Safety Verification of Wait-Only Non-Blocking Broadcast Protocols

Lucie Guillou[1], Arnaud Sangnier[2], and Nathalie Sznajder[3]

[1] IRIF, CNRS, Université Paris Cité, France
[2] DIBRIS, Università di Genova, Italy
[3] LIP6, CNRS, Sorbonne Université, France

Abstract. We study networks of processes that all execute the same finite protocol and communicate synchronously in two different ways: a process can broadcast one message to all other processes or send it to at most one other process. In both cases, if no process can receive the message, it will still be sent. We establish a precise complexity class for two coverability problems with a parameterised number of processes: the state coverability problem and the configuration coverability problem. It is already known that these problems are Ackermann-hard (but decidable) in the general case. We show that when the protocol is *Wait-Only*, i.e., it has no state from which a process can send and receive messages, the complexity drops to P and PSpace, respectively.

Keywords: Parameterised Networks · Broadcast · Verification

1 Introduction

Verification of distributed systems. The ubiquity of distributed and concurrent systems in nowadays applications leads to an increasing need to ensure their correct behaviour. Over the last two decades, the verification of such systems has become a crucial research direction in the field of computer science. Indeed, analysing distributed systems has proven to be challenging. One difficulty is due to the numerous interleavings caused by the concurrent behaviour of the system entities, that make the design and modelling of these systems very complex. Moreover, the number of agents is often not known a priori; in that case, verifying all possible behaviours of such a system amounts to analyse it for any number of agents, i.e. an infinite number of times. The unpredictability of the number of participants in a system makes classical techniques such as model-checking impractical and requires some new techniques.

Parameterised verification. Addressing the challenge of unbounded entities involves designing schematic programs or protocols intended for implementation by multiple identical processes and parameterised by the number of entities. While in general parameterised verification is undecidable [AK86], several realistic restrictions enable automatic verification. Among them, one can highlight systems where entities have no identity, and systems with simple com-

munication mechanism. Several papers have considered synchronous communication means, as rendez-vous [GS92, DRB02, HS20, BER21, GSS23] and broadcast [EFM99, DSTZ12]. Note that, surprisingly, parameterised verification, when decidable, is sometimes significantly easier than the same problem with a fixed number of entities [DEGM17]. In all those models, all the entities execute the same program which is modelled as a finite-state automaton.

Wait-Only Non-Blocking Broadcast protocols In [GSS23], the authors have studied the complexity of several parameterised verification problems in the context of non-blocking rendez-vous. This communication mechanism, motivated by Java Threads programming, involves at most two processes: when a process sends a message, it is received by at most one process ready to receive the message, and both processes jointly change their local state. However, when no process is ready to receive the message, the message is sent anyway and lost, and only the sender changes its local state. This is in contrast with classical rendez-vous as studied for instance in [GS92], where a sender is prevented to send a message if no process can receive it. The model proposed in [GSS23] allows to capture some behaviour of the Threads: when a Thread is suspended in a waiting state, it can be woken up upon the reception of a `notify` message sent by another Thread, but the sender is not blocked if no Thread is suspended; it simply continues its execution and the `notify` message is lost. However, this fails to capture the behaviour of what occurs when a Thread sends a `notifyAll` message that will be received by all the suspended Threads waiting for that message. This, as already highlighted in [DRB02], is modelled by the broadcast mechanism, in which a message sent by a process will be received by all the processes ready to receive it. Observe that broadcast is also a non-blocking means of communication. In this work we consider Non-Blocking Broadcast protocols, that allow for both broadcast and non-blocking rendez-vous. One important problem in parameterised verification is the coverability problem: is it possible that, starting from an initial configuration, (at least) one process reaches a bad state? With classical rendez-vous mechanism, this problem is in P [GS92], while with non-blocking rendez-vous, it is Expspace-complete [GSS23]. For protocols enabling both broadcast and non-blocking rendez-vous, the problem is decidable [EK03] and Ackermann-hard [SS13, EFM99, ARZ15]. In this work we study the coverability problem for a syntactic restriction of the protocols, introduced in [GSS23], namely Wait-Only protocols, in which there is no state from which a process can both send and receive a message. In this context, when processes communicate with non-blocking rendez-vous only, the coverability problem is in P [GSS23].

Our contributions. We show that the coverability problem for Wait-Only Non-Blocking Broadcast protocols is P-complete, and that the *configuration coverability problem* is Pspace-complete. This last problem asks whether it is possible to cover a given configuration (and not simply a bad state) from an initial state. Note that both problems are in P when forbidding broadcasts [GSS23], however, in this work, the P-membership proof is less involved and the Pspace-membership proof uses a different technique.

2 Model and verification problems

We denote by $\mathbb{N}$ the set of natural numbers. For a finite set E, the set $\mathbb{N}^E$ represents the multisets over E. For two elements $s, s' \in \mathbb{N}^E$, we denote by $s + s'$ the multiset such that $(s + s')(e) = s(e) + s'(e)$ for all $e \in E$. We say that s' is bigger than s, denoted $s \preceq s'$ if and only if $s(e) \leq s'(e)$ for all $e \in E$. If $s \preceq s'$, then $s' - s$ is the multiset such that $(s' - s)(e) = s'(e) - s(e)$ for all $e \in E$. Given a subset $E' \subseteq E$ and $s \in \mathbb{N}^E$, we denote by $\|s\|_{E'}$ the sum $\Sigma_{e \in E'} s(e)$ of elements of E' present in s. The size of a multiset s is given by $\|s\| = \|s\|_E$. For $e \in E$, we use sometimes the notation e for the multiset s verifying $s(e) = 1$ and $s(e') = 0$ for all $e' \in E \setminus \{e\}$ and, to represent for instance the multiset with four elements a, b, b and c, we will also use the notations $\wr a, b, b, c \wr$ or $\wr a, 2 \cdot b, c \wr$.

2.1 Networks of Processes using Rendez-Vous and Broadcast

We now present the model under study in this work. We consider networks of processes where each entity executes the same protocol given by a finite state automaton. Given a finite alphabet Σ of messages, the transitions of a protocol are labelled with four types of actions that can be executed by the processes of the network. For $m \in \Sigma$ a process can (1) send a (non-blocking) rendez-vous over the message m with $!m$, (2) send a broadcast over m with $!!m$, (3) receive a rendez-vous or a broadcast over m with $?m$ and (4) perform an internal action with τ (assuming $\tau \notin \Sigma$). In order to refer to these different actions, we denote by $!\Sigma$ the set $\{!m \mid m \in \Sigma\}$, by $!!\Sigma$ the set $\{!!m \mid m \in \Sigma\}$ and by $?\Sigma$ the set $\{?m \mid m \in \Sigma\}$. Finally, we use the notation Op_Σ to represent the set of labels $!!\Sigma \cup !\Sigma \cup ?\Sigma \cup \{\tau\}$ and Act_Σ to represent the set of actions $!!\Sigma \cup !\Sigma \cup \{\tau\}$.

Definition 1. *A* Non-Blocking Broadcast protocol P *(NB-Broadcast protocol) is a tuple* (Q, Σ, q_{in}, T) *such that* Q *is a finite set of states,* Σ *is a finite alphabet,* q_{in} *is an initial state, and* $T \subseteq Q \times \mathsf{Op}_\Sigma \times Q$ *is the transition relation.*

In this work, we are in particular interested in studying some syntactical restrictions on such protocols. We say that a protocol is *Wait-Only* when for all $q \in Q$, either $\{q' \mid (q, \alpha, q') \in T \text{ with } \alpha \in ?\Sigma\} = \varnothing$, or $\{q' \mid (q, \alpha, q') \in T \text{ with } \alpha \in !!\Sigma \cup !\Sigma \cup \{\tau\}\} = \varnothing$. We call a state respecting the first or both conditions an *active* state and a state respecting the second condition a *waiting* state. In the following, we denote by Q_A the set of active states of P and Q_W its set of waiting states.

If the protocol does not contain any broadcast transition of the form $(q, !!m, q')$, we call it a *Non-Blocking Rendez-vous protocol* (NB-Rendez-vous protocol).

Example 1. An example of protocol is depicted on Figure 1. We name P the protocol drawn without the dashed arrow between q_2 and q_3, and P_{dashed} the complete protocol. Note that P is a *Wait-Only* protocol, indeed each state is either an active state (q_{in}, q_2, q_3, q_5 and q_6), or a waiting state, (q_1 and q_4). However, P_{dashed} is not a Wait-Only protocol, since q_2 is neither an active state nor a waiting state as it has an outgoing transition labelled with an action $!!c$, and an outgoing transition labelled with an action $?a$.

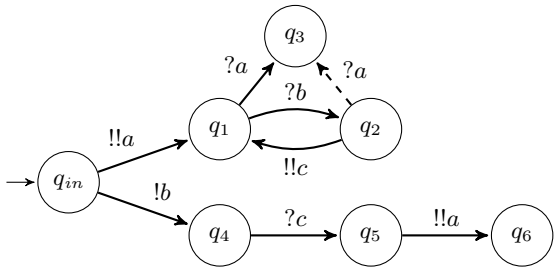

Fig. 1: Example of a protocol denoted P_{dashed} (we note P the protocol P_{dashed} without the dashed arrow between q_2 and q_3)

We shall now present the semantics associated to protocols. Intuitively, we consider networks of processes, each process being in a state of the protocol and changing its state according to the transitions of the protocol with the following assumptions. A process can perform on its own an internal action τ and this does not change the state of the other processes. When a process sends a broadcast with the action $!!m$, then all the processes in the network which are in a state from which the message m can be received (i.e. with an outgoing transition labelled by $?m$) have to take such a transition. And when a process sends a rendez-vous with the action $!m$, then *at most* one process receives it: in fact, if there is at least one process in a state from which the message m can be received, then exactly one of these processes has to change its state, along with the receiver (while the other processes do not move), but if no process can receive the message m, only the sender performs the action $!m$. This is why we call this communication mechanism a *non-blocking* rendez-vous.

We move now to the formal definition of the semantics. Let $P = (Q, \Sigma, q_{in}, T)$ be a protocol.

A *configuration* C over P is a non-empty multiset over Q, it is *initial* whenever $C(q) = 0$ for all $q \in Q \setminus \{q_{in}\}$. We note $\mathcal{C}$ the set of all configurations over P, and $\mathcal{I}$ the set of all initial configurations over P.

For $q \in Q$, we denote by $R(q)$ the set $\{m \in \Sigma \mid \text{there exists } q' \in Q, (q, ?m, q') \in T\}$ of messages that can be received when in the state q. Given a transition $t = (q, \alpha, q') \in T$, we define the relation $\xrightarrow{t} \subseteq \mathcal{C} \times \mathcal{C}$ as follows: for two configurations C, C' we have $C \xrightarrow{t} C'$ iff one of the following conditions holds:

(a) $\alpha = \tau$, and $C(q) > 0$ and $C' = C - \wr q \wr + \wr q' \wr$;
(b) $\alpha = !!m$, and $C = \wr q_1, q_2, \ldots, q_n, q \wr$ for some $n \in \mathbb{N}$, and $C' = \wr q_1', q_2', \ldots, q_n', q' \wr$ where for all $1 \leq i \leq n$, either $m \notin R(q_i)$ and $q_i' = q_i$, or $(q_i, ?m, q_i') \in T$;
(c) $\alpha = !m$, and $C(q) > 0$, and $(C - \wr q \wr)(p) = 0$ for all $p \in Q$ such that $m \in R(p)$, and $C' = C - \wr q \wr + \wr q' \wr$;
(d) $\alpha = !m$ and $C(q) > 0$ and there exists $p \in Q$ such that $(C - \wr q \wr)(p) > 0$ and $(p, ?m, p') \in T$ for some $p' \in Q$, and $C' = C - \wr p, q \wr + \wr q', p' \wr$.

Observe that when $C \xrightarrow{t} C'$, we necessarily have $\|C\| = \|C'\|$.

The case (a) corresponds to the internal action of a single process, the case (b) to the emission of a broadcast hence all the processes that can receive the message have to receive it. The case (c) corresponds to the case where a process sends a rendez-vous and there is no process to answer to it, hence only the sender changes its state. The case (d) corresponds to a classical rendez-vous where a process sends a rendez-vous and another process receives it. Note that for both the broadcast and the rendez-vous, the absence of a receiver does not prevent a sender from its action. We call non-blocking our semantics because of the case (c), which contrasts with the broadcast model of [EFM99] for instance, where this case is not possible.

We write $C \to C'$ whenever there exists $t \in T$ such that $C \xrightarrow{t} C'$, and denote by $\to^*$ [resp. $\to^+$] the reflexive and transitive [resp. transitive] closure of $\to$. An execution ρ is then a finite sequence of the form $C_0 \xrightarrow{t_1} C_1 \xrightarrow{t_2} \dots \xrightarrow{t_n} C_n$, it is said to be initialized when C_0 is an initial configuration in $\mathcal{I}$.

Example 2. We consider the protocol P of Figure 1. We have then the following execution starting at the initial configuration $\wr q_{in}, q_{in}, q_{in}\wr$ with three processes:

$$\wr q_{in}, q_{in}, q_{in}\wr \xrightarrow{(q_{in},!!a,q_1)} \wr q_1, q_{in}, q_{in}\wr \xrightarrow{(q_{in},!b,q_4)} \wr q_2, q_4, q_{in}\wr \xrightarrow{(q_{in},!b,q_4)} \wr q_2, q_4, q_4\wr \xrightarrow{(q_2,!!c,q_1)} \wr q_1, q_5, q_5\wr \xrightarrow{(q_5,!!a,q_6)} \wr q_3, q_6, q_5\wr.$$

It corresponds to the following sequence of events: one of the agents broadcasts message a (not received by anyone), then another agent sends message b which leads to a rendez-vous with the first agent on q_1, the last agent sends message b which is not received by anyone (the sending is possible thanks to the non-blocking semantics), the agent in state q_2 broadcasts message c which is received by the two other agents, and finally, one of the agents in q_5 broadcasts letter a which is received by the process on q_1.

Remark 1. Observe that internal transitions labelled by τ can be replaced by broadcast transitions of the form $!!\tau$. Since no transition is labelled by $?\tau$, when τ is broadcasted, no process is ready to receive it and the semantics is equivalent to the one of an internal transition. Observe also that since $\tau \in \mathsf{Act}_\Sigma$, transforming internal transitions into broadcasts keeps a protocol Wait-Only.

Following this remark, we will omit internal transitions in the rest of this work.

2.2 Verification Problems

We present now the verification problems we are interested in. Both these problems consist in ensuring a safety property: we want to check that, no matter the number of processes in the network, a configuration exhibiting a specific pattern can never be reached. If the answer to the problem is positive, it means in our context that the protocol is not safe.

The state coverability problem StateCover is stated as follows:

StateCover
Input: An NB-Broadcast Protocol P and a state $q_f \in Q$; **Question:** Do there exist $C \in \mathcal{I}$ and $C' \in \mathcal{C}$ such that $C \to^* C'$, and $C'(q_f) > 0$?

When the answer is positive, we say that q_f is *coverable* by P. The second problem, called the configuration coverability problem ConfCover, is a generalisation of the first one where we look for a multi-set to be covered.

ConfCover
Input: An NB-Broadcast Protocol P and a configuration $C_f \in \mathcal{C}$; **Question:** Do there exist $C \in \mathcal{I}$ and $C' \in \mathcal{C}$ such that $C \to^* C'$, and $C_f \leq C'$?

Remark 2. Note that if P is a Wait-Only protocol and its initial state q_{in} is a waiting state, then no state besides q_{in} is coverable and the only coverable configurations are the initial ones. Hence, when talking about Wait-Only protocols, we assume in the rest of this work that the initial state q_{in} is always an *active state*.

Example 3. In the protocol P of Figure 1, configuration $\{\!|q_3, q_6|\!\}$ is coverable as $\{\!|q_{in}, q_{in}, q_{in}|\!\} \to^* \{\!|q_3, q_6, q_5|\!\}$ (see Example 2) and $\{\!|q_3, q_6|\!\} \leq \{\!|q_3, q_6, q_5|\!\}$.

Results. We summarize in Table 1 the results on NB-Broadcast protocols, where our results appear in red. Note that for what concerns the lower bounds for NB-Broadcast protocols, they have been proved in [SS13][Fact16, Remark 17] for Broadcast protocols with a classical "blocking" rendez-vous semantics, i.e. where a process requesting a rendez-vous cannot take the transition if no process answers the rendez-vous. However, it is possible to retrieve the lower bound for NB-Broadcast protocols without rendez-vous by using the fact that "blocking" rendez-vous can be simulated by broadcast as shown in [EFM99, ARZ15].

Type of protocols	StateCover	ConfCover
NB-Broadcast	Decidable [EK03] and Ackermann-hard [SS13, ARZ15, EFM99]	
NB-Rendez-vous	ExpSpace-complete [GSS23]	
Wait-Only NB-Rendez-vous	in P [GSS23], **P-hard**	
Wait-Only NB-Broadcast	**P-complete**	**PSpace-complete**

Table 1: Coverability in NB-Broadcast protocols

In the rest of this work, we will focus on Wait-Only NB-Broadcast protocols which we name Wait-Only protocols for ease of notation.

3 Preliminaries properties

Wait-Only protocols enjoy a nice property on coverable states. The property makes a distinction between active states and waiting states. First, we show that

when an active state is coverable, then it is coverable by a number of processes as big as one wants, whereas this is not true for a waiting state. Indeed, it is possible that a waiting state can be covered by exactly one process at a time, and no more. However, we show that if two active states, or if an active state and a waiting state are coverable, then there is an execution that reaches a configuration where they are both covered.

This property relies on the fact that once the active state has been covered in an execution, it will not be emptied while performing the sequence of actions allowing to cover the second (waiting state), since no reception of message can happen in such a state. As we will see, this phenomenon can be generalised to a subset of active states.

Example 4. Going back to the protocol P of Figure 1, consider the active state q_2. It is coverable as shown by the execution $\wr q_{in}, q_{in} \wr \xrightarrow{(q_{in},!!a,q_1)} \wr q_1, q_{in} \wr \xrightarrow{(q_{in},!b,q_4)} \wr q_2, q_4 \wr$. From this execution, for any integer $n \in \mathbb{N}$, one can build an execution leading to a configuration covering $\wr n \cdot q_2 \wr$. For instance, for $n = 2$, we build the following execution:

$$\wr q_{in}, q_{in}, q_{in}, q_{in} \wr \xrightarrow{(q_{in},!!a,q_1)} \wr q_1, q_{in}, q_{in}, q_{in} \wr \xrightarrow{(q_{in},!b,q_4)} \wr q_2, q_4, q_{in}, q_{in} \wr$$
$$\xrightarrow{(q_{in},!!a,q_1)} \wr q_2, q_4, q_1, q_{in} \wr \xrightarrow{(q_{in},!b,q_4)} \wr q_2, q_4, q_2, q_4 \wr.$$

Furthermore each coverable waiting state is coverable by a configuration that also contains q_2. For instance, $\wr q_2, q_4 \wr$ is coverable as shown by the above execution.

Note that when considering P_{dashed}, which is not Wait-Only, such an execution is not possible as the second broadcast of a should be received by the process on q_2. In fact, q_2 is coverable by only one process and no more. This is because q_1 is coverable by at most one process at a time; every new process arriving in state q_1 will do so by broadcasting a, message that will be received by the process already in q_1. Then any attempt to send two processes in q_2 requires a broadcast of a, hence the reception of a by the process already in q_2. For the same reason, in P_{dashed}, $\wr q_1, q_2 \wr$ is not coverable whereas q_1 is a coverable waiting state and q_2 a coverable active state.

Before stating the main lemma of this section (Lemma 1), we need an additional definition. For each *coverable* state $q \in Q$, let min_q be the minimal number of processes needed to cover q. More formally, $\mathsf{min}_q = \min\{n \mid n \in \mathbb{N}$, there exists $C \in \mathcal{C}$ s. t. $\wr n.q_{in} \wr \to^* C$ and $C(q) > 0\}$. Note that min_q is defined only when q is coverable.

Lemma 1. *Let $P = (Q, \Sigma, q_{in}, T)$ be a Wait-Only protocol, $A = \{q_1, \ldots, q_n\} \subseteq Q_A$ a subset of coverable* active *states and $p \in Q_W$ a coverable* waiting *state. Then, for all $N \in \mathbb{N}$, there exists an execution $C_0 \to^* C_m$ such that $C_0 \in \mathcal{I}$, and $\{N \cdot q_1, \ldots, N \cdot q_n, p\} \preceq C_m$. Moreover, $||C_0|| = N.\sum_{i=1}^{n} \mathsf{min}_{q_i} + \mathsf{min}_p$.*

The proof of this lemma relies on the two following properties on executions of Wait-Only protocols:

Lemma 2. *Given an initialized execution* $C_0 \xrightarrow{t_1} C_1 \xrightarrow{t_2} \dots \dots \xrightarrow{t_k} C_k$ *and another initial configuration* C'_0*, we can build an execution* $\widehat{C_0} \xrightarrow{t_1} \widehat{C_1} \xrightarrow{t_2} \dots \dots \xrightarrow{t_k} \widehat{C_k}$ *with* $\widehat{C_0} = C_0 + C'_0$*. For all* $0 \leq i \leq k$*,* $\widehat{C}_i(q) = C_i(q)$ *for all* $q \in Q \setminus \{q_{in}\}$*, and* $\widehat{C}_i(q_{in}) = C_i(q_{in}) + C'_0(q_{in})$*.*

This property comes from the fact that q_{in} is an active state. Hence, if we start from a bigger configuration, we can take exactly the same transitions as in the initial execution, the additional processes will stay in the initial state.

Lemma 3. *Given an initialized execution* $C_0 \xrightarrow{t_1} C_1 \xrightarrow{t_2} \dots \dots \xrightarrow{t_k} C_k$*, given some* $M \geq 1$*, for all configurations (not necessarily initial)* $\widetilde{C}_0$ *such that* $\widetilde{C}_0(q_{in}) \geq M.C_0(q_{in})$*, we have the execution* $\widetilde{C}_0 \xrightarrow{t_1} \dots \xrightarrow{t_k} \widetilde{C}_k$ *in which, for all* $0 \leq i \leq k$*:* $\widetilde{C}_i(q) \geq \widetilde{C}_0(q) + C_i(q)$ *for all* $q \in Q_A \setminus \{q_{in}\}$*,* $\widetilde{C}_i(q) \geq C_i(q)$ *for all* $q \in Q_W$ *and* $\widetilde{C}_i(q_{in}) \geq (M-1).C_0(q_{in}) + C_i(q_{in})$*.*

This last property states that, if one mimicks an initialized execution from another (non initial) configuration, the processes already present in *active* states (different from the initial state) will not move during the execution.

Proof of Lemma 1. Let $N \in \mathbb{N}$. Using these two properties, we can now prove the lemma. We start by proving that there exists an execution $C_0 \to^* C_m$ such that for all $q \in A$, $C_m(q) \geq N$ and $\|C_0\| = N.\sum_{i=1}^{n} \mathsf{min}_{q_i}$. We prove it by induction on the size of A. If $A = \varnothing$, the property is trivially true. Let $n \in \mathbb{N}$, and assume the property to hold for all subsets $A \subseteq Q_A$ of size n. Take $A = \{q_1, q_2, \dots, q_{n+1}\} \subseteq Q_A$ of size $n+1$ such that all states $q \in A$ are coverable and let $A' = A \setminus \{q_1\}$. Let $C_0 \xrightarrow{t_1} C_1 \xrightarrow{t_2} \dots \dots \xrightarrow{t_k} C_k$ be an execution covering q_1 with $\|C_0\| = \mathsf{min}_{q_1}$. Let $C'_0 \xrightarrow{t'_1} C'_1 \xrightarrow{t'_2} \dots \xrightarrow{t'_m} C'_m$ be an execution such that for all $q' \in A'$, $C'_m(q') \geq N$ and $\|C'_0\| = N.\sum_{i=2}^{n+1} \mathsf{min}_{q_i}$ (it exists by induction hypothesis). We let $C_0^N = \wr(N.\mathsf{min}_{q_1}) \cdot q_{in} \wr$ and $C''_0 = C'_0 + C_0^N$. Thanks to Lemma 2, we can build an execution $C''_0 \xrightarrow{t'_1} C''_1 \xrightarrow{t'_2} \dots \xrightarrow{t'_m} C''_m$, with $C''_m(q) = C'_m(q)$ for all $q \in Q \setminus \{q_{in}\}$ and $C''_m(q_{in}) = C'_m(q_{in}) + C_0^N(q_{in}) = C'_m(q_{in}) + N.\mathsf{min}_{q_1}$. So, for all $q' \in A'$, $C''_m(q') = C'_m(q') \geq N$ and $\|C''_0\| = \|C'_0\| + \|C_0^N\| = N.\sum_{i=2}^{n+1} \mathsf{min}_{q_i} + N.\mathsf{min}_{q_1}$.

Now that we have shown how to build an execution that leads to a configuration with more than N processes on all states in A' and enough processes in the initial state, we show that mimicking N times the execution allowing to cover q_1 allows to obtain the desired result. Let $C_{0,1} = C''_m$. We know that for all $q' \in A'$, $C_{0,1}(q') \geq N$, and $C_{0,1}(q_{in}) \geq N.\mathsf{min}_{q_1}$. Since $\|C_0\| = \mathsf{min}_{q_1}$, using Lemma 3, we can build the execution $C_{0,1} \xrightarrow{t_1} \dots \xrightarrow{t_k} C_{k,1}$ with $C_{k,1}(q_{in}) \geq (N-1).\mathsf{min}_{q_1}$, $C_{k,1}(q') \geq C_{0,k}(q') + C_k(q') \geq N$ for all $q' \in A'$ and $C_{k,1}(q_1) \geq C_{0,k}(q_1) + C_k(q_1) \geq 1$. Iterating this construction and applying each time Lemma 3, we obtain that there is an execution $C_{0,1} \xrightarrow{t_1} \dots \xrightarrow{t_k} C_{k,1} \xrightarrow{t_1} \dots \xrightarrow{t_k} C_{k,2} \dots \xrightarrow{t_1} \dots \xrightarrow{t_k} C_{k,N-1} \xrightarrow{t_1} \dots \xrightarrow{t_k} C_{k,N}$ with $C_{k,i}(q_{in}) \geq (N-i).\mathsf{min}_{q_1}$, $C_{k,i}(q') \geq N$ for all $q' \in A'$ and $C_{k,i}(q_1) \geq C_{k,i-1}(q_1) + 1 \geq i$. Observe that to obtain that $C_{k,i}(q_1) \geq i$ from Lemma 3, we use the fact that $q_1 \in Q_A$. Hence, $C_{k,N}(q_1) \geq N$ and $C_{k,N}(q') \geq N$

for all $q' \in A'$ and we have build an execution where $C_{k,N}(q) \geq N$ for all $q \in A$ and $\|C_{k,N}\| = \|C_0''\| = N.\sum_{i=1}^{|A|} \mathsf{min}_{q_i}$, as expected.

At last, take a subset $A = \{q_1, \ldots, q_n\} \subseteq Q_A$ of coverable active states. Let $C_0 \to^* C_m$ be an execution such that $C_m(q) \geq N$ for all $q \in A$ and $\|C_0\| = N.\sum_{i=1}^{n} \mathsf{min}_{q_i}$. Let $p \in Q_W$ a coverable state and $C_0' \to^* C_k'$ such that $C_k'(p) \geq 1$ and $\|C_0'\| = \mathsf{min}_p$. By Lemma 2, we let $\widehat{C}_0 = C_0 + C_0'$ and we have an execution $\widehat{C}_0 \to^* \widehat{C}_m$ with $\widehat{C}_m(q) = C_m(q)$ for all $q \in Q \setminus \{q_{in}\}$, and $\widehat{C}_m(q_{in}) = C_m(q_{in}) + C_0'(q_{in})$. Hence, $\widehat{C}_m(q) \geq N$ for all $q \in A$ and $\widehat{C}_m(q_{in}) \geq C_0'(q_{in})$, and note that $\|\widehat{C}_m\| = \|\widehat{C}_0\| = \|C_0\| + \|C_0'\| = N.\sum_{i=1}^{n} \mathsf{min}_{q_i} + \mathsf{min}_p$. Then, with $\widetilde{C}_0 = \widehat{C}_m$, by Lemma 3, we have an execution $\widetilde{C}_0 \to^* \widetilde{C}_k$ with $\widetilde{C}_k(q) \geq \widetilde{C}_0(q) + C_k(q) \geq \widetilde{C}_0(q) \geq N$ for all $q \in A$, and $\widetilde{C}_k(p) \geq C_k(p) \geq 1$, and $\|\widetilde{C}_0\| = \|\widehat{C}_0\| = N.\sum_{i=1}^{n} \mathsf{min}_{q_i} + \mathsf{min}_p$.

□

4 StateCover for Wait-Only protocols is P-complete

4.1 Upper bound

We present here a polynomial time algorithm to solve the state coverability problem when the considered protocol is Wait-Only. Our algorithm computes in a greedy manner the set of coverable states using Lemma 1.

Given a Wait-Only protocol $P = (Q, \Sigma, q_{in}, T)$, we compute iteratively a set of states $S \subseteq Q$ containing all the states that are coverable by P, by relying on a family $(S_i)_{i \in \mathbb{N}}$ of subsets of Q formally defined as follows (we recall that $\mathsf{Act}_\Sigma = !!\Sigma \cup !\Sigma$):

$S_0 = \{q_{in}\}$

$S_{i+1} = S_i \ \cup \ \{q \mid \text{there exists } q' \in S_i, (q', \alpha, q) \ \in T, \alpha \in \mathsf{Act}_\Sigma\}$

$\cup \{q_2' \mid \text{there exist } q_1, q_2 \in S_i, q_1' \in Q, a \in \Sigma \text{ s. t. } (q_1, !a, q_1') \in T \text{ and } (q_2, ?a, q_2') \in T\}$

$\cup \{q_2' \mid \text{there exist } q_1, q_2 \in S_i, q_1' \in Q, a \in \Sigma \text{ s. t. } (q_1, !!a, q_1') \in T \text{ and } (q_2, ?a, q_2') \in T\}$

Intuitively at each iteration, we add some control states to S_{i+1} either if they can be reached from a transition labelled with an action (in Act_Σ) starting at a state in S_i or if they can be reached by two transitions corresponding to a communication by broadcast or by rendez-vous starting from states in S_i. We then define $S = \bigcup_{n \in \mathbb{N}} S_n$. Observe that $(S_i)_{i \in \mathbb{N}}$ is an increasing sequence such that $|S_i| \leq |Q|$ for all $i \in \mathbb{N}$. Then we reach a fixpoint $M \leq |Q|$ such that $S_M = S_{M+1} = S$. Hence S can be computed in polynomial time.

The two following lemmas show correctness of this algorithm. We first prove that any state $q \in S$ is indeed coverable by P. Moreover, we show that min_q the minimal number of processes necessary to cover $q \in Q$ is smaller than $2^{|Q|}$.

Lemma 4. *If $q \in S$, then there exists $C \in \mathcal{I}$ and $C' \in \mathcal{C}$ such that $C \to^* C'$, $C'(q) > 0$ and $\|C\| \leq 2^{|Q|}$.*

Proof. Let $M \in \mathbb{N}$ be the first natural such that $S_M = S_{M+1}$. We have then $S_M = S$ and $M \leq |Q|$. We prove by induction that for all $0 \leq i \leq M$, for all $q \in S_i$, there exists $C \in \mathcal{I}$ and $C' \in \mathcal{C}$ such that $C \to^* C'$, $C'(q) > 0$ and $\|C\| \leq 2^i$.

As $S_0 = \{q_{in}\}$, the property trivially holds for $i = 0$, since $\wr q_{in} \wr \in \mathcal{I}$ and $\wr q_{in} \wr(q_{in}) > 0$.

Assume now the property to be true for $i < M$ and let $q \in S_{i+1}$. If $q \in S_i$, then by induction hypothesis, we have that there exists $C \in \mathcal{I}$ and $C' \in \mathcal{C}$ such that $C \to^* C'$, $C'(q) > 0$ and $\|C\| \leq 2^i < 2^{i+1}$. We suppose that $q \notin S_i$ and proceed by a case analysis on the way q has been added to S_{i+1}.

1. there exists $q' \in S_i$ and $t = (q', \alpha, q) \in T$ with $\alpha \in \mathsf{Act}_\Sigma$. By induction hypothesis, there exists an execution $C \to^* C'$ such that $C'(q') > 0$ and $\|C\| \leq 2^i$. But we have then $C' \xrightarrow{t} C''$ with $C''(q) > 0$, and consequently as well $C \to^* C''$. This is true because of the "non-blocking" nature of both broadcast and rendez-vous message in this model. Hence there is no need to check for a process to receive the message to ensure the execution $C \to^* C''$.
2. there exist $q_1, q_2 \in S_i$ and $q_1' \in Q$ and there exists $a \in \Sigma$ such that $(q_1, !a, q_1'), (q_2, ?a, q) \in T$. By induction hypothesis, we have that there exists $C_1, C_2 \in \mathcal{I}$ and $C_1', C_2' \in \mathcal{C}$ such that $C_1 \to^* C_1'$ and $C_2 \to^* C_2'$ and $C_1'(q_1) > 0$ and $C_2'(q_2) > 0$ and $\|C_1\| \leq 2^i$ and $\|C_2\| \leq 2^i$. Note furthermore that by definition q_1 is in Q_A and as $(q_2, ?a, q) \in T$, q_2 do not belong to Q_A. Hence $q_1 \neq q_2$. By Lemma 1, we know that there exist $C \in \mathcal{I}$ and $C' \in \mathcal{C}$ such that $C \to^* C'$ and $C'(q_1) > 0$ and $C'(q_2) > 0$. Furthermore, recall that min_{q_i} for $i \in \{1, 2\}$ is the minimal number of processes needed to cover q_i, by Lemma 1, $\|C\| \leq \mathsf{min}_{q_1} + \mathsf{min}_{q_2}$. By induction hypothesis, $\mathsf{min}_{q_1} + \mathsf{min}_{q_2} \leq 2^i + 2^i$, hence $\|C\| \leq 2^{i+1}$.

   We then have $C' \xrightarrow{(q_1, !a, q_1')} C''$ with $C'' = C' - \wr q_1, q_2 \wr + \wr q_1', q \wr$. Hence $C \to^* C''$ with $C''(q) > 0$.
3. there exist $q_1, q_2 \in S_i$ and $q_1' \in Q$ and there is some $a \in \Sigma$ such that $(q_1, !!a, q_1'), (q_2, ?a, q) \in T$. As above, we obtain the existence of an execution $C \to^* C'$ with $C'(q_1) > 0$ and $C'(q_2) > 0$, and $\|C\| \leq 2^{i+1}$. Then $C' = \wr q_1, q_2, \ldots, q_k \wr$ and $C' \xrightarrow{(q_1, !!a, q_1')} C''$ with $C'' = \wr q_1', q, \ldots, q_k' \wr$ with, for all $3 \leq j \leq k$, either $a \notin R(q_j)$ and $q_j = q_j'$ or $(q_j, ?a, q_j') \in T$. In any case, we have $C \to^* C''$ with $C''(q) > 0$.

So, for any $q \in S$, $q \in S_M$ and we have an execution $C \to^* C'$ with $C \in \mathcal{I}$ such that $C'(q) > 0$ and $\|C\| \leq 2^M \leq 2^{|Q|}$. □

We now prove the completeness of our algorithm by showing that every state coverable by P belongs to S.

Lemma 5. *If there exists $C \in \mathcal{I}$ and $C' \in \mathcal{C}$ such that $C \to^* C'$ and $C'(q) > 0$, then $q \in S$.*

Proof. We consider the initialized execution $C_0 \xrightarrow{t_1} C_1 \xrightarrow{t_2} \ldots \xrightarrow{t_n} C_n$ with $C = C_0$ and $C_n = C'$. We will prove by induction on $0 \leq i \leq n$ that for all q such that $C_i(q) > 0$, we have $q \in S_i$.

For $i = 0$, we have $C_0 = \wr \|C\| \cdot q_{in} \wr$, and $S_0 = \{q_0\}$. Hence the property holds.

Assume the property to be true for $i < n$, and let $q \in Q$ such that $C_{i+1}(q) > 0$. If $C_i(q) > 0$, then by induction hypothesis we have $q \in S_i$ and since $S_i \subseteq S_{i+1}$, we deduce that $q \in S_{i+1}$. Assume now that $C_i(q) = 0$. We proceed by a case analysis.

1. $t_{i+1} = (q', !a, q)$ or $t_{i+1} = (q', !!a, q)$ for some $a \in \Sigma$ and $q' \in Q$. Since $C_i \xrightarrow{t_{i+1}} C_{i+1}$, we have necessarily $C_i(q') > 0$. By induction hypothesis, $q' \in S_i$, and by construction of S_{i+1}, we deduce that $q' \in S_{i+1}$.
2. $t_{i+1} = (q_1, !a, q_1')$ or $t_{i+1} = (q_1, !!a, q_1')$ with $q_1' \neq q$. Since $C_i(q) = 0$ and $C_{i+1}(q) > 0$, there exists a transition of the form $(q_2, ?a, q)$ with $q_1 \neq q_2$ (because $q_1 \in Q_A$ and $(q_2, ?a, q) \in T$ hence $q_2 \notin Q_W$). Consequently, we know that we have $C_i(q_1) > 0$ and $C_i(q_2) > 0$. By induction hypothesis q_1, q_2 belong to S_i and by construction of S_{i+1} we deduce that $q \in S_{i+1}$.

□

The two previous lemmas show the soundness and completeness of our algorithm to solve StateCover based on the computation of the set S. Since this set of states can be computed in polynomial time, we obtain the following result.

Theorem 1. *StateCover is in P for Wait-Only protocols.*

Furthermore, completeness of the algorithm along with the bound on the number of processes established in Lemma 4 gives the following result.

Corollary 1. *Given a Wait-Only protocol $P = (Q, \Sigma, q_{in}, T)$, for all $q \in Q$ coverable by P, then min_q the minimal number of processes necessary to cover q is at most $2^{|Q|}$.*

4.2 Lower Bound

We show that StateCover for Wait-Only protocols is P-hard. For this, we provide a reduction from the Circuit Value Problem (CVP) which is known to be P-complete [Lad75]. CVP is defined as follows: given an acyclic Boolean circuit with n input variables, one output variable, m boolean gates of type *and*, *or*, *not*, and a truth assignment for the input variables, is the value of the output equal to a given boolean value? Given an instance of the CVP, we build a protocol in which the processes broadcast variables (input ones or associated with gates) along with their boolean values. These broadcasts will be received by other processes that will use them to compute boolean value of their corresponding gate, and broadcast the obtained value. Hence, different values are propagated through the protocol representing the circuit, until the state representing the output variable value we look for is covered.

Take for example a CVP instance C with two variables v_1, v_2, and two gates: one *not* gate on variable v_1 denoted $g_1(v_1, \neg, o_1)$ (where o_1 stands for the output variable of g_1), and one *or* gate on variable v_2 and o_1 denoted $g_2(o_1, v_2, \vee, o_2)$ (where o_2 stands for the output variable of gate g_2). Assume the input boolean value for v_1 [resp. v_2] is $\top$ [resp. $\bot$]. The protocol associated to C is displayed

on Figure 2. Assume the output value of C is o_2, we will show that $q^2_\top$ [resp. $q^2_\bot$] is coverable if and only if o_2 evaluates to $\top$ [resp. $\bot$]. Note that with the truth assignment depicted earlier, o_2 evaluates to $\bot$, and indeed one can build an execution covering $q^2_\bot$ with three processes:

$$\wr 3.q_{in}\wr \to \wr 2.q_{in}, q^2_0\wr \to \wr q_{in}, q^1_0, q^2_0\wr \xrightarrow{(q_{in},!!(v_1,\top),q_{in})} \wr q_{in}, q^1_\bot, q^2_0\wr$$
$$\xrightarrow{(q^1_\bot,!!(o_1,\bot),q^1_\bot)} \wr q_{in}, q^1_\bot, q^2_1\wr \xrightarrow{(q_{in},!!(v_2,\bot),q_{in})} \wr q_{in}, q^1_\bot, q^2_\bot\wr$$

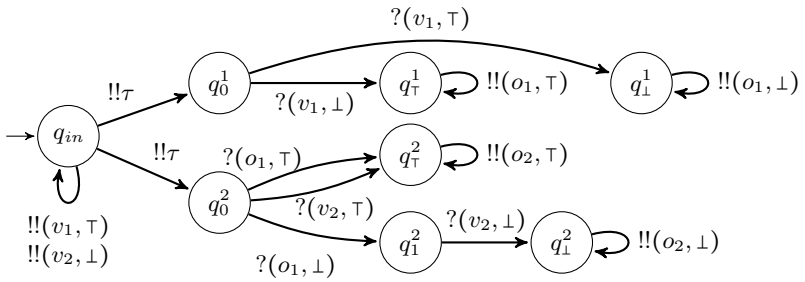

Fig. 2: Protocol for a CVP instance with two variables v_1, v_2, two gates $g_1(\neg, v_1, o_1)$ and $g_2(\vee, o_1, v_2, o_2)$, and input $\top$ for v_1 and $\bot$ for v_2 and output variable o_2. Depending on the truth value of o_2 to test, the state we ask to cover can be $q^2_\bot$ or $q^2_\top$.

Formal proofs of the reduction are given in Appendix B. Together with Theorem 1, we get the following theorem.

Theorem 2. *StateCover for Wait-Only protocols is P-complete.*

This reduction can be adapted to Wait-Only NB-Rendez-vous protocols, which leads to the following theorem, proving that the upper bound presented in [GSS23] is tight.

Theorem 3. *StateCover for Wait-Only NB-Rendez-vous protocols is P-hard.*

5 ConfCover for Wait-Only protocols is PSpace-complete

We present here an algorithm to solve the configuration coverability problem for Wait-Only protocols in polynomial space.

5.1 Main ideas

For the remaining of the section, we fix a Wait-Only protocol $P = (Q, \Sigma, q_{in}, T)$ and a configuration $C_f \in \mathcal{C}$ to cover, and we let $K = \|C_f\|$. The intuition is

the following: we (only) keep track of the K processes that will cover C_f. Of course, they might need other processes to reach the desired configuration, if they need to receive messages. That is why we also maintain the set of reachable states along the execution. An abstract configuration will then be a multiset of K states (concrete part of the configuration) and a set of all the reachable states (abstract part of the configuration). Lemma 1 ensures that it is enough to know which active states are reachable to ensure that both the concrete part and the active states of the abstract part are coverable at the same time. However, there is a case where this abstraction would not be enough: assume that one of the K processes has to send a message, and this message *should not* be received by the other $K-1$ processes. This can happen when the message is received by a process in the part of the configuration that we have abstracted away. In that case, even if the (waiting) state is present in the set of reachable states, Lemma 1 does not guarantee that the entire configuration is reachable, so the transition to an abstract configuration where none of the $K-1$ processes has received the message might be erroneous. This is why in that case we need to precisely keep track of the process that will receive the message, even if in the end it will not participate in the covering of C_f. This leads to the definition of the $\Rightarrow_{\mathsf{switch}}$ transition below.

This proof is structured as follows: we present the formal definitions of the abstract configurations and semantics in Section 5.2. In Section 5.3, we present the completeness proof, Section 5.4 is devoted to prove the soundness of the construction. In the latter, we also give some ingredients to prove an upper bound on the number of processes needed to cover the configuration. In Section 5.5 one can find the main theorem of this section: it states that the ConfCover problem is in PSpace and if the configuration is indeed coverable, it presents an upper bound on the number of processes needed to cover it. In Section 5.6, we prove that this lower bound is tight as the problem is PSpace-hard.

5.2 Reasoning with Abstract Configurations

We present the abstract configurations we rely on. Let us fix $K = \|C_f\|$. An *abstract configuration* γ is a pair (M, S) where M is a configuration in $\mathcal{C}$ such that $\|M\| = K$ and $S \subseteq Q$ is a subset of control states such that $\{q \in Q \mid M(q) > 0\} \subseteq S$. We call M the M-part of γ and S its S-part. We denote by Γ the set of abstract configurations and by γ_{in} the initial abstract configuration $\gamma_{in} = (\{\!|K \cdot q_{in}|\!\}, \{q_{in}\})$. An abstract configuration $\gamma = (M, S)$ represents a set of configurations $[\![\gamma]\!] = \{C \in \mathcal{C} \mid M \preceq C \text{ and } C(q) > 0 \text{ implies } q \in S\}$. Hence in $[\![\gamma]\!]$, we have all the configurations C that are bigger than M as long as the states holding processes in C are stored in S (observe that this implies that all the states in M appear in S).

We now define an abstract transition relation for abstract configurations. For this matter, we define three transition relations $\Rightarrow_{\mathsf{step}}$,$\Rightarrow_{\mathsf{ext}}$ and $\Rightarrow_{\mathsf{switch}}$ and let $\Rightarrow$ be defined by $\Rightarrow_{\mathsf{step}} \cup \Rightarrow_{\mathsf{ext}} \cup \Rightarrow_{\mathsf{switch}}$. Let $\gamma = (M, S)$ and $\gamma' = (M', S')$ be two abstract configurations and $t = (q, \alpha, q')$ be a transition in T with $\alpha = !a$ or

$\alpha = !!a$. For $\kappa \in \{\mathsf{step}, \mathsf{ext}, \mathsf{switch}\}$, we have $\gamma \overset{t}{\Rightarrow}_\kappa \gamma'$ iff all the following conditions hold:

- $S \subseteq S'$, and,
- for all $p \in S' \setminus S$, either $p = q'$ or there exist $p' \in S$ and $(p', ?a, p)$ in T, and,
- one of the following cases is true:
  - $\kappa = \mathsf{step}$ and $M \xrightarrow{t} M'$. This relation describes a message emitted from the M-part of the configuration;
  - $\kappa = \mathsf{ext}$ and $q \in S$ and $M + \wr q \int \xrightarrow{t} M' + \wr q' \int$;
  - $\kappa = \mathsf{ext}$ and there exists $(p, ?a, p')$ in T such that $q, p \in S$ and $M + \wr q, p \int \xrightarrow{t} M + \wr q', p' \int$ (note that in that case $M = M'$). The relation ext hence describes a message emitted from the S-part of the configuration;
  - $\kappa = \mathsf{switch}$ and $q \in S$ and $\alpha = !a$ and there exists $t' = (p, ?a, p') \in T$ such that $\wr p \int \preceq M$ and $\wr q' \int \preceq M'$ and $M - \wr p \int = M' - \wr q' \int$ and $M + \wr q \int \xrightarrow{t} M' + \wr p' \int$. This relation describes a sending from a state in the S-part of the abstract configuration leading to a rendez-vous with one process in the M-part, and a "switch" of processes: we remove the receiver process of the M-part and replace it by the sender.

Note that in any case, q, the state from which the message is sent, belongs to S. We then write $\gamma \overset{t}{\Longrightarrow} \gamma'$ whenever $\gamma \overset{t}{\Rightarrow}_\kappa \gamma'$ for $\kappa \in \{\mathsf{step}, \mathsf{ext}, \mathsf{switch}\}$ and we do not always specify the used transition t (when omitted, it means that there exists a transitions allowing the transition). We denote by $\Rightarrow^*$ the reflexive and transitive closure of $\Rightarrow$.

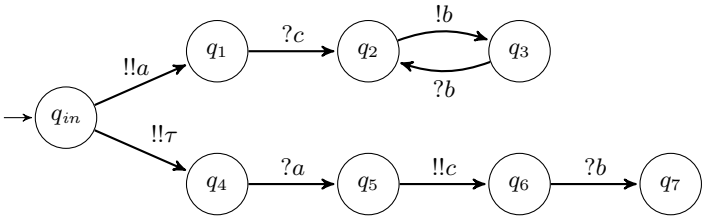

Fig. 3: A Wait-Only protocol P'.

Example 5. We consider the Wait-Only protocol P' depicted on Figure 3 with set of states Q'. We want to cover $C_f = \wr q_3, q_3, q_6 \int$ (hence $K = 3$). In this example, the abstract configuration $\gamma = (\wr q_2, q_2, q_4 \int, \{q_{in}, q_1, q_2, q_4, q_5, q_6\})$ represents all the configurations of P' with at least two processes on q_2 and one on q_4, and no process on q_3 nor q_7.

Considering the following abstract execution, we can cover C_f:

$$\gamma_{in} \xRightarrow[\mathsf{step}]{(q_{in},!!\tau,q_4)} (\wr q_4, q_{in}, q_{in}\int, \{q_{in}, q_4\}) \xRightarrow[\mathsf{step}]{(q_{in},!!\tau,q_4)} (\wr q_4, q_4, q_{in}\int, \{q_{in}, q_4\})$$
$$\xRightarrow[\mathsf{step}]{(q_{in},!!a,q_1)} (\wr q_5, q_5, q_1\int, \{q_{in}, q_1, q_4, q_5\}) \xRightarrow[\mathsf{ext}]{(q_5,!!c,q_6)} (\wr q_5, q_5, q_2\int, Q' \setminus \{q_3, q_7\})$$
$$\xRightarrow[\mathsf{step}]{(q_2,!b,q_3)} (\wr q_5, q_5, q_3\int, Q') \xRightarrow[\mathsf{step}]{(q_5,!!c,q_6)} (\wr q_6, q_5, q_3\int, Q')$$
$$\xRightarrow[\mathsf{switch}]{(q_2,!b,q_3)} (\wr q_3, q_5, q_3\int, Q') \xRightarrow[\mathsf{step}]{(q_5,!!c,q_6)} (\wr q_3, q_6, q_3\int, Q')$$

It corresponds for instance to the following concrete execution:

$$\wr q_{in}, q_{in}, q_{in}\int + \wr q_{in}, q_{in}\int \to^+ \wr q_4, q_4, q_{in}\int + \wr q_4, q_{in}\int \xrightarrow{(q_{in},!!a,q_1)} \wr q_5, q_5, q_1\int + \wr q_5, q_{in}\int$$
$$\xrightarrow{(q_{in},!!a,q_1)} \wr q_5, q_5, q_1\int + \wr q_5, q_1\int \xrightarrow{(q_5,!!c,q_6)} \wr q_5, q_5, q_2\int + \wr q_6, q_1\int$$
$$\xrightarrow{(q_2,!b,q_3)} \wr q_5, q_5, q_3\int + \wr q_7, q_1\int \xrightarrow{(q_5,!!c,q_6)} \wr q_6, q_5, q_3\int + \wr q_7, q_2\int$$
$$\xrightarrow{(q_2,!b,q_3)} \wr q_3, q_5, q_3\int + \wr q_7, q_7\int \xrightarrow{(q_5,!!c,q_6)} \wr q_3, q_6, q_3\int + \wr q_7, q_7\int$$

The M-part of our abstract configuration $\wr q_6, q_5, q_3\int$ reached just before the $\Rightarrow_{\mathsf{switch}}$ transition does not correspond to the set of processes that finally cover C_f, at this point of time, the processes that will finally cover C_f are in states q_2, q_5, and q_3. But here we ensure that the process on q_6 will actually receive the b sent by the process on q_2, leaving the process on q_3 in its state. Once this has been ensured, process on q_6 is not useful anymore, and instead we follow the process that was on q_2 before the sending, hence the $\Rightarrow_{\mathsf{switch}}$ transition.

The algorithm used to solve ConfCover, is then to seek in the directed graph $(\Gamma, \Rightarrow)$ if a vertex of the form (C_f, S) is reachable from γ_{in}.

Before proving that this algorithm is correct, we establish the following property.

Lemma 6. *Let (M, S) and (M', S') be two abstract configurations and $\widetilde{S} \subseteq Q$ such that $S \subseteq \widetilde{S}$. We have:*

1. $[\![(M, S)]\!] \subseteq [\![(M, \widetilde{S})]\!]$.
2. *If $(M, S) \Rightarrow (M', S')$ then there exists $S'' \subseteq Q$ such that $(M, \widetilde{S}) \Rightarrow (M', S'')$ and $S' \subseteq S''$.*

Proof. The first point is a direct consequence of the definition of $[\![\,]\!]$. For the second point, it is enough to take $S'' = S' \cup \widetilde{S}$ and apply the definition of $\Rightarrow$. □

5.3 Completeness of the algorithm

In this subsection we show that if C_f can be covered then there exists an abstract configuration $\gamma = (C_f, S)$ such that $\gamma_{in} \Rightarrow^* \gamma$. We use $\mathcal{C}_{\geq K}$ to represent the set

$\{C \in \mathcal{C} \mid \|C\| \geq K\}$ of configurations with at least K processes and $\mathcal{C}_{=K}$ the set $\{C \in \mathcal{C} \mid \|C\| = K\}$ of configurations with exactly K processes. This first lemma shows the completeness for a single step of our abstract transition relation (note that we focus on the M-part, as it is the one witnessing C_f in the end).

Lemma 7. *Let $C, C' \in \mathcal{C}_{\geq K}$ and $t \in T$ such that $C \xrightarrow{t} C'$. Then for all $M' \in \mathcal{C}_{=K}$ such that $M' \preceq C'$, there exists $M \in \mathcal{C}_{=K}$ and $S' \subseteq Q$ such that $(M, S) \Rightarrow (M', S')$ with $S = \{q \in Q \mid C(q) > 0\}$, $C \in [\![(M, S)]\!]$ and $C' \in [\![(M', S')]\!]$.*

Proof. Let $M' \in \mathcal{C}_{=K}$ such that $M' \preceq C'$. We assume that $t = (q, \alpha, q')$ with $\alpha \in \{!a, !!a\}$. We let $S = \{p \in Q \mid C(p) > 0\}$ and $S' = S \cup \{p \in Q \mid C'(p) > 0\}$. By definition of S and S', for all $p \in S' \setminus S$, either $p = q'$ or there exists $p' \in S$ and $(p', ?a, p)$ in T. In fact, let $p \in S' \setminus S$ such that $p \neq q'$. Since $C \xrightarrow{t} C'$, we have necessarily that there exist $p' \in Q$ such that $C(p') > 0$ (hence $p' \in S$), and $(p', ?a, p)$ in T. We now reason by a case analysis to determine $M \in \mathcal{C}_{=K}$ such that $(M, S) \Rightarrow (M', S')$ and $C \in [\![(M, S)]\!]$. The different cases are: (i) $\alpha = !!a$, (ii) $\alpha = !a$ and the message is *not received*, (iii) $\alpha = !a$ and the message is received by a process. Because of space constraints, we present here only case (iii) as it exhibits the most different abstract behaviours. The two other cases can be found on Appendix C.1.

(iii) if $\alpha = !a$ and the message is received by a process (i.e. it is a rendez-vous), denote by $(p, ?a, p')$ the reception transition issued between C and C'. Using the definition of $\rightarrow$, we get $C' = C - \wr q, p\wr + \wr q', p'\wr$. We consider the four following disjoint cases:
- $M'(q') = 0$ and $M'(p') = 0$. Since $\wr q', p'\wr \preceq C'$ and $M' \preceq C'$, we get that $C' = M' + \wr q', p'\wr + M_2$ for some multiset M_2. We deduce that $C = M' + \wr q, p\wr + M_2$. This allows us to deduce that $M' \preceq C$ and consequently $C \in [\![(M', S)]\!]$. Moreover, $M' + \wr p, q\wr \xrightarrow{t} M' + \wr q', p'\wr$ and $q, p \in S$. Hence $(M', S) \overset{t}{\Rightarrow}_{\mathsf{ext}} (M', S')$.
- $M'(q') = 0$ and $M'(p') > 0$. In that case $C' = M' + \wr q'\wr + M_2$ for some multiset M_2. Let $M = M' - \wr p'\wr + \wr p\wr$. We have then $C = C' + \wr q, p\wr - \wr q', p'\wr = M' + \wr q'\wr + M_2 + \wr q, p\wr - \wr q', p'\wr = M' + M_2 - \wr p'\wr + \wr p\wr + \wr q\wr = M + \wr q\wr + M_2$. This allows us to deduce that $M \preceq C$ and consequently $C \in [\![(M, S)]\!]$. Furthermore $q \in S$ and $M + \wr q\wr \xrightarrow{t} M' + \wr q'\wr$. Hence $(M, S) \overset{t}{\Rightarrow}_{\mathsf{ext}} (M, S')$.
- $M'(q') > 0$ and $M'(p') = 0$. In that case $C' = M' + \wr p'\wr + M_2$ for some multiset M_2. Let $M = M' - \wr q'\wr + \wr p\wr$. We have then $C = C' + \wr q, p\wr - \wr q', p'\wr = M' + \wr p'\wr + M_2 + \wr q, p\wr - \wr q', p'\wr = M' + M_2 - \wr q'\wr + \wr p\wr + \wr q\wr = M + \wr q\wr + M_2$. This allows us to deduce that $q \in S$ and $M \preceq C$ and consequently $C \in [\![(M, S)]\!]$. We also have $\wr p\wr \preceq M$ and $\wr q'\wr \preceq M'$ and $M - \wr p\wr = M' - \wr q'\wr$ and $M + \wr q\wr \xrightarrow{t} M' + \wr p'\wr$. Hence $(M, S) \overset{t}{\Rightarrow}_{\mathsf{switch}} (M', S')$. Observe that we need to use the $\Rightarrow_{\mathsf{switch}}$ transition relation in this case. Assume that $C(s) > 0$ for some state $s \in S$ such that $(s, ?a, s') \in T$, and that any configuration in $\mathcal{C}_{=K}$ such that $M \preceq C$ contains such state s. Then, applying $\Rightarrow_{\mathsf{step}}$ to such a multiset M will take away the process on state s and will lead to an abstract configuration with $M' \not\preceq C'$.

– $M'(q') > 0$ and $M'(p') > 0$. In that case $C' = M' + M_2$ for some multiset M_2. Let $M = M' - \wr p', q' \wr + \wr p, q \wr$. We have then $C = C' + \wr q, p \wr - \wr q', p' \wr = M + M_2$. This allows us to deduce that $M \preceq C$ and consequently $C \in [\![(M, S)]\!]$ and that $M \xrightarrow{t} M'$. Hence $(M, S) \overset{t}{\Rightarrow}_{\mathsf{step}} (M', S')$. □

The two previous lemmas allow us to establish completeness of the construction, by a simple induction on the length of the considered execution. The proof can be found on Appendix C.1.

Lemma 8. *Let $C_{in} \in \mathcal{I}$ and $C \in \mathcal{C}_{\geq K}$ such that $C_{in} \to^* C$. For all $M \in \mathcal{C}_{=K}$ such that $M \preceq C$ there exists $S \subseteq Q$ such that $C \in [\![(M, S)]\!]$ and $\gamma_{in} \Rightarrow^* (M, S)$.*

5.4 Soundness of the algorithm

We now prove that if we have $\gamma_{in} \Rightarrow^* (M, S)$ then the configuration M can be covered. We first establish that the S-part of a reachable abstract configuration stores only states that are reachable in a concrete execution.

Lemma 9. *If $\gamma = (M, S)$ is an abstract configuration such that $\gamma_{in} \Rightarrow^* \gamma$, then all states $q \in S$ are coverable.*

Proof. We suppose that we have $\gamma_{in} = \gamma_0 \Rightarrow \gamma_1 \Rightarrow \ldots \Rightarrow \gamma_n = (M, S)$ and we prove this lemma by induction on n, the length of the abstract execution.

Case $n = 0$**:** In that case $(M, S) = \gamma_0 = (K \cdot \wr q_{in} \wr, \{q_{in}\})$, as q_{in} is trivially coverable, the property holds.

Case $n > 0$**:** We assume that the property holds for all $0 \leq m < n$ and consider the abstract execution $\gamma_0 \overset{t_1}{\Longrightarrow} \gamma_1 \overset{t_2}{\Longrightarrow} \ldots \overset{t_n}{\Longrightarrow} \gamma_n$ where $\gamma_0 = \gamma_{in}$ and $\gamma_i = (M_i, S_i)$ for all $0 \leq i \leq n$. Let $p \in S_n$. If $p \in S_{n-1}$, then by induction hypothesis, p is coverable. Otherwise, $p \in S_n \setminus S_{n-1}$, and let $t_n = (q, \alpha, q')$ with $\alpha \in \{!a, !!a \mid a \in \Sigma\}$. By definition of $\Rightarrow$, $q \in S_{n-1}$ and

– either $q' = p$, and by induction hypothesis, there exists an initialized execution $C_0 \to^* C$ with $C(q) > 0$ and in that case, $C_0 \to^* C \xrightarrow{t} C'$ for a configuration C' such that $C'(p) > 0$ and p is coverable.
– or $\alpha \in \{!a, !!a\}$ for some $a \in \Sigma$ and $(p', ?a, p) \in T$, with $p' \in S_{n-1}$. By induction hypothesis, both q and p' are coverable, with $q \in Q_A$ and $p \in Q_W$. By Lemma 1, there exists an execution $C_0 \to^* C$ such that $C(q) \geq 1$ and $C(p') \geq 1$. We then have $C \xrightarrow{t_n} C'$ with $C'(p) > 0$ (if $\alpha = !a$ then the process on p' can receive the message a and move to p, and if $\alpha = !!a$ then the process on p' will necessary receive the broadcast and move to p), and p is coverable.

□

The next lemma establishes soundness of the algorithm. Moreover, it gives an upper bound on the minimal number of processes needed to cover a configuration.

Lemma 10. *Let (M,S) be an abstract configuration such that $\gamma_{in} \Rightarrow \gamma_1 \Rightarrow \ldots \Rightarrow \gamma_n = (M,S)$. Then, there exist $C_{in} \in \mathcal{I}$, $C \in \mathcal{C}$ such that $M \preceq C$ and $C_{in} \rightarrow^* C$. Moreover, $\|C_{in}\| = \|C\| \leq K + 2^{|Q|} \times n$.*

Proof. We reason by induction on n, the length of the abstract execution.

Case $n = 0$**:** The property trivially holds for $C = \wr K \cdot q_{in} \int$.

Case $n > 0$**:** We assume that the property holds for all $0 \leq m < n$ and consider the abstract execution $\gamma_0 \stackrel{t_1}{\Longrightarrow} \gamma_1 \stackrel{t_2}{\Longrightarrow} \ldots \stackrel{t_n}{\Longrightarrow} \gamma_n$ where $\gamma_0 = \gamma_{in}$ and $\gamma_i = (M_i, S_i)$ for all $0 \leq i \leq n$. By induction hypothesis, we know that there exist $C_{in} \in \mathcal{I}, C_{n-1} \in \mathcal{C}$ such that $M_{n-1} \preceq C_{n-1}$ and $C_{in} \rightarrow^* C_{n-1}$ and $\|C_{in}\| = \|C_{n-1}\| \leq K + 2^{|Q|} \times (n-1)$. If $M_n = M_{n-1}$ then the property holds. Assume now that $M_n \neq M_{n-1}$. We let $t_n = (q, \alpha, q')$ with $\alpha = !a$ or $\alpha = !!a$. By definition of $\Rightarrow$, we know that $S_{n-1} \subseteq S_n$ and that $q \in S_{n-1}$. Thanks to Lemma 9, q is coverable. We now perform a case analysis:

- Assume $\gamma_{n-1} \stackrel{t_n}{\Rightarrow}_{\mathsf{step}} \gamma_n$. Then $M_{n-1} \xrightarrow{t_n} M_n$, and since $M_{n-1} \preceq C_{n-1}$, we have $C_{n-1} = M_{n-1} + M$ for some multiset M.
  - If $\alpha = !!a$, then $M_{n-1} + M = \wr q_1, \ldots q_{K-1}, q \int + \wr p_1, \ldots, p_L \int$ and $M_n = \wr q'_1, \ldots q'_{K-1}, q' \int$ where for all $1 \leq i \leq K-1$, either $(q_i, ?a, q'_i) \in T$ or $a \notin R(q_i)$ and $q_i = q'_i$. For each $1 \leq i \leq L$, define p'_i as $p'_i = p_i$ if $a \notin R(p_i)$ or p'_i is such that $(p_i, ?a, p'_i) \in T$. If we let $M' = \wr p'_1, \ldots, p'_L \int$ and $C_n = M_n + M'$, we have by definition that $C_{n-1} \xrightarrow{t_n} C_n$ with $M_n \preceq C_n$.
  - If $\alpha = !a$ and $(M_{n-1} - \wr q \int)(p) > 0$ for some $p \in Q$ such that $(p, ?a, p') \in T$ (i.e., a rendez-vous occurred), it holds that $M_{n-1} + M \xrightarrow{t} M_n + M$ and we choose $C_n = M_n + M$.
  - If $\alpha = !a$ and $(M_{n-1} - \wr q \int)(p) = 0$ for all $p \in Q$ such that $a \in R(p)$ (i.e. it was a non-blocking sending of a message), then either there exists $(p, ?a, p') \in T$ such that $M(p) > 0$, either for all $p \in Q$ such that $M(p) > 0$, $a \notin R(p)$. In the first case, a rendez-vous will occur in the execution of t_n over C_{n-1}, and we have $M_{n-1} + M \xrightarrow{t_n} M_n + M - \wr p \int + \wr p' \int$. We then let $C_n = M_n + M - \wr p \int + \wr p' \int$. In the latter case, $M_{n-1} + M \xrightarrow{t_n} M_n + M$ and with $C_n = M_n + M$. In both cases, we have , $C_{n-1} \xrightarrow{t} C_n$ and $M_n \preceq C_n$.

  In all cases, we have $C_{in} \rightarrow^* C_{n-1} \rightarrow C_n$ and $\|C_{in}\| = \|C_n\| = \|C_{n-1}\| \leq K + 2^{|Q|} \times (n-1) \leq K + 2^{|Q|} \times n$.
- Assume $\gamma_{n-1} \stackrel{t_n}{\Rightarrow}_{\mathsf{ext}} \gamma_n$ or $\gamma_{n-1} \stackrel{t_n}{\Rightarrow}_{\mathsf{switch}} \gamma_n$. As q is coverable, from Corollary 1, there exists an execution $C^q_{in} \rightarrow^* C^q$ such that $C^q_{in} \in \mathcal{I}$ and $C^q(q) > 0$ and $\|C^q_{in}\| \leq 2^{|Q|}$. From Lemma 2, we have the following execution: $C_{in} + C^q_{in} \rightarrow^* C_{in} + C^q$. Next, from Lemma 3, by taking $M = 1$ and $\tilde{C}_0 = C_{in} + C^q$, we have an execution $C_{in} + C^q \rightarrow^* C_{n-1} + C'^q$ where $C'^q(q) > 0$. We deduce that $C_{n-1} + C'^q = M_{n-1} + \wr q \int + M$ for some multiset M. We now proceed with a case analysis:
  - Case $\gamma_{n-1} \stackrel{t_n}{\Rightarrow}_{\mathsf{ext}} \gamma_n$ where $M_{n-1} + \wr q \int \xrightarrow{t_n} M_n + \wr q' \int$. By definition of $\rightarrow$, we also have $M_{n-1} + \wr q \int + M \rightarrow M_n + \wr q' \int + M'$ for some multiset M'. Letting $C_n = M_n + \wr q' \int + M'$ gives us that that $C_{in} + C^q_{in} \rightarrow^* M_{n-1} + \wr q \int + M \rightarrow C_n$.

- Case $\gamma_{n-1} \overset{t_n}{\Rightarrow}_{\mathsf{switch}} \gamma_n$: by definition of $\Rightarrow_{\mathsf{switch}}$, we know that there exists $(p, ?a, p') \in T$ with $p \in S_{n-1}$ such that $M_{n-1} = M' + \wr p \wr$ and that $M' + \wr p \wr + \wr q \wr \xrightarrow{t_n} M' + \wr q' \wr + \wr p' \wr$ and $M_n = M' + \wr q' \wr$. Furthermore, by definition of $\rightarrow$, we have $M' + \wr p, q \wr + M \rightarrow M' + \wr p', q' \wr + M$. Hence setting $C_n = M' + \wr p', q' \wr + M = M_n + \wr p' \wr + M$ gives us that that $C_{in} + C_{in}^q \rightarrow^* M_{n-1} + \wr q \wr + M \rightarrow C_n$, with $M_n \preceq C_n$.

In both cases, we have shown that there exists C_n such that $M_n \preceq C_n$ and $C_{in} + C_{in}^q \rightarrow^* C_n$. Furthermore we have that $\|C_n\| = \|C_{in} + C_{in}^q\| \leq K + 2^{|Q|} \times (n-1) + 2^{|Q|} \leq K + 2^{|Q|} \times n$. □

5.5 Upper Bound

Using Lemmas 8 and 10, we know that there exists $C \in \mathcal{I}$ and $C' \in \mathcal{C}$ such that $C \rightarrow^* C'$ and $C_f \preceq C'$ iff there exists an abstract execution $\gamma_{in} \Rightarrow \gamma_1 \Rightarrow \cdots \Rightarrow \gamma_n$ with $\gamma_n = (C_f, S)$ for some $S \subseteq Q$, hence the algorithm consisting in deciding reachability of a vertex of the form (C_f, S) from γ_{in} in the finite graph $(\Gamma, \Rightarrow)$ is correct. Note furthermore that the number of abstract configurations $|\Gamma|$ is bounded by $|Q|^{\|C_f\|} \times 2^{|Q|}$. As the reachability of a vertex in a graph is NL-complete, this gives us a NPSpace procedure, which leads to a Pspace procedure thanks to Savitch's theorem.

Theorem 4. *ConfCover is in PSpace.*

Remark 3. Thanks to Lemma 10, we know that $\gamma_{in} \Rightarrow \gamma_1 \Rightarrow \cdots \Rightarrow \gamma_n$ with $\gamma_n = (C_f, S)$ iff there exists $C_{in} \in \mathcal{I}$, $C \in \mathcal{C}$ such that $C_f \preceq C$ and $C_{in} \rightarrow^* C$ and $\|C_{in}\| = \|C\| \leq K + 2^{|Q|} \times n$. But due to the number of abstract configurations, we can assume that $n \leq 2^{|Q|} \times |Q|^{\|C_f\|}$ as it is unnecessary in the abstract execution $\gamma_{in} \Rightarrow \gamma_1 \Rightarrow \cdots \Rightarrow \gamma_n$ to visit twice the same abstract configuration. Hence the configuration C_f is coverable iff there is $C \in \mathcal{I}$ and $C' \in \mathcal{C}$ such $C \rightarrow^* C'$ and $C_f \preceq C'$ and $\|C\| = \|C'\| \leq K + 2^{|Q|} \times 2^{|Q|} \times |Q|^{\|C_f\|}$.

5.6 Lower Bound

To prove PSpace-hardness of the ConfCover problem for Wait-Only protocols, we reduce the intersection non-emptiness problem for deterministic finite automata, which is known to be PSpace-complete [Koz77]. The PSpace-hardness in fact holds when considering Wait-Only protocols without any (non-blocking) rendez-vous transitions, i.e. transitions of the form $(q, !a, q')$.

Let $\mathcal{A}_1, \ldots, \mathcal{A}_n$ be a list of deterministic finite and *complete* automata with $\mathcal{A}_i = (\Sigma, Q_i, q_i^0, \{q_i^f\}, \Delta_i)$ for all $1 \leq i \leq n$. Observe that we restrict our reduction to automata with a unique accepting state, which does not change the complexity of the problem. We note Σ^* the set of words over the finite alphabet Σ and Δ_i^* the function extending Δ_i to Σ^*, i.e, for all $q \in Q_i$, $\Delta_i^*(q, \varepsilon) = q$, and for all $w \in \Sigma^*$ and $a \in \Sigma$, $\Delta_i^*(q, wa) = \Delta_i(\Delta_i^*(q, w), a)$.

We build the protocol P with set of states Q, displayed in Figure 4 where P_i for $1 \leq i \leq n$ is a protocol mimicking the behaviour of the automaton $\mathcal{A}_i$: $P_i =$

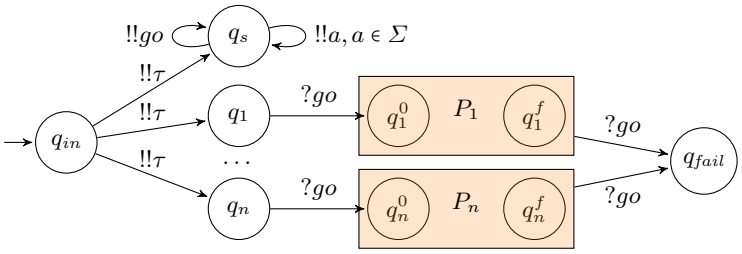

Fig. 4: Protocol P for PSpace-hardness of ConfCover.

$(Q_i, \Sigma, q_i^0, T_i)$, with $T_i = \{(q, ?a, q') \mid (q, a, q') \in \Delta_i\}$. Moreover, from any state $q \in \bigcup_{1 \leq i \leq n} Q_i$, there is an outgoing transition $(q, ?go, q_{fail})$. These transitions are depicted by the outgoing transitions labelled by $?go$ from the orange rectangles.

Note that P is Wait-Only as all states in P_i for all $1 \leq i \leq n$ are waiting states and the only active states are q_{in} and q_s. We show that $\bigcap_{1 \leq i \leq n} L(\mathcal{A}_i) \neq \varnothing$ if and only if there is an initial configuration $C \in \mathcal{I}_P$ and a configuration $C' \in \mathcal{C}_P$ such that $C \to^* C'$ and $C_f \preceq C'$ with $C_f = \{\!\{q_1^f, \ldots, q_n^f\}\!\}$.

The idea is to synchronize (at least) n processes into simulating the n automata. To this end, we need an additional (leader) process that will broadcast a message *go*, which will be received by the n processes, leading each of them to reach a different automaton initial state. Then, the leader process will broadcast a word letter by letter. Since the automata are all complete, these broadcast will be received by all the processes that simulate the automata, mimicking an execution. If the word belongs to all the automata languages, then each process simulating the automata ends the simulation on the unique final state of the automaton. Note that if the leader process broadcasts the message *go* a second time, then all the processes simulating the automata stop their simulation and reach the state q_{fail}. Formal proofs can be found in Appendix C.2.

6 Conclusion

We have proved that when extending the model presented in [GSS23] with broadcasts, StateCover for Wait-Only protocols remains in P, and is even P-complete. We also explained how to retrieve a P lower bound for the model of [GSS23] for both problems restricted to Wait-Only protocols. However ConfCover for Wait-Only protocols is now PSpace-complete. In the future, we wish to study not only coverability problems but extend the analysis of this model to liveness properties. We also wish to expand this model with dynamic creations of messages and processes in order to take a step closer to the modelling of Java Threads programming, where Threads can dynamically create new objects in which they can synchronize with notify and notifyAll messages.

References

AK86. K. R. Apt and D. C. Kozen. Limits for automatic verification of finite-state concurrent systems. *Inf. Process. Lett.*, 22(6):307–309, 1986.

ARZ15. B. Aminof, S. Rubin, and F. Zuleger. On the expressive power of communication primitives in parameterised systems. In *LPAR'15*, volume 9450 of *Lecture Notes in Computer Science*, pages 313–328. Springer, 2015.

BER21. A. R. Balasubramanian, J. Esparza, and M. A. Raskin. Finding cut-offs in leaderless rendez-vous protocols is easy. In *FOSSACS'21*, volume 12650 of *LNCS*, pages 42–61. Springer, 2021.

DEGM17. A. Durand-Gasselin, J. Esparza, P. Ganty, and R. Majumdar. Model checking parameterized asynchronous shared-memory systems. *Formal Methods in System Design*, 50(2-3):140–167, 2017.

DRB02. G. Delzanno, J. F. Raskin, and L. Van Begin. Towards the automated verification of multithreaded java programs. In *TACAS'02*, volume 2280 of *LNCS*, pages 173–187. Springer, 2002.

DSTZ12. G. Delzanno, A. Sangnier, R. Traverso, and G. Zavattaro. On the complexity of parameterized reachability in reconfigurable broadcast networks. In *FSTTCS'12*, volume 18 of *LIPIcs*, pages 289–300. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2012.

EFM99. J. Esparza, A. Finkel, and R. Mayr. On the verification of broadcast protocols. In *LICS'99*, pages 352–359. IEEE Comp. Soc. Press, July 1999.

EK03. E. Allen Emerson and V. Kahlon. Model checking guarded protocols. In *(LICS 2003*, pages 361–370. IEEE, 2003.

GS92. S. M. German and A. P. Sistla. Reasoning about systems with many processes. *Journal of the ACM*, 39(3):675–735, 1992.

GSS23. L. Guillou, A. Sangnier, and N. Sznajder. Safety analysis of parameterised networks with non-blocking rendez-vous. In *CONCUR'23*, volume 279 of *LIPIcs*, pages 7:1–7:17. loss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.

HS20. F. Horn and A. Sangnier. Deciding the existence of cut-off in parameterized rendez-vous networks. In *CONCUR'20*, volume 171 of *LIPIcs*, pages 46:1–46:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

Koz77. Dexter Kozen. Lower bounds for natural proof systems. In *FOCS'77*, pages 254–266. IEEE Computer Society, 1977.

Lad75. Richard E. Ladner. The circuit value problem is log space complete for *P*. *SIGACT News*, 7(1):18–20, 1975.

SS13. S. Schmitz and P. Schnoebelen. The power of well-structured systems. In *CONCUR'13*, volume 8052 of *Lecture Notes in Computer Science*, pages 5–24. Springer, 2013.

A Proofs of Section 3

Proof of Lemma 2.

Formally, we can prove Lemma 2 by induction on $0 \leq i \leq k$. For $i = 0$, it comes from the definitions. Now assume that $\widehat{C}_i(q) = C_i(q)$ for all $q \in Q \setminus \{q_{in}\}$, and $\widehat{C}_i(q_{in}) = C_i(q_{in}) + C'_0(q_{in})$, and let $M = C'_0(q_{in})$.

- If $t'_{i+1} = (p_1, !!m, p_2)$, let $C_i = \wr p_1, q'_1, \ldots, q'_{N'-1} \int$ and $C_{i+1} = \wr p_2, p'_1, \ldots, p'_{N'-1} \int$. For all $1 \leq i \leq N' - 1$, either $m \notin R(q'_i)$, and $p'_i = q'_i$, or $(q'_i, ?m, p'_i) \in T$. By induction hypothesis, $\widehat{C}_i = \wr p_1, q'_1, \ldots, q'_{N'-1}, M \cdot q_{in} \int$. Since q_{in} is an active state, we know that $m \notin R(q_{in})$, hence $\widehat{C}_{i+1} = \wr p_2, p'_1, \ldots, p'_{N'-1}, M \cdot q_{in} \int$, and $\widehat{C}_{i+1}(q) = C_{i+1}(q)$ for all $q \in Q \setminus \{q_{in}\}$, and $\widehat{C}_{i+1}(q_{in}) = C_{i+1}(q_{in}) + C'_0(q_{in})$.
- If $t'_{i+1} = (p_1, !m, p_2)$ and there exist $q, q' \in Q$ such that $(q, ?m, q') \in T$ and $C_i(q) > 0$, then $C_i = \wr p_1, q, q'_1, \ldots q'_{N'-2} \int$ and $C_{i+1} = \wr p_2, q', q'_1, \ldots q'_{N'-2} \int$. By induction hypothesis, $\widehat{C}_i = \wr p_1, q, q'_1, \ldots q'_{N'-2}, M \cdot q_{in} \int$ and hence $\widehat{C}_{i+1} = \wr p_2, q', q'_1, \ldots q'_{N'-2}, M \cdot q_{in} \int$.
- If $t'_{i+1} = (p_1, !m, p_2)$ and for all $q, q' \in Q$ such that $(q, ?m, q') \in T$, $C_i(q) = 0$, then for all $q, q' \in Q$ such that $(q, ?m, q') \in T$, $\widehat{C}_i(q) = C_i(q) = 0$. Indeed, since q_{in} is an active state, $q_{in} \neq q$. Then $C_{i+1} = C_i - \wr p_1 \int + \wr p_2 \int$ and $\widehat{C}_{i+1} = \widehat{C}_i - \wr p_1 \int + \wr p_2 \int$. Hence, $\widehat{C}_{i+1}(q) = C_{i+1}(q)$ for all $q \in Q \setminus \{q_{in}\}$ and $\widehat{C}_{i+1}(q_{in}) = C_{i+1}(q_{in}) + C'_0(q_{in})$.

□

Proof of Lemma 3. Again, we can prove this property by induction on $0 \leq i \leq k$. For $i = 0$, it is obvious since $C_0(q) = 0$ for all $q \in Q \setminus \{q_{in}\}$. Let now $0 \leq i < k$ and assume that the property is true. Let $N_1 = C_0(q_{in})$. Then $\widetilde{C}_i \xrightarrow{t_{i+1}} \widetilde{C_{i+1}}$ is a transition that preserves the property. The proof of it depends on t_{i+1}:

- If $t_{i+1} = (p_1, !!a, p_2)$, then by induction hypothesis, $\widetilde{C}_i(p_1) \geq \widetilde{C_0}(p_1) + C_i(p_1)$ if $p_1 \neq q_{in}$, and $\widetilde{C}_i(p_1) \geq (M-1).N_1 + C_i(p_1)$ if $p_1 = q_{in}$. Moreover, $C_i(p_1) > 0$, hence the transition t_{i+1} can be taken from $\widetilde{C}_i$. Let $C_i = \wr p_1, q'_1, \ldots, q'_{N_1-1} \int$, then $C_{i+1} = \wr p_2, p'_1, \ldots, p'_{N-1} \int$ such that for all $1 \leq i \leq N$, either $a \notin R(q'_i)$ and $p'_i = q'_i$, or $(q'_i, ?a, p'_i) \in T$. Also, $\widetilde{C}_i = \wr p_1, q'_1, \ldots, q'_{N_1-1}, q''_1, \ldots, q''_K \int$ and $\widetilde{C_{i+1}} = \wr p_2, p'_1, \ldots, p'_{N_1-1}, p''_1, \ldots, p''_K \int$, with, for all $1 \leq i \leq K$, either $a \notin R(q''_i)$ and $p''_i = q''_i$, or $(q''_i, ?a, p''_i) \in T$. So it is obvious that for all $q' \in Q$, $\widetilde{C_{i+1}}(q') \geq C_{i+1}(q')$. Let now $q' \in Q_A$. Either $\widetilde{C_{i+1}}(q') = \widetilde{C}_i(q') + \ell$ for some $\ell \geq 0$, or $\widetilde{C_{i+1}}(q') = \widetilde{C}_i(q') - 1$ (and $q' = p_1$). In the first case, $C_{i+1}(q') = C_i(q') + \ell'$, with $\ell \geq \ell' \geq 0$. Indeed, let $\{q^1, \ldots, q^r\}$ be the set of states such that $\widetilde{C}_i(q^j) > 0$ and $(q^j, ?a, q')$ for all $1 \leq j \leq r$. For all $1 \leq j \leq r$, $C_i(q^j) \leq \widetilde{C}_i(q^j)$ by induction hypothesis, so $C_{i+1}(q') = C_i(q') + \ell'$ and $\widetilde{C_{i+1}}(q') = \widetilde{C}_i(q') + \ell$ with $\ell \geq \ell' \geq -1$ (observe that if $\ell' = -1$, it means that $q' = p_1$ and that no process has gone to p_1 when receiving a). Hence, if $q' \neq q_{in}$, by induction hypothesis, $\widetilde{C_{i+1}}(q') \geq \widetilde{C_0}(q') + C_i(q') + \ell = \widetilde{C_0}(q') + C_{i+1}(q') - \ell' + \ell \geq \widetilde{C_0}(q') + C_{i+1}(q')$. If otherwise $q' = q_{in}$, by induction hypothesis, $\widetilde{C_{i+1}}(q_{in}) = \widetilde{C}_i(q_{in}) + \ell \geq (M-1).N_1 + C_i(q_{in}) + \ell = (M-1).N_1 + C_{i+1}(q_{in}) - \ell' + \ell \geq (M-1).N_1 + C_{i+1}(q_{in})$.

In the second case, we have that $\widetilde{C_{i+1}}(p_1) = \widetilde{C}_i(p_1)-1$, so no process receiving message a goes to p_1. Then $C_{i+1}(p_1) = C_i(p_1) - 1$, because for all $q \in Q_W$ such that $(q, ?a, p_1) \in T$, $C''_{m,i}(q) = 0$, and $C''_{m,i}(q) \geq C_i(q)$, so no process in C_i can receive message a and go to p_1 neither. So, if $p_1 \neq q_{in}$, by induction hypothesis, $\widetilde{C_{i+1}}(p_1) \geq \widetilde{C_0}(p_1) + C_i(p_1) - 1 = \widetilde{C_0} + C_{i+1}(p_1)$, and if $p_1 = q_{in}$, $\widetilde{C_{i+1}}(q_{in}) \geq (M-1).N_1 + C_i(q_{in}) - 1 = (M-1).N_1 + C_{i+1}(q_{in})$ and we are done.

- Let $t_{i+1} = (p_1, !a, p_2)$ and there exist $p, p' \in Q$ such that $(p, ?m, p') \in T$ and $C_i(p) > 0$ and $C_i(p_1) > 0$. Then, $C_{i+1} = C_i - \wr p_1, p\wr + \wr p_2, p'\wr$. By induction hypothesis, if $p_1 \neq q_{in}$, $\widetilde{C}_i(p_1) \geq \widetilde{C_0}(p_1) + C_i(p_1)$, if $p_1 = q_{in}$, $\widetilde{C}_i(p_1) \geq (M-1).N_1 + C_i(p_1)$, and $\widetilde{C}_i(p) \geq C_i(p)$ hence $\widetilde{C}_i \xrightarrow{t_{i+1}} \widetilde{C_{i+1}}$. Moreover, $\widetilde{C_{i+1}} = \widetilde{C}_i - \wr p_1, p\wr + \wr p_2, p'\wr$. Hence, if $p_1 \neq q_{in}$, $\widetilde{C_{i+1}}(p_1) = \widetilde{C}_i(p_1) - 1 \geq \widetilde{C_0}(p_1) + C_i(p_1) - 1 = \widetilde{C_0}(p_1) + C_{i+1}(p_1)$, and if $p_1 = q_{in}$, $\widetilde{C_{i+1}}(p_1) = \widetilde{C}_i(p_1) - 1 \geq (M-1).N_1 + C_i(p_1) - 1 = (M-1).N_1 + C_{i+1}(p_1)$. Now, by induction hypothesis, we also have that $\widetilde{C}_i(p) \geq C_i(p)$. Hence, $\widetilde{C_{i+1}}(p) = \widetilde{C}_i(p) - 1 \geq C_i(p) - 1 = C_{i+1}(p)$. Let any other $q \in Q_A$. If $q \notin \{p_2, p', q_{in}\}$, $\widetilde{C_{i+1}}(q) = \widetilde{C}_i(q) \geq \widetilde{C_0}(q) + C_i(q) = \widetilde{C_0}(q) + C_{i+1}(q)$. If $q = q_{in} \notin \{p_2, p'\}$, $\widetilde{C_{i+1}}(q) = \widetilde{C}_i(q) \geq (M-1).N_1 + C_i(q) = (M-1).N_1 + C_{i+1}(q)$. If $q \in \{p_2, p'\}$, then $\widetilde{C_{i+1}}(q) = \widetilde{C}_i(q) + 1 \geq \widetilde{C_0}(q) + C_i(q) + 1 = \widetilde{C_0}(q) + C_{i+1}(q)$ if $q \neq q_{in}$, and $\widetilde{C_{i+1}}(q) = \widetilde{C}_i(q) + 1 \geq (M-1).N_1 + C_i(q) + 1 = (M-1) + N_1 + C_{i+1}(q)$ if $q = q_{in}$. Let $q \in Q_W$ such that $p \neq q$, if $q \notin \{p_2, p'\}$, $\widetilde{C_{i+1}}(q) = \widetilde{C}_i(q) \geq C_i(q) = C_{i+1}(q)$. If $p \in \{p_2, p'\}$, then $\widetilde{C_{i+1}}(q) = \widetilde{C}_i(q) + 1 \geq C_i(q) + 1 = C_{i+1}(q)$.
- If $t_{i+1} = (p_1, !a, p_2)$ and for all $p, p' \in Q$ such that $(p, ?m, p') \in T$, we have that $C'_i(p) = 0$. Then, either $\widetilde{C}_i(p) = 0$ for all $p, p' \in Q$ such that $(p, ?a, p') \in T$, or there exist some $p, p' \in Q$ such that $(p, ?m, p') \in T$ and $\widetilde{C}_i(p) > 0$. In the first case, since $\widetilde{C}_i(p_1) \geq 0$ by induction hypothesis, $\widetilde{C_{i+1}} = \widetilde{C}_i - \wr p_1\wr + \wr p_2\wr$, and $C_{i+1} = C_i - \wr p_1\wr + \wr p_2\wr$. Then it is obvious that $\widetilde{C_{i+1}}(q) \geq \widetilde{C_0}(q) + C_{i+1}(q)$ for all $q \in Q_A \setminus \{q_{in}\}$, $\widetilde{C_{i+1}}(q) \geq C_{i+1}(q)$ for all $q \in Q_W$, and $\widetilde{C_{i+1}}(q_{in}) \geq (M-1).N_1 + C_{i+1}(q_{in})$. In the second case, $\widetilde{C_{i+1}} = \widetilde{C}_i - \wr p_1, p\wr + \wr p_2, p'\wr$. Let $q \in Q_A$. If $q = p_1$, then if $q \neq q_{in}$, $\widetilde{C_{i+1}}(q) = \widetilde{C}_i(q) - 1 \geq \widetilde{C_0}(q) + C_i(q) - 1 = \widetilde{C_0}(q) + C_{i+1}(q)$, and if $q = p_1 = q_{in}$, then $\widetilde{C_{i+1}}(q) = \widetilde{C}_i(q) - 1 \geq (M-1).N_1 + C_i(q) - 1 = (M-1).N_1 + C_{i+1}(q)$. If $q = p_2$, if $q \neq q_{in}$, $\widetilde{C_{i+1}}(q) = \widetilde{C}_i(q) + 1 \geq \widetilde{C_0}(q) + C_i(q) + 1 = \widetilde{C_0}(q) + C_{i+1}(q)$, and if $q = p_2 = q_{in}$, $\widetilde{C_{i+1}}(q) = \widetilde{C}_i(q) + 1 \geq (M-1).N_1 + C_i(q) + 1 = (M-1).N_1 + C_{i+1}(q)$. If $q = p'$, and $q \neq q_{in}$, $\widetilde{C_{i+1}}(q) = \widetilde{C}_i(q) + 1 \geq \widetilde{C_0}(q) + C_i(q) + 1 > \widetilde{C_0}(q) + C_{i+1}(q)$ since $C_{i+1}(p') = C_i(p') + 1$. If $q = p' = q_{in}$, $\widetilde{C_{i+1}}(q) = \widetilde{C}_i(q) + 1 \geq (M-1).N_1 + C_i(q) + 1 = (M-1).N_1 + C_{i+1}(q)$. Otherwise, $\widetilde{C_{i+1}}(q) = \widetilde{C}_i(q) \geq \widetilde{C_0}(q) + C_i(q) = \widetilde{C_0}(q) + C_{i+1}(q)$ if $q \neq q_{in}$, and $\widetilde{C_{i+1}}(q) = \widetilde{C}_i(q) \geq (M-1).N_1 + C_i(q) = (M-1).N_1 + C_{i+1}(q)$ if $q = q_{in}$.

  Finally, let $q \in Q_W$. If $p = q$, then $\widetilde{C_{i+1}}(q) = \widetilde{C}_i(q) - 1 \geq 0$. However, $C_{i+1}(q) = C_i(q) = 0$, then $\widetilde{C_{i+1}}(q) \geq C_{i+1}(q)$. If $q = p_2$, then $\widetilde{C_{i+1}}(q) = \widetilde{C}_i(q) + 1 \geq C_i(q) + 1 = C_{i+1}(q)$. If $q = p'$, then $\widetilde{C_{i+1}}(q) = \widetilde{C}_i(q) + 1 \geq C_i(q) + 1 > C_i(q) = C_{i+1}(q)$. Otherwise, $\widetilde{C_{i+1}}(q) = \widetilde{C}_i(q) \geq C_i(q) = C_{i+1}(q)$.

□

B Proofs of Section 4.2

We give the formal definitions of the protocol and the formal proofs of Section 4.2. We start by introducing some notations. We denote an instance of CVP: $C = (V, o, G, B, b)$ where $V = \{v_1, \ldots, v_n\}$ denotes the n input variables, o is the output variable, $G = \{g_1, \ldots, g_m\}$ the m boolean gates, $B = \{b_1, \ldots, b_n\}$ the boolean assignment such that for all $1 \leq i \leq n$, boolean $b_i \in \{\top, \bot\}$ is the assignment of variable v_i, and b the boolean output value to test. Let $V' = V \cup \{o_1, \ldots, o_m\}$ where o_j is the output variable of gate j for $1 \leq j \leq m$. Wlog we can assume that $o = o_m$. For $1 \leq j \leq m$, we denote gate g_j by $g_j(\diamond, x_1, x_2, o_j)$ with $x_1, x_2 \in V'$ and $\diamond \in \{\vee, \wedge\}$ or by $g_j(\neg, x, o_j)$ with $x \in V'$. As C is acyclic, one can assume that $x_1, x_2, x \in V \cup \{o_1, o_2, \ldots, o_{j-1}\}$.

Let some $x \in V'$, we denote $\mathsf{bv}(x)$ the boolean value of x with respect to the input B. Note that if $x \in V$, there exists $1 \leq i \leq n$ such that $x = v_i$ and so $\mathsf{bv}(x) = b_i$.

Let $1 \leq j \leq m$, we describe $P_j = (Q_j, \Sigma_j, T_j)$ where $\Sigma_j = V' \times \{\top, \bot\}$ as follows:

- if $g_j(\vee, x_1, x_2, o_j)$, $Q_j = \{q_0^j, q_\top^j, q_1^j, q_\bot^j\}$, and $T_j = \{(q_0^j, ?(x_k, \top), q_\top^j) \mid k = 1, 2\}$ $\cup \{(q_0^j, ?(x_1, \bot), q_1^j), (q_1^j, ?(x_2, \bot), q_\bot^j)\} \cup \{(q_\top^j, !!(o_j, \top), q_\top^j), (q_\bot^j, !!(o_j, \bot), q_\bot^j)\}$;
- if $g_j(\wedge, x_1, x_2, o_j)$, $Q_j = \{q_0^j, q_\top^j, q_1^j, q_\bot^j\}$, and $T_j = \{(q_0^j, ?(x_k, \bot), q_\bot^j) \mid k = 1, 2\}$ $\cup \{(q_0^j, ?(x_1, \top), q_1^j), (q_1^j, ?(x_2, \top), q_\top^j)\} \cup \{(q_\top^j, !!(o_j, \top), q_\top^j), (q_\bot^j, !!(o_j, \bot), q_\bot^j)\}$;
- if $g_j(\neg, x, o_j)$, $Q_j = \{q_0^j, q_\top^j, q_\bot^j\}$, and $T_j = \{(q_0^j, ?(x, \bot), q_\top^j), (q_0^j, ?(x, \top), q_\bot^j)\} \cup$ $\{(q_\top^j, !!(o_j, \top), q_\top^j), (q_\bot^j, !!(o_j, \bot), q_\bot^j)\}$.

We are now ready to define the protocol associated to C, $P_C = (Q, \Sigma, q_{in}, T)$:

- $Q = \{q_{in}\} \cup \bigcup_{1 \leq j \leq m} Q_j$;
- $\Sigma = V' \times \{\top, \bot\}$;
- $T = \{(q_{in}, !!(v_j, b_j), q_{in}) \mid 1 \leq j \leq n\} \cup \{(q_{in}, !!\tau, q_0^j) \mid 1 \leq j \leq m\} \bigcup_{1 \leq j \leq m} T_j$.

Observe that P is Wait-Only: q_{in} is an active state, and for all $1 \leq j \leq m$: if g_j is a "not" gate, q_0^j is a waiting state, and $q_\bot^j, q_\top^j$ are active states, and if g_j is an "and" gate or an "or" gate, $q_\bot^j, q_\top^j$ are active states and q_0^j, q_1^j are waiting states.

We show that $\mathsf{bv}(o) = b$ if and only if there is an initial configuration $C_0 \in \mathcal{I}$ and $C_f \in \mathcal{C}$ such that $C_0 \to^* C_f$ and $C_f(q_b^m) > 0$.

Lemma 11. *If $\mathsf{bv}(o) = b$, then there exists $C_0 \in \mathcal{I}$ and $C_f \in \mathcal{C}$ such that $C_0 \to^* C_f$ and $C_f(q_b^m) > 0$.*

Proof. Assume that $\mathsf{bv}(o) = b$, and take $C_0 = \wr(m+1).q_{in}\wr$. There exists an execution $C_0 \to^* C_f$ with $C_f = \wr q_{in}, q_{y_1}^1, q_{y_2}^2, \ldots q_{y_m}^m \wr$ where $y_j = \mathsf{bv}(o_j)$ with the input boolean values B for $1 \leq j \leq m$. By definition, $y_m = \mathsf{bv}(o) = b$. The execution is: $C_0 \to^+ C_1 \to^+ \ldots \to^+ C_m$ where $C_j = \wr q_{in}, q_{y_1}^1, \ldots q_{y_j}^j, q_{in}, \ldots, q_{in}\wr$ for all $0 \leq j < m$. Between C_j and C_{j+1}, the sequence of transitions is:

– if $g_{j+1}(\vee, x_1, x_2, o_{j+1})$ with $\mathsf{bv}(x_k) = \top$ for some $k \in \{1,2\}$, then $\mathsf{bv}(o_{j+1}) = \top$. Either $x_k = v_i$ (and $b_i = \top$) for some $1 \leq i \leq n$, or $x_k = o_i$ (and $y_i = \top$) for some $1 \leq i \leq j$. In the first case, as $C_j(q_{in}) \geq 2$, then consider the sequence $C_j \xrightarrow{(q_{in}, !!\tau, q_0^{j+1})} C'_j \xrightarrow{(q_{in}, !!(v_i,\top), q_{in})} C_{j+1}$. It holds that $C'_j = C_j - \{\!| q_{in} |\!\} + \{\!| q_0^{j+1} |\!\}$ and $C_{j+1} = C'_j - \{\!| q_0^{j+1} |\!\} + \{\!| q_\top^{j+1} |\!\}$. Hence $C_{j+1} = \{\!| q_{in}, q_{y_1}^1, \dots q_{y_j}^j, q_{y_{j+1}}^{j+1}, \dots, q_{in} |\!\}$.
In the second case, $C_j(q_\top^i) > 0$ as $i \leq j$ and $\top = \mathsf{bv}(o_i)$, and $C_j(q_{in}) > 0$. Consider the sequence $C_j \xrightarrow{(q_{in}, !!\tau, q_0^{j+1})} C'_j \xrightarrow{(q_\top^i, !!(o_i,\top), q_\top^i)} C_{j+1}$. It holds that $C'_j = C_j - \{\!| q_{in} |\!\} + \{\!| q_0^{j+1} |\!\}$ and $C_{j+1} = C'_j - \{\!| q_0^{j+1} |\!\} + \{\!| q_\top^{j+1} |\!\}$. Hence $C_{j+1} = \{\!| q_{in}, q_{y_1}^1, \dots q_{y_j}^j, q_{y_{j+1}}^{j+1}, \dots, q_{in} |\!\}$.
– if $g_{j+1}(\wedge, x_1, x_2, o_{j+1})$ with $\mathsf{bv}(x_k) = \bot$ for some $k = 1,2$, then $\mathsf{bv}(o_{j+1}) = \bot = y_{j+1}$. The sequence of transitions is built in an analogous way than the previous case, however this time the broadcast messages are τ and $(x_k, \bot)$ and the reached state is $q_\bot^{j+1}$.
– if $g_{j+1}(\vee, x_1, x_2, o_{j+1})$ with $\mathsf{bv}(x_1) = \mathsf{bv}(x_2) = \bot$, then $\mathsf{bv}(o_{j+1}) = \bot$. Either $x_1 = v_i$ (and $b_i = \bot$) for some $1 \leq i \leq n$, or $x_1 = o_i$ (and $y_i = \bot$) for some $1 \leq i \leq j$. In the first case, as $C_j(q_{in}) \geq 2$, then consider the sequence $C_j \xrightarrow{(q_{in}, !!\tau, q_0^{j+1})} C_{j,1} \xrightarrow{(q_{in}, !!(v_i,\bot), q_{in})} C_{j,2}$. It holds that $C_{j,1} = C_j - \{\!| q_{in} |\!\} + \{\!| q_0^{j+1} |\!\}$ and $C_{j,2} = C_{j,1} - \{\!| q_0^{j+1} |\!\} + \{\!| q_1^{j+1} |\!\}$.
If $x_1 = o_i$ for some $i \leq j$, $C_j(q_\bot^i) > 0$ as $i \leq j$ and $\bot = \mathsf{bv}(o_i)$, and $C_j(q_{in}) > 0$. Consider the sequence $C_j \xrightarrow{(q_{in}, !!\tau, q_0^{j+1})} C_{j,1} \xrightarrow{(q_\bot^i, !!(o_i,\bot), q_\bot^i)} C_{j,2}$. It holds that $C_{j,1} = C_j - \{\!| q_{in} |\!\} + \{\!| q_0^{j+1} |\!\}$ and $C_{j,2} = C_{j,1} - \{\!| q_0^{j+1} |\!\} + \{\!| q_1^{j+1} |\!\}$.
In both cases, $C_{j,2} = \{\!| q_{in}, q_{y_1}^1, \dots q_{y_j}^j, q_1^{j+1}, \dots, q_{in} |\!\}$.
We make the same cases distinctions in order to build the configuration C_{j+1} such that $C_{j,2} \xrightarrow{(q, !!(x_2,\bot), q)} C_{j+1}$ where $q = q_{in}$ if $x_2 \in V$, and otherwise $x_2 = o_i$ for some $i \leq j$ and $q = q_\bot^i$.
It holds that $C_{j+1} = C_{j,2} - \{\!| q_1^{j+1} |\!\} + \{\!| q_\bot^{j+1} |\!\}$, hence $C_{j+1} = \{\!| q_{in}, q_{y_1}^1, \dots q_{y_j}^j, q_{y_{j+1}}^{j+1}, \dots, q_{in} |\!\}$.
– if $g_{j+1}(\wedge, x_1, x_2, o_{j+1})$ with $\mathsf{bv}(x_1) = \mathsf{bv}(x_2) = \top$, then $\mathsf{bv}(o_{j+1}) = \top$. The sequence of transitions is built in an analogous way than the previous case, however this time the broadcast messages are τ, $(x_1, \top)$ and $(x_2, \top)$ and the reached state is $q_\top^{j+1}$.
– if $g_{j+1}(\neg, x, o_{j+1})$ with $\mathsf{bv}(x) = \top$ (resp. $\bot$), then $\mathsf{bv}(o_{j+1}) = \bot$ (resp. $\top$). Either $x = v_i$ for some $1 \leq i \leq n$, and as $C_j(q_{in}) \geq 2$, we build the following sequence: $C_j \xrightarrow{(q_{in}, !!\tau, q_0^{j+1})}, C'_j \xrightarrow{(q_{in}, (v_i, b_i), q_{in})} C_{j+1}$. It holds that $C'_j = C_j - \{\!| q_{in} |\!\} + \{\!| q_0^{j+1} |\!\}$ and $C_{j+1} = C'_j - \{\!| q_0^{j+1} |\!\} + \{\!| q_{\bar{b}_i}^{j+1} |\!\}$ where $\bar{b}_i = \bot$ (resp. $\bar{b}_i = \top$).
Otherwise, $x = o_i$ for some $i \leq j$, and so $C_j(q_\top^i) > 0$ (resp. $C_j(q_\bot^i) > 0$) and $C_j(q_{in}) > 0$. Hence, we can build the following sequence: $C_j \xrightarrow{(q_{in}, !!\tau, q_0^{j+1})}, C'_j \xrightarrow{(q_{in}, (o_i, \mathsf{bv}(o_i)), q_{in})} C_{j+1}$. It holds that $C'_j = C_j - \{\!| q_{in} |\!\} + \{\!| q_0^{j+1} |\!\}$ and $C_{j+1} = C'_j - \{\!| q_0^{j+1} |\!\} + \{\!| q_{\bar{y}_i}^{j+1} |\!\}$ where $\bar{y}_i = \bot$ (resp. $\bar{y}_i = \top$).

In both cases, $C_{j+1} = \wr q_{in}, q^1_{y_1}, \dots q^j_{y_j}, q^{j+1}_{y_{j+1}}, \dots, q_{in}\wr$.

Hence, $C_0 \to^+ C_m$ where $C_m(q^m_{y_m}) > 0$, and so if $b = y_m$, $C_m(q^m_b) > 0$. □

Lemma 12. *If there exists $C_0 \in \mathcal{I}$ and $C_f \in \mathcal{C}$ such that $C_0 \to^* C_f$ and $C_f(q^m_b) > 0$, then $\mathsf{bv}(o) = b$.*

Proof. Let $C_0 \in \mathcal{I}$ and $C_f \in \mathcal{C}$ such that $C_0 \to^* C_f$ and $C_f(q^m_b) > 0$.

First we show that all broadcast messages $(x, b_x) \in V' \times \{\top, \bot\}$ are such that $b_x = \mathsf{bv}(x)$. We start by proving it for $x \in V$, and then proceed to prove it for $x \in \{o_1, \dots, o_m\}$ by induction on m.

Let (x, b_x) a broadcast message such that $x \in V$, we note $x = v_i$ with $1 \leq i \leq n$. By construction of the protocol P, the only broadcast transition labelled with first element v_i is the transition $(q_{in}, !!(v_i, b_i), q_{in})$ where b_i is the input boolean value for variable v_i, i.e. $\mathsf{bv}(v_i) = b_i$. Hence, all broadcast messages (x, b_x) with $x \in V$, are such that $b_x = \mathsf{bv}(x)$.

We prove now than for all $(x, b_x) \in \{o_1, \dots, o_m\} \times \{\top, \bot\}$, $b_x = \mathsf{bv}(x)$ and we do so by induction on m. For $m = 1$, we have that $x = o_1$. Note that tuples containing o_1 can only be broadcast from $q^1_\bot$ or $q^1_\top$. Denote $g_1(\diamond, x_1, x_2, o_1)$ or $g_1(\neg, x_3, o_1)$ with $\diamond \in \{\vee, \wedge\}$ and $x_1, x_2, x_3 \in V$ by acyclicity.

- if $\diamond = \vee$, then a process reaching $q^1_\top$ has necessarily received $(x_1, \top)$ or $(x_2, \top)$, and a process reaching $q^1_\bot$ has necessarily received $(x_1, \bot)$ and $(x_2, \bot)$. As we proved, all broadcast messages containing x_1 (resp. x_2) are of the form $(x_1, \mathsf{bv}(x_1))$ (resp. $(x_2, \mathsf{bv}(x_2))$). Hence, only one state between $q^1_\top$ and $q^1_\bot$ is reachable. If it is $q^1_\top$ (resp. $q^1_\bot$), the only messages containing o_1 which can be broadcast are $(o_1, \top)$ (resp. $(o_1, \bot)$) and it holds that $\top = \mathsf{bv}(x_1) \vee \mathsf{bv}(x_2) = \mathsf{bv}(o_1)$ (resp. $\bot$) as the process on $q^1_\top$ (resp. $q^1_\bot$) received either $(x_1, \top)$ or $(x_2, \top)$ (resp. $(x_1, \bot)$ and $(x_2, \bot)$);
- if $\diamond = \wedge$, the argument is analogous to the previous case;
- if $\diamond = \neg$, then a process reaching $q^1_\top$ has necessarily received $(x_3, \bot)$ and a process reaching $q^1_\bot$ has necessarily received $(x_3, \top)$. As we proved, all broadcast messages containing x_3 are of the form $(x_3, \mathsf{bv}(x_3))$. Hence, only one state between $q^1_\top$ and $q^1_\bot$ is reachable. If it is $q^1_\top$ (resp. $q^1_\bot$), the only messages containing o_1 which can be broadcast are $(o_1, \top)$ (resp. $(o_1, \bot)$) and it holds that $\top = \neg\mathsf{bv}(x_3) = \mathsf{bv}(o_1)$ (resp. $\bot$) as the process on $q^1_\top$ (resp. $q^1_\bot$) received $(x_3, \bot)$ (resp. $(x_3, \top)$).

Assume the property true for m gates, and let (o_{m+1}, b_{m+1}) a broadcast message and note $g_{m+1}(\diamond, x_1, x_2, o_{m+1})$ with $\diamond \in \{\wedge, \vee\}$, or $g_{m+1}(\neg, x_3, o_{m+1})$ with $x_1, x_2, x_3 \in V \cup \{o_1, \dots, o_m\}$. By induction hypothesis, the only broadcast messages containing x_1, x_2 or x_3 are $(x_1, \mathsf{bv}(x_1))$, $(x_2, \mathsf{bv}(x_2))$, and $(x_3, \mathsf{bv}(x_3))$. The arguments are then the same than in the case $m = 1$. □

Hence, we proved with the two previous lemmas that $\mathsf{bv}(o) = b$ if and only if q^m_b is coverable and we get the following theorem.

Theorem 5. *StateCover for Wait-Only protocols is P-hard.*

Remark 4. If we transform the Wait-Only protocol presented here into a Wait-Only NB-Rendez-vous protocol (by transforming all the broadcast transitions into sending transitions on the same message), the reduction remains sound and complete. Indeed, in the execution of P_C built in Lemma 11, all the broadcasts are received by only one process. Hence, the same execution is possible by replacing broadcasts transitions by sending transitions. The proof of Lemma 12 works exactly the same way if the broadcasts transitions become sending transitions. Hence, the StateCover and ConfCover problems for Wait-Only NB-Rendez-vous are P-hard, closing a lower bound open in [GSS23].

C Proofs of Section 5

C.1 Proofs of Section 5.3

Proof of Lemma 7. We present here the two missing cases in the proof of Lemma 7.

(i) if $\alpha = !!a$, then by definition of $\to$, $C = \wr q_1, \dots, q_n, q \wr$ and $C' = \wr q'_1, \dots, q'_n, q' \wr$ such that for all $1 \leq i \leq n$, either $(q_i, ?a, q'_i) \in T$ or $a \notin R(q_i)$ and $q_i = q'_i$. We get the two following disjoint cases:
   - $M'(q') = 0$. In this case, $M' = \wr q'_{i_1}, q'_{i_2}, \dots q'_{i_K} \wr$ where $1 \leq i_j \leq n$ for all $1 \leq j \leq K$ and $i_j \neq i_\ell$ if $j \neq \ell$. Let $M = \wr q_{i_1}, q_{i_2}, \dots q_{i_K} \wr$. Note that we have $M \leq C$, hence $C \in [\![(M, S)]\!]$. Using the definition of $\to$, we have as well $M + \wr q \wr \to M' + \wr q' \wr$. Furthermore since $C(q) > 0$, we have $q \in S$ by definition of S. Applying the definition of $\Rightarrow_{\mathsf{ext}}$ we have $(M, S) \overset{t}{\underset{\mathsf{ext}}{\Longrightarrow}} (M', S')$.
   - $M'(q') > 0$. In this case, $M' = \wr q'_{i_1}, q'_{i_2}, \dots q'_{i_{K-1}}, q' \wr$ where $1 \leq i_j \leq n$ for all $1 \leq j < K$ and $i_j \neq i_\ell$ if $j \neq \ell$. Let $M = \wr q_{i_1}, q_{i_2}, \dots q_{i_{K-1}}, q \wr$. Note that we have $M \leq C$, hence $C \in [\![(M, S)]\!]$. Using the definition of $\to$, we have as well $M \to M'$. Applying the definition of $\Rightarrow_{\mathsf{step}}$, we get $(M, S) \overset{t}{\underset{\mathsf{step}}{\Longrightarrow}} (M', S')$.

(ii) if $\alpha = !a$ and the message is *not* received (i.e. it is a non blocking sending), then using the definition of $\to$, we have $C' = C - \wr q \wr + \wr q' \wr$ and $a \notin R(p)$ for all $p \in Q$ such that $(C - \wr q \wr)(p) > 0$. We obtain the two following disjoint cases:
   - $M'(q') = 0$. Since $C'(q') > 0$ and $M' \leq C'$, we deduce that $C' = M' + \wr q' \wr + M_2$ for some multiset M_2. By definition of C', we have as well $C = C' + \wr q \wr - \wr q' \wr$, hence $C = M' + \wr q \wr + M_2$. We deduce that $M' \leq C$ and hence $C \in [\![(M', S)]\!]$. Furthermore we have $M' + \wr q \wr \to M' + \wr q' \wr$ as $a \notin R(q)$ for all states $q \in Q$ such that $M'(q) > 0$. Consequently, $(M', S) \overset{t}{\underset{\mathsf{ext}}{\Longrightarrow}} (M', S')$.
   - $M'(q') > 0$. Since $M' \leq C'$, we have $C' = M' + M_2$ for some multiset M_2. Let $M = M' + \wr q \wr - \wr q' \wr$. We have hence $C = C' + \wr q \wr - \wr q' \wr = M' + \wr q \wr - \wr q' \wr + M_2 = M + M_2$. Hence $M \leq C$ and $C \in [\![(M, S)]\!]$. Furthermore

we have that $a \notin R(p)$ for all $p \in Q$ such that $(M - \wr q \wr)(p) > 0)$. We hence deduce that $M \xrightarrow{t} M'$. Applying the definition of $\Rightarrow_{\mathsf{step}}$, we get $(M,S) \overset{t}{\underset{\mathsf{step}}{\Longrightarrow}} (M',S')$.

□

Proof of Lemma 8. Suppose we have $C_0 \xrightarrow{t_1} C_1 \xrightarrow{t_2} \dots \xrightarrow{t_n} C_n$ with $C_0 = C_{in}$ and $C_n = C$. We show by induction on $0 \leq i \leq n$ that for all $M_i \in \mathcal{C}_{=K}$ such that $M_i \preceq C_i$ there exists $S_i \subseteq Q$ such that $C_i \in [\![(M_i,S_i)]\!]$ and $\gamma_{in} \Rightarrow^* (M_i,S_i)$.

For $i = 0$, we have $C_{in} = \wr ||C|| \cdot q_{in} \wr$ and $\wr K \cdot q_{in} \wr$ is the unique configuration of size K smaller than C_{in}. Since $\gamma_{in} = (\wr K \cdot q_{in} \wr, \{q_{in}\})$, we have $C_{in} \in [\![\gamma_{in}]\!]$.

Assume now that the property holds for $0 \leq i < n$. Let $M_{i+1} \in \mathcal{C}_{=K}$ such that $M_{i+1} \preceq C_{i+1}$. Since $C_i \xrightarrow{t_{i+1}} C_{i+1}$, by applying Lemma 7, there exists $M_i \in \mathcal{C}_{=K}$, $S_{i+1} \subseteq Q$ such that $(M_i,S_i) \Rightarrow (M_{i+1},S_{i+1})$ where $S_i = \{q \in Q \mid C_i(q) > 0\}$, $C_i \in [\![(M_i,S_i)]\!]$ and $C_{i+1} \in [\![(M_{i+1},S_{i+1})]\!]$. By induction hypothesis, there exists $S'_i \subseteq Q$ such that $C_i \in [\![(M_i,S'_i)]\!]$ and $\gamma_{in} \Rightarrow^* (M_i,S'_i)$. But since $C_i \in [\![(M_i,S'_i)]\!]$ and $S_i = \{q \in Q \mid C_i(q) > 0\}$, we have $S_i \subseteq S'_i$. Using Lemma 6.2, we deduce that there exists S'_{i+1} such that $S_{i+1} \subseteq S'_{i+1}$ and $(M_i,S'_i) \Rightarrow (M_{i+1},S'_{i+1})$. Hence $\gamma_{in} \Rightarrow^* (M_{i+1},S'_{i+1})$ and thanks to Lemma 6.1, $C_{i+1} \in [\![(M_{i+1},S'_{i+1})]\!]$. □

C.2 Proofs of Section 5.6

Assume that there exists a word $w = a_1 \dots a_k \in \bigcap_{1 \leq i \leq n} L(\mathcal{A}_i)$, i.e., $q_i^f = \Delta^*(q_i^0, a_1 \dots a_k)$ for all $1 \leq i \leq n$. Then take $C = \wr (n+1).q_{in} \wr$. There exists an execution $C \to^* C'$ with $C' = \wr q_s, q_1^f, \dots, q_n^f \wr \succeq C_f$. This execution is $C \xrightarrow{(q_{in},!!\tau,q_s)} \tilde{C}_1 \xrightarrow{(q_{in},!!\tau,q_1)} \tilde{C}_2 \dots \xrightarrow{(q_{in},!!\tau,q_n)} \tilde{C}_0 \xrightarrow{(q_s,!!go,q_s)} C_0 \xrightarrow{(q_s,!!a_1,q_s)} C_1 \xrightarrow{(q_s,!!a_2,q_s)} C_2 \dots \xrightarrow{(q_s,!!a_k,q_s)} C'$.

One can check that $C_0(q_i^0) = 1$ for all $1 \leq i \leq n$, and hence, by definition of $(T_i)_{1 \leq i \leq n}$, $C_k(q_i^f) = 1$ for all $1 \leq i \leq n$. Hence $C' = \wr q_s, q_1^f, \dots, q_n^f \wr \succeq C_f$.

Reciprocally, assume that there exists an initial configuration C and an execution $C \to^* C'$ with $C' \succeq C_f$. We first make easy observations about the executions of this protocol.

Observation 1 *Let $C_1, C_2 \in \mathcal{C}_P$. We write $C_1 \xrightarrow{\tau}{}^* C_2$ if there exists a sequence of k transitions $C_1 \xrightarrow{t_1} \dots \xrightarrow{t_k} C_2$ such that $t_i \in \{(q_{in}, !!\tau, q) \mid q \in \{q_s, q_1, \dots, q_n\}\}$ for all $1 \leq i \leq k$. Then $C_2(q) \geq C_1(q)$ for all $q \in \{q_s, q_1, \dots, q_n\}$, $C_2(q_{in}) \leq C_1(q_{in})$, and $C_2(q) = C_1(q)$ for all other $q \in Q$.*

Observation 2 *Let $C_1, C_2 \in \mathcal{C}_P$ such that $C_1 \to^* C_2$ with no transition $(q_s, !!go, q_s)$. If there is some $1 \leq i \leq n$ with $C_1(q) = 0$ for all $q \in Q_i$, then $C_2(q) = 0$ for all $q \in Q_i$.*

Observation 3 *Let $C_1, C_2 \in \mathcal{C}_P$ such that $C_1 \xrightarrow{(q_s,!!go,q_s)} C_2$. Then, $C_2(q_s) = C_1(q_s) > 0$, and $C_2(q) = 0$ for all $q \in \bigcup_{1 \leq i \leq n}(Q_i \setminus \{q_i^0\})$.*

We can deduce from these observations that the execution $C \to^* C'$ can be decomposed in $C \to^* \hat{C} \xrightarrow{(q_s,!!go,q_s)} C_0 \to^* C'$. Indeed, since $C(q) = 0$ for all $q \in \bigcup_{1\leq i\leq n} Q_i$, if no transition $(q_s, !!go, q_s)$ appears in the execution, Observation 2 allows to conclude that $C'(q_i^f) = 0$ for all $1 \leq i \leq n$, which contradicts the fact that $C' \geq C_f$. Assume now that this transition is the last transition where action go is sent, i.e., $C_0 \to^* C'$ has no transition $(q_s, !!go, q_s)$. By Observation 3, $C_0(q_s) > 0$ and $C_0(q) = 0$ for all $q \in \bigcup_{1\leq i\leq n}(Q_i \setminus \{q_i^0\})$. By Observation 2, we also deduce that $C_0(q_i^0) > 0$ for all $1 \leq i \leq n$. Otherwise, if there exists $1 \leq j \leq n$ such that $C_0(q_j^0) = 0$, then $C'(q_j^f) = 0$ which is a contradiction with $C' \geq C_f$.

Now the execution $C_0' \to^* C'$ is of the form $C_0 \xrightarrow{\tau}{}^* C_0' \xrightarrow{(q_s,!!a_1,q_s)} C_1 \xrightarrow{\tau}{}^* C_1' \xrightarrow{(q_s,!!a_2,q_s)} C_2 \dots \xrightarrow{(q_s,!!a_k,q_s)} C_k \xrightarrow{\tau}{}^* C'$. Using Observation 1, we can obtain a new execution $C_0 \xrightarrow{(q_s,!!a_1,q_s)} \tilde{C}_1 \xrightarrow{(q_s,!!a_2,q_s)} \tilde{C}_2 \dots \xrightarrow{(q_s,!!a_k,q_s)} \tilde{C}_k$ such that for all $1 \leq j \leq k$, we let $\tilde{C}_j(q_s) = C_0(q_s)$, $\tilde{C}_j(q_i) = C_0(q_i)$ for all $1 \leq i \leq n$, $\tilde{C}_j(q_{in}) = C_0(q_{in})$ and $\tilde{C}_j(q) = C_j(q)$ for all other q. Moreover, $\tilde{C}_k(q_i^f) = C_k(q_f^i) = C'(q_f^i) \geq C_f$.

We can show now that the word $a_1 \dots a_k$ belongs to $\bigcap_{1\leq i\leq n} L(\mathcal{A}_i)$. Let $1 \leq i \leq n$. It is easy to see that for all $1 \leq j \leq k$, there exists a unique $q_i^j \in Q_i$ such that $\tilde{C}_j(q_i^j) > 0$ and $\tilde{C}_j(q) = 0$ for all $q \in Q_i \setminus \{q_i^j\}$. Moreover, for all $1 \leq j \leq k$, $q_i^j = \Delta_i^*(q_i^0, a_1 \dots a_j)$. As $\tilde{C}_k(q_i^f) = C'(q_i^f) > 0$, we get that $q_i^f = \Delta_i^*(q_i^0, a_1 \dots a_k)$, and hence $a_1 \dots a_k \in L(\mathcal{A}_i)$.