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 [12], 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.

A. Muscholl (Ed.): FOSSACS 2014, LNCS 8412, pp. 134–148, 2014.

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 [12], 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 [11] 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 [13] or acyclic directed graphs [1].

We propose here to extend the model of reconfigurable broadcast networks studied in [11] 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 [14,8]. 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 [2,15] (where a monotonic well-quasi order is associated to the set of configurations), a class to which belong the broadcast reconfigurable networks of [12], 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 [9]). 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 details and proofs can be found in [7].

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 $\mathcal{M} = (\Gamma, \Gamma^{(1)}, \Gamma^{(p)}, \to, prob)$ where Γ is a denumerable set of configurations (or vertices) partitioned into the configurations of Player 1 $\Gamma^{(1)}$ and the probabilistic configurations $\Gamma^{(p)}$; $\to$: $\Gamma^{(1)} \mapsto \Gamma$ is the non deterministic transition relation; $prob : \Gamma^{(p)} \mapsto \mathsf{Dist}(\Gamma^{(1)})$ is the probabilistic transition relation.*

For a tuple $(\gamma, \gamma') \in \to$, we will sometimes use the notations $\gamma \to \gamma'$. A *finite path* in the game $\mathcal{M} = (\Gamma, \Gamma^{(1)}, \Gamma^{(p)}, \to, prob)$ is a finite sequence of configurations $\gamma_0\gamma_1\dots\gamma_k$ such that for all $0 \leq i \leq k-1$, if $\gamma_i \in \Gamma^{(1)}$ then $\gamma_i \to \gamma_{i+1}$ and otherwise $prob(\gamma_i)(\gamma_{i+1}) > 0$; moreover we will say that such a path starts from the configuration γ_0. An *infinite path* is an infinite sequence $\rho \in \Gamma^\omega$ such that any finite prefix of ρ is a finite path. Furthermore we will say that a path ρ is *maximal* if it is infinite or it is finite and there does not exist a configuration γ 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 Ω.

A *scheduler* in the game $\mathcal{M} = (\Gamma, \Gamma^{(1)}, \Gamma^{(p)}, \to, prob)$ 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 $\gamma \to \gamma'$. We denote by Π the set of schedulers for $\mathcal{M}$. Given a scheduler $\pi \in \Pi$, we say that a finite path $\gamma_0\gamma_1\dots\gamma_n$ *respects* the scheduler π if for every $i \in \{0\dots n-1\}$, we have that if $\gamma_i \in \Gamma^{(1)}$ then $\pi(\gamma_0\dots\gamma_i) = \gamma_{i+1}$. Similarly we say that an infinite path $\rho = \gamma_0\gamma_1\dots$ *respects* the scheduler π if every finite prefix of ρ respects π.

Remark 1. Alternatively, a scheduler in the game $\mathcal{M} = (\Gamma, \Gamma^{(1)}, \Gamma^{(p)}, \rightarrow, prob)$ 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* $\pi_M : \Gamma^{(1)} \times M \rightarrow \Gamma$, an *update function* $U_M : \Gamma^{(1)} \times M \times \Gamma \rightarrow M$, and an initialization function $I_M : \Gamma^{(0)} \rightarrow M$. 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 π. We define then extremal probabilities of the event $\mathcal{A}$ starting from configuration γ 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 [11] with probability and can be defined in two steps. First, a configuration of the network is represented by 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; Σ is a finite message alphabet; $\Delta \subseteq (Q^{(1)} \times \{!!a, ??a \mid a \in$*

$\Sigma\} \times Q^{(1)}) \cup (Q^{(1)} \times \{\varepsilon\} \times Q)$ *is the transition relation;* $\Delta^{int} : Q^{(P)} \mapsto \mathsf{Dist}(Q^{(1)})$ *is the internal probabilistic transition relation.*

The label $!!a$ [resp. $??a$] represents the broadcast [resp. reception] of the message $a \in \Sigma$, whereas ε 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 $\{q \in Q \mid \exists (q, !!a, q') \in \Delta \text{ or } \exists (q, \varepsilon, q') \in \Delta\}$ 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 terms of an infinite state $1 - \frac{1}{2}$ player game. We will represent the network by 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 is a 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 terms of the $1 - \frac{1}{2}$ player game $\mathcal{M}_{\mathcal{P}} = (\Gamma_{\mathcal{P}}, \Gamma_{\mathcal{P}}^{(1)}, \Gamma_{\mathcal{P}}^{(p)}, \to_{\mathcal{P}}, prob_{\mathcal{P}})$ where:

- $\to_{\mathcal{P}} \subseteq \Gamma_{\mathcal{P}}^{(1)} \times \Gamma_{\mathcal{P}}$ 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 $\gamma \to_{\mathcal{P}} \gamma'$ 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 $q \in \mathit{ActStates}$ 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 step E' is arbitrarily defined and this is what induces reconfiguration);

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. Furthermore, note that since the topology can change arbitrarily at any reconfiguration step, we could have considered an equivalent semantics without topology but with a set of possible receivers for each emitted message.

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.

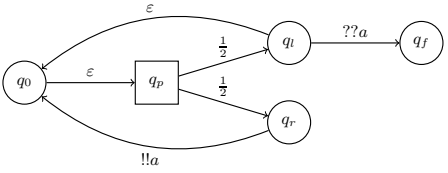

Fig. 1. Simple example of probabilistic protocol

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 $\textsc{Reach}^{\sim \mathtt{b}}_{\mathtt{opt}}$ be the following problem:

Input: A probabilistic protocol $\mathcal{P}$, and a control state $q_f \in Q$.
Question: Does there exist an initial configuration γ_0 such that $\mathbb{P}_{\mathtt{opt}}(\mathcal{M}_{\mathcal{P}}, \gamma_0, \Diamond q_f) \sim \mathtt{b}$?

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 [12]). 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 γ_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, 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* Λ *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* ρ 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 λ_0. An *infinite path* is an infinite sequence $\rho \in \Lambda^\omega$ such that any finite prefix of ρ is a finite path. Similarly to paths in $1 - \frac{1}{2}$ player game, *maximal paths* in 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 ρ 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 (σ, τ) and an initial configuration λ_0, the game 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 $\lambda_{i+1} = \sigma(\lambda_0 \dots \lambda_i)$, otherwise $\lambda_{i+1} = \tau(\lambda_0 \dots \lambda_i)$.

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 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. Note that in the context of games played over a finite graph the safety-parity objective can easily be turned into a parity objective, by removing the unsafe edges and by adding an even parity self-loop on deadlock states; However when the game is played on an infinite graph, this transformation is difficult because one first has to be able to detect deadlock configurations. Finally, we say that a play ρ 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 σ for Player 1 is a *winning strategy* from configuration λ_0 if for every strategy τ 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;* Σ *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. 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 (the complete definition can be found in [7]). 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. The set of configurations $\Lambda_{\mathbf{P}}$ of the network built over a parity protocol $\mathbf{P} = (Q, Q^{(1)}, Q^{(2)}, q_0, \Sigma, \Delta, \mathsf{col}, \mathsf{safe})$ 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 $\Lambda_{\mathbf{P}}^{(1)} = \mathcal{G}_{Q\times\{\bot\}} \cup \{(V, E, L) \in \Lambda_{\mathbf{P}} \mid card(\{v \in V \mid L(v) \in Q^{(1)} \times \{\top\}\}) = 1\}$and $\Lambda_{\mathbf{P}}^{(2)} = \{(V, E, L) \in \Lambda_{\mathbf{P}} \mid card(\{v \in V \mid L(v) \in Q^{(2)} \times \{\top\}\}) = 1\}$. 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 $\mathsf{G}_{\mathbf{P}} = (\Lambda_{\mathbf{P}}, \Lambda_{\mathbf{P}}^{(1)}, \Lambda_{\mathbf{P}}^{(2)}, T_{\mathbf{P}}, \mathsf{col}_{\mathbf{P}}, \mathsf{safe}_{\mathbf{P}})$ where $T_{\mathbf{P}} \subseteq \Lambda_{\mathbf{P}} \times \Lambda_{\mathbf{P}}$ is defined using reconfiguration and process choices for Player 1 and internal and communication rules as the one defined in the case of probabilistic protocols, whereas $\mathsf{col}_{\mathbf{P}} : T_{\mathbf{P}} \to \mathbb{N}$ and $\mathsf{safe}_{\mathbf{P}} \subseteq T_{\mathbf{P}}$ are defined following col and safe lifting the definition from states to configurations. 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 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 λ_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.

We now 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 $b : (Q^{(2)} \cap ActStates) \mapsto \Delta$ such that for all $q \in Q^{(2)} \cap ActStates$, $b(q) \in \{(q, \varepsilon, q') \mid (q, \varepsilon, q') \in \Delta\}$. Such a local behavior induces what we will call a *local strategy* τ_b for Player 2 in $\mathsf{G}_{\mathbf{P}}$ defined as follows: let ρ 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 λ' is the unique configuration obtained from λ 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 $\mathcal{S}_{\mathsf{I}}^{(2)}$ 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 $\mathcal{S}_{\mathsf{I}}^{(2)}$ is thus finite and contains at most $card(\Delta)$ 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* $\exists \lambda_0 \in \Lambda_{\mathbf{P},0}.\ \exists \sigma \in \mathcal{S}^{(1)}.\ \forall \tau \in \mathcal{S}^{(2)},\ \rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau) \in \mathsf{Win} \iff \forall \tau \in \mathcal{S}^{(2)}_I.\ \exists \lambda_0 \in \Lambda_{\mathbf{P},0}.\ \exists \sigma \in \mathcal{S}^{(1)}, \rho(\mathsf{G}_{\mathbf{P}}, \lambda_0, \sigma, \tau) \in \mathsf{Win}$

The proof of this lemma shares some similarities with the one to establish that memoryless strategies are sufficient for Player 2 in energy parity games [10]. It is performed by induction on the number of states of Player 2 in the parity protocol. In the induction step, a configuration is split into several sub-configurations (one for each local strategy of Player 2) and Player 1 navigates among the sub-configurations each time Player 2 changes strategy. For instance if Player 2 has two choices, say left and right, then at the beginning Player 1 plays in the "left"-sub-configuration and when Player 2 decides to choose right instead of left, then the associated node is moved to the "right"-sub-configuration and the game proceeds in this sub-configuration, and so on. It can be shown that if Player 1 wins against the strategy which chooses always left and against the one which chooses always right, then it wins agains any strategy of Player 2.

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 and 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 col' is the restriction of col to Δ' 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* ρ *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 $L(\lambda_n) \subseteq (Q \times \{\bot\}) \setminus (\mathit{ActStates} \times \{\bot\})$. In [11], 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* Win_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 [11]. Then from [11] 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 values is 0 (i.e. a cycle whose edge's labels sum to 0) and this is can be done in polynomial time thanks to [16].

Lemma 4. *The problem of deciding whether there exists an infinite path* ρ *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 Win_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 Win_p instead of Win_s (use Lemma 3 with all the transitions considered as safe) and there is an NP algorithm for the same problem with Win_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 Win_p we will have an NP algorithm and for Win_s and Win_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* Π_2^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 $\text{Reach}^{=1}_{\max}$

Let us now discuss the most involved case, $\text{REACH}^{=1}_{\max}$, 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:

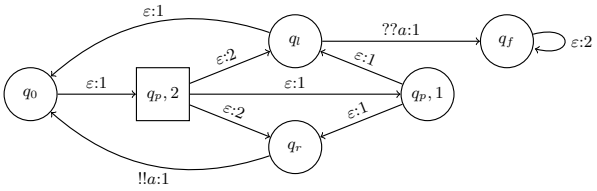

Fig. 2. Parity protocol for the probabilistic protocol from Figure 1

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 π ensures to reach q_f with probability 1, one builds a winning strategy σ 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 λ 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 π from γ to a configuration containing q_f, where γ is defined as λ but the label of v is q. Assuming that π reaches q_f with probability 1, such a path must exist for every reachable configuration in the game. This definition of σ 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 σ, we define a scheduler π that mimics the choices of σ 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. $\text{REACH}^{=1}_{\max}$ *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 $\text{REACH}^{=1}_{\max}$. From φ a formula in conjunctive normal form, we define a probabilistic protocol $\mathcal{P}_\varphi$ and a control state q_f such that φ is unsatisfiable if and only if there exists an initial configuration $\gamma_0 \in \Gamma_{\mathcal{P},0}$ and a scheduler π 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 φ is unsatisfiable, is to generate a random assignment of the variables (using the gadgets represented bottom of the Figure), which will necessarily violate a clause of φ. Choosing then this clause in the above part of the protocol allows to reach state 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 φ is satisfiable, there is a positive probability to generate a valuation satisfying it, and then not to be able to reach 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 $\text{REACH}^{=0}_{\min}$ [resp. $\text{REACH}^{<1}_{\min}$] 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 $\text{REACH}^{=0}_{\min}$, 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 safe. For $\text{REACH}^{<1}_{\min}$, $\mathbf{P}$ 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 are replaced with choices of Player 2. Also, at any time, one can move from the

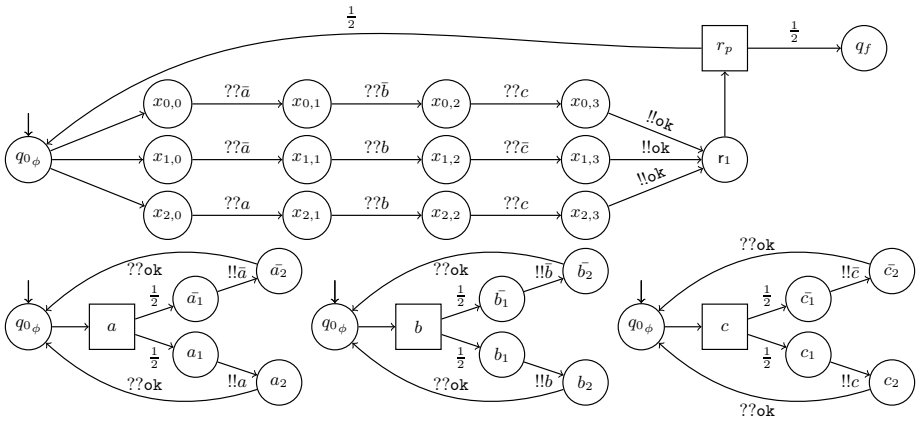

Fig. 3. Probabilistic protocol for the formula $\varphi = (a \vee b \vee \bar{c}) \wedge (a \vee \bar{b} \vee c) \wedge (\bar{a} \vee \bar{b} \vee \bar{c})$

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. $\text{REACH}^{=0}_{\min}$ *and* $\text{REACH}^{<1}_{\min}$ *are in* Π^{P}_2.

The decidability and complexity of the remaining cases are established directly, without reducing to games on parity protocols. First of all, $\text{REACH}^{>0}_{\max}$ is interreducible to the reachability problem in *non-probabilistic* reconfigurable broadcast networks, known to be P-complete [11]. 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. $\text{REACH}^{>0}_{\max}$, $\text{REACH}^{=0}_{\max}$, $\text{REACH}^{<1}_{\max}$, $\text{REACH}^{=1}_{\min}$ *and* $\text{REACH}^{>0}_{\min}$ *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 coNP^{NP}, as summarized in the table below.

Problem	$\text{REACH}^{=0}_{\min}$	$\text{REACH}^{<1}_{\min}$	$\text{REACH}^{=1}_{\max}$	others
Complexity	Π^{P}_2	Π^{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. Abdulla, P.A., Atig, M.F., Rezine, O.: Verification of directed acyclic ad hoc networks. In: Beyer, D., Boreale, M. (eds.) FMOODS/FORTE 2013. LNCS, vol. 7892, pp. 193–208. Springer, Heidelberg (2013)
2. Abdulla, P.A., Cerans, K., Jonsson, B., Tsay, Y.-K.: General decidability theorems for infinite-state systems. In: LICS 1996, pp. 313–321. IEEE Computer Society (1996)
3. Abdulla, P.A., Henda, N.B., Mayr, R.: Decisive Markov chains. Logical Methods in Computer Science 3(4) (2007)
4. Baier, C., Bertrand, N., Schnoebelen, P.: Verifying nondeterministic probabilistic channel systems against ω-regular linear-time properties. ACM Transactions on Computational Logic 9(1) (2007)
5. Baier, C., Katoen, J.-P.: Principles of Model Checking. The MIT Press (2008)
6. Bertrand, N., Fournier, P.: Parameterized verification of many identical probabilistic timed processes. In: FSTTCS 2013. LIPIcs, vol. 24, pp. 501–513. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2013)
7. Bertrand, N., Fournier, P., Sangnier, A.: Playing with probabilities in Reconfigurable Broadcast Networks. Research Report HAL-00929857, HAL, CNRS, France (2014)
8. Brázdil, T., Kučera, A., Stražovský, O.: On the decidability of temporal properties of probabilistic pushdown automata. In: Diekert, V., Durand, B. (eds.) STACS 2005. LNCS, vol. 3404, pp. 145–157. Springer, Heidelberg (2005)
9. Chatterjee, K., de Alfaro, L., Faella, M., Legay, A.: Qualitative logics and equivalences for probabilistic systems. Logical Methods in Computer Science 5(2) (2009)
10. Chatterjee, K., Doyen, L.: Energy parity games. Theoretical Computer Science 458, 49–60 (2012)
11. Delzanno, G., Sangnier, A., Traverso, R., Zavattaro, G.: On the complexity of parameterized reachability in reconfigurable broadcast networks. In: FSTTCS 2012. LIPIcs, vol. 18, pp. 289–300. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2012)
12. Delzanno, G., Sangnier, A., Zavattaro, G.: Parameterized verification of ad hoc networks. In: Gastin, P., Laroussinie, F. (eds.) CONCUR 2010. LNCS, vol. 6269, pp. 313–327. Springer, Heidelberg (2010)
13. Delzanno, G., Sangnier, A., Zavattaro, G.: On the power of cliques in the parameterized verification of ad hoc networks. In: Hofmann, M. (ed.) FOSSACS 2011. LNCS, vol. 6604, pp. 441–455. Springer, Heidelberg (2011)
14. Etessami, K., Yannakakis, M.: Recursive Markov decision processes and recursive stochastic games. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 891–903. Springer, Heidelberg (2005)
15. Finkel, A., Schnoebelen, P.: Well-structured transition systems everywhere! Theoretical Computer Science 256(1-2), 63–92 (2001)
16. Kosaraju, S.R., Sullivan, G.F.: Detecting cycles in dynamic graphs in polynomial time (preliminary version). In: STOC 1988, pp. 398–406. ACM (1988)