acknowledgements. In particular all 0-tests must succeed, as otherwise the next leader in the cycle would receive less messages than the previous one has sent.

We consider the sequence of transitions $\delta_1, \dots, \delta_k$ of the Minsky machine described by the sequence of messages sent by a_1. Let a_j be the next leader on the cycle (possibly a_1 itself if it is the only one). A straightforward induction shows that after a_1 receives its nth acknowledgement, the number of agents in state $(\mathtt{x}_i, 1)$ in $\{a_2, \dots, a_{j-1}\}$ is exactly the counter value of the Minsky machine after applying $\delta_1, \dots, \delta_n$ (which can be applied as we argued that all 0-tests were successful). As a_1 reaches q_f, the sequence of transitions $\delta_1, \dots, \delta_k$ must end in ℓ_f, and is a valid execution of the Minsky machine. This concludes our reduction. ◀

K Proofs of Section 4

We simplify notations as in this section we do not need to consider local tests: with a single register every local equality test is satisfied and every local disequality test is not. Hence we can delete transitions with disequality tests and replace equality tests with internal transitions that have no effect (which may in turn be encoded in our model using broadcasts of dummy messages).

Furthermore, the register argument in receptions and broadcasts is always 1, hence we remove it. Our new set of operations is $Op^{\mathcal{M}} = \{\mathbf{br}(m), \mathbf{rec}(m, *), \mathbf{rec}(m, \downarrow), \mathbf{rec}(m, =), \mathbf{rec}(m, \neq) \mid m \in \mathcal{M}\}$. Finally, given a configuration γ, we write $\mathsf{data}(\gamma)(a)$ for $\mathsf{data}(\gamma)(a, 1)$.

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

First, to simplify the proofs, we use a lemma allowing us to only consider protocols which do not contain any reception transition 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 process 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 '$*$'.

K.1 Removing Disequality Tests

We start with a basic observation that adding agents to a configuration cannot decrease the set of reachable configurations.

▶ **Definition 39.** *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 40.** *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 $\tilde{\gamma}_{k-1} \rightarrow \tilde{\gamma}_k$. 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 $\pi : \tilde{\mathbb{A}} \rightarrow \mathbb{A}$ 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 $\tilde{\gamma}_{k-1} \rightarrow \tilde{\gamma}_k$ 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 $\tilde{\gamma}_{k-1} \rightarrow \tilde{\gamma}_k$). Otherwise, write $\tilde{\mathbb{A}}_{\neq}$ the subset of $\tilde{\mathbb{A}}$ corresponding to agents taking in $\tilde{\gamma}_{k-1} \rightarrow \tilde{\gamma}_k$ 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 $\tilde{\gamma}_{k-1} \rightarrow \tilde{\gamma}_k$. 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 $\tilde{\gamma}_{k-1} \rightarrow \tilde{\gamma}_k$. 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 $\tilde{\gamma}_{k-1} \rightarrow \tilde{\gamma}_k$, hence the configuration γ_k reached by the constructed run is such that $\tilde{\gamma}_k \trianglelefteq \gamma_k$. ◀

K.2 Abstraction

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

▶ **Definition 41.** *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 $\rho = \gamma_0 \rightarrow \gamma_1 \rightarrow \cdots \rightarrow \gamma_k$ be a run and $v \in \mathsf{val}(\rho)$. The gang of value v in ρ, written $\mathsf{gang}_v(\rho)$, is the gang (b, K) 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 $\mathsf{b} := \mathsf{st}(\gamma_k)(a_0)$, otherwise $\mathsf{b} := \bot$,*
- *$\mathsf{K} := \{q \in Q \mid \exists i \leq k, \exists a \in \mathbb{A} \setminus \{a_0\}, \gamma_i(a) = (q, v)\}$*

We define abstract runs as follows:

▶ **Definition 42.** *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 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*:

(1) There exist $m \in \mathcal{M}$ and $q_{\textbf{br}} \in \mathsf{K}, q'_{\textbf{br}} \in \mathsf{K}'$ s.t. $(q_{\textbf{br}}, \textbf{br}(m), q'_{\textbf{br}}) \in \Delta$.

(2) Either $\mathsf{b} = \mathsf{b}'$ or there exists $(\mathsf{b}, \textbf{rec}(m, \alpha), \mathsf{b}') \in \Delta$ for some action α.

(3) $(\mathsf{K} \cup \{q'_{\textbf{br}}\}) \subseteq \mathsf{K}'$ and, for all $q' \in \mathsf{K}' \setminus (\mathsf{K} \cup \{q'_{\textbf{br}}\})$, there exists q s.t. $(q, \textbf{rec}(m, \alpha), q') \in \Delta$ where:

- *$\alpha =$ '$=$' or '$*$' and $q \in \mathsf{K}$, or*
- *$\alpha =$ '$\downarrow$' and $q \in S$.*

(4) $(S \cup \{q'_{\textbf{br}}\}) \subseteq S'$ and, for all $q' \in S' \setminus (S \cup \{q'_{\textbf{br}}\})$, there exists q s.t. $(q, \textbf{rec}(m, \alpha), q') \in \Delta$ where:

- *$\alpha =$ '$=$' and $q \in \mathsf{K}$, or*
- *$\alpha =$ '$\downarrow$' or '$*$' and $q \in S$.*

2. *Broadcast from boss*:

(1) there exists $m \in \mathcal{M}$ such that $(\mathsf{b}, \textbf{br}(m), \mathsf{b}') \in \Delta$

(2) $\mathsf{b}, \mathsf{b}' \neq \bot$ (technically implied by (1) but written here to match other cases)

(3) $\mathsf{K} \subseteq \mathsf{K}'$ and, for all $q' \in \mathsf{K}' \setminus \mathsf{K}$, there exists q s.t. $(q, \textbf{rec}(m, \alpha), q') \in \Delta$ where:

- *$\alpha =$ '$=$' or '$*$' and $q \in \mathsf{K}$, or*
- *$\alpha =$ '$\downarrow$' and $q \in S$.*

(4) $S \cup \{\mathsf{b}'\} \subseteq S'$ and, for all $q' \in S' \setminus (S \cup \{\mathsf{b}'\})$, there exists q s.t. $(q, \textbf{rec}(m, \alpha), q') \in \Delta$ where:

- *$\alpha =$ '$=$' and $q \in \mathsf{K}$, or*
- *$\alpha =$ '$\downarrow$' or '$*$' and $q \in S$.*

3. *External broadcast*:

(1) There exists $m \in \mathcal{M}$ and $q_{\textbf{br}} \in S, q'_{\textbf{br}} \in S'$ s.t. $(q_{\textbf{br}}, \textbf{br}(m), q'_{\textbf{br}}) \in \Delta$.

(2) Either $\mathsf{b} = \mathsf{b}'$ or:

- *$\mathsf{b}' \neq \bot$ and there exists $(\mathsf{b}, \textbf{rec}(m, *), \mathsf{b}') \in \Delta$, or*
- *$\mathsf{b}' = \bot$ and there exists $(\mathsf{b}, \textbf{rec}(m, \downarrow), \mathsf{b}') \in \Delta$.*

(3) $\mathsf{K} \subseteq \mathsf{K}'$ and, for all $q' \in \mathsf{K}' \setminus \mathsf{K}$, there exists $q \in \mathsf{K}$ s.t. $(q, \textbf{rec}(m, *), q') \in \Delta$.

(4) $(S \cup \{q'_{\textbf{br}}\}) \subseteq S'$ and, for all $q' \in S' \setminus (S \cup \{q'_{\textbf{br}}\})$, there exists $q \in S$ s.t. $(q, \textbf{rec}(m, \alpha), q') \in \Delta$ where $\alpha =$ '$\downarrow$' or $\alpha =$ '$*$'.

4. *Gang reset: $S' = S$, $\mathsf{K}' = \emptyset$ 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 $\nu = \sigma_0, \dots, \sigma_k$ 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 gang reset will help us to do the following, if one wants to check if one state is reachable, one should check the presence of an abstract run leading to an abstract configuration in which the state appears. Once it is done, the reachable state is added to S, and we can now check something else, for example, if we want to check the reachability of another state. This way, it can do a gang reset step in order to restart with some new boss but it keeps in mind the states it knows to be reachable, so it can use it later on (for instance, if it needs to receive a message from a state in S).

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

▶ **Lemma 43.** *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. We start with some intuition.

To prove the completeness, 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 $(\mathsf{cov}(\rho), \mathsf{gang}_v(\rho))$. Note that $\mathsf{cov}(\rho)$ is the set of all states appearing in ρ and $\mathsf{gang}_v(\rho) = (\mathsf{b}, \mathsf{K})$ is the gang of agents with value v at the end of ρ, i.e., if the agent starting with value v has still the same value, b denotes its state, and otherwise $\mathsf{b} = \bot$, and K is the set of states on which there is an agent with value v at the end of ρ.

To construct the abstract run, we will first show that, for all concrete run ρ and for all value v, $(\mathsf{cov}(\rho), q_0, \emptyset) \xrightarrow{*} (\mathsf{cov}(\rho), \mathsf{gang}_v(\rho))$. To do this, it suffices to follow steps by steps how agents with value v evolve and find the fitting abstract step such that we follow exactly the evolution of the gang associated to value v. Then, it is left to show that for all concrete run ρ and for all value v, $\sigma_0 \xrightarrow{*} (\mathsf{cov}(\rho), q_0, \emptyset)$. We shall prove this by induction on the size of $\mathsf{cov}(\rho)$. The base case is handled by the first property we proved. For the induction case, we consider a prefix run ρ' of ρ that stops just before all $S(\rho)$ is covered, and we take $v_{\mathbf{br}}$ the value of the broadcast in following step, that covers the rest of $S(\rho)$. By induction hypothesis and our first property, we prove that $\sigma_0 \xrightarrow{*} (\mathsf{cov}(\rho'), q_0, \emptyset) \xrightarrow{*} (\mathsf{cov}(\rho'), \mathsf{gang}_{v_{\mathbf{br}}}(\rho'))$. By going one step forward and finding the fitting case in the abstract semantics, we prove that $(\mathsf{cov}(\rho'), \mathsf{gang}_{v_{\mathbf{br}}}(\rho')) \to (\mathsf{cov}(\rho), \mathsf{G})$ for some G. Finally, $\sigma_0 \xrightarrow{*} (\mathsf{cov}(\rho), q_0, \emptyset)$ by the gang reset step of the abstract semantics. Putting everything together and applying the first property, we finally get: $\sigma_0 \xrightarrow{*} \mathsf{abs}_v(\rho)$.

It is left to prove that our abstraction is sound, in order to do this, consider an abstract run $\sigma_0 \xrightarrow{*} \sigma = (S, \mathsf{b}, \mathsf{K})$. We shall prove that there exists a concrete run $\rho : \gamma_0 \xrightarrow{*} \gamma$ such that γ covers all states in S, there exists a special agent on state b with value v and for each state q in K, there is (at least) one agent on q with register value v. In fact, we prove something stronger: in each of states $s \in S$, we shall prove that we can put an exponential number (in the size of the protocol) of agents in each state $s \in S$ and in each state $k \in \mathsf{K}$ with value v. This stronger property will allow us to do a proof by induction on the length of the run.

As we proved our abstraction to be sound and complete, by Lemma 43, we can guess an abstract run of at most $(|Q|+2)^3$ steps leading to an abstract configuration in which q_f appears. As a result, we obtain an NP-algorithm for COVER in 1BNRA.

K.2.1 Completeness

This subsection is devoted to proving Lemma 46. The following remark formalizes the intuition that, because the only way to take a new register value is to receive another agent's value, no new register values are created in a run.

▶ Remark 44. Because a register may only change its value to one that is broadcast from another register, we have that for all $\rho : \gamma_0 \xrightarrow{*} \gamma_f$, $\mathsf{val}(\rho) = \{v \in \mathbb{N} \mid \exists a, \mathsf{data}(\gamma_0)(a) = v\}$.

▶ **Lemma 45.** *For all initial runs $\rho : \gamma_0 \xrightarrow{*} \gamma$ and $v \in \mathsf{val}(\rho)$, $(\mathsf{cov}(\rho), q_0, \emptyset) \xrightarrow{*} \mathsf{abs}_v(\rho)$.*

Proof. Let $S := \mathsf{cov}(\rho)$ and $\mathbb{A} = \mathsf{ag}(\rho)$. Thanks to Remark 44, v appears 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, and write $\sigma^i := (S, \mathsf{gang}_v(\rho_i))$. Note that $\mathsf{gang}_v(\rho_0) = (q_0, \emptyset)$ hence $\sigma^0 = (S, q_0, \emptyset)$. Also, we write $(S, \mathsf{b}_i, \mathsf{K}_i) := \sigma^i$.

We prove by induction on i that $\sigma^0 \xrightarrow{*} \sigma^i$. The statement is trivially true for $i = 0$.

Suppose now that $(S, \emptyset, \bot) \xrightarrow{*} \sigma^i$. If suffices to prove that $\sigma^i \to \sigma^{i+1}$. First of all we clearly have, by definition, $K_{i+1} \subseteq S$ and $\mathsf{b}_{i+1} \in S \cup \{\bot\}$. 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, $a_{\mathsf{boss}} \neq a_{\mathbf{br}}$.

We now prove the other conditions:

(1) In case 1, since $a_{\mathbf{br}}$ has value v in γ_i and γ_{i+1} but not in every γ_j for $j \leq i$, we directly have $q_{\mathbf{br}} \in \mathsf{K}_i$ and $q'_{\mathbf{br}} \in \mathsf{K}_{i+1}$. In case 2, it suffices to note that $\mathsf{b}_i = \mathsf{st}(\gamma)(a_{\mathbf{br}})$ and $\mathsf{b}_{i+1} = \mathsf{st}(\gamma_{i+1})(a_{\mathbf{br}})$. In case 3, it suffices to note that $q_{\mathbf{br}}, q'_{\mathbf{br}} \in S$ as both states appear in ρ.

(2) 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 may not 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.

(3) We have $\mathsf{K}_i \subseteq \mathsf{K}_{i+1}$ because ρ_i is a prefix of ρ_{i+1}; also, in case 1, $q'_{\mathbf{br}} \in \mathsf{K}_{i+1}$ because of $a_{\mathbf{br}}$. Let $q' \in \mathsf{K}_{i+1} \setminus \mathsf{K}_i$ with, in case 1, $q' \neq q'_{\mathbf{br}}$. There must exist an agent a that takes a reception transition $(q, \mathbf{rec}(m, \alpha, q')$ in s_{i+1} and has value v in γ_{i+1}, with $q \in S$ as it appears in ρ. In cases 1 and 2, the broadcast has value v hence if $\alpha =$ '$=$' or '$*$' then a has value v in γ_i and $q \in \mathsf{K}_i$. In case 3, the broadcast has value $\neq v$ hence a may have value v in γ_{i+1} only when $\alpha =$ '$*$' and a had value v in γ_i, which implies $q \in \mathsf{K}_i$.

(4) It suffices to note that the first components of σ^i and σ^{i+1} are equal to S and $q'_{\mathbf{br}} \in S$, and $\mathsf{b}_{i+1} \in S \cup \{\perp\}$.

Overall, we have proven that $\sigma^i \to \sigma^{i+1}$, which concludes the induction step. Applying the result with $i = k$ proves Lemma 45. ◀

We may now prove completeness of the abstraction.

▶ **Lemma 46.** *If ρ is an initial run and $v \in \mathsf{val}(\rho)$, then $\sigma_0 \xrightarrow{*} \mathsf{abs}_v(\rho)$.*

Proof. Let ρ an initial run. We proceed by induction on the size of the set $\mathsf{cov}(\rho)$. First, if $\mathsf{cov}(\rho)$ is of size 1 then $\mathsf{cov}(\rho) = \{q_0\}$. Applying Lemma 45 directly gives $\sigma_0 = (\{q_0\}, q_0, \perp) \xrightarrow{*} \mathsf{abs}_v(\rho)$.

Suppose now that the statement is true for any ρ such that $\mathsf{cov}(\rho)$ is of size k, and suppose that we have a run ρ such that $\mathsf{cov}(\rho)$ is of size $k+1$. Let $v \in \mathsf{val}(\rho)$. Let $\rho_p : \gamma_0 \xrightarrow{*} \gamma_p$ the longest suffix of ρ such that $\mathsf{cov}(\rho) \neq \mathsf{cov}(\rho_p)$. By induction hypothesis, we know that for all $v \in \mathsf{val}(\rho)$, $\sigma_0 \xrightarrow{*} \mathsf{abs}_v(\rho_p)$. Write s the step immediatly after ρ_p in ρ. By maximality of ρ_p, ρ_p covers all states in $\mathsf{cov}(\rho) \setminus \mathsf{cov}(\rho_p)$.

Write $a_{\mathbf{br}}$ the agent broadcasting in s, $(q_{\mathbf{br}}, \mathbf{br}(m), q'_{\mathbf{br}})$ the corresponding transition and v the broadcast value. By induction hypothesis applied to ρ_p and v, $\sigma_0 \xrightarrow{*} \mathsf{abs}_v(\rho_p)$. Applying Lemma 45 on ρ and v gives that $(\mathsf{cov}(\rho), q_0, \emptyset) \xrightarrow{*} \mathsf{abs}_v(\rho)$. Therefore, it remains to prove that $\mathsf{abs}_v(\rho_p) \xrightarrow{*} (\mathsf{cov}(\rho), q_0, \emptyset)$. It suffices to prove that there exist b, K such that $\mathsf{abs}_{v_{\mathbf{br}}}(\rho_p) \to (\mathsf{cov}(\rho), \mathsf{b}, \mathsf{K})$, a gang reset allowing us to then reach $(\mathsf{cov}(\rho), q_0, \emptyset)$. In other words, we prove that, from $\mathsf{abs}_{v_{\mathbf{br}}}(\rho_p)$, one may cover all states in $\mathsf{cov}(\rho)$ in just one abstract step.

We aim at proving that $\mathsf{abs}_{v_{\mathbf{br}}}(\rho_p) \to (\mathsf{cov}(\rho), \mathsf{b}, \mathsf{K})$ for well-chosen b and K. Just like in the proof of Lemma 45, we make a case disjunction to prove that the broadcast in s may be mimicked in the abstraction from $\mathsf{abs}_{v_{\mathbf{br}}}(\rho_p)$. The type obtained is either a broadcast from clique or a broadcast from boss, because by hypothesis the agent broadcasting has value $v_{\mathbf{br}}$ and therefore its state before the broadcast is either the boss or in the clique in $\mathsf{abs}_{v_{\mathbf{br}}}(\rho_p)$.

Because we may choose b and K freely, the only challenging condition is (4). Let $q' \in \mathsf{cov}(\rho) \setminus \mathsf{cov}(\rho_p)$, $q' \neq q'_{\mathbf{br}}$. There exists an agent a that takes a reception transition $(q, \mathbf{rec}(m, \alpha), q')$ in s. If $\alpha =$ '=', then agent a has value $v_{\mathbf{br}}$ at the end of ρ_p hence q is in the clique of $\mathsf{abs}_{v_{\mathbf{br}}}(\rho_p)$ and condition (4) is satisfied. Otherwise, one has $q \in \mathsf{cov}(\rho_p)$ and condition (4) is satisfied. In the end, there exist b, K such that $\mathsf{abs}_{v_{\mathbf{br}}}(\rho_p) \to (\mathsf{cov}(\rho), \mathsf{b}, \mathsf{K})$. By a gang reset, this implies that $\mathsf{abs}_{v_{\mathbf{br}}}(\rho_p) \xrightarrow{*} (\mathsf{cov}(\rho), q_0, \emptyset)$; we have proven that $\sigma_0 \xrightarrow{*} \mathsf{abs}_{v_{\mathbf{br}}}(\rho_p) \xrightarrow{*} (\mathsf{cov}(\rho), q_0, \emptyset) \xrightarrow{*} \mathsf{abs}_v(\rho)$ which concludes the proof. ◀

K.2.2 Soundness

▶ **Lemma 47.** *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 47.

▶ **Lemma 48.** *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 $\mathbb{A}_i$, a configuration γ_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 $v_0, \ldots, v_n \in \mathbb{N}$ 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* $d = (q, \mathbf{br}(m), q')$ *from the clique* 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 $k' \in K_{i+1} \setminus K_i$ 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.

For all $s' \in S_{i+1} \setminus (S_i \cup K_{i+1})$ there exists a transition $d' = (s, \mathbf{rec}(m, \alpha), s')$. If $\alpha =$ '$=$' we then make all agents of $\mathbb{A}^{d'}_{i,s}$ follow that transition. Otherwise it means that $s \in K_i$, hence this transition is possible as they all have value v_i.

We set $v_{i+1} = v_i$.

Case 2: If $\sigma_i \to \sigma_{i+1}$ is a *broadcast* $d = (b_i, \mathbf{br}(m), b_{i+1})$ *from the boss*, 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.

For all $s' \in S_{i+1} \setminus (S_i \cup K_{i+1} \cup \{b_{i+1}\})$ there exists a transition $d' = (s, \mathbf{rec}(m, \alpha), s')$ (the operation cannot be $\downarrow$ or $=$ as otherwise s would be in K_{i+1}). We then make all agents of $\mathbb{A}^{d'}_s$ follow 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.

For all $s' \in S_{i+1} \setminus (S_i \cup K_{i+1})$ there exists a transition $d' = (s, \mathbf{rec}(m, \alpha), s')$ with $\alpha \in \{*, \downarrow\}$. We then make all agents of $\mathbb{A}^{d'}_s$ follow 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}.
- 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 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 process has value v_i, thus all those agents keep value $v_i = v_{i+1}$.
- For all $s \in S_i$, the agents of $\mathbb{A}^{\varepsilon}_{i,s}$ do not move between configurations γ_i and γ_{i+1}, hence they have state s in γ_{i+1}.
- If the step is not a gang reset, for all $s' \in S_{i+1} \setminus (S_i \cup \{b_{i+1}\})$, there exists $d \in \Delta$ from some $s \in S_i$ to s' such that all agents of $\mathbb{A}^{d}_{i,s}$ take that transition.
- 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. ◀

We simply apply Lemma 48 to an abstract run $\sigma_0 \to \cdots \sigma_n = \sigma$ from σ_0 to σ by setting $i = n$.

K.3 Conclusion

▶ **Proposition 49.** *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 right-to-left direction is given by Lemma 47. For the left-to-right direction, let ρ be an initial run ending in a configuration γ covering q_f. Let $v \in \mathbb{N}$ be such that γ has some agents with value v in state q_f.

We construct a suitable abstract run as follows: by Lemma 46 there exists an abstract run from some $\sigma_0 \in \Sigma_{\mathsf{init}}$ to an abstract configuration $\mathsf{abs}_v(\rho) = (S, b, K')$ for some S, b, K such that $q_f \in S$ and $q_f \in \{b\} \cup K$.

◀

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

▶ **Proposition 50.** *The cover problem is NP-hard.*

Proof. Let $x_1, \ldots, x_n$ 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 9. Our alphabet of messages is the set of literals $\{x_i, \neg x_i \mid 1 \leq i \leq n\}$. Each agent 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 one of $\ell_j^1, \ell_j^2, \ell_j^3$ for each j, with its own register value.

Suppose ϕ is satisfiable, let ν be a satisfying assignment, then we set $\mathbb{A} = \{a, a_1, \ldots, a_n\}$ as our set of agents. First for each i we make a broadcast x_i if $\nu(x_i) = \top$ and $\neg x_i$ otherwise, this broadcast shall be received only by agent a_i. Agent a_i will store a's register value. Then

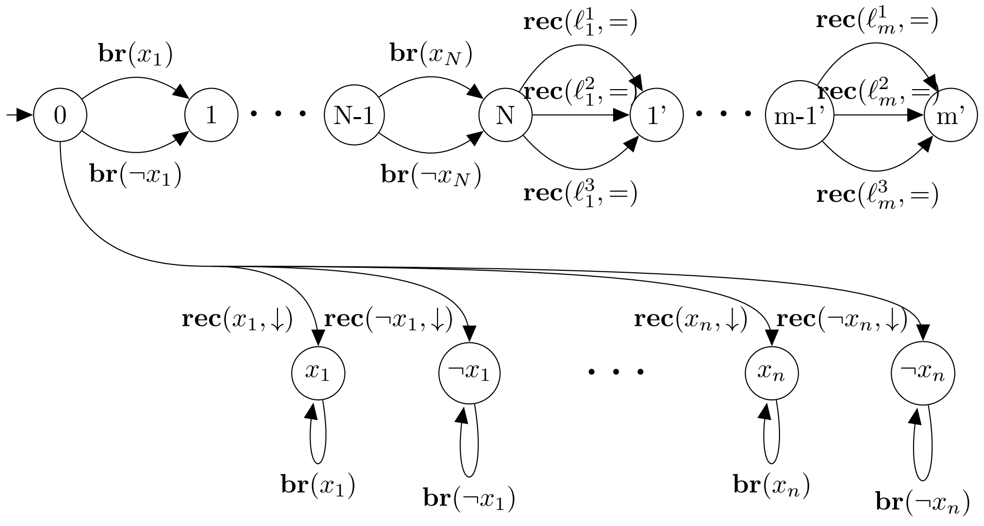


Figure 9 The protocol used for the NP-hardness proof.

for each j we select some ℓ_j^p satisfied by ν. There exists i such that a_i is in state ℓ_j^p. It 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 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. The register of a cannot have changed its value throughout the run. For each j it has received one of $\ell_j^1, \ell_j^2, \ell_j^3$ along with its own initial register value (which we call r). Let p_j be such that a has received $\ell_j^{p_j}$. Hence for all j there exists an agent $a_j \in \mathbb{A}$ such that at some point in the run the register value of a_j is r and a_j broadcasts $\ell_j^{p_j}$. This agent a_j must be in state $\ell_j^{p_j}$ after receiving the broadcast $\mathbf{br}(\ell_j^{p_j})$ from a (as all agents start with different register values). Hence $\ell_j^{p_j}$ 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 26.** *The coverability problem is* NP*-complete for protocols with one register.*

Proof. The lower bound is given by Proposition 50. For the upper bound, say we are given a protocol $\mathcal{P} = (Q, \mathcal{M}, \Delta, q_0, r)$ and a state q_f. We have to verify that there exists a reachable configuration γ_f convering q_f. By Proposition 49, it is the case if and only if there is an abstract run to an abstract configuration (S, b, K) with $q_f \in S$. Furthermore, by Lemma 43 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. ◀