# Playing with probabilities in Reconfigurable Broadcast Networks

Nathalie Bertrand, Paulin Fournier, Arnaud Sangnier

# Playing with probabilities in Reconfigurable Broadcast Networks

Nathalie Bertrand[1], Paulin Fournier[2], and Arnaud Sangnier[3]

[1] Inria Rennes Bretagne Atlantique
[2] ENS Cachan Antenne de Bretagne
[3] LIAFA, Univ Paris Diderot, Sorbonne Paris Cité, CNRS

**Abstract.** We study verification problems for a model of network with the following characteristics: the number of entities is parametric, communication is performed through broadcast with adjacent neighbors, entities can change their internal state probabilistically and reconfiguration of the communication topology can happen at any time. The semantics of such a model is given in term of an infinite state system with both non deterministic and probabilistic choices. We are interested in qualitative problems like whether there exists an initial topology and a resolution of the non determinism such that a configuration exhibiting an error state is almost surely reached. We show that all the qualitative reachability problems are decidable and some proofs are based on solving a 2 player game played on the graphs of a reconfigurable network with broadcast with parity and safety objectives.

## 1 Introduction

Providing methods to analyze and verify distributed systems is a complex task and this for several reasons. First there are different families of distributed systems depending on the communication means (shared memory or message passing), on the computing power of the involved entities, on the knowledge of the system provided to the entities (full knowledge, or local knowledge of their neighbors, or no knowledge at all) or on the type of communication topology that is considered (ring, tree, arbitrary graph, etc). Second, most of the protocols developed for distributed systems are supposed to work for an unbounded number of participants, hence in order to verify that a system behaves correctly, one needs to develop methods which allow to deal with such a parameter.

In [11], the authors propose a model which allows to take into account the main features of a family of distributed networks, namely ad-hoc networks. It characterizes the following aspects of such systems: the nodes in the network can only communicate with their neighbors using broadcast communication and the number of participants is unbounded. In this model, each entity behaves similarly following a protocol which is represented by a finite state machine performing three kinds of actions (1) broadcast of a message, (2) reception of a message and (3) internal action. Furthermore, the communication topology does not change during an execution and no entity is deleted or added during an execution. The

*control state reachability problem* consists then in determining whether there exists an initial number of entities in a communication topology such that it is possible to reach a configuration where at least one process is in a specific control state (considered for instance as an error state). The main difficulty in solving such a problem lies in the fact that both the number of processes and the initial communication topology are parameters, for which one wishes to find an instantiation. In [11], it is proven that this problem is undecidable but becomes decidable when considering non-deterministic *reconfiguration* of the communication topology, i.e. when at any moment the nodes can move and change their neighborhood. In [10] this latter problem is shown to be P-complete. An other way to gain decidability in such so called broadcast networks consists in restricting the set of communication topologies to complete graphs (aka cliques) or bounded depth graphs [12] or acyclic directed graphs [1].

We propose here to extend the model of reconfigurable broadcast networks studied in [10] by allowing probabilistic internal actions, that is, a process can change its internal state according to a probabilistic distribution. Whereas the semantics of reconfigurable broadcast networks was given in term of an infinite state system with non-determinism (due to the different possibility of sending messages from different nodes and also to the non-determinism of the protocol itself), we obtain here an infinite state system with probabilistic and non-deterministic choices. On such a system we study the probabilistic version of the *control state reachability* by seeking for the existence of a scheduler resolving non-determinism which minimizes or maximizes the probability to reach a configuration exhibiting a specific state. We focus on the qualitative aspects of this problem by comparing probabilities only with 0 and 1. Note that another model of broadcast networks with probabilistic protocols was defined in [6]; it was however different: the communication topologies were necessarily cliques and decidability of qualitative probabilistic reachability only holds when the network size evolves randomly over time.

For finite state systems with non-determinism and probabilities (like finite state Markov Decision Processes), most verification problems are decidable [5], but when the number of states is infinite, they are much harder to tackle. The introduction of probabilities might even lead to the undecidability, for problems that are decidable in the non-probabilistic case. For instance for extensions of pushdown systems with non-deterministic and probabilistic choices, the model-checking problems of linear time or branching time logic are undecidable [13,7]. On the other hand, it is not always the case that the introduction of probabilistic transitions leads to undecidability but then dedicated verification methods have to be invented, as it is the case for instance for nondeterministic probabilistic lossy channel systems [4]. Even if for well-structured infinite state systems [2,14] (where a monotonic well-quasi order is associated to the set of configurations), a class to which belong the broadcast reconfigurable networks of [11], a general framework for the extension to purely probabilistic transitions has been proposed in [3], it seems hard to adapt such a framework to the case with probabilistic and non-deterministic choices.

In this paper, we prove that the qualitative versions of the *control state reachability problem* for reconfigurable broadcast networks with probabilistic internal choices are all decidable. For some of these problems, like finding a scheduler such that the probability of reaching a control state is equal to 1, our proof technique is based on a reduction to a 2 player game played on infinite graphs with safety and parity objectives. This translation is inspired by a similar translation for finite state systems (see for instance [8]). However when moving to infinite state systems, two problems raise: first whether the translation is correct when the system has an infinite number of states, and then whether we can solve the game. In our translation, we answer the first question in Section 3 and the second one in Section 4. We also believe that the parity game we define on broadcast reconfigurable networks could be used to verify other properties on such systems. Due to lack of space, omitted proofs can be found in Appendix.

## 2 Networks of probabilistic reconfigurable protocols

### 2.1 Preliminary definitions

For a finite or denumerable set $E$, we write $\mathsf{Dist}(E)$, for the set of discrete probability distributions over $E$, that is the set of functions $\delta : E \mapsto [0,1]$ such that $\Sigma_{e\in E}\delta(e) = 1$. We now give the definition of a $1-\frac{1}{2}$ player game, which will be later used to provide the semantics of our model.

**Definition 1 ($1-\frac{1}{2}$ player game).** *A $1-\frac{1}{2}$ player game is a tuple $\mathcal{M} = (\Gamma, \Gamma^{(1)}, \Gamma^{(p)}, \rightarrow, prob)$ where $\Gamma$ is a denumerable set of configurations (or vertices) partitioned into the configurations of Player 1 $\Gamma^{(1)}$ and the probabilistic configurations $\Gamma^{(p)}$; $\rightarrow$: $\Gamma^{(1)} \times \Gamma$ is the non deterministic transition relation; $prob : \Gamma^{(p)} \mapsto \mathsf{Dist}(\Gamma^{(1)})$ is the probabilistic transition relation.*

For a tuple $(\gamma, \gamma') \in \rightarrow$, we will sometimes use the notations $\gamma \rightarrow \gamma'$. A *finite path* in the game $\mathcal{M} = (\Gamma, \Gamma^{(1)}, \Gamma^{(p)}, \rightarrow, prob)$ is a finite sequence of configurations $\gamma_0\gamma_1\ldots\gamma_k$ such that for all $0 \leq i \leq k-1$, if $\gamma_i \in \Gamma^{(1)}$ then $\gamma_i \rightarrow \gamma_{i+1}$ and otherwise $prob(\gamma_i)(\gamma_{i+1}) > 0$; moreover we will say that such a path starts from the configuration $\gamma_0$. An *infinite path* is an infinite sequence $\rho \in \Gamma^\omega$ such that any finite prefix of $\rho$ is a finite path. Furthermore we will say that a path $\rho$ is *maximal* if it is infinite or it is finite and there does not exist a configuration $\gamma$ such that $\rho\gamma$ is a finite path (in other words a finite maximal path ends up in a deadlock configuration). The set of maximal paths is denoted $\Omega$.

A *scheduler* in the game $\mathcal{M} = (\Gamma, \Gamma^{(1)}, \Gamma^{(p)}, \rightarrow, prob)$ is a function $\pi : \Gamma^* \cdot \Gamma^{(1)} \mapsto \Gamma$ that assigns, to a finite sequence of configurations ending with a configuration in $\Gamma^{(1)}$, a successor configuration such that for all $\rho \in \Gamma^*$, $\gamma \in \Gamma^{(1)}$ and $\gamma' \in \Gamma$, if $\pi(\rho \cdot \gamma) = \gamma'$ then $\gamma \rightarrow \gamma'$. We denote by $\Pi$ the set of schedulers for $\mathcal{M}$. Given a scheduler $\pi \in \Pi$, we say that a finite path $\gamma_0\gamma_1\ldots\gamma_n$ *respects* the scheduler $\pi$ if for every $i \in \{0\ldots n-1\}$, we have that if $\gamma_i \in \Gamma^{(1)}$ then $\pi(\gamma_0\ldots\gamma_i) = \gamma_{i+1}$. Similarly we say that an infinite path $\rho = \gamma_0\gamma_1\ldots$ *respects* the scheduler $\pi$ if every finite prefix of $\rho$ respects $\pi$.

*Remark 1.* Alternatively, a scheduler in the game $\mathcal{M} = (\Gamma, \Gamma^{(1)}, \Gamma^{(p)}, \to, prob)$ can be defined as what is often called a *scheduler with memory*. It is given by a set $M$ called the *memory* together with a *strategic function* $\pi_M : \Gamma^{(1)} \times M \to \Gamma$, an *update function* $U_M : \Gamma^{(1)} \times M \times \Gamma \to M$, and an initialization function $I_M : \Gamma^{(0)} \to M$. Intuitively, the update function updates the memory state given the previous configuration, the current memory state and the current configuration. The two definitions for schedulers coincide, and we will use one or the other, depending on what is more convenient.

The set of paths starting from a configuration and respecting a scheduler represents a stochastic process. Given a measurable set of paths $\mathcal{A} \subseteq \Omega$, we denote by $\mathbb{P}(\mathcal{M}, \gamma, \pi, \mathcal{A})$ the probability of event $\mathcal{A}$ for the infinite paths starting from the configuration $\gamma \in \Gamma$ and respecting the scheduler $\pi$. We define then extremal probabilities of the event $\mathcal{A}$ starting from configuration $\gamma$ as follows:

$$\mathbb{P}_{\inf}(\mathcal{M}, \gamma, \mathcal{A}) = \inf_{\pi \in \Pi} \mathbb{P}(\mathcal{M}, \gamma, \pi, \mathcal{A}) \text{ and } \mathbb{P}_{\sup}(\mathcal{M}, \gamma, \mathcal{A}) = \sup_{\pi \in \Pi} \mathbb{P}(\mathcal{M}, \gamma, \pi, \mathcal{A})$$

### 2.2 Networks of probabilistic reconfigurable protocols

We introduce in this section our model to represent the behavior of a communication protocol in a network. This model has three main features : the communication in the network is performed via broadcast communication, each node in the network can change its internal state probabilistically and the communication topology can change dynamically. This model extends the one proposed in [10] with probability and can be defined in two steps. First, a configuration of the network is represented thanks to a labelled graph in which the edges characterize the communication topology and the label of the nodes give the state and whether they are the next node which will perform an action or not.

**Definition 2 ($\mathcal{L}$-graph).** *Given $\mathcal{L}$ a set of labels, an* $\mathcal{L}$-graph *is a labelled undirected graph $G = (V, E, L)$ where: $V$ is a finite set of nodes, $E \subseteq V \times V \setminus \{(v, v) \mid v \in V\}$ is a finite set of edges such that $(v, v') \in E$ iff $(v', v) \in E$, and $L : V \mapsto \mathcal{L}$ is a labelling function.*

We denote by $\mathcal{G}_{\mathcal{L}}$ the infinite set of $\mathcal{L}$-graphs and for a graph $G = (V, E, L)$, let $L(G) \subseteq \mathcal{L}$ be the set of all the labels present in $G$, i.e. $L(G) = \{L(v) \mid v \in V\}$. For an edge $(v, v') \in V$, we use the notation $v \sim_G v'$ to denote that the two vertices $v$ and $v'$ are adjacent in $G$. When the considered graph $G$ is made clear from the context, we may omit $G$ and write simply $v \sim v'$.

Then, in our model, each node of the network behaves similarly following a protocol whose description is given by what can be seen as a finite $1 - \frac{1}{2}$ player game labelled with a communication alphabet.

**Definition 3 (Probabilistic protocol).** *A* probabilistic protocol *is a tuple $\mathcal{P} = (Q, Q^{(1)}, Q^{(P)}, q_0, \Sigma, \Delta, \Delta^{\mathsf{int}})$ where $Q$ is a finite set of control states partitioned into $Q^{(1)}$, the states of Player 1, and $Q^{(P)}$ the probabilistic states;*

*$q_0 \in Q^{(1)}$ is the initial control state; $\Sigma$ is a finite message alphabet; $\Upsilon$ is a finite set of internal actions; $\Delta \subseteq (Q^{(1)} \times \{!!a, ??a \mid a \in \Sigma\} \times Q^{(1)}) \cup (Q^{(1)} \times \{\varepsilon\} \times Q)$ is the transition relation; $\Delta^{\mathsf{int}} : Q^{(P)} \mapsto \mathsf{Dist}(Q^{(1)})$ is the internal probabilistic transition relation.*

The label $!!a$ [resp. $??a$] represents the broadcast [resp. reception] of the message $a \in \Sigma$, whereas $\varepsilon$ represents an internal action. Given a state $q \in Q$ and a message $a \in \Sigma$, we define the set $R_a(q) = \{q' \in Q \mid (q, ??a, q') \in \Delta\}$ containing the control states that can be reached in $\mathcal{P}$ from the state $q$ after receiving the message $a$. We also denote by *ActStates* the set of states $\{q \in Q \mid \exists (q, !!a, q') \in \Delta$ or $\exists (q, \varepsilon, q') \in \Delta\}$ from which a broadcast or an internal action can be performed.

The semantics associated to a protocol $\mathcal{P} = (Q, Q^{(1)}, Q^{(P)}, q_0, \Sigma, \Delta, \Delta^{\mathsf{int}})$ is given in term of an infinite state $1-\frac{1}{2}$ player game. We will represent the network thanks to labelled graphs. The intuition is that each node of the graph runs the protocol and the semantics respect the following rules: first the Player 1 chooses non deterministically a communication topology (i.e. the edge relation) and a node which will then perform either a broadcast or an internal change; if the node broadcasts a message, all the adjacent nodes able to receive it will change their states, and if the node performs an internal move, then it will be the only one to change its state to a new state, if it's probabilistic state a probabilistic move will then follow. Observe that the topology can hence possibly change at each step of the Player 1. Finally, in our model, there is no creation neither deletion of nodes, hence along a path in the associated game the number of nodes in the graphs is fixed. We now formalize this intuition.

Let $\mathcal{P} = (Q, Q^{(1)}, Q^{(P)}, q_0, \Sigma, \Delta, \Delta^{\mathsf{int}})$ be a probabilistic protocol. The set of configurations $\Gamma_{\mathcal{P}}$ of the network built over $\mathcal{P}$ is a set of $(Q \times \{\bot, \top\})$-graphs formally defined as follows: $\Gamma_{\mathcal{P}}^{(1)} = \{(V, E, L) \in \mathcal{G}_{Q^{(1)} \times \{\bot, \top\}} \mid card(\{v \in V \mid L(v) \in Q^{(1)} \times \{\top\}\}) \leq 1\}$ and $\Gamma_{\mathcal{P}}^{(p)} = \{(V, E, L) \in \mathcal{G}_{Q \times \{\bot\}} \mid card(\{v \in V \mid L(v) \in Q^{(P)} \times \{\bot\}\}) = 1\}$ and $\Gamma_{\mathcal{P}} = \Gamma^{(1)} \cup \Gamma^{(p)}$. Hence in the configurations of Player 1, there is no node labelled with probabilistic state and at most one node labelled with $\top$ (it is the chosen node for the action to be performed) and in the probabilistic configurations no node is labelled with $\top$ and exactly one node is labelled with a probabilistic state. For this last set of configurations, the intuition is that when in the network one node changes its state to a probabilistic one then the network goes in a configuration in $\Gamma_{\mathcal{P}}^{(p)}$ from which it performs a probabilistic choice for the next possible state of the considered node.

The semantics of the network built over $\mathcal{P}$ is then given in term of the $1 - \frac{1}{2}$ player game $\mathcal{M}_{\mathcal{P}} = (\Gamma_{\mathcal{P}}, \Gamma_{\mathcal{P}}^{(1)}, \Gamma_{\mathcal{P}}^{(p)}, \rightarrow_{\mathcal{P}}, prob_{\mathcal{P}})$ where:

- $\rightarrow_{\mathcal{P}} \subseteq \Gamma_{\mathcal{P}}^{(1)} \times \Gamma_{\mathcal{P}}$ is defined as follows, for all $\gamma = (V, E, L)$ in $\Gamma_{\mathcal{P}}^{(1)}$, all $\gamma' = (V', E', L')$ in $\Gamma_{\mathcal{P}}$, we have $\gamma \rightarrow_{\mathcal{P}} \gamma'$ iff one of the following conditions hold:
  **Reconfiguration and process choice:** $\gamma \in \mathcal{G}_{Q^{(1)} \times \{\bot\}}$, $V' = V$ and there exists a vertex $v \in V$ and a state $q \in$ *ActStates* such that $L(v) = (q, \bot)$ and $L'(v) = (q, \top)$ and for all $v' \in V \setminus \{v\}$, $L(v') = L'(v')$;

**Internal:** $\gamma \in \Gamma_{\mathcal{P}}^{(1)}$, $V' = V$, $E' = E$ and there exists $v \in V$, $q \in Q^{(1)}$ and $q' \in Q$ such that $L(v) = (q, \top)$, $L'(v) = (q', \bot)$ and $(q, \varepsilon, q') \in \Delta$, and for all $v' \in V \setminus \{v\}$, $L'(v') = L(v')$;

**Communication:** $\gamma' \in \Gamma_{\mathcal{P}}^{(1)}$, $V' = V$, $E' = E$ and there exists $v \in V, q, q' \in Q^{(1)}$ and $a \in \Sigma$ such that $L(v) = (q, \top)$, $L'(v) = (q', \bot)$, $(q, !!a, q') \in \Delta$ and for every $v' \in V \setminus \{v\}$ with $L(v') = (q'', \bot)$, if $v \sim v'$ and $R_a(q'') \neq \emptyset$ then $L'(v') = (q''', \bot)$ with $q''' \in R_a(q'')$ and otherwise $L'(v') = L(v')$;

– $prob_{\mathcal{P}} : \Gamma_{\mathcal{P}}^{(p)} \mapsto \mathsf{Dist}(\Gamma_{\mathcal{P}}^{(1)})$ is defined as follows, for all $\gamma = (V, E, L) \in \Gamma_{\mathcal{P}}^{(p)}$, we have : if $v \in V$ is the unique vertex such that $L(v) \in Q^{(P)} \times \{\bot\}$ and if $\Delta^{\mathsf{int}}(L(v)) = \mu$, then for all $\gamma' = (V', E', L') \in \Gamma_{\mathcal{P}}$, if $V' = V$ and $E' = E$ and for all $v' \in V \setminus \{v\}$, $L'(v') = L(v)$ and then $prob_{\mathcal{P}}(\gamma)(\gamma') = \mu(q')$ where $(q, \bot) = L(v)$ and $(q', \bot) = L'(v)$ , and otherwise $prob_{\mathcal{P}}(\gamma)(\gamma') = 0$.

Finally we will denote by $\Gamma_{\mathcal{P},0}$ the set of initial configurations in which all the vertices are labelled with $(q_0, \bot)$. We point out the fact that since we do not impose any restriction on the size of the $Q$-graphs, the $1-\frac{1}{2}$ player game $\mathcal{M}_{\mathcal{P}}$ has hence an infinite number of configurations. However the number of configurations reachable from an initial configuration $\gamma \in \Gamma_{\mathcal{P},0}$ since the number of states in a probabilistic protocol is finite.

A simple example of probabilistic protocol is represented on Figure 1. The initial state is $q_0$ and the only probabilistic state is $q_p$. From $q_r$ the broadcast of a message $a$ leads back to $q_0$, and this message can be received from $q_l$ to reach the target $q_f$.

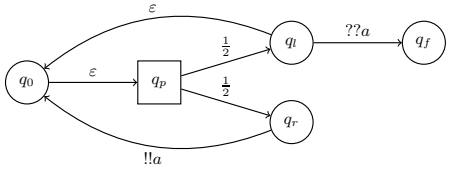


Fig. 1: Simple example of probabilistic protocol

### 2.3 Qualitative reachability problems

The problems we propose to investigate are qualitative ones where we will compare the probability of reaching a particular state in a network built over a probabilistic protocol with 0 or 1. Given a probabilistic protocol $\mathcal{P} = (Q, Q^{(1)}, Q^{(P)}, q_0, \Sigma, \Delta, \Delta^{\mathsf{int}})$ and a state $q_f \in Q$, we denote by $\Diamond q_f$ the set of all maximal paths of $\mathcal{M}_{\mathcal{P}}$ of the form $\gamma_0 \cdot \gamma_1 \cdots$ such that there exists $i \in \mathbb{N}$ verifying $(q_f, \bot) \in L(\gamma_i)$, i.e. the set of paths which eventually reach a graph where a node is labelled with the state $q_f$. It is well known that such a set of paths is measurable (see [5] for instance). We are now ready to provide the definition of the different qualitative reachability problems that we will study. Given $\mathtt{opt} \in \{\min, \max\}$, $\mathtt{b} \in \{0, 1\}$ and $\sim \in \{<, =, >\}$, let $\textsc{Reach}_{\mathtt{opt}}^{\sim \mathtt{b}}$ be the following problem:

**Input**: A probabilistic protocol $\mathcal{P}$, and a control state $q_f \in Q$.
**Question**: Does there exists an initial configuration $\gamma_0$ such that $\mathbb{P}_{\mathtt{opt}}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \Diamond q_f) \sim \mathsf{b}$ ?

Remark that this problem is parameterized by the initial configuration and this is the point that make this problem difficult to solve (and that leads to undecidability in the case with no probabilistic choice and no reconfiguration in the network [11]). However for a fixed given initial configuration, the problem boils down to the analysis of a finite $1-\frac{1}{2}$ player game as already mentioned. As a consequence, the minimum and maximum (rather than infimum and supremum) probabilities are well-defined when an initial configuration $\gamma_0$ is fixed; moreover, these extremal values are met for memoryless schedulers.

## 3 Networks of parity reconfigurable protocols

### 3.1 Parity, safety and safety/parity games

We first introduce 2 player turn-based zero-sum games with various winning objectives. For technical reasons that will become clear in the sequel, our definition differs from the classical one: colors (or parities) label the edges rather than the vertices.

**Definition 4 (2 player game).** *A* 2 player game *is a tuple* $\mathsf{G} = (\Lambda, \Lambda^{(1)}, \Lambda^{(2)}, T, \mathsf{col}, \mathsf{safe})$ *where* $\Lambda$ *is a denumerable set of configurations, partitioned into* $\Lambda^{(1)}$ *and* $\Lambda^{(2)}$*, configurations of Player 1 and* 2*, respectively;* $T \subseteq \Lambda \times \Lambda$ *is a set of directed edges;* $\mathsf{col} : T \to \mathbb{N}$ *is the coloring function such that* $\mathsf{col}(T)$ *is finite;* $\mathsf{safe} \subseteq T$ *is a subset of safe edges.*

As in the case of $1-\frac{1}{2}$ player game, we define the notions of paths and the equivalent to schedulers: strategies. A *finite path* $\rho$ is a finite sequence of configurations $\lambda_0\lambda_1\cdots\lambda_n \in \Lambda^*$ such that $(\lambda_i, \lambda_{i+1}) \in T$ for all $0 \leq i < n$. Such a path is said to start at configuration $\lambda_0$. An *infinite path* is an infinite sequence $\rho \in \Lambda^\omega$ such that any finite prefix of $\rho$ is a finite path. Similarly to paths in $1-\frac{1}{2}$ player game, *maximal paths* in $\mathsf{G}$ are infinite paths or finite paths ending in a deadlock configuration.

A *strategy* for Player 1 dictates its choices in configurations of $\Lambda^{(1)}$. More precisely, a strategy for Player 1 in the game $\mathsf{G} = (\Lambda, \Lambda^{(1)}, \Lambda^{(2)}, T, \mathsf{col}, \mathsf{safe})$ is a function $\sigma : \Lambda^*\Lambda^{(1)} \mapsto \Lambda$ such that for every finite path $\rho$ and $\lambda \in \Lambda^{(1)}$, $(\lambda, \sigma(\rho\lambda)) \in T$. Strategies $\tau : \Lambda^*\Lambda^{(2)} \to \Lambda$ for Player 2 are defined symmetrically, and we write $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(2)}$ for the set of strategies for each player. A *strategy profile* is a pair of strategies, one for each player. Given a strategy profile $(\sigma, \tau)$ and an initial configuration $\lambda_0$, the game $\mathsf{G}$ gives rise to the following maximal path, aka the *play*, $\rho(\mathsf{G}, \lambda_0, \sigma, \tau) = \lambda_0\lambda_1\cdots$ such that for all $i \in \mathbb{N}$, if $\lambda_i \in \Lambda^{(1)}$ then $\lambda_{i+1} = \sigma(\lambda_0 \ldots \lambda_i)$, otherwise $\lambda_{i+1} = \tau(\lambda_0 \ldots \lambda_i)$.

*Remark 2.* Similarly to the case of schedulers in $1-\frac{1}{2}$ player game, (see Remark 1), when convenient the players' strategies can be alternatively defined as strategies with memory. In this case, a strategy for Player 1 with memory $M$ is given by means of a *strategic function* $\sigma_M : \Lambda^{(1)} \times M \to \Lambda$, an *update function* $U_M : \Lambda^{(1)} \times M \times \Lambda \to M$, and an initialization function $I_M : \Lambda \to M$.

The winning condition for Player 1 is a subset of plays $\mathsf{Win} \subseteq \Lambda^* \cup \Lambda^\omega$. In this paper, we characterize winning conditions through safety, parity objectives and combinations of these two objectives, respectively denoted by $\mathsf{Win_s}$, $\mathsf{Win_p}$ and $\mathsf{Win_{sp}}$, and defined as follows:

$$\mathsf{Win_s} = \{\rho \in \Lambda^* \cup \Lambda^\omega \mid \forall 0 \leq i < |\rho| - 1.(\rho(i), \rho(i+1)) \in \mathsf{safe} \text{ and } \rho \text{ is maximal}\}$$
$$\mathsf{Win_p} = \{\rho \in \Lambda^\omega \mid \max\{n \in \mathbb{N} \mid \forall i \geq 0.\exists j \geq i.\mathsf{col}((\rho(j), \rho(j+1))) = n\} \text{is even}\}$$
$$\mathsf{Win_{sp}} = (\mathsf{Win_p} \cap \mathsf{Win_s}) \cup (\Lambda^* \cap \mathsf{Win_s})$$

The safety objective denotes the maximal path that use only edges in $\mathsf{safe}$, the parity objective denotes the infinite paths for which the maximum color visited infinitely often is even and the safety-parity objective denotes the set of safe maximal paths which have to respect the parity objectives when they are infinite. We say that a play $\rho$ is *winning for Player 1* for an objective $\mathsf{Win} \subseteq \Lambda^* \cup \Lambda^\omega$ if $\rho \in \mathsf{Win}$, in the other case it is winning for Player 2. Last, a strategy $\sigma$ for Player 1 is a *winning strategy* from configuration $\lambda_0$ if for every strategy $\tau$ of Player 2, the play $\rho(\mathsf{G}, \lambda_0, \sigma, \tau)$ is winning for Player 1.

### 3.2 Networks of parity reconfigurable protocols

We now come to the definition of networks of parity reconfigurable protocols, introducing their syntax and semantics. The main differences with the probabilistic protocol introduced previously lies in the introduction of states for Player 2, the use of colors associated to the transition relation and the removal of the probabilistic transitions.

**Definition 5 (Parity protocol).** *A* parity protocol *is as a tuple* $\mathbf{P} = (Q, Q^{(1)}, Q^{(2)}, q_0, \Sigma, \Delta, \mathsf{col}, \mathsf{safe})$ *where $Q$ is a finite set of control states partitioned into $Q^{(1)}$ and $Q^{(2)}$; $q_0 \in Q^{(1)}$ is the initial control state; $\Sigma$ is a finite message alphabet; $\Delta \subseteq \left(Q^{(1)} \times \left(\{!!a, ??a \mid a \in \Sigma\} \cup \{\varepsilon\}\right) \times Q\right) \cup \left(Q^{(2)} \times \{\varepsilon\} \times Q\right)$ is the transition relation;* $\mathsf{col} : \Delta \to \mathbb{N}$ *is the coloring function;* $\mathsf{safe} \subseteq \Delta$ *is a set of safe edges.*

Note that the roles of Player 1 and Player 2 are not symmetric: only Player 1 can initiate a communication, and Player 2 performs only internal actions. As for probabilistic protocols, given a state $q \in Q$ and a message $a \in \Sigma$, we define the set $R_a(q) = \{q' \in Q \mid (q, ??a, q') \in \Delta\}$ containing the control states that can be reached in $\mathcal{P}$ from the state $q$ after receiving the message $a$. We also denote by *ActStates* the set of states $\{q \in Q \mid \exists.(q, !!a, q') \in \Delta \text{ or } \exists.(q, \varepsilon, q') \in \Delta\}$ of states from which a broadcast or an internal action can be performed.

The semantics associated to a parity protocol is given in term of a 2 player game whose definition is similar to the $1-\frac{1}{2}$ player game associated to a probabilistic protocol. Here also the Player 1 has the ability to choose a communication topology and a node which will perform an action, and according to the control state labelling this node either Player 1 or Player 2 will then perform the next move. Let $\mathbf{P} = (Q, Q^{(1)}, Q^{(2)}, q_0, \Sigma, \Delta, \mathsf{col}, \mathsf{safe})$ be parity protocol. The set of configurations $\Lambda_{\mathbf{P}}$ of the network built over $\mathbf{P}$ is defined as follows: $\Lambda_{\mathbf{P}} = \{(V, E, L) \in \mathcal{G}_{Q\times\{\bot,\top\}} \mid card(\{v \in V \mid L(v) \in Q \times \{\top\}\}) \leq 1\}$ and then we have $\Lambda^{(1)}_{\mathbf{P}} = \mathcal{G}_{Q\times\{\bot\}} \cup \{(V, E, L) \in \Lambda_{\mathbf{P}} \mid card(\{v \in V \mid L(v) \in Q^{(1)} \times \{\top\}\}) = 1\}$and $\Lambda^{(2)} = \{(V, E, L) \in \Lambda_{\mathbf{P}} \mid card(\{v \in V \mid L(v) \in Q^{(2)} \times \{\top\}\}) = 1\}$. We observe that Player 1 owns vertices where no node is tagged $\top$, and Player $i$ owns the vertices where the node tagged $\top$ is in a Player $i$ control state.

The semantics of the network built over $\mathbf{P}$ is then given in term of the 2 player game $\mathsf{G}_{\mathbf{P}} = (\Lambda_{\mathbf{P}}, \Lambda^{(1)}_{\mathbf{P}}, \Lambda^{(2)}_{\mathbf{P}}, T_{\mathbf{P}}, \mathsf{col}_{\mathbf{P}}, \mathsf{safe}_{\mathbf{P}})$ where $T_{\mathbf{P}} \subseteq \Lambda_{\mathbf{P}} \times \Lambda_{\mathbf{P}}$, $\mathsf{col}_{\mathbf{P}} : T_{\mathbf{P}} \to \mathbb{N}$ and $\mathsf{safe}_{\mathbf{P}} \subseteq T_{\mathbf{P}}$ are defined as follows: for all $\lambda = (V, E, L)$ and $\lambda' = (V', E', L')$ in $\Lambda_{\mathbf{P}}$, we have $(\lambda, \lambda') \in T_{\mathbf{P}}$ iff one the following conditions is satisfied:

**Reconfiguration and process choice** $\lambda \in \mathcal{G}_{Q\times\{\bot\}}$ and there exists a vertex $v \in V$ and a state $q \in \mathit{ActStates}$ such that $L(v) = (q, \bot)$ and $L'(v) = (q, \top)$ and for all $v' \in V \setminus \{v\}$, $L(v') = L'(v')$; in this case, $\mathsf{col}_{\mathbf{P}}(\lambda, \lambda') = 0$ and $(\lambda, \lambda') \in \mathsf{safe}_{\mathbf{P}}$;

**Internal** $V' = V$, $E' = E$ and there exists $v \in V$, $q, q' \in Q$ such that $L(v) = (q, \top)$, $L'(v) = (q', \bot)$ and $(q, \varepsilon, q') \in \Delta$, and for all $v' \in V \setminus \{v\}$, $L'(v') = L(v')$; in this case, $\mathsf{col}_{\mathbf{P}}(\lambda, \lambda') = \mathsf{col}(q, \varepsilon, q')$ and $(\lambda, \lambda') \in \mathsf{safe}_{\mathbf{P}}$ iff $(q, \varepsilon, q') \in \mathsf{safe}$;

**Communication** $V' = V$, $E' = E$ and there exists $v \in V$, $q, q' \in Q$ and $a \in \Sigma$ such that $L(v) = (q, \top)$, $L'(v) = (q', \bot)$, $(q, !!a, q') \in \Delta$ and for every $v' \in V \setminus \{v\}$ with $L(v') = (q'', \bot)$, if $v \sim v'$ and $R_a(q'') \neq \emptyset$ then $L'(v') = (q''', \bot)$ with $q''' \in R_a(q'')$ and otherwise $L'(v') = L(v')$ ; in this case, $\mathsf{col}_{\mathsf{G}}(\lambda, \lambda') = \mathsf{col}(q, !!a, q')$ and $(\lambda, \lambda') \in \mathsf{safe}_{\mathbf{P}}$ iff we have $(q, !!a, q') \in \mathsf{safe}$ and for all the reception transitions $(q'', ??a, q''')$ used, $(q'', ??a, q''') \in \mathsf{safe}$.

Finally, we will say that a configuration $\lambda = (V, E, L)$ is initial if $L(v) = (q_0, \bot)$ for all $v \in V$ and we will write $\Lambda_{\mathbf{P},0}$ the set of initial configurations. Note that here also the number of initial configuration is infinite. We are now able to define the game problem for parity protocol as follows:

**Input**: A parity protocol $\mathbf{P}$, and a winning condition $\mathsf{Win}$.
**Question**: Does there exists an initial configuration $\lambda_0 \in \Lambda_{\mathbf{P},0}$ such that Player 1 has a winning strategy in $\mathsf{G}_{\mathbf{P}}$ from $\lambda_0$?

### 3.3 Restricting the strategies of Player 2

In order to solve the game problem for parity protocols, we first show that we can restrict the strategies of Player 2 to strategies that always choose from a

given control state the same successor, independently of the configuration, or the history in the game.

In the sequel we consider a parity protocol $\mathbf{P} = (Q, Q_1, Q_2, q_0, \Sigma, \Delta, \mathsf{col}, \mathsf{safe})$. We begin by defining what are the local positional strategies for Player 2 in $\mathsf{G}_\mathbf{P}$. A *local behavior* for Player 2 in $\mathsf{G}_\mathbf{P}$ is a function $b : (Q_2 \cap \mathit{ActStates}) \mapsto \Delta$ such that for all $q \in Q_2 \cap \mathit{ActStates}$, $b(q) \in \{(q, \varepsilon, q') \mid (q, \varepsilon, q') \in \Delta\}$. Such a local behavior induced what we will call a *local strategy* $\tau_b$ for Player 2 in $\mathsf{G}_\mathbf{P}$ defined as follows: let $\rho$ be a finite path in $\Lambda^*_\mathbf{P}$ and $\lambda = (V, E, L) \in \Lambda^{(2)}$, if $v$ is the unique vertex in $V$ such that $L(v) \in Q^{(2)} \times \{\top\}$ and if $L(v) = (q, \top)$, we have $\tau_b(\rho\lambda) = \lambda'$ where $\lambda'$ is the unique configuration obtained from $\lambda$ by applying accordingly to the definition of $\mathsf{G}_\mathbf{P}$ the rule corresponding to $b(q)$ (i.e. the internal action initiated from vertex $v$). We denote by $\mathcal{S}_\mathsf{l}^{(2)}$ the set of local strategies for Player 2. Note that there are a finite number of states and of edges in $\mathbf{P}$, the set $\mathcal{S}_\mathsf{l}^{(2)}$ is thus finite and contain at most $\mathit{card}(\Delta)$ strategies. The next lemma shows that we can restrict Player 2 to follow only local strategies in order to solve the game problem for $\mathbf{P}$ when considering the previously introduced winning objectives.

**Lemma 1.** *For* $\mathsf{Win} \in \{\mathsf{Win_s}, \mathsf{Win_p}, \mathsf{Win_{sp}}\}$*, we have* $\exists \lambda_0 \in \Lambda_{\mathbf{P},0}.\ \exists \sigma \in \mathcal{S}^{(1)}.\ \forall \tau \in \mathcal{S}^{(2)},\ \rho(\mathsf{G}_\mathbf{P}, \lambda_0, \sigma, \tau) \in \mathsf{Win} \iff \forall \tau \in \mathcal{S}_l^{(2)}.\ \exists \lambda_0 \in \Lambda_{\mathbf{P},0}.\ \exists \sigma \in \mathcal{S}^{(1)}, \rho(\mathsf{G}_\mathbf{P}, \lambda_0, \sigma, \tau) \in \mathsf{Win}$

### 3.4 Solving the game against local strategies

In this section, we explain how to decide whether there exists an initial configuration and a strategy for Player 1 which is winning against a fixed local strategy. We consider a parity protocol $\mathbf{P} = (Q, Q_1, Q_2, q_0, \Sigma, \Delta, \mathsf{col}, \mathsf{safe})$ and a local behavior $b$. From this parity protocol we build a parity protocol $\mathbf{P}' = (Q, q_0, \Sigma, \Delta', \mathsf{col}', \mathsf{safe}')$ by removing the choices of Player 2 not corresponding to $b$ and by merging states of Player 1 an states of Player 2; this protocol is formally defined as follows: $\Delta' \subseteq \Delta$ and $(q, \mathbf{a}, q') \in \Delta'$ iff $q \in Q^{(1)}$ and $(q, \mathbf{a}, q') \in \Delta'$, or, $q \in Q^{(2)}$ and $b(q)$ is defined and equal to $(q, \mathbf{a}, q')$, furthermore $\mathsf{col}'$ is the restriction of $\mathsf{col}$ to $\Delta'$ and $\mathsf{safe}' = \Delta' \cap \mathsf{safe}$. The following lemma states the relation between $\mathbf{P}$ and $\mathbf{P}'$.

**Lemma 2.** *For* $\mathsf{Win} \in \{\mathsf{Win_s}, \mathsf{Win_p}, \mathsf{Win_{sp}}\}$*, there exists a path* $\rho$ *in* $\mathsf{G}_{\mathbf{P}'}$ *starting from an initial configuration and such that* $\rho \in \mathsf{Win}$ *iff* $\exists \lambda_0 \in \Lambda_{\mathbf{P},0}.\ \exists \sigma \in \mathcal{S}^{(1)}, \rho(\mathsf{G}_\mathbf{P}, \lambda_0, \sigma, \tau_b) \in \mathsf{Win}$.

We will now show how to decide the two following properties on $\mathsf{G}_{\mathbf{P}'}$: whether there exists a maximal finite path in $\mathsf{Win_s}$ starting from an initial configuration in $\mathsf{G}_{\mathbf{P}'}$ and whether there exists an infinite path $\in \mathsf{Win_p} \cap \mathsf{Win_s}$ starting from an initial configuration. Once, we will have shown how to solve these two problems, this will provide us, for each winning condition, an algorithm to decide whether there exists a winning path in $\mathsf{G}_{\mathbf{P}'}$.

We now provide the idea to solve the first problem. By definition, a finite path $\rho = \lambda_0 \lambda_1 \cdots \lambda_n$ in the game $\mathsf{G}_{\mathbf{P}'}$ is maximal if there does not exist a

configuration $\lambda \in \Lambda_{\mathbf{P}'}$ such that $(\lambda_n, \lambda) \in T'_{\mathbf{P}}$ and according to the semantics of the parity protocol $\mathbf{P}'$, this can be the case if and only if $\lambda_n = (V, E, L)$ where $L(\lambda_n) \subseteq (Q \times \{\perp\}) \setminus (\mathit{ActStates} \times \{\perp\})$. In [10], it is shown that, for reconfigurable broadcast protocol, one can decide in NP whether, given a set of protocol states, there exists a path starting from an initial configuration reaching a configuration in which no vertices are labelled by the given states. We deduce the next lemma.

**Lemma 3.** *The problem of deciding whether there exists in* $\mathsf{G}_{\mathbf{P}'}$ *a finite maximal path belonging to* $\mathsf{Win}_\mathsf{s}$ *starting from an initial configuration is in* NP.

We now show how to decide in polynomial time whether there exists an infinite path in $\mathsf{Win}_\mathsf{p} \cap \mathsf{Win}_\mathsf{s}$ starting from an initial configuration. The idea is the following. We begin by removing in $\mathbf{P}'$ the unsafe edges. Then we compute in polynomial time all the reachable control states using an algorithm of [10]. Then from [10] we also know that there exists a reachable configuration exhibiting as many reachable states as we want. Finally, we look for an infinite loop respecting the parity condition from such a configuration. This is done by using a counting abstraction method which translates the system into a Vector Addition System with States (VASS) and then by looking in this VASS for a cycle whose effect on each of the manipulated value is 0 (i.e. a cycle whose edge's labels sum to 0) and this is can be done in polynomial time thanks to [15].

**Lemma 4.** *The problem of deciding whether there exists an infinite path* $\rho$ *starting from an initial configuration in* $\mathsf{G}_{\mathbf{P}'}$ *such that* $\rho \in \mathsf{Win}_\mathsf{p} \cap \mathsf{Win}_\mathsf{s}$ *is in* PTIME.

By Lemma 2 we know hence that: there is an NP algorithm to decide whether $\exists \lambda_0 \in \Lambda_{\mathbf{P},0}.\ \exists \sigma \in \mathcal{S}^{(1)}, \rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau_b) \in \mathsf{Win}_\mathsf{s}$ (in fact this reduces to looking for a finite maximal path belonging to $\mathsf{Win}_\mathsf{s}$ and use Lemma 3 or an infinite safe path, in this case we put all the colors to 0 and we use Lemma 4); there is a polynomial time algorithm to decide the same problem with $\mathsf{Win}_\mathsf{p}$ instead of $\mathsf{Win}_\mathsf{s}$ (use Lemma 3 with all the transitions considered as safe) and there is an NP algorithm for the same problem with $\mathsf{Win}_\mathsf{sp}$ (here again we look either for a finite maximal safe path and use Lemma 3 or for an infinite safe path satisfying the parity condition and we use Lemma 4).

So now since the number of local strategies is finite, this gives us non deterministic algorithms to solve whether $\exists \tau \in \mathcal{S}_\mathsf{l}^{(2)}.\ \forall \lambda_0 \in \Lambda_{\mathbf{P},0}.\ \forall \sigma \in \mathcal{S}^{(1)}, \rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau) \notin \mathsf{Win}$ with $\mathsf{Win} \in \{\mathsf{Win}_\mathsf{s}, \mathsf{Win}_\mathsf{p}, \mathsf{Win}_\mathsf{sp}\}$. Note that for $\mathsf{Win}_\mathsf{p}$ we will have an NP algorithm and for $\mathsf{Win}_\mathsf{s}$ and $\mathsf{Win}_\mathsf{sp}$, an NP algorithm using an NP oracle (i.e. an algorithm in $\mathrm{NP}^{\mathrm{NP}} = \Sigma_2^P$). Hence thanks to Lemma 1, we are able to state the main result of this section.

**Theorem 1.** *For safety and safety-parity objectives, the game problem for parity protocol is decidable and in* $\Pi_2^{\mathrm{P}}$ *(=co-*$\mathrm{NP}^{\mathrm{NP}}$*), and in co-*NP *for parity objectives.*

## 4 Solving probabilistic networks

In this section we solve the qualitative reachability problems for probabilistic reconfigurable broadcast networks. The most involved case is $\textsc{Reach}^{=1}_{\max}$ for which we reduce to games on parity protocols with a parity winning condition.

### 4.1 $\mathbf{Reach}^{=1}_{\max}$

Let us now discuss the most involved case, $\textsc{Reach}^{=1}_{\max}$, and show how to reduce it to the game problem for parity protocols with a parity winning condition. From $\mathcal{P} = (Q, Q^{(1)}, Q^{(P)}, q_0, \Sigma, \Delta, \Delta^{\mathsf{int}})$ a probabilistic protocol and $q_f \in Q$ a control state, we derive the parity protocol $\mathbf{P} = (Q_{\mathbf{P}}, Q^{(1)}_{\mathbf{P}}, Q^{(2)}_{\mathbf{P}}, q_{0\mathbf{P}}, \Sigma_{\mathbf{P}}, \Delta_{\mathbf{P}}, \mathsf{col}, \mathsf{safe})$ as follows: $Q_{\mathbf{P}} = Q^{(1)}_{\mathbf{P}} \cup Q^{(2)}_{\mathbf{P}}$, $Q^{(1)}_{\mathbf{P}} = Q^{(1)} \cup Q^{(P)} \times \{1\}$, $Q^{(2)}_{\mathbf{P}} = Q^{(P)} \times \{2\}$, and $q_{0\mathbf{P}} = q_0$; $\Sigma_{\mathbf{P}} = \Sigma$; $\Delta_{\mathbf{P}} = \big(Q^{(1)} \times \{!!a, ??a \mid a \in \Sigma\} \times Q^{(1)} \cap \Delta\big) \cup \{(q_f, \varepsilon, q_f)\} \cup \{(q, \varepsilon, (q', 2)), ((q', i), \varepsilon, q'), ((q, 2), \varepsilon, (q, 1)) \mid (q, \varepsilon, q') \in \Delta, i \in \{1, 2\}\} \cup \{((q, i), \varepsilon, q') \mid \Delta^{\mathsf{int}}(q)(q') > 0, i \in \{2, 3\}\}$; and last $\mathsf{col}((q_f, \varepsilon, q_f)) = 2$, $\mathsf{col}(((q, 2), \varepsilon, q') = 2$ and otherwise $\mathsf{col}(\delta) = 1$.

Intuitively, all random choices corresponding to internal actions in $\mathcal{P}$ are replaced in $\mathbf{P}$ with choices for Player 2, where either he decides the outcome of the probabilistic choice, or he lets Player 1 choose. Only transitions where Player 2 makes the decision corresponding to a probabilistic choice and the self loop on the state $q_f$ have parity 2. Figure 2 illustrates this reduction on the example probabilistic protocol from Figure 1. This construction ensures:

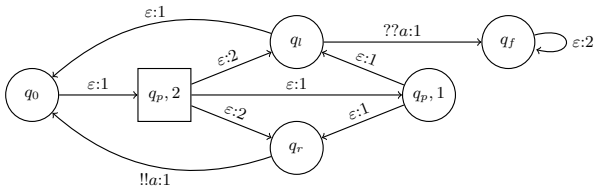


Fig. 2: Parity protocol for the probabilistic protocol from Figure 1.

**Proposition 1.** *$\exists \lambda_0 \in \Lambda_{\mathbf{P},0}.\ \exists \sigma \in \mathcal{S}^{(1)}.\ \forall \tau \in \mathcal{S}^{(2)},\ \rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau) \in \mathsf{Win}_{\mathsf{p}}$ if and only if $\exists \gamma_0 \in \Gamma_{\mathcal{P},0}.\ \mathbb{P}_{\max}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \Diamond q_f) = 1$.*

*Proof (sketch).* The easiest direction is from left to right. Assuming that some scheduler $\pi$ ensures to reach $q_f$ with probability 1, one builds a winning strategy $\sigma$ for the parity objective as follows. When Player 2 makes a decision corresponding to a probabilistic choice in $\mathcal{P}$, the strategy chooses to play this probabilistic transition. Now, when Player 1 needs to make a decision in some configuration $\lambda$ where there is a vertex $v$ labelled by $((q, 1), \top) \in Q^{(P)} \times \{1\} \times \{\top\}$, the strategy is to play along a shortest path respecting $\pi$ from $\gamma$ to a configuration containing $q_f$, where $\gamma$ is defined as $\lambda$ but the label of $v$ is $q$. Assuming that $\pi$ reaches $q_f$ with probability 1, such a path must exist for every reachable configuration in the game. This definition of $\sigma$ ensures to eventually reach $q_f$ under the assumption that Player 2, from some point on, always lets Player 1 decide in configurations corresponding to probabilistic states of $\mathcal{P}$.

Let us now briefly explain how the right to left implication works. Notice that if Player 2 always chooses transitions with parity 1 (thus letting Player 1

decide the outcome of probabilistic choices), the only way for Player 1 to win is to reach $q_f$, and from there use the self loop to ensure the parity condition. As a consequence, from any reachable configuration, the target state $q_f$ must be reachable.
From a winning strategy $\sigma$, we define a scheduler $\pi$ that mimics the choices of $\sigma$ on several copies of the network. The difficulty comes from the transformation of choices of Player 1 in states of the form $(q, 1) \in Q^{(P)} \times \{1\}$ into probabilistic choices. Indeed, the outcome of these random choices cannot surely match the decision of Player 1. The idea is the following: when a probabilistic choice in $\mathcal{P}$ does not agree with the decision of Player 1 in $\mathbf{P}$, this "wrong choice" is attributed to Player 2. The multiple copies thus account for memories of the "wrong choices", and a process performing such a choice is moved to a copy where the choice was made by Player 2. With probability 1, eventually a "good choice" is made, and the 1-1/2 player game can continue in the original copy of the network. Therefore, almost-surely the play will end in a given copy, where Player 1 always decides, and thus $q_f$ is reached. □

**Theorem 2.** $\textsc{Reach}^{=1}_{\max}$ *is co-*NP*-complete.*

*Proof (sketch).* The co-NP membership is a consequence of Proposition 1 and Theorem 1, and we now establish the matching lower-bound. To establish the coNP-hardness we reduce the unsatisfiability problem to $\textsc{Reach}^{=1}_{\max}$. From $\varphi$ a formula in conjunctive normal form, we define a probabilistic protocol $\mathcal{P}_\varphi$ and a control state $q_f$ such that $\varphi$ is unsatisfiable if and only if there exists an initial configuration $\gamma_0 \in \Gamma_{\mathcal{P},0}$ and a scheduler $\pi$ such that $\mathbb{P}(\mathcal{M}_\mathcal{P}, \gamma_0, \pi, \Diamond q_f) = 1$.

We provide here the construction on an example in Figure 3, the general definition is given in Appendix. For simplicity, the initial state $q_0$ of the probabilistic protocol is duplicated in the picture. The idea, if $\varphi$ is unsatisfiable, is to generate a random affectation of the variables (using the gadgets represented bottom of the Figure), which will necessarily violate a close of $\varphi$. Choosing then this clause in the above part of the protocol allows to reach state $\mathsf{r}_1$, and from there to reach $q_f$ with probability half. Iterating this process, the target can be almost-surely reached. The converse implication relies on the fact that if $\varphi$ is satisfiable, there is a positive probability to generate a valuation satisfying it, and then not to be able to reach $\mathsf{r}_1$, a necessary condition to reach $q_f$. Therefore, the maximum probability to reach the target is smaller than 1 in this case. □

### 4.2 Other cases

The decision problems $\textsc{Reach}^{=0}_{\min}$ [resp. $\textsc{Reach}^{<1}_{\min}$] can be reduced to a game problem for parity protocols with a safety [resp. safety/parity] winning condition. From a probabilistic protocol $\mathcal{P}$, for $\textsc{Reach}^{=0}_{\min}$, we build a parity protocol $\mathbf{P}$ where all random choices in $\mathcal{P}$ are replaced in $\mathbf{P}$ with choices for Player 2. The transitions with target $q_f$ are the only ones that do not belong to the safe set $\mathsf{safe}$. For $\textsc{Reach}^{<1}_{\min}$, $\mathbf{P}$ consists of two copies of $\mathcal{P}$. In the first copy, all random choices are replaced with choices of Player 1, whereas in the second copy they

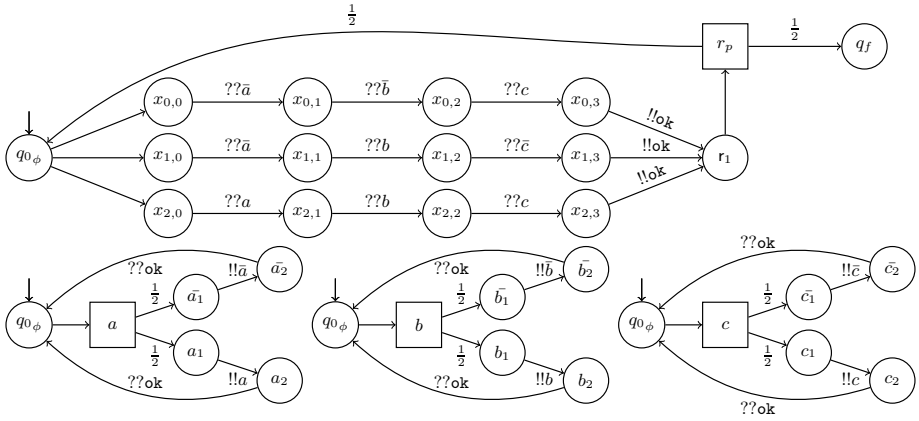


Fig. 3: Probabilistic protocol for the formula $\varphi = (a \vee b \vee \bar{c}) \wedge (a \vee \bar{b} \vee c) \wedge (\bar{a} \vee \bar{b} \vee \bar{c})$

are replaced with choices of Player 2. Also, at any time, one can move from the first to the second copy. The parity of transitions with target in the second copy is 2, and otherwise it is 1. Moreover, the only unsafe transitions are those with targer $q_f$. In these two cases, using Theorem 1, we obtain:

**Theorem 3.** $\textsc{Reach}^{=0}_{\min}$ *and* $\textsc{Reach}^{<1}_{\min}$ *are in* $\Pi^{\mathrm{P}}_2$.

The decidability and complexity of the remaining cases are established directly, without reducing to games on parity protocols. First of all, $\textsc{Reach}^{>0}_{\max}$ is interreducible to the reachability problem in *non-probabilistic* reconfigurable broadcast networks, known to be P-complete [10]. For the other decision problems we use a monotonicity property: intuitively, with more nodes, the probability to reach the target can only increase. The problems are then reduced to qualitative reachability problems in the finite state MDP for the network with a single process, and thus belong to PTIME.

**Theorem 4.** $\textsc{Reach}^{>0}_{\max}$, $\textsc{Reach}^{=0}_{\max}$, $\textsc{Reach}^{<1}_{\max}$, $\textsc{Reach}^{=1}_{\min}$ *and* $\textsc{Reach}^{>0}_{\min}$ *are in* PTIME.

## 5 Conclusion

In this paper we introduced probabilistic reconfigurable broadcast networks and studied parameterized qualitative reachability questions. The decidability of these verification questions are proved by a reduction to a 2-player games played on an infinite graphs, for which we provide decision algorithms. The complexities range from PTIME to $\textsc{coNP}^{\mathrm{NP}}$, as summarized in the table below.

| Problem | $\textsc{Reach}^{=0}_{\min}$ | $\textsc{Reach}^{<1}_{\min}$ | $\textsc{Reach}^{=1}_{\max}$ | others |
|---|---|---|---|---|
| Complexity | $\Pi^{\mathrm{P}}_2$ | $\Pi^{\mathrm{P}}_2$ | coNP-complete | PTIME |

In the future, we would like to find the precise complexity for $\text{REACH}^{=0}_{\min}$ and $\text{REACH}^{<1}_{\min}$ either by determining matching lower bounds or by improving the decision procedures. We will also study quantitative versions of the reachability problem. Finally we also believe that we could use our games played over reconfigurable broadcast protocols either to decide other properties on this family of systems or to analyze new models.

## References


1. P. A. Abdulla, M. F. Atig, and O. Rezine. Verification of directed acyclic ad hoc networks. In *FMOODS/FORTE'13*, volume 7892 of *LNCS*, pages 193–208. Springer, 2013.
2. P. A. Abdulla, K. Cerans, B. Jonsson, and Y.-K. Tsay. General decidability theorems for infinite-state systems. In *LICS'96*, pages 313–321. IEEE Computer Society, 1996.
3. P. A. Abdulla, N. B. Henda, and R. Mayr. Decisive markov chains. *Logical Methods in Computer Science*, 3(4), 2007.
4. C. Baier, N. Bertrand, and P. Schnoebelen. Verifying nondeterministic probabilistic channel systems against $\omega$-regular linear-time properties. *ACM Trans. Comput. Log.*, 9(1), 2007.
5. C. Baier and J.-P. Katoen. *Principles of Model Checking*. The MIT Press, 2008.
6. N. Bertrand and P. Fournier. Parameterized verification of many identical probabilistic timed processes. In *FSTTCS'13*, LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2013. To appear.
7. T. Brázdil, A. Kucera, and O. Strazovský. On the decidability of temporal properties of probabilistic pushdown automata. In *STACS*, volume 3404 of *LNCS*, pages 145–157. Springer, 2005.
8. K. Chatterjee, L. de Alfaro, M. Faella, and A. Legay. Qualitative logics and equivalences for probabilistic systems. *Logical Methods in Computer Science*, 5(2), 2009.
9. K. Chatterjee and L. Doyen. Energy parity games. *Theoretical Computer Science*, 458:49–60, 2012.
10. 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 fuer Informatik, 2012.
11. G. Delzanno, A. Sangnier, and G. Zavattaro. Parameterized verification of ad hoc networks. In *CONCUR'10*, volume 6269 of *LNCS*. Springer, 2010.
12. G. Delzanno, A. Sangnier, and G. Zavattaro. On the power of cliques in the parameterized verification of ad hoc networks. In *FoSSaCS'11*, volume 6604 of *LNCS*. Springer, 2011.
13. K. Etessami and M. Yannakakis. Recursive markov decision processes and recursive stochastic games. In *ICALP'05*, volume 3580 of *Lecture Notes in Computer Science*, pages 891–903. Springer, 2005.
14. A. Finkel and P. Schnoebelen. Well-structured transition systems everywhere! *Theor. Comput. Sci.*, 256(1-2):63–92, 2001.
15. S. R. Kosaraju and G. F. Sullivan. Detecting cycles in dynamic graphs in polynomial time (preliminary version). In *STOC'88*. ACM, 1988.

# Appendix

## A Details for Section 3

### A.1 Proof of Lemma 1

We first prove:

$$\exists \lambda_0 \in \Lambda_{\mathbf{P},0}.\ \exists \sigma \in \mathcal{S}^{(1)}.\ \forall \tau \in \mathcal{S}^{(2)},\ \rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau) \in \mathsf{Win}$$
$$\iff$$
$$\exists \lambda_0 \in \Lambda_{\mathbf{P},0}.\ \exists \sigma \in \mathcal{S}^{(1)}.\ \forall \tau \in \mathcal{S}^{(2)}_{\mathsf{l}},\ \rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau) \in \mathsf{Win}$$

Only the right-to-left implication deserves a proof, since $\mathcal{S}^{(2)}_{\mathsf{l}} \subseteq \mathcal{S}^{(2)}$. The proof shares some similarities with the one to establish that memoryless strategies are sufficient for Player 2 in energy parity games [9]. It is done by induction on $|Q_2|$, the number of states of Player 2 in the parity protocol.

We assume that there exists an initial configuration $\lambda_0 \in \Lambda_{\mathbf{P},0}$ and a strategy $\sigma$ for Player 1 such that for all local strategies $\tau$ of Player 2, $\rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau) \in \mathsf{Win}$.

If $Q_2 = \emptyset$, Player 2 makes no decision in the game, hence, the set of local strategies agrees with the set of all strategies and they are empty. Therefore, the base case is obvious.

Assume now that the property holds for $|Q_2| < n$, and let us consider a parity protocol $\mathbf{P}$ with $|Q_2| = n$. We suppose that $Q_2 = \{q_1, \cdots, q_n\}$. We first rule out the case where in $q_n$ a single internal transition is enabled, since in that case, Player 2 has no real choice in $q_n$, and $q_n$ can as well belong to $Q_1$, so that the induction hypothesis applies. Without loss of generality, we consider that two transitions $\delta_\ell = (q_n, \varepsilon, q_\ell)$ and $\delta_r = (q_n, \varepsilon, q_r)$, are enabled in $q_n$. From $\mathbf{P}$, we derive two variants of the parity protocol $\mathbf{P}_\ell$ and $\mathbf{P}_r$, where only the left transition $\delta_\ell$ (resp. the right transition $\delta_r$) is present and in which $q_n$ belongs to $Q_1$. We write $\mathsf{G}_{\mathbf{P}_\ell}$ and $\mathsf{G}_{\mathbf{P}_r}$ for the corresponding games. Any local strategy of Player 2 in say $\mathsf{G}_{\mathbf{P}_\ell}$ is also a local strategy in $\mathsf{G}_{\mathbf{P}}$, and similarly for $\mathsf{G}_{\mathbf{P}_r}$. Therefore, the strategy $\sigma$ wins against all local strategies in both $\mathsf{G}_{\mathbf{P}_l}$ and $\mathsf{G}_{\mathbf{P}_r}$ from the initial configuration $\lambda_0$.

Because the number of Player 2 states in $\mathbf{P}_\ell$ [resp. $\mathbf{P}_r$] is at most $n-1$, the induction hypothesis applies, and there exists an initial configuration $\lambda_{0,\ell}$ [resp. $\lambda_{0,r}$] and a strategy $\sigma_\ell$ [resp. $\sigma_r$] for Player 1 such that $\sigma_\ell$ [resp. $\sigma_r$] wins against all strategies of Player 2 in $\mathsf{G}_{\mathbf{P}_\ell}$ [resp. $\mathsf{G}_{\mathbf{P}_r}$] from the configuration $\lambda_{0,\ell}$ [resp. $\lambda_{0,r}$]. We now explain how to define an initial configuration $\lambda'_0$ and a winning strategy $\sigma$ for Player 1 in the game $\mathsf{G}_{\mathbf{P}}$.

First we define $\lambda'_0$ as the disjoint union of the two graphs $\lambda_{0,\ell}$ and $\lambda_{0,r}$, hence the initial graph is composed of two disconnected graphs to which will refer as the "left" part and the "right" part. Now we explain how $\sigma$ is defined, the idea being that this strategy simulates $\sigma_\ell$ on the "left" part of the graph and $\sigma_r$ on the "right" part of the graph. Also, $\sigma$ sometimes exchanges nodes from each part: for instance a node labelled with $(q_n, \top)$ present in the "left" part and

from which Player 2 will choose to perform $\delta_r$. Note that moving such a node is possible because the actions of Player 2 are only internal action (labelled with $\varepsilon$) and consequently they only change the states of a single node; this is not the case for broadcast actions which change the states of the node performing the broadcast but also of all the adjacent nodes able to receive the associated message.

The global strategy $\sigma$ starts by playing as $\sigma_r$ on the processes in the "right" part, until the decision in $\sigma_r$ is to change the label of a node $v_r \in V$ from $(q_n, \bot)$ to $(q_n, \top)$, meaning that the next move for Player 2 will be to decide which transition to perform. Note that, if this never happens, the play is clearly winning for Player 1 since $\sigma_r$ is winning in $\mathsf{G}_{\mathbf{P}_r}$. Just before to change the label of $v_r$, the simulation of $\sigma_r$ is suspended, and $\sigma$ changes mode to play as $\sigma_\ell$ in the "left" part. Similarly as above, if $\sigma_\ell$ never dictates to change the label of a node $v_\ell$ from $(q_n, \bot)$ to $(q_n, \top)$, then $\sigma$ sticks to $\sigma_\ell$, and the play is winning. Otherwise, Player 1 has to move one process to a configuration where $v_\ell$ is labelled with $(q_n, \top)$. Assume it does so, in the "left" part. Assuming further that the next move of Player 2 corresponds to choosing transition $\delta_\ell$, then $\sigma$ continues to simulate $\sigma_\ell$ on the processes of the "left" part. Else, if Player 2 chooses to play transition $\delta_r$, the vertex $v_l$ and the vertex $v_r$ are exchanged (remember that $v_r$ is labelled as $v_\ell$ just before this sequence of actions i.e. with $(q_n, \bot)$) and $\sigma$ changes mode and now simulates $\sigma_r$ on the right part.

If we consider now a strategy $\tau$ for player 2, the play $\rho(\mathsf{G}_{\mathbf{P}}, \lambda'_0, \sigma, \tau)$ can have the three following forms where we add a superscript $(\ell)$ or $(r)$ to denote whether for each state whether the game is played as in $\mathsf{G}_{\mathbf{P}_\ell}$ or in $\mathsf{G}_{\mathbf{P}_r}$:

1. $\lambda_0'^{(r)} \dots \lambda_i^{(\ell)} \dots \lambda_j^{(r)} \lambda_{j+1}^{(r)} \lambda_{j+2}^{(r)} \dots$ in which after some points $\sigma$ plays only as $\sigma_r$ in $\mathsf{G}_{\mathbf{P}_r}$;
2. $\lambda_0'^{(r)} \dots \lambda_i^{(\ell)} \dots \lambda_j^{(\ell)} \lambda_{j+1}^{(\ell)} \lambda_{j+2}^{(\ell)} \dots$ in which after some points $\sigma$ plays only as $\sigma_\ell$ in $\mathsf{G}_{\mathbf{P}_\ell}$;
3. $\lambda_0'^{(r)} \dots \lambda_i^{(\ell)} \lambda_{i+1}^{(\ell)} \lambda_{i+2}^{(\ell)} \dots \lambda_j^{(r)} \lambda_{j+1}^{(r)} \lambda_{j+2}^{(r)} \dots \lambda_k^{(\ell)} \lambda_{k+1}^{(\ell)} \lambda_{k+2}^{(\ell)} \dots$ in which $\sigma$ plays alternatively infinitely often as $\sigma_r$ in $\mathsf{G}_{\mathbf{P}_r}$ and as $\sigma_\ell$ in $\mathsf{G}_{\mathbf{P}_\ell}$.

From the fact that $\sigma_r$ and $\sigma_\ell$ are winning strategies in their respective game, it is obvious that in the two first cases $\rho(\mathsf{G}_{\mathbf{P}}, \lambda'_0, \sigma, \tau)$ is a winning play. For the third case, from how we build our strategies $\sigma$, it is clear that the play obtained from $\rho(\mathsf{G}_{\mathbf{P}}, \lambda'_0, \sigma, \tau)$ by removing the configurations with superscripts $(r)$ correspond to a play in $\mathsf{G}_{\mathbf{P}_\ell}$ which respects a strategy profile $(\sigma_\ell, \tau')$ for some strategy $\tau'$ of Player 2, and consequently the play is winning. Symmetrically, the play obtained by removing the configurations with superscripts $(\ell)$ is winning in $\mathsf{G}_{\mathbf{P}_r}$. Consequently the "composition" of these two plays is necessarily winning: for the safety objective it never uses "unsafe" actions and for the parity objective, the maximal color seen infinitely often is either in the $(\ell)$ part or in the $(r)$ part and in both cases it is even. This shows that $\rho(\mathsf{G}_{\mathbf{P}}, \lambda'_0, \sigma, \tau)$ is a winning play for Player 1.

In order to complete the proof of the lemma, we still need to prove:

$$\exists \lambda_0 \in \Lambda_{\mathbf{P},0}.\ \exists \sigma \in \mathcal{S}^{(1)}.\ \forall \tau \in \mathcal{S}_{\mathsf{I}}^{(2)},\ \rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau) \in \mathsf{Win}$$
$$\Longleftrightarrow$$
$$\forall \tau \in \mathcal{S}_{\mathsf{I}}^{(2)}.\ \exists \lambda_0 \in \Lambda_{\mathbf{P},0}.\ \exists \sigma \in \mathcal{S}^{(1)}, \rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau) \in \mathsf{Win}$$

Here again only the right-to-left implication deserves a proof.

Without loss of generality, when the winning objective is $\mathsf{Win} = \mathsf{Win}_{\mathsf{p}}$, we assume that $\mathsf{safe} = \Delta$.

Assume that for every local strategy $\tau$ of Player 2 there exists $\lambda_{0,\tau} \in \Lambda_{\mathbf{P},0}$ and $\sigma_\tau \in \mathcal{S}^{(1)}$ such that $\rho(\mathsf{G}_{\mathbf{P}}, \lambda_{0,\tau}, \sigma_\tau, \tau) \in \mathsf{Win}$. The intuition is to define a strategy $\sigma$ for Player 1 that guesses which local strategy Player 2 is playing, and plays the adequate counter strategy $\sigma_\tau$. When the guess is incorrect, and the error is detected, $\sigma$ starts again on an other graph and guesses an other local strategy for Player 2.

In order to define $\sigma$, we let $\mathcal{S}_{\mathsf{I},ns}^{(2)}$ denote the set of *unsafe* local strategies for Player 2, for which there is no risk for Player 1 to guess them, since they fire unsafe transitions. Formally, $\mathcal{S}_{\mathsf{I},ns}^{(2)}$ is the set of strategies $\tau_b$ induced by the local behaviors $b$ such that $\forall q \in Q^{(2)}$ if there exists $\delta = (q, \varepsilon, q') \in \Delta$ with $\delta \notin \mathsf{safe}$ then $b(q) \notin \mathsf{safe}$. Hence $\mathcal{S}_{\mathsf{I},ns}^{(2)}$ is the set of strategies that always choose a non-safe transition if possible.

We define $\lambda_0$ as the disjoint union of the graphs $\lambda_{0,\tau}$ for all local unsafe $\tau \in \mathcal{S}_{\mathsf{I},ns}^{(2)}$. For each strategy $\tau \in \mathcal{S}_{\mathsf{I},ns}^{(2)}$ we refer to the corresponding sub-graph as the graph $\tau$.

The strategy $\sigma$ for Player 1 is defined as the strategy that starts playing as $\sigma_\tau$ on the graph $\tau$ for some $\tau \in \mathcal{S}_{\mathsf{I},ns}^{(2)}$. When $\sigma$ detects a wrong guess (when the last move was $(q, \varepsilon, q')$ whereas $\tau$ should have done $(q, \varepsilon, q'')$), $\sigma$ changes for an other graph $\tau'$ in which he did not play already and switches to $\sigma_{\tau'}$.

Notice that when $\sigma$ detects that the current game graph is the wrong one, since the strategy he is playing wins against some strategy of $\mathcal{S}_{\mathsf{I},ns}^{(2)}$, necessarily the transition $(q, \varepsilon, q')$ is safe. Also notice that before detecting that $\tau$ is a wrong copy, $\sigma$ is playing a strategy winning for some $\tau'$ hence never fires a non-safe transition.

Let $\tau \in \mathcal{S}_{\mathsf{I}}^{(2)}$ be any strategy that Player 2 plays against $\sigma$. Let us consider the play $\rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau)$. It can be written $\rho_1.\rho_2$ where $\rho_2$ is the longest suffix of the play that remains in the same graph, say $\tau'$ graph. Then, $\rho_2'$ be defined as the projection of $\rho_2$ on the graph $\tau'$. Since $\sigma$ is playing as $\sigma_{\tau'}$ in this suffix, and since it never detects a wrong guess in this suffix, we know that $\rho_2' = \rho(\mathsf{G}_{\mathbf{P}}, \lambda_{0,\tau'}, \sigma_{\tau'}, \tau) = \rho(\mathsf{G}_{\mathbf{P}}, \lambda_{0,\tau'}, \sigma_{\tau'}, \tau') \in \mathsf{Win}$. Therefore $\rho_2 \in \mathsf{Win}$. Last, because $\rho_1$ never uses unsafe transitions, we obtain $\rho_1.\rho_2 \in \mathsf{Win}$.

### A.2 Proof of Lemma 3

From $\mathbf{P}'$, we build a broadcast protocol $\mathbf{P}''$ by deleting all the edges of the form $(q, !!a, q')$ or $(q, \varepsilon, q')$ which do not belong to $\mathsf{safe}$. Then using the result of

Theorem 4.3 of [10], we can decide in NP whether there exists a path in $\mathsf{G}_{\mathbf{P}''}$ such that none of the vertices is labelled with states in $(ActStates \times \{\bot\})$ (note that in [10] the semantics is a bit different since the authors do not use labels with $\bot$ and $\top$ but this does not involve any significant changes on the behavior of the broadcast protocol). Such a path exists if and only if in $\mathsf{G}_{\mathbf{P}'}$ there exists a finite maximal path belonging to $\mathsf{Win_s}$ starting from an initial configuration in $\mathsf{G}_{\mathbf{P}'}$.

### A.3 Proof of Lemma 4

As in the previous proof we begin to build from $\mathbf{P}'$ a protocol $\mathbf{P}'' = (Q, q_0, \Sigma, \Delta'', \mathsf{col}'')$ obtained by removing from $\mathbf{P}'$ all the edges of the form $(q, !!a, q')$ or $(q, \varepsilon, q')$ which do not belong to $\mathsf{safe}$ (this because we do not want to consider the paths obtained by using these edges) and $\mathsf{col}''$ is then the restriction of $\mathsf{col}'$ to $\Delta''$. Note that we do not need to precise a safe set in $\mathbf{P}''$ since all its transitions will be considered as safe.

We now introduce the following notation, let $\mathcal{R}_{\mathbf{P}''} \subseteq Q$ be the set of control states of $\mathbf{P}''$ reachable in $\mathsf{G}_{\mathbf{P}''}$ which can be defined as: $q \in \mathcal{R}_{\mathbf{P}''}$ if and only if there exists a finite path $\lambda_0\lambda_1 \cdots \lambda_n \in \Lambda^*_{\mathbf{P}''}$ starting from an initial configuration such that $\{(q, \bot), (q, \top)\} \cap L(\lambda_n) \neq \emptyset$. Using Algorithm 1 presented in [10], we know that this set can be computed in polynomial time and using the proof of Lemma 1 of the same paper, we also know that for all natural $k$ there is a witness reachable configurations where each state of $\mathcal{R}_{\mathbf{P}''}$ label at least $k$ vertex. These two properties are summarized in the following lemma.

**Lemma 5.** *[10]*

1. *$\mathcal{R}_{\mathbf{P}''}$ can be computed in polynomial time.*
2. *For all $k \in \mathbb{N}$, there exists a finite path $\lambda_0\lambda_1 \cdots \lambda_n \in \Lambda^*_{\mathbf{P}''}$ such that $\lambda_n = (V, E, L)$ and for all $q \in \mathcal{R}_{\mathbf{P}''}$, $card(\{v \in V \mid L(v) = (q, \bot)\}) \geq k$.*

In order to detect, if we have an infinite path $\rho$ starting from an initial configuration in $\mathsf{G}_{\mathbf{P}'}$ such that $\rho \in \mathsf{Win_p} \cap \mathsf{Win_s}$, we only need to show that in $\mathsf{G}_{\mathbf{P}''}$ there exists a cycle which begins with a reachable configuration and end up with this next configuration and so that the maximum color seen on this cycle is even. To perform this task we will use the counter abstraction techniques which consists in abstracting away the vertices and remember for each control states of $\mathbf{P}''$ how many vertices are labelled with such a vertex in a configuration. Note that using such a technique in our context is feasible because only the number of control states matters, the reconfiguration rule allows us in fact to forget completely the communication topology. For this matter we will translate the behavior of $\mathsf{G}_{\mathbf{P}''}$ into a VASS.

We begin by recalling what is a VASS of dimension $n$ with parity, it can be defined as a tuple $\mathcal{V} = (S, T, \mathsf{col})$ where $S$ is a finite set of control states, $\Rightarrow \subseteq S \times \mathbb{Z}^n \times S$ is the transition relation labelled with vector of integers and $\mathsf{col} : \Rightarrow \mapsto \mathbb{N}$ is the coloring function. To a vass of this form we associate a transition system $TS(\mathcal{V}) = (S \times \mathbb{N}^n, \Rightarrow, \mathsf{col})$ where $S \times \mathbb{N}^n$ is the set of configurations, $\Rightarrow \subseteq (S \times$

$\mathbb{N}^n) \times (S \times \mathbb{N}^n)$ is the transition relation and $\mathsf{col} : \Rightarrow \mapsto \mathbb{N}$ is the coloring function which are defined as followed : $(s, \mathbf{v}) \Rightarrow (s', \mathbf{v}')$ and $\mathsf{col}((s, \mathbf{v}) \Rightarrow (s', \mathbf{v}')) = c$ if and only if there exists a transition $(s, \mathbf{b}, s') \in T$ such that $\mathbf{v}' = \mathbf{v} + \mathbf{b}$ and $\mathsf{col}((s, \mathbf{b}, s')) = c$. An *infinite run* of $\mathcal{V}$ starting from a configuration $(s_0, \mathbf{v}_0)$ is an infinite sequence of the form $(s_0, \mathbf{v}_0) \Rightarrow (s_1, \mathbf{v}_1) \Rightarrow (s_2, \mathbf{v}_2) \dots$ and it is *valid* with respect to $\mathsf{col}$ if $\max\{c \in \mathbb{N} \mid \forall i \geq 0 \; \exists j \geq i \text{ s.t. } \mathsf{col}((s_j, \mathbf{v}_j) \Rightarrow (s_{j+1}, \mathbf{v}_{j+1}))) = c\}$ is even.

We now provide the construction of a VASS $\mathcal{V}_{\mathbf{P}''}$ with parity from $\mathbf{P}'' = (Q, q_0, \Sigma, \Delta'', \mathsf{col}'')$ such that there will exists a configuration $(s_0, \mathbf{v}_0)$ and a valid infinite run starting from this configuration if and only if there exists an infinite path $\rho$ starting from an initial configuration in $\mathsf{G}_{\mathbf{P}''}$ such that $\rho \in \mathsf{Win}_{\mathsf{p}}$. We first assume that $\mathcal{R}_{\mathbf{P}''} = \{q_1, \dots, q_n\}$. We build the VASS of dimensions $2n$ $\mathcal{V}_{\mathbf{P}''} = (S_{\mathbf{P}''}, T_{\mathbf{P}''}, \mathsf{col}')$ as follows:

- $S_{\mathbf{P}''} = \{s_0\} \cup \{s_e^1, s_e^2 \mid e \in \Delta'' \text{ is of the form } (q_i, !!a, q_j) \text{ or } (q_i, \varepsilon, q_j)\}$
- $T_{\mathbf{P}''}$ is the smallest relation satisfying the following conditions (and we define $\mathsf{col}'$ simultaneously), for all $e = (q_i, \mathbf{a}, q_j) \in \Delta''$:
  - if $\mathbf{a} = \varepsilon$ or $\mathbf{a} = !!a$, then $(s_0, \mathbf{b}, s_e^1), (s_e^1, \mathbf{0}, s_e^2), (s_e^2, \mathbf{b}', s_0) \in T_{\mathbf{P}''}$ (and $\mathsf{col}'(s_0, \mathbf{b}, s_e^1) = \mathsf{col}'(s_e^1, \mathbf{0}, s_e^2) = \mathsf{col}'(s_e^2, \mathbf{b}', s_0) = \mathsf{col}(e)$) with:
    - $\mathbf{b}[i] = -1$ and $\mathbf{b}[k] = 0$ for all $k \in \{1, \dots, 2n\} \setminus \{i\}$;
    - $\mathbf{b}'[j] = 1$ and $\mathbf{b}[k] = 0$ for all $k \in \{1, \dots, 2n\} \setminus \{j\}$;
  - if $\mathbf{a} = ??a$, then for all edges $e' = (q_k, !!a, q_\ell)$ in $\Delta$, $(s_{e'}^1, \mathbf{b}, s_{e'}^1), (s_{e'}^2, \mathbf{b}', s_{e'}^2) \in T_{\mathbf{P}''}$ (and $\mathsf{col}'(s_{e'}^1, \mathbf{b}, s_{e'}^1) = \mathsf{col}'(s_{e'}^2, \mathbf{b}', s_{e'}^2) = 0$) with:
    - $\mathbf{b}[i] = -1$ and $\mathbf{b}[n+j] = 1$ and $\mathbf{b}[k] = 0$ for all $k \in \{1, \dots, 2n\} \setminus \{i, n+j\}$;
    - $\mathbf{b}'[j] = 1$ and $\mathbf{b}'[n+j] = -1$ and $\mathbf{b}[k] = 0$ for all $k \in \{1, \dots, 2n\} \setminus \{j, n+j\}$;

The intuition behind this construction is that each vector value in a configuration of $\mathcal{V}_{\mathbf{P}''}$ characterizes the number of nodes in each state, then if a broadcast corresponding to an edge $e = (q_k, !!a, q_\ell)$ is performed we begin by decreasing of one unit the number of nodes in state $q_k$ by taking a transition from $s_0$ to $s_e^1$ then we can take some reception rules of the form $(q_i, ??a, q_j)$ and this is simulated by decrementing of one unit the number of nodes in $q_i$ and incrementing the number of nodes in $q_j$. Note that here we use the $n+j$-extra counter, because we do not want in a same simulation of a reception that a node goes to $q_j$ and then receive again the message $a$, this is the reason why we cut the simulation of the broadcast into the state $s_e^1$ and $s_e^2$; and finally, when the simulation of the reception is over, we increment of one unit the number of nodes in states $q_\ell$ thanks to the transition from $q_e^2$ to $q_0$. Note that the value of the counter $n+j$ is not necessarily put to 0 but this can be done later, this is authorized since thanks to reconfiguration we do not ask in the simulation the node to react to the broadcast at the precise moment, it could be some nodes that will then be completely disconnected and come back later. The next Lemma states the formal property of this intuition. The proof of this Lemma is a direct consequence of the transition relation of $\mathsf{G}''_{\mathbf{P}}$ and of the way we define $\mathcal{V}_{\mathbf{P}''}$.

**Lemma 6.** *Let $\lambda = (V, E, L)$ be a configuration of $\mathsf{G}_{\mathbf{P}''}$ such that $L(\lambda) \in \mathcal{R}_{\mathbf{P}''} \times \{\bot\}$ and $\mathbf{b} \in \mathbb{N}^{2n}$ such that $\mathbf{b}[i] = card(\{v \in V \mid L(v) = (q_i, \bot)\})$ for all $i \in \{1, \ldots, n\}$ and $\mathbf{b}[i] = 0$ for all $i \in \{n+1, \ldots, 2n\}$. Then, in $\mathsf{G}_{\mathbf{P}''}$ there exists an infinite path in $\mathsf{Win}_{\mathsf{p}}$ starting from $\lambda$ if and only if in $\mathcal{V}_{\mathbf{P}''}$ there exists an infinite run starting from $(s_0, \mathbf{b})$ valid with respect to $\mathsf{col}'$.*

The definition of $\mathcal{V}_{\mathbf{P}''}$ allows us to state that, there exists a vector $\mathbf{b} \in \mathbb{N}^{2n}$ with $\mathbf{b}[i] = 0$ for all $i \in \{n+1, \ldots, 2n\}$ such that in $\mathcal{V}_{\mathbf{P}''}$ there exists an infinite run starting from $(s_0, \mathbf{b})$ valid with respect to $\mathsf{col}'$ if and only if there exists a finite cycle in $T^+_{\mathbf{P}''}$ of the form $(s_0, \mathbf{b}_0, s_1)(s_1, \mathbf{b}_1, s_2) \ldots (s_k, \mathbf{b}_k, s_{k+1})$ with $s_{k+1} = s_0$ such that $\Sigma_{i \in \{0,\ldots,k\}} \mathbf{b}_i \geq \mathbf{0}$ and such that $\max\{\mathsf{col}'(s_i, \mathbf{b}_i, s_{i+1}) \mid i \in \{0, \ldots, k\}\}$ is even. The following lemma tells us that finding such a cycle can be done in polynomial time, it uses a result presented in [15] to detect in a graph labeled with vector of integers a cycle whose sum on the different vectors seen on the cycle is 0.

**Lemma 7.** *Deciding whether there exists a finite cycle in $T^+_{\mathbf{P}''}$ of the form $(s_0, \mathbf{b}_0, s_1)(s_1, \mathbf{b}_1, s_2) \ldots (s_k, \mathbf{b}_k, s_{k+1})$ with $s_{k+1} = s_0$ such that $\Sigma_{i \in \{0,\ldots,k\}} \mathbf{b}_i \geq \mathbf{0}$ and such that $\max\{\mathsf{col}'(s_i, \mathbf{b}_i, s_{i+1}) \mid i \in \{0, \ldots, k\}\}$ is even can be done in polynomial time.*

*Proof.* Let $c \in \mathbb{N}$ be a color such that there exists an edge $(s, \mathbf{b}, s') \in T_{\mathbf{P}''}$ verifying $\mathsf{col}'(s, \mathbf{b}, s') = c$. First note that by construction of $\mathcal{V}_{\mathbf{P}''}$ there exists a finite cycle in $T^+_{\mathbf{P}''}$ of the form $(s_0, \mathbf{b}_0, s_1)(s_1, \mathbf{b}_1, s_2) \ldots (s_k, \mathbf{b}_k, s_{k+1})$ with $s_{k+1} = s_0$ and $\Sigma_{i \in \{0,\ldots,k\}} \mathbf{b}_i \geq \mathbf{0}$ if and only if there exists a cycle in $T^+_{\mathbf{P}''}$ of the form $(s'_0, \mathbf{b}'_0, s'_1)(s'_1, \mathbf{b}'_1, s'_2) \ldots (s'_k, \mathbf{b}'_k, s'_{k+1})$ with $s'_{k+1} = s'_0$ and $\Sigma_{i \in \{0,\ldots,k\}} \mathbf{b}'_i \geq \mathbf{0}$, i.e. a cycle that do not necessarily begin with in state $s_0$. This because we cannot have a zero sum cyle looping only on a state of the form $s^1_e$ or $s^2_e$ since such loop make strictly decrease a counter, hence any cycle will necessarily passes through $s_0$. Then we know thanks to [15] that the existence of such a cycle can be detected in polynomial time. The problem is that we do not know whether for such a cycle $\max\{\mathsf{col}'(s'_i, \mathbf{b}'_i, s'_{i+1}) \mid i \in \{0, \ldots, k\}\}$ is even.

To solve this for each color $c \in \mathbb{N}$ present in $\mathcal{V}''_{\mathbf{P}}$, we proceed as follows. We remove all the edges with colors strictly bigger than $c$, then we add a dimension (i.e. the $2n + 1$ counter) such that for all edges $e$ of the form $(q_i, !!a, q_j)$ or $(q_i, \varepsilon, q_j)$, if $\mathsf{col}(e) \neq c$ then the transition $(s_0, \mathbf{b}, s^1_e)$ decrement this $2n + 1$ counter of 1 unit and if $\mathsf{col}(e) = c$ then we add a self loop on $s^1_e$ that increments this counter, the other transitions remaining unchanged. Consequently the only way to a have a cycle with zero sum in such a VASS is to take at least a transition of color $c$. We can do this for any even color and since there is linear number of such color in $\mathcal{V}_{\mathbf{P}''}$, this allows us to conclude. □

As said previously, there exists a vector $\mathbf{b} \in \mathbb{N}^{2n}$ with $\mathbf{b}[i] = 0$ for all $i \in \{n+1, \ldots, 2n\}$ such that in $\mathcal{V}_{\mathbf{P}''}$ there exists an infinite run starting from $(s_0, \mathbf{b})$ valid with respect to $\mathsf{col}'$ if and only if there exists a finite cycle in $T^+_{\mathbf{P}''}$ of the form $(s_0, \mathbf{b}_0, s_1)(s_1, \mathbf{b}_1, s_2) \ldots (s_k, \mathbf{b}_k, s_{k+1})$ with $s_{k+1} = s_0$ such that $\Sigma_{i \in \{0,\ldots,k\}} \mathbf{b}_i \geq \mathbf{0}$ and such that $\max\{\mathsf{col}'(s_i, \mathbf{b}_i, s_{i+1}) \mid i \in \{0, \ldots, k\}\}$ is even, but note that if

such a vector $\mathbf{b}$ exists then the property still holds for all vector $\mathbf{b}' \in \mathbb{N}^{2n}$ with $\mathbf{b}'[i] = 0$ for all $i \in \{n+1, \ldots, 2n\}$ and $\mathbf{b}'[i] \geq \mathbf{b}[i]$ for all $i \in \{1, \ldots, n\}$. So if we resume Lemma 5 tells that we can compute in polynomial time $\mathcal{R}_{\mathbf{P}''}$ and that for any natural $k$ we can find a reachable configuration where each label of $\mathcal{R}_{\mathbf{P}''}$ is present at least $k$ times. then using Lemma 6 and 7 we know that we can decide in polynomial time whether there exists such a configuration from which there exists an infinite path in $\mathsf{Win}_\mathsf{p}$ in $\mathsf{G}_{\mathbf{P}''}$. By the way we build $\mathbf{P}''$ (removing the unsafe action from $\mathbf{P}'$), we can hence deduce the Lemma 4.

## B Details for Section 4

We define urgent strategies for Player 1: $\sigma \in \mathcal{S}^{(1)}$ is *urgent* if for every play $\rho\lambda$ with $\lambda = (V, E, L) \in \mathcal{G}_{Q^\mathbf{P} \times \{\bot\}}$ if there exists $v \in V$ such that $L(v) \in Q^{(2)} \times \{\bot\}$ then $\sigma(\rho\lambda) = \lambda'$ where $\lambda' = (V, E, L')$, and $L'(v) \in Q^{(2)} \times \{\top\}$. In words, the urgent strategies are those where Player 1 activates (*i.e.* puts to $\top$) the processes in states belonging to Player 2 as soon as possible. The set of urgent strategies for Player 1 is denoted $\mathcal{S}_u^{(1)}$. Note that if $\sigma_u$ is urgent, and $\rho\lambda$ respects $\sigma_u$ with $\lambda = (V, E, L)$ we have $card(\{v \in V \mid L(v) \in Q^{(2)} \times \{\bot, \top\}\}) \leq 1$.

We prove that there is a winning strategy for Player 1 if and only if there is an urgent one.

**Lemma 8.**

$$\exists \lambda_0 \in \Lambda_{\mathbf{P},0}.\ \exists \sigma \in \mathcal{S}^{(1)}.\ \forall \tau \in \mathcal{S}^{(2)},\ \rho(\mathsf{G}_\mathbf{P}, \lambda_0, \sigma, \tau) \in \mathsf{Win}$$
$$\Longleftrightarrow$$
$$\exists \lambda_0 \in \Lambda_{\mathbf{P},0}.\ \exists \sigma \in \mathcal{S}_u^{(1)},\ \forall \tau \in \mathcal{S}^{(2)},\ \rho(\mathsf{G}_\mathbf{P}, \lambda_0, \sigma, \tau) \in \mathsf{Win}$$

*Proof.* Only the right to left implication deserves a proof, since $\mathcal{S}_u^{(1)} \subseteq \mathcal{S}^{(1)}$.

We assume that there exists $\lambda_0 \in \Lambda_{\mathbf{P},0}$ and $\sigma \in \mathcal{S}^{(1)}$ such that $\forall \tau \in \mathcal{S}^{(2)},\ \rho(\mathsf{G}_\mathbf{P}, \lambda_0, \sigma, \tau) \in \mathsf{Win}$. Without loss of generality we can assume that $\sigma$ is memoryless, since given the initial configuration $\lambda_0 \in \Lambda_{\mathbf{P},0}$ the game is finite, hence there is a winning strategy if and only if there is a memoryless winning strategy.

Intuitively, the urgent $\sigma'$ plays similarly to $\sigma$ but delays the moment the processes enter in state of $Q_2$ until $\sigma$ would have played them. Strategy $\sigma'$ is then urgent, but has memory.

More formally, one the one hand, in $\lambda$ if $\sigma$ would have move a process to a state $q \in Q^{(2)}$ then for the paths $\rho$ that $\sigma'$ identifies to $\lambda$, $\sigma'$ remembers that this process should be in $q$. One the other hand, in $\lambda$ if $\sigma$ would have put a process in a state $q' \in Q^{(2)}$ to $\top$, then for the paths $\rho$ that $\sigma'$ identifies to $\lambda$, $\sigma'$ first moves the process to $q'$ and then puts it immediately to $\top$.

Let $\tau$ be a strategy for Player 2. To a play $\rho(\mathsf{G}_\mathbf{P}, \lambda_0, \sigma', \tau)$ we can associate the play $\tilde{\rho}(\mathsf{G}_\mathbf{P}, \lambda_0, \sigma', \tau)$ of configurations as stored in the memory of $\sigma'$. Notice that $\tilde{\rho}(\mathsf{G}_\mathbf{P}, \lambda_0, \sigma', \tau)$ corresponds to a play for $\sigma$, and hence satisfies the wining condition $\mathsf{Win}$. Moreover the transitions visited along that play are the same as $\rho(\mathsf{G}_\mathbf{P}, \lambda_0, \sigma', \tau)$. Hence the urgent strategy $\sigma'$ built above is winning: for every strategy $\tau$ for Player 2, $\rho(\mathsf{G}_\mathbf{P}, \lambda_0, \sigma', \tau) \in \mathsf{Win}$. □

### B.1 Proof of Proposition 2

*Proof.* Since the parity protocol $\mathbf{P}_{\mathcal{P}}$ and the probabilistic protocol $\mathcal{P}$ have the same set of states, there is a direct correspondence between the configurations $\gamma$ of $\mathcal{M}$ and the configurations $\lambda$ of $\mathsf{G}_{\mathbf{P}_{\mathcal{P}}}$. Hence there is a direct correspondence between schedulers on $\mathcal{M}_{\mathcal{P}}$ and urgent strategies for Player 1.

For $\gamma_0$ an initial configuration and $\pi$ a scheduler such that $\mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \Diamond q_f) = 0$, for every reachable configuration $\gamma$ we know that $q_f \notin \gamma$. We define $\lambda_0$ as the configuration corresponding to $\gamma_0$. The strategy $\sigma_\pi$ that corresponds to $\pi$, never uses a transition leading to $q_f$ nor visits a state $q \in Q^{(2)}$ that may lead to $q_f$. Therefore $\sigma_\pi$ ensures the safety objective from $\lambda_0$.

For $\lambda_0$ an initial configuration and $\sigma$ a strategy such that $\forall \tau \in \mathcal{S}^{(2)},\ \rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau) \in \mathsf{Win}_{\mathsf{s}}$, for every reachable configuration $\lambda$ we know that $q_f \notin \lambda$, since the safety objective prevents using action leading to $q_f$. Hence the scheduler $\pi_\sigma$ corresponding to $\sigma$ never reaches a configuration containing $q_f$. Hence $\mathbb{P}(\mathcal{M}_{\mathcal{P}}, \lambda_0, \pi_\sigma, \Diamond q_f) = 0$, where $\gamma_0$ corresponds to $\lambda_0$. □

### B.2 Proof of Proposition 3

*Proof.* We start by proving:

$$\begin{array}{c} \exists \gamma_0 \in \Lambda_{\mathcal{P},0}.\ \exists \pi \in \Pi, \mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \Diamond q_f) < 1 \\ \Downarrow \\ \exists \lambda_0 \in \Lambda_{\mathbf{P},0}.\ \exists \sigma \in \mathcal{S}^{(1)}.\ \forall \tau \in \mathcal{S}^{(2)},\ \rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau) \in \mathsf{Win}_{\mathsf{sp}} \end{array}$$

Let $\gamma_0 \in \Lambda_{\mathcal{P},0}$ be an initial configuration and $\pi$ be a scheduler such that $\mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \Diamond q_f) < 1$. Hence, there exists a finite path $\rho$ that respects $\pi$ and such that $\mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \mathcal{A}_\rho \cap \Diamond q_f) = 0$, where $\mathcal{A}_\rho$ is the cylinder of paths starting with prefix $\rho$.

We define $\lambda_0$ as the initial configuration similar to $\gamma_0$ except that all processes are labelled by $(q_0, 1)$ instead of $q_0$. We also define $\sigma$ as the strategy that plays the finite prefix $\rho$ with all processes in the first copy, and after $\rho$, moves them all to the second copy and then plays according to $\pi$ (as the proof of Proposition 2).

For any finite play respecting $\sigma$ the safety objective is satisfied because we never use a transition that enters $(q_f, 1)$ during the prefix $\rho$, and because the same arguments as the proof of Proposition 2 apply for the suffix in the second copy. Moreover, any infinite play respecting $\sigma$ only stays for a finite number of steps in the first copy and then ends is in the second copy. Hence the play satisfies the parity objective. All together, $\sigma$ ensures the combined safety/parity objective.

We now prove:

$$\begin{array}{c} \exists \lambda_0 \in \Lambda_{\mathbf{P},0}.\ \exists \sigma \in \mathcal{S}^{(1)}.\ \forall \tau \in \mathcal{S}^{(2)},\ \rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau) \in \mathsf{Win}_{\mathsf{sp}} \\ \Downarrow \\ \exists \gamma_0 \in \Lambda_{\mathcal{P},0}.\ \exists \pi \in \Pi, \mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \Diamond q_f) < 1 \end{array}$$

Let $\lambda_0 \in \Lambda_{\mathbf{P},0}$ be an initial configuration and $\sigma$ be a strategy for Player 1 such that $\forall \tau \in \mathcal{S}^{(2)},\ \rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau) \in \mathsf{Win}_{\mathsf{sp}}$.

We define $\gamma_0$ as $\lambda_0$ but in which the labels $(q_0, 1)$ are replaced with $q_0$. Also, if we merge states $(q, 1)$ and $(q, 2)$, a configuration $\lambda$ of $\mathsf{G}_{\mathbf{P}}$ can be seen as a configuration of $\mathcal{M}_{\mathcal{P}}$. We define $\pi$ as the scheduler that mimicks $\sigma$ on these merged configurations, and that skips the actions leading from the first copy to the second one (since they have no meaning in $\mathcal{P}$).

Let $\mathcal{A}$ by the set of all plays that correspond to a play $\rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau)$ where we look at merged configurations and in which we removed the actions leading from copy 1 to copy 2. Since $\sigma$ is winning for the safety objective, we know that $\mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \mathcal{A} \cap \Diamond q_f) = 0$. Moreover, since $\sigma$ ensures also the parity objective, we obtain that $\sigma$ only fixes a finite number of probabilistic choices (the choices made in the first copy). Hence there is a positive probability that these probabilistic choices are all correct in $\mathcal{M}_{\mathcal{P}}$ under $\pi$, hence $\mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \mathcal{A}) > 0$.

The combination of $\mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \mathcal{A} \cap \Diamond q_f) = 0$ and $\mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \mathcal{A}) > 0$ gives us that $\mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \Diamond q_f) < 1$. □

### B.3 Proof of Proposition 1

*Proof (Proposition 1).* Let us first prove the following implication:

$$\begin{array}{c} \exists \gamma_0 \in \Lambda_{\mathcal{P},0}.\ \exists \pi \in \Pi, \mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \Diamond q_f) = 1 \\ \Downarrow \\ \exists \lambda_0 \in \Lambda_{\mathbf{P},0}.\ \exists \sigma \in \mathcal{S}_u^{(1)}.\ \forall \tau \in \mathcal{S}^{(2)},\ \rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau) \in \mathsf{Win}_{\mathsf{p}} \end{array}$$

First remark that we can embed configurations of $\mathcal{M}_{\mathcal{P}}$ as configurations of $\mathsf{G}_{\mathbf{P}}$ by identifying $q \in Q^{(P)}$ to $(q, 2) \in Q_{\mathbf{P}}^{(2)}$, so that $Q_{\mathcal{P}} \subseteq Q_{\mathbf{P}}$. In particular, we define $\lambda_0$ as the configuration that corresponds to $\gamma_0$.

Let $\gamma_0$ and $\pi$ be such that $\mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \Diamond q_f) = 1$. We can assume, without loss of generality that $\pi$ is memoryless, because, given the initial configuration $\gamma_0$, the number of configurations is finite.

Now we explain how to define $\sigma$ from $\pi$. For a configuration $\lambda = (V, E, L)$:

- If there exists $v \in V$ with $L(v) = (q_f, \bot)$, then $\sigma$ plays forever the two actions $(q_f, \bot) \to (q_f, \top) \to (q_f, \bot)$ using the loop on $q_f$, which has parity 2.
- Otherwise, if there exists $v \in V$ with $L(v) = ((q, 2), \bot)$, for some $q \in Q^{(P)}$, then $\sigma$ moves to the configuration where this node is labelled $((q, 2), \top)$.
- Otherwise, if there are no vertices $v \in V$ with $L(v) = ((q, 1), \top)$ or $L(v) = ((q, 1), \bot)$ for some $q \in Q^{(P)}$, $\lambda$ can be identified to a configuration $\gamma$, and $\sigma$ in $\lambda$ mimics $\pi$ in $\gamma$.
- Otherwise, there exists $v \in V$ with $L(v) = ((q, 1), \_)$ for some state $q \in Q$. In this case, we let $\gamma_1$ be the configuration corresponding to $\lambda$ except that $v$ is labelled by state $q$. We then pick a shortest path $\gamma_1 \gamma_2 \dots \gamma_n$ that respects $\pi$ and reaches $q_f$, and define $\sigma(\lambda)$ as the configuration corresponding to $\gamma_2$.

Strategy $\sigma$ is urgent by definition, and we show now that it is winning. Let $\tau$ be any strategy for Player 2, and consider the play $\rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau)$. There are two possibilities: either $q_f$ is reached, and then $\sigma$ ensures to loop on $q_f$ with parity 2

so that $\rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau) \in \mathsf{Win}_{\mathsf{p}}$; or $q_f$ is not reached. By definition of $\sigma$, notice that every configuration that can be reached under $\sigma$ and has no processes labelled with a state $(q, 1)$ (for $q \in Q^{(P)}$) corresponds to a configuration reachable under $\pi$. Moreover, since $\mathbb{P}_\pi(\mathcal{M}_{\mathcal{P}}, \gamma_0, \Diamond q_f) = 1$, from every configuration reachable under $\pi$ there exists a path to $q_f$. Second notice that if, at some point, $\tau$ always choose to move processes to states of the form $(q, 1)$, then $\sigma$ would ensure to reach $q_f$ by following a shortest path. Hence infinitely often $\tau$ chooses not to move the process to $(q, 1)$ and this implies that the play has parity 2. Therefore $\rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau) \in \mathsf{Win}_{\mathsf{p}}$.

Let us now prove the other direction:

$$\exists \lambda_0 \in \Lambda_{\mathbf{P},0}.\ \exists \sigma \in \mathcal{S}_u^{(1)}.\ \forall \tau \in \mathcal{S}^{(2)},\ \rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau) \in \mathsf{Win}_{\mathsf{p}}$$
$$\Downarrow$$
$$\exists \gamma_0 \in \Lambda_{\mathcal{P},0}.\ \exists \pi \in \Pi, \mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \Diamond q_f) = 1$$

Let $\lambda_0 \in \Lambda_{\mathbf{P},0}$ be an initial configuration and $\sigma \in \mathcal{S}_u^{(1)}$ be an urgent strategy for Player 1 such that $\forall \tau \in \mathcal{S}^{(2)},\ \rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau) \in \mathsf{Win}_{\mathsf{p}}$.

Notice that since $\mathcal{P}$ and $\mathbf{P}$ share the same initial state, we can see $\lambda_0$ as a configuration of $\mathcal{M}_{\mathcal{P}}$. We define $\gamma_0$ as the disjoint union of $n$ graphs $\lambda_0$, where $n$ is the number of words on the alphabet $Q^{(P)}$, where each letter appears at most once. The initial graph is composed of $n$ disconnected graphs, to which we will refer as the graph $w$ with $w = w_1 \cdots w_k$ such that for all $i, j \in \{1 \dots k\}$ we have $w_i, w_j \in Q^{(P)}$ and if $i \neq j$ then $w_i \neq w_j$. We use the prefix relation to compare two such words, *i.e.* $w$ is smaller than $w'$, written $w \preceq w'$ if $w$ is a prefix of $w'$. Last, for a state $w \in Q^{(P)}$, we will say that $q$ belongs to $w$, written $q \in w$ if there exists $i \in \{1 \dots k\}$ such that $w_i = q$.

We now explain how to define the scheduler $\pi$, that mimics $\sigma$ in every graph $w$ and that starts in the graph $\varepsilon$. It is not straightforward to define a scheduler from the strategy, since in $\mathbf{P}$, every state $q \in Q^{(P)}$ is duplicated in $\mathcal{P}$ into $(q, 1)$ and $(q, 2)$. Hence we cannot establish a direct correspondence between the configurations of $\mathcal{M}_{\mathcal{P}}$ and $\mathsf{G}_{\mathbf{P}}$. Instead we use the memory of $\pi$ to make this correspondence feasible. The idea is that $\pi$ plays in graph $w$ as if for every $q \notin w$ Player 2 always chooses in the future to move to $(q, 1)$, and for $q \in w$ as if Player 2 always chooses not to move to $(q, 1)$.

Let us consider a fixed probabilistic state $q \in Q^{(P)}$ such that $\Delta^{\mathsf{int}}(q) = \mu$, we can assume w.l.o.g. that there are only two successors to $q$, the left one $q_l$ and the right one $q_r$, and hence that $\mu(q_r) + \mu(q_l) = 1$. The scheduler $\pi$ we build has memory. We will explain how to update the memory states. Assume that $w$ is the graph in which $\pi$ is currently playing, and that its memory state is $mem_\pi$. Two cases arise.

- Assume first that $q \notin w$. The normal behavior of the scheduler $\pi$ is to mimic what $\sigma$ does on the state that $mem_\pi$ indicates it should be, when in the graph $w$. If at some point, $\sigma$ moves a parity protocol to the state $(q, 2)$, then $\pi$ moves the corresponding node $v$, to state $q$. The consequential probabilistic choice then leads to either $q_l$ or $q_r$. Assume, for example, that it is $q_r$. In this

case, we store in the memory state $mem_\pi$ that $v$ should be in state $(q, 1)$ and the computation continue.
After some time, $\sigma$ may decide to move this process out of $(q, 1)$. There are two possible cases.

- The easy case is when the strategy dictates to move to $q_r$. Since $v$ is already in $q_r$, nothing needs to be done, and the computation resumes.
- Otherwise, the process should have been moved to $q_l$. This somehow contradicts the result of the probabilistic choice that already happened in the probabilistic protocol. In this case, $mem_\pi$ remembers that the graph $w$ needs a process in state $q_l$. Also, the scheduler $\pi$ changes its working graph to the graph $w'$, where $w'$ is a shortest word such that $w.q \preceq w'$ and such that the graph $w'$ does not already need a process in a any state.

– Assume now that $q \in w$. As in the other case $\pi$ plays as $\sigma$ would have play on the state that that $mem_\pi$ indicates it should be, when in the graph $w$. If at some point, $\sigma$ moves a parity protocol to the state $(q, 2)$, then $\pi$ moves the corresponding node $v$ to state $q$. The subsequent probabilistic choice then leads to either $q_l$ or $q_r$. Assume for example that it is $q_r$.
If there exists[4] $w' \preceq w$ such that $w'$ needs a process in state $q_r$, then $\pi$ changes its working graph to $w'$. Also $\pi$ switches the process labelled by $q_r$ in $w$ and the one "in need" in $w'$. Doing so, the configuration in $w'$ corresponds exactly to a configuration reachable by $\sigma$, and the configuration in $w$ corresponds to a configuration where Player 2 has chosen $q_l$ instead of $q_r$.
If there is no such $w' \preceq w$ and $w'$ needs a process in state $q_r$ (*e.g.* they may all need a process in state $q_l$), then $\pi$ continues to mimic $\sigma$ in $w$ assuming Player 2 chooses to move directly to $q_r$.

First, notice that in every graph $w$, $\pi$ plays accordingly to $\sigma$. Second, notice that if $\pi$ is playing in graph $w$ then for every $q \in w$ there must exist $w' \preceq w$ such that the graph $w'$ needs either $q_r$ or $q_l$. Last, notice that for every play, there exists a smallest graph $w$ such that the play visits infinitely often $w$. Indeed, to reach from $w$ another graph $w'$ with neither $w \preceq w'$ nor $w' \preceq w$, the play necessarily visits a graph $w''$ such that of $w'' \preceq w$ and $w'' \preceq w'$.

We now show that $\pi$ as defined above is winning, that is: $\mathbb{P}(\mathcal{M}_\mathcal{P}, \gamma_0, \pi, \Diamond q_f) = 1$. Toward a contradiction, suppose that there exists a set $\mathcal{A}$ of plays respecting $\pi$, such that $\mathbb{P}(\mathcal{M}_\mathcal{P}, \gamma_0, \pi, \mathcal{A}) > 0$ and $\mathbb{P}(\mathcal{M}_\mathcal{P}, \gamma_0, \pi, \mathcal{A} \cap \Diamond q_f) = 0$. Since there are only finitely many graphs, we can assume without loss of generality that all plays of $\mathcal{A}$ share the same smallest graph visited infinitely often $w$.

For $q \in w$, let $\mathcal{A}'_q \subseteq \mathcal{A}$ be the subset of plays in which infinitely often one of state $q$ is visited. Every play of $\mathcal{A}'_q$ is composed of a finite prefix, followed by infinitely many probabilistic choice in $q$ that always have the same outcome. Indeed, otherwise $\pi$ would have moved to a prefix of $w$ that needs $q_r$ or $q_l$. Therefore $\mathbb{P}(\mathcal{M}_\mathcal{P}, \gamma_0, \pi, \mathcal{A}'_q) = 0$, and letting $\mathcal{A}' = \cup_{q \in w} \mathcal{A}'_q$, $\mathbb{P}(\mathcal{M}_\mathcal{P}, \gamma_0, \pi, \mathcal{A}') = 0$.

[4] Remark that if such a $w'$ exists, it is unique.

Consider now a play $\rho$ in $\mathcal{A} \setminus \mathcal{A}'$. After a finite prefix, $\rho$ never visits any state of $w$. Hence, in the graph $w$, the scheduler $\pi$ plays as $\sigma$ when Player 2 always move to the states $Q^{(P)} \times \{1\}$. The only way to win when Player 2 always move to the states $Q^{(P)} \times \{1\}$ is to reach $q_f$, and then to achieve the parity objective by looping on the state $q_f$. Since $\sigma$ is winning, we deduce that $\rho$ eventually reaches $q_f$. Hence we have: $\mathbb{P}(\mathcal{M}_\mathcal{P}, \gamma_0, \pi, (\mathcal{A} \setminus \mathcal{A}') \cap \Diamond q_f) = \mathbb{P}(\mathcal{M}_\mathcal{P}, \gamma_0, \pi, \mathcal{A} \setminus \mathcal{A}')$, and since $\mathbb{P}(\mathcal{M}_\mathcal{P}, \gamma_0, \pi, \mathcal{A}) = \mathbb{P}(\mathcal{M}_\mathcal{P}, \gamma_0, \pi, (\mathcal{A} \setminus \mathcal{A}')) + \mathbb{P}(\mathcal{M}_\mathcal{P}, \gamma_0, \pi, \mathcal{A}')$ we obtain $\mathbb{P}(\mathcal{M}_\mathcal{P}, \gamma_0, \pi, \mathcal{A} \cap \Diamond q_f) > 0$. This contradicts the definition of $\mathcal{A}$. Therefore, $\mathbb{P}(\mathcal{M}_\mathcal{P}, \gamma_0, \pi, \Diamond q_f) = 1$. □

### B.4 Proof of Theorem 2

*Proof.* To establish the co-NP hardness we reduce the unsatisfiability problem to $\textsc{Reach}^{=1}_{\max}$.

Let $\mathcal{V}$ be a set of variables, and $\varphi = \bigwedge_{0 \leq i \leq l} x_{i,1} \vee x_{i,2} \vee x_{i,3}$ a formula in conjunctive normal form where each litteral $x_{i,j}$ is either a variable $y \in \mathcal{V}$ or its negation $\bar{y}$. From $\varphi$ we define the following probabilistic protocol $\mathcal{P}_\varphi = (Q, Q^{(1)}, Q^{(P)}, q_0, \Sigma, \Delta, \Delta^{\mathsf{int}})$, where:

- $Q^{(1)} = \{x_{i,j} \mid 0 \leq i \leq l,\ 1 \leq j \leq 3\} \cup \{q_0, \mathsf{r}_1, q_f\} \cup \{y_1, \bar{y_1}, y_2, \bar{y_2} \mid y \in \mathcal{V}\}$
- $Q^{(P)} = \mathcal{V} \cup \{\mathsf{r}_P\}$;
- $\Sigma = \{y, \bar{y} \mid y \in \mathcal{V}\} \cup \{\mathtt{ok}\}$;
- $\Delta = \{(x_{i,j}, ??\bar{x}_{i,j+1}, x_{i,j+1}) \mid 0 \leq i \leq l,\ 1 \leq j \leq 3\}$
  $\cup \{(q_0, \varepsilon, x_{i,0}), (x_{i,3}, !!\mathtt{ok}, \mathsf{r}_1) \mid 0 \leq i \leq l\}$
  $\cup \{(q_0, \varepsilon, y), (y_1, !!y, y_2), (\bar{y_1}, !!\bar{y}, \bar{y_2}), (y_2, ??\mathtt{ok}, q_0), (\bar{y_2}, ??\mathtt{ok}, q_0) \mid y \in \mathcal{V}\}$
  $\cup \{(\mathsf{r}_1, \varepsilon, \mathsf{r}_P)\}$;
- for every $y \in \mathcal{V}$, $\Delta^{\mathsf{int}}(y)(y_1) = \Delta^{\mathsf{int}}(y)(\bar{y_1}) = \frac{1}{2}$,
  and $\Delta^{\mathsf{int}}(\mathsf{r}_P)(q_f) = \Delta^{\mathsf{int}}(\mathsf{r}_P)(q_0) = \frac{1}{2}$.

The construction is illustrated on an example in Figure 4, where, for simplicity, the initial state $q_0$ of the probabilistic protocol is duplicated.

Let us now prove that $\varphi$ is unsatisfiable if and only if there exists an initial configuration $\gamma_0 \in \Gamma_{\mathcal{P},0}$ and a scheduler $\pi$ such that $\mathbb{P}(\mathcal{M}_\mathcal{P}, \gamma_0, \pi, \Diamond q_f) = 1$.

Assume first that $\varphi$ is unsatisfiable. Without loss of generality, we assume that each clause in $\varphi$ do not contain two literals with the same variable ($y \vee \bar{y}$ can be replaced with $\mathsf{true}$, and $y \vee y$ with $y$). We set $N = \mathit{card}(\mathcal{V}) + 1$ and $\gamma_0 = (V, E, L)$ with $\mathit{card}(V) = N$, $E = V \times V \setminus \{(v, v) \mid v \in V\}$ and $L(v) = (q_0, \bot)$ for every $v \in V$. We define a scheduler $\pi$ for $\mathcal{M}_\mathcal{P}$ as follows. From the initial configuration $\gamma_0$, and for each variable $y \in \mathcal{V}$, $\pi$ decides to perform transition $(q_0, \varepsilon, y)$ followed by the corresponding probabilistic transition, thus moving one process to either state $y$ or state $\bar{y}$. These processes are called *variable processes* and at this step, their position defines a valuation $\mathbf{v} : \mathcal{V} \to \{0, 1\}$. Alternatively, $\mathbf{v}$ can be seen as the set of variables or negated variables that fulfill the valuation. Since $\varphi$ is unsatisfiable, there exists a clause $i$ such that for every literal $x_{i,j}$, $x_{i,j} \notin \mathbf{v}$. At this stage, $\pi$ decides to move the remaining process, the *formula process*, along the transition $(q_0, \varepsilon, x_{i,0})$ corresponding to choosing the clause

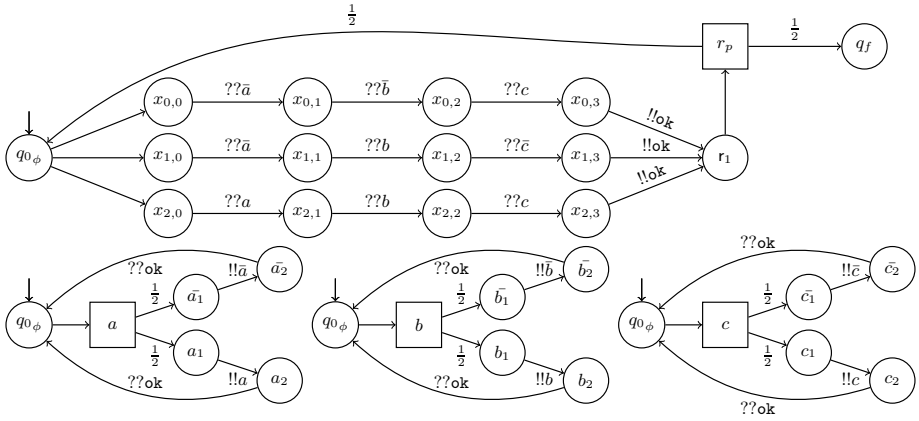


Fig. 4: Probabilistic protocol for the formula $\varphi = (a \vee b \vee \bar{c}) \wedge (a \vee \bar{b} \vee c) \wedge (\bar{a} \vee \bar{b} \vee \bar{c})$

which is violated by $\mathbf{v}$. Next, the variable processes in state $x_{i,1}^{-}, x_{i,2}^{-}$ and $x_{i,3}^{-}$ broadcast in turn to the formula process, so that the latter moves to state $x_{i,3}$. The other variable processes broadcast to no one. Then, the formula process broadcasts to all variable processes the $\mathtt{ok}$-message: the $N-1$ variable processes are back to their initial state $q_0$, and the formula process reaches state $\mathtt{r}_P$, where a probabilistic choice happens. With probability half, the state $q_f$ is reached, and with the remaining probability the formula process is back in the initial state $q_0$, so that all processes are back in the initial configuration $\gamma_0$. In this first round, $\pi$ thus ensures to reach $q_f$ with probability $\frac{1}{2}$. Iterating this process, $\pi$ ensures to reach $q_f$ almost-surely: $\mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \Diamond q_f) = 1$.

Assume now that $\varphi$ is satisfiable, and let $\mathbf{v} : \mathcal{V} \to \{0,1\}$ be a satisfying assignment of the variables. We fix $\pi$ an arbitrary scheduler and $\gamma_0$ an arbitrary initial configuration, say with $N$ processes. A probabilistic choice from state $y$ is said *consistent* with $\mathbf{v}$ if it sets $y$ to its truth value in $\mathbf{v}$. We aim at showing that $\mathbb{P}(\mathcal{M}_{\mathcal{P}_\phi}, \gamma_0, \pi, \Diamond q_f)$ is bounded away from 1 by some factor independent of $\pi$. We partition the set of paths $\Omega$ into incompatible events:

$$\Omega = \mathcal{A}_0 \sqcup \mathcal{A}_1 \sqcup \cdots \sqcup \mathcal{A}_N \sqcup \mathcal{A}_{N+1} \sqcup \mathcal{A}'_1 \sqcup \cdots \sqcup \mathcal{A}'_N, \text{ where}$$

- for $0 \leq k \leq N$, $\mathcal{A}_k$ is the set of paths that contain exactly $k$ probabilistic choices, and all of them are consistent with $\mathbf{v}$;
- $\mathcal{A}_{N+1}$ is the set of paths that contain at least $N+1$ probabilistic choices, and the first $N+1$ are consistent with $\mathbf{v}$;
- for $1 \leq k \leq N$, $\mathcal{A}'_k$ is the set of paths that contain at least $k$ probabilistic choices, the first $k-1$ are consistent with $\mathbf{v}$ and the $k$-th is not.

Clearly enough, $\mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \mathcal{A}'_k) \leq \frac{1}{2^k}$, for every $1 \leq k \leq N$. Therefore, $\mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \sqcup_{k=1}^{N} \mathcal{A}'_k) \leq 1 - \frac{1}{2^N}$. Observe also that $\mathcal{A}_{N+1}$ is empty. Indeed,

if the first probabilistic choices are consistent with $\mathbf{v}$, then the formula processes, if any, stay stuck in the initial state $q_0$ and the variable processes cannot receive an $\mathtt{ok}$ to get back to $q_0$. Therefore, starting with $N$ processes, only $N$ consecutive probabilistic choices consistent with $\mathbf{v}$ are possible, and $\mathcal{A}_{N+1}$ is empty. We thus derive a bound for the remaining paths in the partition: $\mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \sqcup_{k=0}^{N} \mathcal{A}_k) \geq \frac{1}{2^N}$. It remains to prove that all paths in the sets $\mathcal{A}_k$ end in a deadlock before reaching $q_f$. Indeed, if the $k$ first probabilistic choices for variable processes are consistent with $\mathbf{v}$, then the formula processes cannot progress further than states $x_{i,2}$'s. As a consequence, the formula processes cannot move back to $q_0$ and get stuck. Therefore, $\mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \neg\Diamond q_f) \geq \frac{1}{2^N}$, or equivalently, $\mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \Diamond q_f) \leq 1 - \frac{1}{2^N}$. Because this holds for every scheduler, we finally obtain $\inf_\pi \mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \Diamond q_f) < 1$. □

### B.5 Proof of Theorem 3

**$\mathbf{Reach}^{=\mathbf{0}}_{\mathbf{min}}$** From $\mathcal{P} = (Q, Q^{(1)}, Q^{(P)}, q_0, \Sigma, \Delta, \Delta^{\mathsf{int}})$ a probabilistic protocol and $q_f \in Q$ a control state, we derive the parity protocol $\mathbf{P} = (Q_{\mathbf{P}}, Q^{(1)}_{\mathbf{P}}, Q^{(2)}_{\mathbf{P}}, q_{0\mathbf{P}}, \Sigma_{\mathbf{P}}, \Delta_{\mathbf{P}}, \mathsf{col}, \mathsf{safe})$ as follows:

- $Q_{\mathbf{P}} = Q$, $Q^{(1)}_{\mathbf{P}} = Q^{(1)}$, $Q^{(2)}_{\mathbf{P}} = Q^{(P)}$, and $q_{0\mathbf{P}} = q_0$;
- $\Sigma_{\mathbf{P}} = \Sigma$;
- $\Delta_{\mathbf{P}} = \Delta \cup \{(q, \varepsilon, q') \mid \Delta^{\mathsf{int}}(q)(q') > 0\}$;
- $\mathsf{safe} = \Delta \setminus \{\delta \mid \delta \in Q \times \big(\{!!a, ??a \mid a \in \Sigma\} \cup \{\varepsilon\}\big) \times \{q_f\}\}$.

Notice that we consider only safety objective, hence the coloring function $\mathsf{col}$ needs not be defined. Intuitively, all random choices in $\mathcal{P}$ are replaced in $\mathbf{P}$ with choices for Player 2. This construction ensures the following equivalence:

**Proposition 2.** *$\exists \lambda_0 \in \Lambda_{\mathbf{P},0}.\ \exists \sigma \in \mathcal{S}^{(1)}.\ \forall \tau \in \mathcal{S}^{(2)},\ \rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau) \in \mathsf{Win}_{\mathsf{s}}$ if and only if $\exists \gamma_0 \in \Gamma_{\mathcal{P},0}.\ \mathbb{P}_{\min}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \Diamond q_f) = 0$.*

*Proof (sketch).* Transitions with target $q_f$ are the only ones that do not belong to the safe set $\mathsf{safe}$. Hence, a winning strategy for the safety objective must ensure that $q_f$ is avoided. Similarly, in order to reach $q_f$ with probability 0, a scheduler also has to ensure that $q_f$ is avoided. Since the parity protocol $\mathbf{P}$ and the probabilistic protocol $\mathcal{P}$ only differ on the type of adversarial states (belonging to Player 2, or probabilistic), the correspondence between strategies and schedulers is immediate. □

From Proposition 2 and Theorem 1 we deduce the decidability of $\textsc{Reach}^{=0}_{\min}$ and establish a complexity upper-bound.

**Corollary 1.** *$\textsc{Reach}^{=0}_{\min}$ is in $\Pi^{\mathrm{P}}_2$.*

**$\mathbf{Reach}^{<1}_{\mathbf{min}}$** For $\textsc{Reach}^{<1}_{\min}$ we reduce to a game problem for parity protocols with a combination of safety and parity winning conditions. From $\mathcal{P} = (Q, Q^{(1)}, Q^{(P)}, q_0, \Sigma, \Delta, \Delta^{\mathsf{int}})$ a probabilistic protocol and $q_f \in Q$ a control state, we define the parity protocol $\mathbf{P} = (Q_{\mathbf{P}}, Q^{(1)}_{\mathbf{P}}, Q^{(2)}_{\mathbf{P}}, q_{0\mathbf{P}}, \Sigma_{\mathbf{P}}, \Delta_{\mathbf{P}}, \mathsf{col}, \mathsf{safe})$ as:

- $Q^{(1)}_{\mathbf{P}} = \big(Q^{(1)} \times \{1,2\}\big) \cup \big(Q^{(P)} \times \{1\}\big)$, $Q^{(2)}_{\mathbf{P}} = Q^{(P)} \times \{2\}$, and $q_{0\mathbf{P}} = (q_0, 1)$;
- $\Sigma_{\mathbf{P}} = \Sigma$;
- $\Delta_{\mathbf{P}} = \{((q,i), \_, (q',i)) | (q, \_, q') \in \Delta, i \in \{1,2\}\}$
  $\cup\ \{((q,i), \varepsilon, (q',i)) | \Delta^{\mathsf{int}}(q)(q') > 0, i \in \{1,2\}\} \cup\ \{((q,1), \varepsilon, (q,2)) | q \in Q\}$;
- $\mathsf{col}((q,i), \_, (q',i'))) = 2$ if $i' = 2$, and 1 otherwise;
- $\mathsf{safe} = \Delta \setminus \{\delta \mid \delta \in Q \times \big(\{!!a, ??a \mid a \in \Sigma\} \cup \{\varepsilon\}\big) \times \{q_f\}\}$.

Intuitively, $\mathbf{P}$ consists of two copies of $\mathcal{P}$. In the first copy, all random choices are replaced with choices of Player 1, whereas in the second copy they are replaced with choices of Player 2. Also, at any time, one can move from the first to the second copy. This construction ensures the following equivalence:

**Proposition 3.** $\exists \lambda_0 \in \Lambda_{\mathbf{P},0}.\ \exists \sigma \in \mathcal{S}^{(1)}.\ \forall \tau \in \mathcal{S}^{(2)},\ \rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau) \in \mathsf{Win}_{\mathsf{sp}}$ *if and only if* $\exists \gamma_0 \in \Gamma_{\mathcal{P},0}.\ \mathbb{P}_{\min}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \Diamond q_f) < 1$.

*Proof (sketch).* Since the only unsafe edges are the ones leading to $q_f$, the safety parity winning condition forces a winning strategy to avoid $q_f$. Moreover, the parity condition ensures that only a finite number of "probabilistic" choices (*i.e.* choices that correspond to probabilistic choices in $\mathcal{P}$) are made by Player 1. On one hand, the existence of a winning strategy implies that with a fixed finite number of probabilistic choices for Player 1, $q_f$ can be avoided. On the other hand, any scheduler for which the probability to reach $q_f$ is less than one, guarantees that after a finite prefix (hence a finitely many probabilistic choices) $q_f$ is avoided with probability one (thus whatever the other probabilistic choices).

**Corollary 2.** $\textsc{Reach}^{<1}_{\min}$ *is in* $\Pi^{\mathrm{P}}_2$.

### B.6 Proof of Theorem 4

The decidability of the last decision problems relies on a monotonicity property.

**Lemma 9.** $\forall \pi\ \forall \gamma_0, \forall \gamma'_0 \supseteq \gamma_0\ \exists \pi'.\ \mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \pi, \Diamond q_f) \leq \mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma'_0, \pi', \Diamond q_f)$.

*Proof.* Intuitively, $\pi'$ behaves as $\pi$ on the group of processes present in $\gamma_0$, and in case of a deadlock of these processes, it uses the remaining processes in an arbitrary way. □

Due to Lemma 9 it is sufficient to consider the initial configuration with a single node. We then solve these decision problems on the corresponding finite-state MDP.

In this proof we write $\gamma_0$ for the initial configuration with a single node.

If $\mathbb{P}_{\max}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \Diamond q_f) = 0$, then, trivially, the answer to $\textsc{Reach}^{=0}_{\max}$ is positive. Otherwise, $\mathbb{P}_{\max}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \Diamond q_f) > 0$, and for every other initial configuration $\gamma_0'$, since $\gamma_0' \supseteq \gamma_0$, by Lemma 9, we obtain that there exists a scheduler $\pi'$ with $\mathbb{P}(\mathcal{M}_{\mathcal{P}}, \gamma_0', \pi', \Diamond q_f) \geq \mathbb{P}_{\max}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \Diamond q_f) > 0$. Therefore, $\mathbb{P}_{\max}(\mathcal{M}_{\mathcal{P}}, \gamma_0', \Diamond q_f) > 0$, and the answer to $\textsc{Reach}^{=0}_{\max}$ is negative. Exactly the same reasoning applies for $\textsc{Reach}^{<1}_{\max}$.

If $\mathbb{P}_{\min}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \Diamond q_f) = 1$, then, the answer to $\textsc{Reach}^{=1}_{\min}$ is positive. Otherwise, $\mathbb{P}_{\min}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \Diamond q_f) < 1$, and for $\gamma_0' \supseteq \gamma_0$, we consider the class of schedulers that always choose a completely disconnected topology of the network, *i.e.* no edge in the graph. For this class of schedulers, the behavior of each node is independent of the others, and hence simulate what happens from $\gamma_0$, the initial configuration with a single node. Therefore, $\mathbb{P}_{\min}(\mathcal{M}_{\mathcal{P}}, \gamma_0', \Diamond q_f) \leq \mathbb{P}_{\min}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \Diamond q_f) < 1$, and the answer to $\textsc{Reach}^{=1}_{\min}$ is negative. Exactly the same reasoning applies for $\textsc{Reach}^{>0}_{\min}$.