Model Checking Population Protocols

Javier Esparza[1], Pierre Ganty[2], Jérôme Leroux[3], and Rupak Majumdar[4]

1 **Technical University of Munich, Germany**
2 **IMDEA Software Institute, Madrid, Spain**
3 **LaBRI, CNRS, Univ. Bordeaux, Talence, France**
4 **MPI-SWS, Kaiserslautern, Germany**

Abstract

Population protocols are a model for parameterized systems in which a set of identical, anonymous, finite-state processes interact pairwise through rendezvous synchronization. In each step, the pair of interacting processes is chosen by a random scheduler. Angluin et al. (PODC 2004) studied population protocols as a distributed computation model. They characterized the computational power in the limit (semi-linear predicates) of a subclass of protocols (the well-specified ones). However, the modeling power of protocols go beyond computation of semi-linear predicates and they can be used to study a wide range of distributed protocols, such as asynchronous leader election or consensus, stochastic evolutionary processes, or chemical reaction networks. Correspondingly, one is interested in checking specifications on these protocols that go beyond the well-specified computation of predicates.

In this paper, we characterize the decidability frontier for the model checking problem for population protocols against probabilistic linear-time specifications. We show that the model checking problem is decidable for qualitative objectives, but as hard as the reachability problem for Petri nets—a well-known hard problem without known elementary algorithms. On the other hand, model checking is undecidable for quantitative properties.

1998 ACM Subject Classification D.2.4 Software/Program Verification, F.1.1 Models of Computation

Keywords and phrases parameterized systems, population protocols, probabilistic model checking, probabilistic linear-time specifications, decidability

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2016.27

1 Introduction

Population protocols [3, 4, 6] are a model of distributed computation by anonymous, identical, finite-state agents interacting in pairs. Given an initial configuration—an initial distribution of agents specifying the number of agents at each state—a random scheduler repeatedly chooses a pair of processes and one of the interactions enabled by their current states. This naturally assigns to each initial configuration a semantics in terms of a finite Markov Chain. Thus, the protocol defines infinitely many finite Markov Chains.

Population protocols were originally introduced to study which predicates on an initial configuration of agents (like, for example, "does the configuration contain more processes of type A than of type B") can be computed by the agents themselves in a distributed way. For this, Angluin et al. [3] introduced a definition of computation by consensus: a configuration computes a value if all its agents eventually converge to that value (represented by a particular state or set of states) with probability 1.

Population protocols have been used to model systems beyond their initial motivation in distributed computing. In particular, they can also model trust propagation [15], evolutionary dynamics [23], or chemical reaction systems [24, 25]. These systems do not aim at the computation of predicates, or they do not compute in the way defined by Angluin et al. [3]. With more diverse applications of population protocols comes also new properties one would like to reason about. For instance, Delporte-Gallet et al. [14] studied privacy in population protocols. They proved (by hand) different properties of specific protocols, like "the system can reach a good configuration without any interaction involving a distinguished agent p_0". Another example is the work of Clement et al. [13] who use the probabilistic model checker PRISM to check properties including the aforementioned convergence. However, PRISM is a finite state model checker and therefore the verification was limited to a finite number of individual initial configurations.

In this paper, we solve the general model checking problem for population protocols against probabilistic linear-time (LTL) specifications. As opposed to previous work, we can reason fully automatically starting from sets of initial configurations which can be infinite as long as they are Presburger definable. We show that the qualitative problem (i.e., deciding if the formula holds with probability 1) is decidable, but as hard as the reachability problem for Petri nets (*reachability hard*). The quantitative problem (deciding if the property holds with at least a given probability) is undecidable. In particular, our proof of undecidability for quantitative LTL uses a novel simulation of counter machines by Petri nets with arbitrarily small error. Additionally, we prove (§4) undecidability for both the qualitative problem for broadcast protocols (a stronger model where interactions involve all processes, not just a fixed number) and a natural variant of LTL specifications where atomic propositions are defined on configurations rather than actions.

From the verification point of view, in this paper we study the decidability of parameterized verification for a class of probabilistic systems. Our work can be seen as an extension of the approach of German and Sistla [20] to probabilistic verification. Our results establish certain decidability frontiers for parameterized verification of probabilistic programs, an area of increasing interest [2, 10, 11, 1, 12]. The question whether all instances of a parameterized system satisfy a property with probability 1 was also studied by Pnueli and Zuck with different co-authors [26, 7] and also by Esparza et al. [17]. The emphasis in these papers was on deductive proof systems, and they do not contain decidability results.

2 Definitions and Examples

A *multiset* on a finite non-empty set E is a mapping $M : E \to \mathbb{N}$. Given $e \in E$, let $M(e)$ denote the number of elements of type e in the multiset M. Operations on $\mathbb{N}$, like addition, subtraction, or comparison are implicitly defined on multisets by defining the operation componentwise. The set of all multisets over E is denoted $\mathbb{N}^E$. Given $e \in E$, we denote by $\mathbf{e} \in \mathbb{N}^E$ the multiset consisting of one occurrence of element e, that is, the multiset satisfying $\mathbf{e}(e) = 1$ and $\mathbf{e}(e') = 0$ for every $e' \neq e$. We write $\{x, y, y, z\}$ to denote the multiset $\mathbf{x} + \mathbf{y} + \mathbf{y} + \mathbf{z}$. Every set S on E is also a multiset which maps E into $\{0, 1\}$. The *size* of a multiset $M \in \mathbb{N}^E$, denoted $|M|$, is defined as $\sum_{e \in E} M(e)$.

A set of multisets $\mathcal{M} \subseteq \mathbb{N}^E$ is said to be *Presburger* if it can be denoted by a formula in *Presburger arithmetic*, i.e., in the first-order theory of addition $FO(\mathbb{N}, +)$. A *population* P on E is a multiset on E with two or more elements: $P \in \mathbb{N}^E$ and $|P| \geq 2$.[1] The set of all

[1] Since, as we will see later, the atomic semantic step of population protocols is a pairwise interaction,

populations on E is denoted by $\mathrm{Pop}(E)$.

2.1 Labeled Population Protocols

A (labeled) *protocol scheme* $\mathcal{A} = (Q, \Sigma, \Delta)$ consists of a finite non-empty set Q of states, a finite set Σ of action labels, and a set $\Delta \subseteq Q^2 \times \Sigma \times Q^2$. If $(q_1, q_2, a, q_1', q_2') \in \Delta$, we write $(q_1, q_2) \overset{a}{\mapsto} (q_1', q_2')$ and call it an a-labeled *transition*. The populations of $\mathrm{Pop}(Q)$ are called *configurations*. Intuitively, a configuration C describes a collection of identical finite-state *agents* with Q as set of states, containing $C(q)$ agents in state q for every $q \in Q$. Pairs of agents interact using labeled transitions from Δ. Formally, given two configurations C and C' and a transition $(q_1, q_2) \overset{a}{\mapsto} (q_1', q_2')$, we write $C \xrightarrow{a} C'$ and call it a *step* if

$$C \geq (\mathbf{q}_1 + \mathbf{q}_2) \text{ holds, and } C' = C - (\mathbf{q}_1 + \mathbf{q}_2) + (\mathbf{q}_1' + \mathbf{q}_2') \ .$$

We write $C \xrightarrow{w} C'$ for a sequence $w = a_1 \dots a_k \in \Sigma^*$ of labels if there exists a sequence $C_0, \dots, C_k$ of configurations satisfying $C = C_0 \xrightarrow{a_1} C_1 \cdots \xrightarrow{a_k} C_k = C'$. We call this sequence a *finite execution*. The notation $C \xrightarrow{w}$ for an infinite sequence w and the notion of an infinite execution are defined analogously. The ω-language of $\mathcal{A}$ from configuration C, denoted $L(\mathcal{A}, C)$, is the set $\{w \in \Sigma^\omega \mid C \xrightarrow{w}\} \subseteq \Sigma^\omega$.

In what follows we assume every protocol scheme has its set of states Q and transitions Δ satisfying the following condition: for every q_1, q_2 in Q, there exists q_1', q_2' and label a such that $(q_1, q_2, a, q_1', q_2') \in \Delta$. It follows that every configuration enables a transition.

The *configuration graph* of a protocol scheme $\mathcal{A}$ is the infinite labeled, directed graph $(\mathrm{Pop}(Q), \Sigma, \rightarrow)$ having the populations over Q as nodes and labeled edges (C, a, C') whenever $C \xrightarrow{a} C'$ holds. Consider the partition $\{\mathrm{Pop}(Q)_i\}_{i \geq 2}$ of $\mathrm{Pop}(Q)$, where $\mathrm{Pop}(Q)_i = \{C \in \mathrm{Pop}(Q) \mid |C| = i\}$. (Note that i starts at 2 because every population contains at least two agents.) Since interactions do not create or destroy agents, the set $\{\rightarrow_i\}_{i \geq 2}$, where $\rightarrow_i = \rightarrow \cap \mathrm{Pop}(Q)_i \times \Sigma \times \mathrm{Pop}(Q)_i$, is also a partition of $\rightarrow$. Therefore $(\mathrm{Pop}(Q), \rightarrow)$ consists of the infinitely many disjoint and finite subgraphs $\{(\mathrm{Pop}(Q)_i, \Sigma, \rightarrow_i)\}_{i \geq 2}$.

A *strongly connected component* (SCC) of the configuration graph is a maximal set of mutually reachable configurations. An SCC is a *bottom SCC* if it is closed under reachability, i.e., if C belongs to the SCC and C' is reachable from C, then C' also belongs to the SCC. A configuration is a *bottom configuration* if it belongs to a bottom SCC of the graph.

We define a *population protocol* as a protocol scheme equipped with a possibly infinite Presburger set $\mathcal{I}$ of *initial configurations* and denote it as a pair $(\mathcal{A}, \mathcal{I})$. The original paper [3] introducing population protocols considered, instead of Presburger sets of initial configurations, a restricted class called simple sets defined by means of *input variables*—a subset of the states of the protocol scheme. Given a set $S \subseteq Q$ of input variables, the set $\mathcal{I}$ of initial configurations thereof is given by $\{C \in \mathrm{Pop}(Q) \mid \forall q \in Q \colon q \in S \vee C(q) = 0\}$. When $\mathcal{I}$ is definable using *input variables* then $\mathcal{I}$ is said to be *simple*.[2]

The complexity of parameterized verification can differ based on the presence or absence of a leader, a specific agent starting in a given state. Simple sets of initial configurations are essential to define *leaderless protocols*: protocols with no distinguished leader agent. (If we want to have a distinguished leader we can design a protocol whose set of states is the disjoint union of two sets $Q_l \cup Q_a$ of leader and agent states, and having a distinguished initial state

we require at least two agents in every population.

[2] To sum up, unless we state $\mathcal{I}$ is simple then it is assumed to be described by a Presburger formula.

$q_0 \in Q_l$ for the leader. Sets $\mathcal{I}$ containing only configurations C satisfying $C(q_0) = 1$ ensure that there is a unique leader.)

We give a population protocol a semantics as an infinite family of finite-state Markov Chains, one for each initial configuration. We assume that, given a configuration C, a probabilistic scheduler picks a pair of agents of C uniformly at random, and then picks one of the transitions they can execute according to some fixed probability distribution satisfying two properties: the probability of a transition depends only on the current states of the agents, and every transition has nonzero probability. This associates with each step $C \xrightarrow{a} C'$ a probability. Since $C \xrightarrow{a} C'$ implies $|C| = |C'|$, the number of configurations reachable from any configuration C is finite. Thus, for every C, the Markov Chain rooted at C has finitely many states.

▶ **Example 1.** Consider a protocol with two states q_1, q_2 and a configuration $C = (C(q_1), C(q_2)) = (1, 4)$. The scheduler picks two agents at state q_1 with probability 0, two agents at states q_1 and q_2 with probability $^2/_5$, and two agents at state q_2 with probability $^3/_5$. If the protocol has three transitions $(q_1, q_2) \overset{a}{\mapsto} (q_2, q_2)$, $(q_1, q_2) \overset{a}{\mapsto} (q_1, q_1)$, $(q_1, q_2) \overset{b}{\mapsto} (q_2, q_2)$, each with probability $^1/_3$, then the steps $(1, 4) \xrightarrow{a} (0, 5)$, $(1, 4) \xrightarrow{a} (2, 3)$ and $(1, 4) \xrightarrow{b} (0, 5)$ have probability $^2/_{15}$ each. The probability of doing an a from $(1, 4)$ is $^4/_{15}$, and the probability of moving from $(1, 4)$ to $(0, 5)$ by means of some action is also $^4/_{15}$.

Once the set of initial configurations $\mathcal{I}$ is fixed, we can talk about the probability of a measurable set of infinite paths of a Markov Chain rooted at some $C \in \mathcal{I}$. We write $\Pr[\mathcal{A}, C \models \mathcal{E}]$ to denote the probability that the stochastic process assigns to an event $\mathcal{E}$ for some $C \in \mathcal{I}$. Later, events will correspond to formulas of the linear temporal logic.

▶ **Example 2. Computation by consensus.** Angluin et al. [3] study protocols as a computation model. In their model, each state has an *output*, either 0 or 1. A protocol is well-specified if for every initial configuration, all agents eventually output the same value $b = 0, 1$ and stay committed to this value forever. Being well-specified further requires that the commitment of a population of agents to a value exclusively depends on their initial distribution. A well-specified protocol computes a predicate over its input variables: for each initial configuration, the predicate has value 0 (resp. 1) iff agents commit to 0 (resp. output 1) with probability 1.

▶ **Example 3. Birth-death processes.** A Moran process [23] is a stochastic process for evolution in a population with $N - 1$ residents and a mutant. In each step, one agent is randomly chosen for reproduction and one agent is randomly chosen for death. In the next step, the agent who dies is replaced with a copy of the agent that reproduces, therefore keeping the size of the population constant. In this setting, an interesting question is fixation: whether a single mutant can eventually replace all residents.

Probability vs. fairness

Instead of the probabilistic semantics, the semantics of population protocols is usually defined using fairness [3, 6]. An execution is *fair* if it is infinite and for every configuration C appearing in it infinitely often, and for every possible labelled step $C \xrightarrow{a} C'$, the step also appears infinitely often in the execution. We show later in Proposition 7 that the fair semantics and the probability semantics are indistinguishable for the questions we are studying: For every initial configuration C and every property φ expressible in LTL, some fair execution starting at C satisfies φ iff the set of executions of the protocol (fair or not) starting at C and satisfying φ has *positive* probability.

2.2 Probabilistic Linear Temporal Logic

Let Σ be a finite set of actions labels. The formulas of linear temporal logic (LTL) are defined by the grammar

$$\varphi ::= a \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \mathbf{X}\varphi \mid \varphi \mathbf{U} \varphi \mid \varphi \mathbf{W} \varphi \qquad \text{where } a \in \Sigma.$$

The semantics of LTL formulas are given in the usual way over traces [9]: an LTL formula φ defines an ω-language $L(\varphi) \subseteq \Sigma^\omega$. We define the following derived symbols: $\neg a \equiv \bigvee_{\sigma\in\Sigma\setminus\{a\}} \sigma$, $\mathit{true} \equiv \bigvee_{\sigma\in\Sigma} \sigma$, $\mathbf{F}\varphi \equiv \mathit{true}\mathbf{U}\varphi$ and $\mathbf{G}\varphi \equiv \varphi\mathbf{W}\mathit{false}$. The derived symbols can be eliminated with at most a polynomial cost in the size of the formula.

Qualitative and quantitative model checking. Let us now introduce the probabilistic interpretation for LTL. Given a configuration C, we say that $(\mathcal{A}, C)$ *satisfies the LTL formula* φ *with probability* p if $\Pr[\mathcal{A}, C \models \varphi] = p$. The *(qualitative) model checking problem* consists of, given a population protocol $(\mathcal{A}, \mathcal{I})$, and an LTL formula φ, deciding if $\Pr[\mathcal{A}, C \models \varphi] = 1$ for all $C \in \mathcal{I}$. We often work with the complement problem (deciding if $\Pr[\mathcal{A}, C \models \neg\varphi] > 0$ for some $C \in \mathcal{I}$). Abusing language, we also call it the *model checking problem*; the context should determine which problem we refer to. The *quantitative model checking problem* has an additional input p between 0 and 1 and asks whether $\Pr[\mathcal{A}, C \models \varphi] \geq p$ for all $C \in \mathcal{I}$.

▶ **Example 4. Cont'd from Ex.2** Let Σ be the set of all actions and let Σ_{ij}, for $i, j \in \{0, 1\}$, be the set of all transitions in which the output of the first process is i and the second process is j. Then, if the property $\Pr[\mathcal{A}, C \models \mathbf{F}(\mathbf{G}\Sigma_{00} \vee \mathbf{G}\Sigma_{11})] = 1$ holds for all $C \in \mathcal{I}$, we have that from any initial configuration, an execution of the protocol stabilizes to an output (0 or 1) with probability 1. Being well-specified actually requires more: all the executions starting at a configuration must converge *to the same value* with probability 1. This property can be expressed as a LTL formula but on a *modified* protocol instead of the original one. Intuitively, the modified protocol is the result of running two copies of the protocol side-by-side as we explained in a previous work [18].

Delporte-Gallet et al. [14] study which predicates can be computed by privacy-preserving protocols that do not reveal information on the initial configuration to a curious adversary. They identify a sufficient condition for a protocol to be privacy preserving, expressed as the conjunction of two properties (plus two other minor conditions). The first one is the existence from each initial configuration of an execution leading to a given set of configurations $\mathcal{G}$, and containing no interaction involving a distinguished agent. For the cases studied in the paper $\mathcal{G}$ can be expressed by an LTL-formula φ, and the property can be reduced to the model-checking problem for the formula $(\bigvee_{a\in A} a)\mathbf{U}\varphi$ for a suitable $A \subseteq \Sigma$. The second one, called $\mathcal{G}$*-imitability*, requires that for every action a of the distinguished agent and for every configuration of $\mathcal{G}$ there is a finite execution leading to a configuration of $\mathcal{G}$ and containing exactly one occurrence of the action. This can be verified by taking $\mathcal{G}$ as set of initial configurations and checking the formula $\bigwedge_{a\in\Sigma} (\neg a\,\mathbf{U}(a \wedge \mathbf{X}(\neg a\,\mathbf{U}\varphi)))$.

▶ **Example 5.** For the Moran process, the property that a mutant takes over the population is $\mathbf{FG}\Sigma_0$, where Σ_0 is the set of actions where a mutant reproduces.

2.3 Deterministic Rabin Automata

A *deterministic Rabin automaton* (DRA) $\mathcal{R} = (Q, \Sigma, \delta, q_0, \mathcal{F})$ consists of a finite set Q of states including an initial state q_0, an alphabet Σ, a transition function $\delta : Q \times \Sigma \to Q$, and an acceptance condition $\mathcal{F} \subseteq 2^Q \times 2^Q$.

An input for $\mathcal{R}$ is an infinite string in Σ^ω. The run of $\mathcal{R}$ on $w = w_0w_1 \ldots \in \Sigma^\omega$ is an infinite sequence $\rho = r_0r_1 \ldots$ of states in Q such that $r_0 = q_0$, the initial state, and for each $i \geq 0$, we have $r_{i+1} = \delta(r_i, w_i)$. Since $\mathcal{R}$ is deterministic, we can extend the transition function δ to finite words as follows: $\delta^*(q, \varepsilon) = q$ for every $q \in Q$ and $\delta^*(q, a\, w) = \delta^*(\delta(q, a), w)$. Let $\mathrm{Inf}(\rho)$ be the set of states that appear infinitely often in ρ.

Let $\mathcal{F} = \{\langle F_1, G_1\rangle, \ldots, \langle F_k, G_k\rangle\}$ be the set of *Rabin pairs.* A run ρ is accepting if there exists an $i \in \{1, \ldots, k\}$ such that $\mathrm{Inf}(\rho) \cap F_i = \emptyset$ and $\mathrm{Inf}(\rho) \cap G_i \neq \emptyset$; that is, no state from F_i is seen infinitely often and some state from G_i is seen infinitely often. A word w is accepted by $\mathcal{R}$ if the unique run of $\mathcal{R}$ on w is accepting. The language of $\mathcal{R}$, written $L(\mathcal{R})$, is the set of all words accepted by $\mathcal{R}$.

For each LTL formula φ, it is well-known that there is a DRA $\mathcal{R}_\varphi$ of size at most doubly exponential in the size of φ such that $L(\varphi) = L(\mathcal{R}_\varphi)$.

2.4 Labeled Petri Nets

A *labeled Petri net* $N = (P, T, F, \Sigma, \lambda)$ consists of a finite set P of *places*, a finite set T of *transitions*, a *flow function* $F \colon (P \times T) \cup (T \times P) \to \mathbb{N}$, an alphabet Σ of *actions* and a *labeling function* $\lambda \colon T \to \Sigma$. Abusing language, we also use λ to denote the homomorphic extension of the labeling function to $T^* \cup T^\omega \to \Sigma^* \cup \Sigma^\omega$. Given a transition $t \in T$, the multiset ${}^\bullet t$ of *input places* of t is defined by ${}^\bullet t(p) = F(p, t)$, and the multiset $t^\bullet$ of *output places* by $t^\bullet(p) = F(t, p)$. A Petri net is a labeled Petri net without a labeling function and alphabet of actions.

A *marking* M of a net N is a multiset of places. Given a place p, we say that M puts $M(p)$ *tokens* in p. A transition $t \in T$ is *enabled at* a marking M, written $M\,[t\rangle$, if ${}^\bullet t \leq M$. A transition t enabled at M can *fire*, yielding the marking $M' = M - {}^\bullet t + t^\bullet$. We write this fact as $M\,[t\rangle\,M'$. We extend enabledness and firing to sequences of transitions as follows. Let $\sigma = t_1 \ldots t_k$ be a finite sequence of transitions $t_j \in T$. We write $M\,[\sigma\rangle\,M'$ and call it a *firing sequence* if there exists a sequence $M_0, \ldots, M_k$ of markings such that $M = M_0\,[t_1\rangle\,M_1 \cdots [t_k\rangle\,M_k = M'$. In that case, we say that M' is *reachable from* M and denote by $\mathit{Reach}(N, M)$ the set of markings reachable from M. Given an infinite sequence $\sigma = t_1t_2 \ldots$, we write $M\,[\sigma\rangle$ if, and only if, there exists an infinite sequence $M_0, M_1, \ldots$ of markings such that $M = M_0$ and $M_i\,[t_{i+1}\rangle\,M_{i+1}$ for every $i \geq 0$. Finally, given $w \in \Sigma^* \cup \Sigma^\omega$, we write $M\,[w\rangle$ and $M\,[w\rangle\,M'$ if $M\,[\sigma\rangle$ and $M\,[\sigma\rangle\,M'$ for some sequence σ of transitions such that $\lambda(\sigma) = w$.

3 Model-checking LTL

In Section 3.1 we first show that the qualitative model checking problem is decidable. We then prove that it is as hard as the reachability problem for Petri nets, even for simple sets of initial configurations (that is, for leaderless protocols) and for the formula $\mathbf{FG}a$. Finally, in Section 3.2, we show the quantitative version is undecidable.

3.1 The Model Checking Problem Is Decidable but Reachability Hard

Our solution to the model checking problem is based on a product construction that, given a protocol scheme $\mathcal{A} = (Q, \Sigma, \Delta)$ and a DRA $\mathcal{R} = (Q', \Sigma, \delta, q_0', \mathcal{F})$ such that $Q \cap Q' = \emptyset$, produces a labeled Petri net $N(\mathcal{A}, \mathcal{R}) = (P, T, F, \Sigma, \lambda)$, defined as follows:

- $P = Q \cup Q'$.
- T contains a transition $t_{\delta_\mathcal{A}, q}$ for each $\delta_\mathcal{A} = (q_1, q_2) \overset{a}{\mapsto} (q_3, q_4) \in \Delta$ and $q \in Q'$.

- For each $t_{\delta_{\mathcal{A}},q} \in T$ with $\delta_{\mathcal{A}} = (q_1, q_2) \overset{a}{\mapsto} (q_3, q_4) \in \Delta$: ${}^\bullet(t_{\delta_{\mathcal{A}},q}) = \{q_1, q_2, q\}$ and $(t_{\delta_{\mathcal{A}},q})^\bullet = \{q_3, q_4, \delta(q, a)\}$. Further, $\lambda(t_{\delta_{\mathcal{A}},q}) = a$.

It follows immediately from the definition of $N(\mathcal{A}, \mathcal{R})$ that $(C+\mathbf{q'})\,[w\rangle\, M$ for some marking M and word $w \in \Sigma^*$ iff $M = C' + \mathbf{q''}$, $C \xrightarrow{w} C'$, and $\delta^*(q', w) = q''$, that is, $N(\mathcal{A}, \mathcal{R})$ captures the action-based synchronized product of $\mathcal{A}$ and $\mathcal{R}$.

A marking M of $N(\mathcal{A}, \mathcal{R})$ is *proper* if $M = C+\mathbf{q}$ where C is a configuration of $\mathcal{A}$ and q is a state of $\mathcal{R}$. In particular, every marking reachable from a proper marking is proper. The *proper reachability graph* of $N(\mathcal{A}, \mathcal{R})$ contains the proper markings of $N(\mathcal{A}, \mathcal{R})$ as nodes and the steps $(C+\mathbf{q'})\,[a\rangle\,(C'+\mathbf{q''})$ for $a \in \Sigma$ as edges. We say that an SCC $\mathcal{S}$ of the reachability graph of $N(\mathcal{A}, \mathcal{R})$ is *accepting* if there is a *Rabin pair* $\langle F, G\rangle$ of $\mathcal{R}$ such that $q' \in F$ for no marking $(C+\mathbf{q'}) \in \mathcal{S}$, and $q' \in G$ for some marking $(C+\mathbf{q'}) \in \mathcal{S}$.

Next, Proposition 6 reduces the qualitative model checking problem to a topological problem about the (typically infinitely many) SCCs of $N(\mathcal{A}, \mathcal{R})$.

▶ **Proposition 6.** *Given a configuration C of $\mathcal{A}$ and a Rabin automaton $\mathcal{R}_\varphi$ for a LTL formula φ: $\Pr[\mathcal{A}, C \models \varphi] > 0$ iff some bottom SCC of the proper reachability graph of $N(\mathcal{A}, \mathcal{R}_\varphi)$ is accepting and reachable from $(C+\mathbf{q_0'})$.*

Proof. For simplicity, we conduct the proof for the special case in which every transition of $\mathcal{A}$ is labeled with a different action. The extension to the general case is straightforward.

Observe that, since the interactions on $\mathcal{A}$ do not change the size of a configuration, for every two markings $(C_1+\mathbf{q_1}), (C_2+\mathbf{q_2})$ of the proper reachability graph of $N(\mathcal{A}, \mathcal{R}_\varphi)$ we have $|C_1| = |C_2|$. Since $\mathcal{R}_\varphi$ has finitely many states, the number of markings reachable from $(C_1+\mathbf{q_1})$ is at most $K(C_1) := (n^{|C_1|+1}) \cdot m$, where n and m are the number of states of $\mathcal{A}$ and $\mathcal{R}_\varphi$, respectively.

We introduce the following notation. Let $\tau = C \xrightarrow{a_1 \cdots a_n} C_n$ be a finite execution of $\mathcal{A}$. As usual, the *cylinder* $\mathit{Cyl}(\tau)$ denotes all the infinite executions of $\mathcal{A}$ starting with τ, and its probability $\Pr(\mathit{Cyl}(\tau))$ is the product of the probabilities of the steps $C \xrightarrow{a_1} C_1, \ldots, C_{n-1} \xrightarrow{a_n} C_n$. Since $\mathcal{R}_\varphi$ is complete and deterministic, the unique state $q_n' = \delta^*(q_0', a_1 \ldots a_n)$ is such that $(C+\mathbf{q_0'})\,[a_1 \cdots a_n\rangle\,(C_n+\mathbf{q_n'})$ is a firing sequence of $N(\mathcal{A}, \mathcal{R}_\varphi)$. We call the run of $\mathcal{R}_\varphi$ the *matching run* of τ, and denote it by $\overline{\tau}$. Further, we denote the firing sequence by $(\tau, \overline{\tau})$. We extend the notation to infinite executions, runs, and firing sequences.

We first prove a preliminary claim. An infinite execution σ of $\mathcal{A}$ starting at a configuration C satisfies the following property with probability 1: the infinite firing sequence $(\sigma, \overline{\sigma})$ of $N(\mathcal{A}, \mathcal{R}_\varphi)$ from the marking $(C+\mathbf{q_0'})$ eventually reaches a bottom SCC of the proper reachability graph of $N(\mathcal{A}, \mathcal{R}_\varphi)$, and visits all its markings infinitely often.

For the first part, observe that for every finite prefix σ_1 of σ there is, by definition, a finite execution σ_2 of length at most $K(C)$ such that the marking reached by $(\sigma_1\sigma_2, \overline{\sigma_1\sigma_2})$ belongs to a bottom SCC of the proper reachability graph of $N(\mathcal{A}, \mathcal{R}_\varphi)$. Therefore, there is a bound $p(C) > 0$, depending only on σ_2 and in particular on C, such that the probability that for $\sigma \in \mathit{Cyl}(\sigma_1)$ the firing sequence $(\sigma, \overline{\sigma})$ reaches a bottom marking is at least $p(C)$. By elementary probability theory, the probability of the infinite execution σ such that $(\sigma, \overline{\sigma})$ eventually visits a bottom SCC is equal to 1. For the second part, assume that σ has a prefix τ such that $(\tau, \overline{\tau})$ leads to a bottom SCC, say $\mathcal{S}$, and let $(C'+\mathbf{q'})$ be an arbitrary marking of $\mathcal{S}$. Then, for every finite execution $\tau\tau_1$ of $\mathcal{A}$, there is τ_2 of length at most $K(C)$ such that the firing sequence $(\tau\tau_1\tau_2, \overline{\tau\tau_1\tau_2})$ leads to $(C'+\mathbf{q'})$. So the probability that an infinite execution of $\mathit{Cyl}(\tau\tau_1)$ eventually reaches $(C'+\mathbf{q'})$ is equal to 1. This concludes the proof of the claim.

We now proceed to prove the proposition. Assume that some bottom SCC $\mathcal{S}$ of the proper reachability graph of $N(\mathcal{A}, \mathcal{R}_\varphi)$ is accepting for some Rabin pair $\langle F, G\rangle$, and is also reachable from $(C+\mathbf{q_0'})$. Let $(\sigma, \overline{\sigma})$ be a firing sequence leading to some marking of $\mathcal{S}$, and let $(C_G+\mathbf{q_G})$ be a marking of $\mathcal{S}$. Because $\mathcal{S}$ is accepting we have $q_G \in G$. For the same reason, no marking of $\mathcal{S}$ is of the form $(C_F+\mathbf{q_F})$ with $q_F \in F$. By the claim, an infinite execution $\tau \in \mathit{Cyl}(\sigma)$ satisfies with probability 1 that the infinite firing sequence $(\tau, \overline{\tau})$ visits $(C_G+\mathbf{q_G})$ infinitely often and visits no marking $(C_F+\mathbf{q_F})$ with $q_F \in F$. So $\Pr[\mathcal{A}, C \models \varphi] \geq \Pr(\mathit{Cyl}(\sigma)) > 0$.

Assume now that no bottom SCC of the proper reachability graph of $N(\mathcal{A}, \mathcal{R}_\varphi)$ is accepting. Then, for every Rabin pair $\langle F, G\rangle$ and every bottom SCC $\mathcal{S}$, either $\mathcal{S}$ contains a marking $(C'+\mathbf{q}')$ such that $q' \in F$, or it contains no marking $(C'+\mathbf{q}')$ such that $q' \in G$. By the claim, for every infinite execution τ of $\mathcal{A}$ starting at C, the infinite firing sequence $(\tau, \overline{\tau})$ gets eventually trapped in a bottom SCC, say $\mathcal{S}$, with probability 1. If $\mathcal{S}$ contains a marking $(C'+\mathbf{q}')$ such that $q' \in F$, then, again by the claim, $(\tau, \overline{\tau})$ visits $(C'+\mathbf{q}')$ infinitely often with probability 1, and so it is non-accepting with probability 1. If $\mathcal{S}$ contains no marking $(C'+\mathbf{q}')$ such that $q' \in G$, then with probability 1 it visits configurations $(C'+\mathbf{q}')$ such that $q' \in G$ only finitely often, and so it is not accepting with probability 1. So C satisfies φ with probability 0. ◀

Using this result we now proceed to prove the indistinguishability of the fair and the probability semantics we announced in Section 2.1:

▶ **Proposition 7.** *Let $(\mathcal{A}, \mathcal{I})$ be a population protocol, $C \in \mathcal{I}$, and let φ be an LTL formula. We have:* $\Pr[\mathcal{A}, C \models \varphi] > 0$ *iff some fair execution of $\mathcal{A}$ starting at C satisfies φ.*

Proof. It is easy to show (see Esparza et al. [18, 19]) that every fair execution of $\mathcal{A}$ starting at a configuration C gets eventually trapped in a bottom SCC of the configuration graph of $\mathcal{A}$, and crosses all its edges infinitely often. Using the same arguments as in Proposition 6, we show that a fair execution starting at C satisfies φ iff its unique counterpart firing sequence reaches an accepting bottom SCC of the proper reachability graph of $N(\mathcal{A}, \mathcal{R}_\varphi)$. So, by Proposition 6, some fair executions starting at C satisfy φ iff $\Pr[\mathcal{A}, C \models \varphi] > 0$. ◀

We need the following fact about Petri nets from [22] to prove decidability.

▶ **Lemma 8.** *[22] Let N be a Petri net. The set of pairs of markings (M, M') such that M and M' are mutually reachable (i.e., M' is reachable from M and M is reachable from M') is Presburger, and a Presburger formula* $\mathtt{MR(M,M')}$ *denoting it can be effectively constructed.*

▶ **Theorem 9.** *Let $(\mathcal{A}, \mathcal{I})$ be a population protocol, and let φ be an LTL formula. The problems whether there exists a configuration $C \in \mathcal{I}$ satisfying* $\Pr[\mathcal{A}, C \models \varphi] > 0$ *and* $\Pr[\mathcal{A}, C \models \varphi] < 1$ *can be reduced to the reachability problem for Petri nets.*

Proof. Let $\mathcal{R}_\varphi$ be a DRA for φ. Assume for simplicity that $\mathcal{R}_\varphi$ has only one Rabin pair $\langle F, G\rangle$ (the generalization to multiple pairs is straightforward). We first show that the set of markings of $N(\mathcal{A}, \mathcal{R}_\varphi)$ that belong to accepting bottom SCCs of the proper reachability graph is Presburger, and that a formula `BA(M)` denoting it can be constructed .

Let `P(M)` be the Presburger formula characterizing the proper markings of $N(\mathcal{A}, \mathcal{R}_\varphi)$. Further, let $\mathtt{FO(M,M')}$ be a Presburger formula that holds if M' can be reached from M by firing one transition. Then we can take for `BA(M)` the formula

$$\begin{array}{lll} \mathtt{P(M)} & \wedge & \forall \mathtt{M'}, \mathtt{M''} \colon (\mathtt{MR(M,M')} \wedge \mathtt{FO(M',M'')}) \Rightarrow \mathtt{MR(M,M'')} \\ & \wedge & \forall \mathtt{M'} \colon \mathtt{MR(M,M')} \Rightarrow \left(\bigwedge_{q \in F} \mathtt{M'}(q) = 0\right) \\ & \wedge & \exists \mathtt{M'} \colon \mathtt{MR(M,M')} \wedge \bigvee_{q \in G} \mathtt{M'} \geq \mathbf{q} \end{array}$$

where $\mathtt{MR(M,M')}$ is obtained following Lemma 8. Similarly, we obtain a formula $\mathtt{BR(M)}$ for the markings that belong to a non-accepting bottom SCCs of $N(\mathcal{A}, \mathcal{R}_\varphi)$. By Proposition 6, the problem whether some configuration $C \in \mathcal{I}$ satisfies $\Pr[\mathcal{A}, C \models \varphi] > 0$ reduces to deciding whether some marking M' satisfying $\mathtt{BA(M')}$ is reachable from some proper marking M in the Presburger set given by $\{(C+\mathbf{q}_0') \mid C \in \mathcal{I}\}$. Similarly, $\Pr[\mathcal{A}, C \models \varphi] < 1$ reduces to reachability of $\mathtt{BR(M')}$. ◀

We show that the problems of Theorem 9 are *reachability hard*, i.e., at least as hard as the reachability problem of Petri nets. In previous works [18, 19], we proved reachability hardness for the well-specification problem (whether a given protocol computes a predicate). However, the protocols given by the reduction from the reachability problem always had a leader (formally, they always had a state such that at all initial configurations that state was inhabited by *exactly* one agent, and this was crucial for the proof). When population protocols are used to compute by consensus (see Example 2), leaderless protocols turn out to have the same computational power as protocols with leader; the only difference is that the latter can be faster [3, 5]. Therefore, the question arises whether verification problems have lower complexity for the special case of leaderless protocols. A positive answer would mean that one can trade-off efficiency for ease of verification, without losing computational power. We now show that, unfortunately, this is not the case: the qualitative model-checking problem for the basic liveness property $\mathbf{FG}a$ is reachability hard for leaderless protocols. The same technique also proves hardness of the well-specification problem.

▶ **Theorem 10.** *The reachability problem for Petri nets can be reduced in linear time to the following problem: given a population protocol* $(\mathcal{A}, \mathcal{I})$ *where* $\mathcal{I}$ *is simple, decide if some configuration* $C \in \mathcal{I}$ *satisfies* $\Pr[\mathcal{A}, C \models \mathbf{FG}a] > 0$.

Sketch of proof. The proof constructs a sequence of reductions from the Petri net reachability problem. Each step in the sequence transform a problem on Petri net into an equivalent problem closer to the model of population protocols. We first use a result of Hack [21] that reduces the reachability problem to the problem of deciding for a given Petri net N and a marking M_0 with a distinguished place $\hat{p}$ if some marking $M \in \mathit{Reach}(N, M_0)$ satisfies $M(\hat{p}) = 0$. Then we introduce a "normal form" for nets: a net $N = (P, T, F)$ is in normal form if $F(x, y) \in \{0, 1\}$ for every $x, y \in (P \times T) \cup (T \times P)$, and every transition t satisfies $1 \le |{}^\bullet t| \le 2$ and $1 \le |t^\bullet| \le 2$. Transitions of Petri nets in normal form can be simulated by transitions of population protocols, which only involve two agents. We prove that the reachability problem reduces to: given a net N in normal form, a place p_0 and a set of places $\hat{P}$, decide if some marking $M \in \mathit{Reach}(N, \mathbf{p_0})$ satisfies $M(\hat{P}) = 0$. The next step applies a simple but key observation: for any two markings M, M' of a net, M' is reachable from M iff M is reachable from M' in the *reverse* net obtained by reversing the arcs. This allows to reduce the reachability problem to: given a net N in normal form, a place p_0 and a set of places $\hat{P}$, decide if $\mathbf{p_0} \in \mathit{Reach}(N, M)$ for some marking M satisfying $M(\hat{P}) = 0$. The crucial point is that this set of markings corresponds to a *simple* set of initial configurations of the protocol simulating the net. So, starting from this simple set, the protocol can reach a certain configuration iff $\mathbf{p_0}$ is reachable. It still remains to ensure that the protocol satisfies $\Pr[\mathcal{A}, C \models \mathbf{FG}a] > 0$ for some initial configuration iff the marking $\mathbf{p_0}$ is reachable. This is achieved by adding "probing" transitions to the protocol, labeled by an action different from b, ensuring that the protocol can always do a b as long as it has not reached $\mathbf{p_0}$. ◀

3.2 Quantitative Model Checking Is Undecidable

We prove that, contrary to the qualitative case in the previous section, the quantitative model checking problem is undecidable. Our proof uses the simulation of deterministic two counter machines with arbitrarily small error.

Recall that a counter machine is a triple $M = (L, \mathit{Co}, \mathit{In})$ where L is a finite set of program labels, Co is a finite set of counters, and In is a set of program instructions, one for each label. The program instruction for label ℓ is of one of the following types: $\ell\colon c := c + 1;$ `goto` ℓ' (increment), $\ell\colon c := c - 1;$ `goto` ℓ' (decrement), $\ell\colon$ `if` $c = 0$ `then goto` ℓ' `else goto` ℓ'' (zero-test), or $\ell\colon$ `halt` (termination). Only one label ℓ_h has an instruction of the last type, and there is also a distinguished initial label ℓ_0. The *termination problem for counter machines* consists of deciding if a given machine, starting at ℓ_0 with all counters initially set to 0, eventually halts, i.e., reaches ℓ_h.

▶ **Lemma 11.** *Given a counter machine M, we can construct in polynomial time a protocol scheme $\mathcal{A}$ with a distinguished state q_h, and a set $\mathcal{I}$ of initial configurations such that M halts iff some configuration $C \in \mathcal{I}$ satisfies the following property: starting at C, the protocol eventually reaches a configuration with one agent in q_h with probability at least $^1/_2$.*

Proof. It is convenient to define first the set of states of $\mathcal{A}$, then the set $\mathcal{I}$ of initial configurations, and then the transitions of $\mathcal{A}$.

- $\mathcal{A}$ has a state for each label and for each counter of M, three distinguished states *Store*, D (for *Dummy*), and *Stop*, and two auxiliary states ℓ_1, ℓ_2 for each zero-test label ℓ. We set $q_h := \ell_h$, i.e., choose the state q_h as the one corresponding to the halting label.
- $\mathcal{I}$ contains the configurations that put one agent in the initial label ℓ_0, one agent in D, arbitrarily many agents in *Store*, and no agent elsewhere.

Before defining the transitions of $\mathcal{A}$ we give some intuition. First, the transitions guarantee that every reachable configuration puts one agent in exactly one of the states corresponding to the programs labels L. This models that the next instruction executed by the machine is the one with label ℓ. We call this agent the *control agent.* The number of agents at a state c models the current value of the counter. The transitions also guarantee that the one agent at D never moves elsewhere (its role is only to enable some transitions).

Increasing and decreasing a counter c is modeled by transitions that transfer an agent from *Store* to c, and from c to *Store*, respectively. Therefore, if an initial configuration puts, say, K agents in *Store*, then from that configuration the protocol cannot always simulate the complete execution of the machine, only the prefix during which the sum of the values of all counters does not exceed K. $\mathcal{A}$ has the following transitions:

- For each instruction $\ell\colon c := c + 1;$ `goto` ℓ', a transition $(\ell, \mathit{Store}) \xmapsto{inc} (\ell', c)$.
- For each instruction $\ell\colon c := c - 1;$ `goto` ℓ', a transition $(\ell, c) \xmapsto{dec} (\ell', \mathit{Store})$.
- For each instruction $\ell\colon$ `if` $c = 0$ `then goto` ℓ' `else goto` ℓ'', the following transitions:
  - $(\ell, D) \xmapsto{go} (\ell_1, D)$, $(\ell, c) \xmapsto{nonzero} (\ell'', c)$
  - $(\ell_1, \mathit{Store}) \xmapsto{back} (\ell, \mathit{Store})$, $(\ell_1, D) \xmapsto{zero} (\ell_2, D)$
  - $(\ell_2, D) \xmapsto{zero'} (\ell', D)$ and $(\ell_2, c) \xmapsto{gameover} (\mathit{Stop}, c)$.

It is convenient to think of the *go* and *back* transitions as a loop moving an agent from ℓ to ℓ_1 and back, and of the transitions *zero* and *nonzero* as the two possible exits of this loop.

Before proving the lemma, we make an observation. Assume that the control agent of a configuration C is at the state of a zero-test label $\ell\colon$ `if` $c = 0$ `then goto` ℓ' `else goto` ℓ'', and assume further $C(\mathit{Store}) \geq 1$. We consider two cases. In the first case, C puts no agents in counter c. Then the *nonzero* exit of the loop is not enabled at C. Therefore, the scheduler will eventually move the control agent to ℓ_2 with probability 1 (after executing the "loop" *go back* a number of times). Further, since the *game over* action is also not enabled, the scheduler will eventually move the control agent to ℓ' with probability 1.

In the second case, C puts at least one agent in c. Then both exits are enabled at C, and the scheduler eventually chooses one of them with probabilities p_z and p_{nz}, respectively. The key point is that these probabilities depend on the number of agents in *Store* and in c. Indeed, for every value of c, increasing the number N of agents in *Store* also increases p_{nz}, since it makes it more likely that the scheduler picks agents at *Store*. In particular, we have $p_{nz} \to 1$ when $N \to \infty$. So the scheduler eventually moves the control agent to ℓ'' with probability that tends to 1 when N tends to infinity.

We prove the left-to-right direction of the lemma. Assume that M halts. We show that there is a configuration $C \in \mathcal{I}$ from which the protocol eventually reaches a configuration with the control agent in q_h with probability at least $^1/_2$.

Let k be the length of the halting computation of M. Clearly, during the computation no counter ever has a value larger than k. So the probability of the protocol taking the wrong exit when the control agent visits a zero-test label is always bounded by a value $p(N)$ that tends to 0 as N tends to infinity. Since the computation of M visits zero-test labels at most k times, the probability that at all these visits the protocol chooses the right exit is at least $(1 - p(N))^k$, which tends to 1 as N tends to infinity. So, by making N sufficiently large, we obtain an initial configuration for which the protocol faithfully simulates the computation of M with probability at least $^1/_2$. Since M halts, that computation visits q_h, and we are done.

We now prove the converse direction, for which we need the *game over* actions, which have played no role so far. Assume that M does not halt. We prove that for every configuration $C \in \mathcal{I}$ the probability that, starting at C, the protocol eventually reaches a configuration with the control agent in q_h is at most $^1/_2$.

We say that the protocol "cheats" during the simulation of M if, after reaching a configuration with the control agent at a zero-test label ℓ and a strictly positive number of agents at c, the protocol moves the control agent to ℓ', and not to ℓ'', as indicated by the instruction.

Let C be an arbitrary initial configuration and let the protocol produce an execution. Since M does not halt, in order to move the control agent to q_h the protocol must cheat at least once. For this, the scheduler must move the control agent to ℓ' at a moment at which there is at least one agent in counter c. But then exactly one pair of agents can move the control agent to ℓ'—namely the agents at ℓ_2 and D—and at least one pair of agents can move it to *Stop* (the agent at ℓ_2 and one of the agents at c). Since after reaching *Stop* the protocol cannot reach q_h anymore, the probability that after moving to ℓ_2 the control agent eventually reaches q_h is at most $^1/_2$. So the probability of reaching q_h is bounded from above by $^1/_2$ times the probability that an execution cheats at least once. Hence, in particular, the probability is at most $^1/_2$, and we are done. ◀

▶ **Proposition 12.** *The quantitative model checking problem for population protocols is already undecidable for specifications of the form* $\mathbf{G}(\bigvee_{a\in A} a)$ *for some set* A *of action labels.*

Proof. We show that the problem is already undecidable for a formula of the form $\mathbf{G}(\bigvee_{a\in A} a)$ for some set A of action labels, and the probability bound $^1/_2$. We proceed by reduction from the non-termination problem for counter machines. Given a machine M, we construct a protocol $\mathcal{A}$ and a set of initial configurations $\mathcal{I}$ such that M halts iff $\Pr[\mathcal{A}, C \models \mathbf{G}(\bigvee_{a\in A} a)] \geq 1/2$ for every $C \in \mathcal{I}$. Almost all the work has been done in Lemma 11. Consider the protocol $\mathcal{A}$ and the set $\mathcal{I}$ defined there, and add a transition $(q_h, D) \xmapsto{halt} (q_h, D)$. Then, an execution of $\mathcal{A}$ satisfies $\mathbf{F}\, halt$ iff it eventually moves the control agent to state q_h. Applying the lemma, we obtain that M does not halt iff $\Pr[\mathcal{A}, C \models \mathbf{F}\, halt] < 1/2$ for every $C \in \mathcal{I}$. Taking A as the set of all actions of $\mathcal{A}$ but *halt*, we get: M does not halt iff $\Pr[\mathcal{A}, C \models \mathbf{G}(\bigvee_{a \in A} a)] \geq 1/2$ for every $C \in \mathcal{I}$. ◀

In fact, Lemma 11 also implies undecidability of the problem whether the probability of a property can be made arbitrarily close to 1 by increasing the number of agents. Indeed, a look at the proof of the lemma shows that the reduction produces a protocol satisfying the following property: the counter machine halts iff there a bound $\rho < 1$ (in the lemma, $\rho = 1/2 + \varepsilon$) such that $\Pr[\mathcal{A}, C \models \varphi] \leq \rho$ for every $C \in \mathcal{I}$.

4 Discussion and Further Undecidability Results

We have characterized the decidability frontier for LTL model checking of population protocols: qualitative model checking is decidable, and quantitative is undecidable. We have also shown that, though decidable, qualitative model checking is as hard as the reachability problem for Petri nets (for which no primitive-recursive algorithm is known) even for leaderless protocols and for the simple formula $\mathbf{FG}a$. Essentially the same proof shows that the well-specification problem is also as hard as the Petri net reachability problem even in the leaderless case, removing the assumption of a leader from our previous hardness proof [19].

In the rest of the section we briefly discuss other undecidability results showing that Theorem 9 is rather close to the "decidabiliy border."

LTL on Configurations. Note that we have defined LTL on actions. An alternate definition could take the set of configurations as atomic propositions. Configuration-based LTL model checking is known to be decidable for Well-Structured Transition Systems (WSTS)—a general class which includes population protocols and much more—provided the reasoning can be restricted to upward-closed sets. This is the key idea used by Baier et al. [8] who prove that a state-based fragment of μ-calculus is decidable for all WSTS. It is not known whether this result can be made more general when focusing on population protocols instead of the whole class of WSTS. Unfortunately, the model checking problem for population protocols is undecidable even for very simple classes of atomic propositions.

Given a state q of a protocol, let $q_{\geq 1}$ denote the atomic proposition that holds for a configuration C if $C(q) \geq 1$. We call $q_{\geq 1}$ a *flag*, since it flags that state q is inhabited.

▶ **Proposition 13.** *The qualitative model checking problem for population protocols and LTL specifications over flags is undecidable.*

Proof. Instead of using the construction of Lemma 11 for zero-tests, we enable the zero-test transition always, and use an LTL formula over configurations to catch "cheating", i.e., the population protocol taking a zero-test when it should not. Indeed, every zero-test instruction ℓ: `if` $c = 0$ `then goto` ℓ' `else goto` ℓ'' yields the formula:

$$\mathbf{G}(q_{\geq 1} \wedge c_{\geq 1} \Rightarrow q_{\geq 1} \mathbf{U} q''_{\geq 1}) \qquad (1)$$

where q, q'' are the states modelling the locations ℓ, ℓ' and c the state modelling the counter. The final formula is the conjunction of the formulas given at (1) (one conjunct for each zero-test) together with $\mathbf{F}(q_h)_{\geq 1}$. If the counter machine halts, then this formula holds with nonzero probability from an initial configuration with exactly one agent in the control location and sufficiently many agents in the *Store* location. If the counter machine rejects, then the probability of the formula is zero for every initial configuration. ◀

Broadcast Protocols. Adding broadcasts makes the qualitative model checking undecidable as well. Consider an extension of population protocols with *broadcast actions* [16] where, in addition to a set Δ of transitions involving two agents, also a set $\Delta^* \subseteq Q \times \Sigma \times Q \times 2^{Q \times Q}$ of "broadcast" transitions is allowed. Given a broadcast transition (q, a, q', δ) and a configuration C satisfying $C(q) > 0$, if the scheduler picks an agent in state q, then the agent changes its state to q' and simultaneously, *all* other agents update their states according to the function δ. The qualitative model checking problem for this model is undecidable, because the model can weakly simulate counter machines by slightly modifying the proof of [16, Theorem 5.1].

Acknowledgements. This research was sponsored in part by the ERC Synergy project ImPACT. Pierre Ganty has been supported by the Madrid Regional Government project S2013/ICE-2731, *N-Greens Software - Next-GeneRation Energy-EfficieNt Secure Software*, and the Spanish Ministry of Economy and Competitiveness project No. TIN2015-71819-P, *RISCO - RIgorous analysis of Sophisticated COncurrent and distributed systems.*

References

1 Parosh Aziz Abdulla, Radu Ciobanu, Richard Mayr, Arnaud Sangnier, and Jeremy Sproston. Qualitative analysis of VASS-induced MDPs. In *FoSSaCS '16*, LNCS, pages 319–334. Springer, 2016. `doi:10.1007/978-3-662-49630-5_19`.

2 Parosh Aziz Abdulla, Noomene Ben Henda, and Richard Mayr. Decisive Markov chains. *Logical Methods in Computer Science*, 3(4), 2007. `doi:10.2168/LMCS-3(4:7)2007`.

3 Dana Angluin, James Aspnes, Zoë Diamadi, Michael J. Fischer, and René Peralta. Computation in networks of passively mobile finite-state sensors. In *PODC '04*, pages 290–299. ACM, 2004. `doi:10.1145/1011767.1011810`.

4 Dana Angluin, James Aspnes, and David Eisenstat. Stably computable predicates are semilinear. In *PODC '06*, pages 292–299. ACM, 2006. `doi:10.1145/1146381.1146425`.

5 Dana Angluin, James Aspnes, and David Eisenstat. Fast computation by population protocols with a leader. *Distributed Computing*, 21(3):183–199, 2008. `doi:10.1007/s00446-008-0067-z`.

6 Dana Angluin, James Aspnes, David Eisenstat, and Eric Ruppert. The computational power of population protocols. *Distributed Computing*, 20(4):279–304, 2007. `doi:10.1007/s00446-007-0040-2`.

7 Tamarah Arons, Amir Pnueli, and Lenore D. Zuck. Parameterized verification by probabilistic abstraction. In *FOSSACS '03*, volume 2620 of *LNCS*, pages 87–102. Springer, 2003. `doi:10.1007/3-540-36576-1_6`.

8 Christel Baier, Nathalie Bertrand, and Philippe Schnoebelen. On computing fixpoints in well-structured regular model checking, with applications to lossy channel systems. In *LPAR '06*, volume 4246 of *LNCS*, pages 347–361. Springer, 2006. `doi:10.1007/11916277_24`.

9 Christel Baier and Joost-Pieter Katoen. *Principles of model checking.* MIT Press, 2008.

10 Nathalie Bertrand and Paulin Fournier. Parameterized verification of many identical probabilistic timed processes. In *FSTTCS '13*, volume 24 of *LIPIcs*, pages 501–513. Schloss Dagstuhl, 2013. `doi:10.4230/LIPIcs.FSTTCS.2013.501`.

11 Nathalie Bertrand, Paulin Fournier, and Arnaud Sangnier. Playing with probabilities in reconfigurable broadcast networks. In *FoSSaCS '14*, volume 8412 of *LNCS*, pages 134–148. Springer, 2014. doi:10.1007/978-3-642-54830-7_9.

12 Patricia Bouyer, Nicolas Markey, Mickael Randour, Arnaud Sangnier, and Daniel Stan. Reachability in networks of register protocols under stochastic schedulers. In *ICALP '16*, volume 55 of *LIPIcs*, pages 106:1–106:14. Schloss Dagstuhl, 2016. doi:10.4230/LIPIcs.ICALP.2016.106.

13 J. Clement, C. Delporte-Gallet, H. Fauconnier, and M. Sighireanu. Guidelines for the verification of population protocols. In *ICDCS '11*, pages 215–224, 2011. doi:10.1109/ICDCS.2011.36.

14 Carole Delporte-Gallet, Hugues Fauconnier, Rachid Guerraoui, and Eric Ruppert. Secretive birds: Privacy in population protocols. In *OPODIS '07*, volume 4878 of *LNCS*, pages 329–342. Springer, 2007. doi:10.1007/978-3-540-77096-1_24.

15 Zoë Diamadi and Michael J. Fischer. A simple game for the study of trust in distributed systems. *Wuhan University Journal of Natural Sciences*, 6(1):72–82, 2001. doi:10.1007/BF03160228.

16 Javier Esparza, Alain Finkel, and Richard Mayr. On the verification of broadcast protocols. In *LICS '99*, pages 352–359. IEEE Computer Society, 1999. doi:10.1109/LICS.1999.782630.

17 Javier Esparza, Andreas Gaiser, and Stefan Kiefer. Proving termination of probabilistic programs using patterns. In *CAV'12*, volume 7358 of *LNCS*, pages 123–138. Springer, 2012. doi:10.1007/978-3-642-31424-7_14.

18 Javier Esparza, Pierre Ganty, Jérôme Leroux, and Rupak Majumdar. Verification of population protocols. In *CONCUR '15*, volume 42 of *LIPIcs*, pages 470–482. Schloss Dagstuhl, 2015. doi:10.4230/LIPIcs.CONCUR.2015.470.

19 Javier Esparza, Pierre Ganty, Jérôme Leroux, and Rupak Majumdar. Verification of population protocols. *Acta Informatica*, pages 1–25, 2016. doi:10.1007/s00236-016-0272-3.

20 Steven M. German and A. Prasad Sistla. Reasoning about systems with many processes. *J. ACM*, 39(3):675–735, 1992. doi:10.1145/146637.146681.

21 Michel Henri Théodore Hack. Decidability questions for Petri nets. Technical Report 161, MIT, 1976.

22 Jérôme Leroux. Vector addition system reversible reachability problem. In *CONCUR '11*, volume 6901 of *LNCS*, pages 327–341. Springer, 2011. doi:10.1007/978-3-642-23217-6_22.

23 P. A. P. Moran. Random processes in genetics. *Mathematical Proceedings of the Cambridge Philosophical Society*, 54:60–71, 1 1958. doi:10.1017/S0305004100033193.

24 Saket Navlakha and Ziv Bar-Joseph. Distributed information processing in biological and computational systems. *Commun. ACM*, 58(1):94–102, December 2014. doi:10.1145/2678280.

25 David Soloveichik, Matthew Cook, Erik Winfree, and Jehoshua Bruck. Computation with finite stochastic chemical reaction networks. *Natural Computing*, 7(4):615–633, 2008. doi:10.1007/s11047-008-9067-y.

26 Lenore D. Zuck, Amir Pnueli, and Yonit Kesten. Automatic verification of probabilistic free choice. In *VMCAI '02*, volume 2294 of *LNCS*, pages 208–224. Springer, 2002. doi:10.1007/3-540-47813-2_15.