Acta Informatica manuscript No.
(will be inserted by the editor)

Verification of Population Protocols

Javier Esparza · Pierre Ganty · Jérôme Leroux · Rupak Majumdar

Received: date / Accepted: date

Abstract Population protocols (Angluin et al., *PODC*, 2004) are a formal model of sensor networks consisting of identical mobile devices. Two devices can interact and thereby change their states. Computations are infinite sequences of interactions satisfying a strong fairness constraint.

A population protocol is well specified if for every initial configuration C of devices, and every computation starting at C, all devices eventually agree on a consensus value depending only on C. If a protocol is well specified, then it is said to compute the predicate that assigns to each initial configuration its consensus value.

While the computational power of well-specified protocols has been extensively studied, the two basic verification problems remain open: Is a given protocol well specified? Does a given protocol compute a given predicate? We prove that both problems are decidable by reduction to the reachability problem of Petri nets. We also give a new proof of the fact that the predicates computed by well-defined protocols are those definable in Presburger arithmetic (Angluin et al., *PODC*, 2006).

Keywords population protocols · Petri nets · parametrized verification

CR Subject Classification C.2.2 · D.2.4 · F.3.1

1 Introduction

Population protocols [2] are a model of distributed computation by anonymous, identical finite-state agents. While they were initially introduced to model networks of passively mo-

Javier Esparza
TUM, Germany

Pierre Ganty
IMDEA Software Institute, Spain

Jérôme Leroux
LaBRI, CNRS & Univ. Bordeaux, France

Rupak Majumdar
MPI-SWS, Germany

bile sensors [2], they capture the essence of distributed computation in diverse areas such as trust propagation [9] and chemical reactions [19].

In each computation step of a population protocol, a fixed number of agents are chosen nondeterministically, and their states are updated according to a joint transition function. Since agents are anonymous and identical, the global state of a protocol is completely determined by the number of agents at each local state, called a configuration. A protocol computes a boolean value for a given initial configuration if in all fair executions starting at it, all agents eventually agree to this value—so, intuitively, population protocols compute by reaching consensus. An execution is fair if it is finite and cannot be extended, or it is infinite and satisfies the following condition: if the execution visits a configuration C infinitely often, then it also visits every configuration reachable from C infinitely often. Given a set of inputs (typically a set of vectors of natural numbers), and a mapping that assigns to each input an initial configuration, the predicate computed by a protocol is the function that assigns to each input the boolean value computed by the protocol from the corresponding initial configuration. If the protocol does not reach consensus for some input, then we say it is ill specified and does not compute any predicate.

In each computation step of a population protocol, a fixed number of agents are chosen nondeterministically, and their states are updated according to a joint transition function. Since agents are anonymous and identical, the global state of a protocol is completely determined by the number of agents at each local state, called a configuration. A protocol computes a boolean value for a given initial configuration if in all fair executions starting at it, all agents eventually agree to this value—so, intuitively, population protocols compute by reaching consensus. An execution is fair if it is finite and cannot be extended, or it is infinite and every configuration reachable infinitely often (that is, reachable from infinitely many of the configurations reached along the execution) is also reached infinitely often. Given a set of inputs (typically a set of vectors of natural numbers), and a mapping that assigns to each input an initial configuration, the predicate computed by a protocol is the function that assigns to each input the boolean value computed by the protocol from the corresponding initial configuration. If the protocol does not reach consensus for some input, then we say it is ill specified and does not compute any predicate.

Much of the work on population protocols has concentrated on characterizing the predicates computable by well-specified protocols. In particular, Angluin et al. [2] gave explicit well-specified protocols to compute every predicate definable in Presburger arithmetic, and showed in a later paper (with a different set of authors) that they cannot compute anything else, i.e., well-specified population protocols compute exactly the Presburger-definable predicates [4].

Since it is easy to erroneously design protocols that are not well specified, one can ask two natural verification questions: Given a population protocol, is it well specified? Given a population protocol and a Presburger predicate (represented by a Presburger formula), does the protocol compute the predicate? We call them the *well-specification* and *fitting* problems.

The semantics of a population protocol is an infinite family of finite-state transition systems, one for each possible input. Deciding if the protocol reaches consensus for a fixed input only requires to inspect one of these finite transition systems, and can be done automatically using a model checker. This approach has been followed in [20,21,6,7], but it only proves the correctness of a protocol for a finite number of (typically small) inputs. Alternatively, one can also formalize a proof of well specification in a theorem prover [8], but this approach is not automatic: a human prover must first come up with a proof for each particular protocol.

Since the well-specification problem asks if consensus is reached for all inputs, and there are infinitely many of them, it is not obviously decidable; in fact, similar questions are undecidable for many parameterized systems [5]. Moreover, techniques based on algorithms for the coverability problem of Petri nets, or on well-quasi-orders—which have been used to prove decidability of many parameterized verification problems [1, 12]—cannot be directly applied to the well specification and fitting problems. Loosely speaking, the reason is that the set of initial configurations from which all agents eventually agree on a value is not necessarily upward- nor downward-closed.

Despite these difficulties, in the first part of the paper we show that the well-specification and fitting problems are decidable and recursively equivalent to the reachability problem for Petri nets. Our reductions show that both problems have elementary complexity iff the reachability problem for Petri nets has elementary complexity, a problem that remains open since the early 80s.

In the second part of the paper we study the *tailor* problem: Given a well-specified protocol, returns a Presburger formula for the predicate computed by it. To solve the problem, we introduce a notion of certificate (of well-specification) of a protocol. We provide algorithms that, given a protocol and an advice string decide if the string is a certificate, and extract from it a Presburger formula of the predicate computed by the protocol. The overall algorithm for the tailor problem just enumerates all advice strings, checks if they are a certificate, and if so computes a formula. However, this algorithm may not terminate if a protocol happens to have no certificates. So we also show that this is not the case: every well-specified protocol has at least one certificate. The proof relies on several recent results from the theory of Petri nets: the existence of Presburger-definable inductive sets that separate unreachable markings [15], the effective Presburger-definability of the mutual reachability relation [16], and a result from the theory of accelerations [18]. Finally, along the way we obtain a new proof of the main theorem of [4] showing that well-specified protocols can only compute Presburger-definable predicates.

The paper is organized as follows. Section 2 presents some preliminaries. Section 3 introduces population protocols and defines the well-specification, fitting, and tailor problems. Section 4 describes the connection between population protocols and Petri nets. Sections 5 and 6 reduce the well-specification and fitting problems to the reachability problem for Petri nets, and Section 7 presents reductions in the other direction. Finally, Section 8 presents our certificate-based algorithm for the tailor problem.

2 Preliminaries: Presburger Sets, Semilinear Sets, Multisets

Presburger Arithmetic and Presburger sets. Presburger arithmetic is the first-order theory of addition, i.e., the first-order theory of the natural numbers with addition as only function, and equality as only predicate. A set $\mathbf{S} \subseteq \mathbb{N}^d$ is a *Presburger-definable*, or just a *Presburger set*, if there exists a formula $\mathtt{F}(x_1, \ldots, x_d)$ with free variables $x_1, \ldots, x_d$ such that $\mathtt{F}(n_1, \ldots, n_d)$ is true iff $(n_1, \ldots, n_d) \in \mathbf{S}$.

Semi-linear sets. A set $\mathbf{L} \subseteq \mathbb{N}^d$ is *linear* if there is a *base* or *root vector* $\mathbf{b}$ and a finite set $\mathbf{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_n\}$ of *periods* such that $\mathbf{L} = \{\mathbf{b} + \sum_{i=1}^{n} \lambda_i \mathbf{p}_i \mid (\lambda_1, \ldots, \lambda_n) \in \mathbb{N}^d\}$. We write $\mathbf{L} = (\mathbf{b}; \mathbf{P})$, and say that the pair $(\mathbf{b}; \mathbf{P})$ is a *linear representation* of $\mathbf{L}$. A set $\mathbf{S}$ is *semi-linear set* if it is a finite union of linear sets, and the set of its linear representations is called a *semi-linear representation* of $\mathbf{S}$.

It is well known that the semi-linear sets and the Presburger sets coincide [13]. In particular, semi-linear sets are effectively closed under Boolean operations and emptiness, inclusion, and equivalence of semi-linear sets are all decidable.

Multisets. A *multiset* on a finite set E is a mapping $M\colon E \to \mathbb{N}$. For $e \in E$, $M(e)$ denotes the number of occurrences of element e in M. Operations on $\mathbb{N}$ like addition, subtraction, or comparison, are extended to multisets by defining them component wise. The set of all multisets over E is denoted $\mathbb{N}^E$. Given $e \in E$, we denote $\mathbf{e} \in \mathbb{N}^E$ 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 $e' \neq e$. The *support* of a multiset $M \in \mathbb{N}^E$, denoted by $\mathrm{Sup}(M)$, is the set $\{e \in E \mid M(e) > 0\}$.

Given a total order $e_1 \prec e_2 \prec \cdots \prec e_n$ on E, a multiset M can be represented by the vector $(M(e_1), \ldots, M(e_n))$, and a set $\mathcal{M}$ of multisets by a set of vectors. A set of multisets over a finite set E is Presburger (resp. linear, semi-linear) if its corresponding set of vectors is Presburger (resp. linear, semi-linear)

3 Population Protocols

A *population* P on a finite set E is a non-empty multiset on E, i.e., $P \in \mathbb{N}^E$ and $P \neq \emptyset$. Thus $P(e) > 0$ for some $e \in E$, which is equivalent to $\Sigma_{e \in E} P(e) > 0$. The set of all populations on E is denoted by $\mathrm{Pop}(E)$.

Example 1 Let $E = \{a, b\}$. The set of populations $\{P \in \mathrm{Pop}(E) \mid P(a) \geq P(b)\}$ is Presburger, since it is denoted by the Presburger formula $\exists Y\colon X_a = Y + X_b \wedge X_b > 0$. It is easy to see that the set of populations $\{P \in \mathrm{Pop}(E) \mid P(a) = P(b)^2\}$ is not Presburger. □

3.1 Protocol Schemes

A *protocol scheme* $\mathcal{A} = (Q, \Delta)$ consists of a finite non-empty set Q of states and a set $\Delta \subseteq Q^4$. If $(q_1, q_2, q_1', q_2') \in \Delta$, we write $(q_1, q_2) \mapsto (q_1', q_2')$ and call it a *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 transitions from Δ.[1] Formally, given two configurations C and C' and a transition $\delta = (q_1, q_2) \mapsto (q_1', q_2')$, we write $C \xrightarrow{\delta} C'$ if

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

From the definition of step, it is easily seen that, inside the tuple $(q_1, q_2, q_1', q_2') \in \Delta$, the ordering between q_1 and q_2 and between q_1' and q_2' is irrelevant. We write $C \xrightarrow{w} C'$ for a sequence $w = \delta_1 \ldots \delta_k$ of transitions if there exists a sequence $C_0, \ldots, C_k$ of configurations satisfying $C = C_0 \xrightarrow{\delta_1} C_1 \cdots \xrightarrow{\delta_k} C_k = C'$. We also write $C \to C'$ if $C \xrightarrow{\delta} C'$ for some transition $\delta \in \Delta$, and call $C \to C'$ a *step*. We say that C' is *reachable from* C if $C \xrightarrow{w} C'$ for some (possibly empty) sequence w of transitions. Further, two configurations C, C' are *mutually reachable* if C is reachable from C' and C' is reachable from C. We have:

Lemma 1 *For every configuration C, the set of configurations reachable from C is finite.*

[1] While protocol schemes model pairwise interactions only, one can model k-way interactions for a fixed $k > 2$ by adding additional states.

Proof Follows immediately from the fact that an interaction does not create or destroy agents, just changes their current states. Since Q is finite, there are only finitely many configurations C' satisfying $\sum_{q\in Q} C(q) = \sum_{q\in Q} C'(q)$. $\square$

Example 2 (Debating philosophers.) We consider a protocol scheme $\mathcal{A} = (Q, \Delta)$ with agents called "philosophers". A group of philosophers debate about a thesis, say, "do animals have rights?". At each point in time a philosopher is tired or rested, denoted by T and R, respectively, and is for or against the thesis, denoted by F and A. The set Q contains four states

$$Q = \{\mathtt{T}, \mathtt{R}\} \times \{\mathtt{F}, \mathtt{A}\} \ .$$

The interactions between the philosophers model the following behavior. When two philosophers meet, they compare their positions and update their current state as follows:

(*i*) Philosophers with the same opinion do not debate and stay in their current state.
(*ii*) Rested philosophers convince tired opponents of anything.
(*iii*) If two philosophers in the same physical condition debate, the one for animal rights convinces the one against and they both are tired after the debate.

Accordingly, the set Δ is defined as follows where $\alpha, \beta \in \{\mathtt{F}, \mathtt{A}\}$, $\alpha \neq \beta$ and $\mathtt{X}, \mathtt{Y} \in \{\mathtt{R}, \mathtt{T}\}$:

$$(\mathtt{X}\alpha, \mathtt{Y}\alpha) \mapsto (\mathtt{X}\alpha, \mathtt{Y}\alpha) \qquad (\mathtt{R}\alpha, \mathtt{T}\beta) \mapsto (\mathtt{R}\alpha, \mathtt{T}\alpha) \qquad (\mathtt{X}\alpha, \mathtt{X}\beta) \mapsto (\mathtt{TF}, \mathtt{TF}) \ .$$

We represent configurations as tuples indicating the number of philosophers in each state. Here is a possible infinite step sequence of the protocol scheme.

$$\begin{array}{cccccc} & \mathtt{RF} & \mathtt{RA} & \mathtt{TF} & \mathtt{TA} & \\ (& 3 & 3 & 0 & 0 &) \end{array} \to \begin{array}{cccccc} & \mathtt{RF} & \mathtt{RA} & \mathtt{TF} & \mathtt{TA} & \\ (& 2 & 2 & 2 & 0 &) \end{array} \to \begin{array}{cccccc} & \mathtt{RF} & \mathtt{RA} & \mathtt{TF} & \mathtt{TA} & \\ (& 2 & 2 & 1 & 1 &) \end{array} \to \begin{array}{cccccc} & \mathtt{RF} & \mathtt{RA} & \mathtt{TF} & \mathtt{TA} & \\ (& 1 & 1 & 3 & 1 &) \end{array} \quad \to$$

$$\begin{array}{cccccc} & \mathtt{RF} & \mathtt{RA} & \mathtt{TF} & \mathtt{TA} & \\ (& 0 & 0 & 5 & 1 &) \end{array} \to \begin{array}{cccccc} & \mathtt{RF} & \mathtt{RA} & \mathtt{TF} & \mathtt{TA} & \\ (& 0 & 0 & 6 & 0 &) \end{array} \to \begin{array}{cccccc} & \mathtt{RF} & \mathtt{RA} & \mathtt{TF} & \mathtt{TA} & \\ (& 0 & 0 & 6 & 0 &) \end{array} \to \begin{array}{cccccc} & \mathtt{RF} & \mathtt{RA} & \mathtt{TF} & \mathtt{TA} & \\ (& 0 & 0 & 6 & 0 &) \end{array} \cdots \quad \square$$

3.2 Configuration Graphs

The *configuration graph* of a protocol scheme $\mathcal{A}$ is the infinite directed graph $(\mathrm{Pop}(Q), \to)$ having the populations over Q as nodes and the pairs (C, C') such that $C \to C'$ as edges. Consider the partition $\{\mathrm{Pop}(Q)_i\}_{i\geq 1}$ of $\mathrm{Pop}(Q)$, where $\mathrm{Pop}(Q)_i = \{C \in \mathrm{Pop}(Q) \mid \sum_{q\in Q} C(q) = i\}$. (Note that i starts at 1 because every population contains at least one agent.) Since interactions do not create or destroy agents, the set $\{\to_i\}_{i\geq 1}$, where $\to_i = \to \cap \mathrm{Pop}(Q)_i^2$, is also a partition of $\to$. Therefore $(\mathrm{Pop}(Q), \to)$ consists of the infinitely many disjoint and finite subgraphs $\{(\mathrm{Pop}(Q)_i, \to_i)\}_{i\geq 1}$.

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 configuration graph.

Example 3 (Debating philosophers.) In the step sequence of Example 2 the number of philosophers remains constant at 6, and so all its configurations belong to $\mathrm{Pop}(Q)_6$.

We prove that the set $\mathcal{B}$ of bottom configurations of the debating philosophers is $\mathcal{B} = \mathcal{B}_{\mathtt{F}} \cup \mathcal{B}_{\mathtt{A}}$, where

$$\mathcal{B}_{\mathtt{F}} = \{\, (rf, 0, tf, 0) \mid rf + tf > 0 \,\} \quad \text{and} \quad \mathcal{B}_{\mathtt{A}} = \{\, (0, ra, 0, ta) \mid ra + ta > 0 \,\} \ .$$

(Observe that in the configurations of $\mathcal{B}$ all philosophers have the same opinion: in $\mathcal{B}_F$ they are all for animal rights, and in $\mathcal{B}_A$ against them.) If $C \in \mathcal{B}$, then the only possible step is $C \to C$. So $\{C\}$ is a bottom SCC, and C is a bottom configuration. It remains to prove that if $C \notin \mathcal{B}$ then C is not a bottom configuration. For this it suffices to exhibit a configuration C' reachable from C such that C is not reachable from C' (i.e., C' is reachable from C, but C and C' are not mutually reachable). Let $C = (rf, ra, tf, ta) \notin \mathcal{B}$. We consider several cases:

- $rf > 0$ and $ra > 0$. Then $C \to C'$ for $C' = (rf-1, ra-1, tf+2, ta)$ (two rested philosophers debate and get tired) but, since no rules turn a tired philosopher into a rested one, C is not reachable from C'.
- $rf > 0$ and $ra = 0$. Since $C \notin \mathcal{B}$ we have $ta > 0$, and so $C' = (rf, 0, tf + ta, 0)$ is reachable from C (a rested philosopher for animal rights convinces all tired philosophers against them), but not vice versa.
- $rf = 0$ and $ra > 0$. Since $C \notin \mathcal{B}$ we have $tf > 0$, and so $C' = (0, ra, 0, tf + ta)$ is reachable from C (a rested philosopher against animal rights convinces all tired philosophers in favor of them), but not vice versa.
- $rf = 0$ and $ra = 0$. Since $C \notin \mathcal{B}$ we have $tf, ta > 0$, and so $C' = (0, 0, tf + ta, 0)$ is reachable from C (a tired philosopher for animal rights convinces all tired philosophers against them), but not vice versa. □

3.3 Executions and Fair Executions

An *execution* of $\mathcal{A}$ is a finite or infinite sequence of configurations $C_0, C_1, \ldots$ such that $C_i \to C_{i+1}$ for each $i \geq 0$. Following Angluin et al. [2], we introduce a notion of fair execution. Loosely speaking, if a fair execution is infinite then every step that is enabled infinitely often by visiting infinitely often a configuration C must be taken infinitely often from C. Formally, an execution $C_0, C_1, \ldots$ is *fair* if it is finite and cannot be extended, or it is infinite and for every step $C \to C'$, if $C_i = C$ for infinitely many indices $i \geq 0$, then $C_j = C$ and $C_{j+1} = C'$ for infinitely many indices $j \geq 0$. Thanks to Lemma 1 we show in Lemma 2 that every fair execution reaches a strongly connected component (SCC) of $(\mathrm{Pop}(Q), \to)$ and never leaves it.

Remark 1 Our notion of fairness is based on configurations, and does not coincide with transition-based weak fairness (if a transition is enabled at infinitely many configurations, then it is also taken infinitely often). To illustrate the difference, consider the protocol scheme with states $\{q_1, q_2, r_1, r_2, s\}$ and transitions

$$\delta_{qr} = (q_1, r_1) \mapsto (q_1, r_1) \qquad \begin{array}{l} \delta_{q_1} = (q_1, s) \mapsto (q_2, s) \\ \delta_{q_2} = (q_2, s) \mapsto (q_1, s) \end{array} \qquad \begin{array}{l} \delta_{r_1} = (r_1, s) \mapsto (r_2, s) \\ \delta_{r_2} = (r_2, s) \mapsto (r_1, s) \end{array} .$$

Consider the configuration that puts one agent in each of q_1, r_2, and s, that is $\mathbf{q}_1 + \mathbf{r}_2 + \mathbf{s}$. From this configuration we can execute the infinite sequence of transitions $w = (\delta_{q_1}\delta_{r_2}\delta_{r_1}\delta_{q_2})^\omega$. It is easy to see that during the execution of w the transition δ_{qr} is never enabled, and so the execution is weakly fair in the transition-based sense. However, it is not fair in the configuration-based sense. Indeed, while $\mathbf{q}_1 + \mathbf{r}_2 + \mathbf{s}$ is visited infinitely often, only δ_{q_1} is executed from it, even though it also enables δ_{r_2}. □

The following lemma 2 formalizes a fundamental property of fair executions: they eventually reach a bottom SCC of the configuration graph, and then visit each of its states infinitely often (actually, if the execution is finite, then the bottom SCC consists of just one state without successors; intuitively, the execution reaches this state, and stays there "forever").

Lemma 2 *For every fair execution $C_0, C_1, \ldots$ there is an index $i \geq 0$ such that C_i is a bottom configuration, and the set $\{C_j \mid j \geq i\}$ is a bottom SCC of the configuration graph.*

Proof If the execution is finite, then, since it cannot be extended, its last configuration is a bottom SCC with one single node and no outgoing transitions. If the execution is infinite, then the fairness condition forces it to eventually leave every non-bottom SCC it enters. So there is an index $i \geq 0$ such that C_j is a bottom configuration for every $j \geq i$, and so $\{C_j \mid j \geq i\}$ is included in a bottom SCC $\mathcal{S}$. Now let C be an arbitrary configuration of $\mathcal{S}$. By Lemma 1 the set $\mathcal{S}$ is finite, and so there is a number k such that for every $j \geq i$, the configuration C is reachable from C_j in at most k steps. A simple induction on k shows that, by fairness, C is contained in the execution. So $\mathcal{S} = \{C_j \mid j \geq i\}$. □

Example 4 (Debating philosophers.) It is easy to see that the infinite sequence of steps shown in Example 2 is a fair execution. Many other executions are not fair (for example, the infinite execution $(2, 1, 0, 0)^\omega$ where no two philosophers with diverging opinions get to debate). A less trivial example is $(3, 3, 0, 0)\,(\,(2, 2, 2, 0)\,(2, 2, 1, 1)\,)^\omega$. □

3.4 Population Protocols

We define what it means for a protocol scheme to compute a predicate $\Pi\colon \text{Pop}(\Sigma) \to \{0, 1\}$, where Σ is a non-empty, finite set of *input variables*. Before presenting formal definitions, we give some intuition.

The first step is to add to a protocol scheme an *input mapping* $I\colon \text{Pop}(\Sigma) \to \text{Pop}(Q)$ and an *output mapping* $O\colon \text{Pop}(Q) \to \{0, \bot, 1\}$. The input mapping assigns to an input $X \in \text{Pop}(\Sigma)$ an initial configuration $I(X)$ of the protocol scheme, and the output mapping assigns to a configuration C an output, which can be either 0, 1, or $\bot$. Here $\bot$ stands for "undefined" or "no output".

Intuitively, imagine that an operator is in charge of computing a boolean for each input $X \in \text{Pop}(\Sigma)$ with the help of a machine implementing the protocol scheme. Upon receiving X, the operator first applies the input mapping to it, obtains the configuration $C = I(X)$, allocates $C(q)$ agents to each state q of the scheme/machine, and runs it from this initial configuration, letting it produce a fair execution. The machine has two lamps for the outputs 1 and 0. The b-lamp is switched on whenever the current configuration C satisfies $O(C) = b$, and switched off otherwise. By definition, the execution of the machine outputs $b \in \{0, 1\}$ if it eventually stabilizes to b, meaning that from some moment on the b-lamp stays on forever (that is, from some moment on the execution only visits configurations C such that $O(C) = b$).

For a given input X some fair execution starting at $C(X)$ may not stabilize to 0 or 1. Or two different fair executions starting at $C(X)$ may stabilize to 0 and 1, respectively. Then we say that the scheme is *ill specified*. More precisely: If there is at least one input for which at least one fair execution does not stabilize to 0 or 1, or for which two fair executions stabilize to 0 and to 1, respectively, then the scheme is ill specified, and "does not compute any predicate".

If a scheme is well specified, then for every input X all fair computations from $I(X)$ stabilize to the same boolean b_X, and we define the predicate computed by the protocol as the mapping Π given by $\Pi(X) = b_X$.

Example 5 (Debating philosophers.) We define input and output mappings for the debating philosophers. For the set of inputs we choose $\Sigma = \{\mathtt{F}, \mathtt{A}\}$ (For and Against). So a population over Σ models a population of philosophers, specifying how many are for and against animal rights. We represent a population with f philosophers for and a philosophers against animal rights by the pair (f, a).

As input mapping we choose the function $I\colon \mathrm{Pop}(\Sigma) \to \mathrm{Pop}(Q)$ given by

$$I(f,a) = \begin{array}{cccc} \mathtt{RF} & \mathtt{RA} & \mathtt{TF} & \mathtt{TA} \\ (\; f & a & 0 & 0 \;) \end{array}$$

In other words, the mapping assigns to (f, a) a population with f rested philosophers supporting animal rights, a rested philosophers against animal rights, and no tired philosophers.

As output mapping we choose the function $O\colon \mathrm{Pop}(Q) \to \{0, \perp, 1\}$ given by

$$O(rf, ra, tf, ta) = \begin{cases} 1 & \text{if } ra + ta = 0 \text{ (all philosophers are for animal rights)} \\ 0 & \text{if } rf + tf = 0 \text{ (all philosophers are against animal rights)} \\ \perp & \text{otherwise} \end{cases}$$

□

After this informal introduction, we now present some formal definitions.

Input and Output Mappings. Formally, an *input mapping* of a protocol scheme $\mathcal{A} = (Q, \Delta)$ is a function $I\colon \mathrm{Pop}(\Sigma) \to \mathrm{Pop}(Q)$ that maps each input population X to a configuration of $\mathcal{A}$. The set of *initial configurations* is $\mathcal{I} = \{I(X) \mid X \in \mathrm{Pop}(\Sigma)\}$. An *output mapping* of O is a function $O\colon \mathrm{Pop}(Q) \to \{0, \perp, 1\}$ that associates to each configuration C of $\mathcal{A}$ an output value in $\{0, \perp, 1\}$. A configuration C on Q such that $O(C) = b$ for some $b \in \{0, \perp, 1\}$ is called a *b-configuration*.

If input and output mappings can be arbitrary functions, even non computable ones, then any problem involving them is bound to be undecidable. For this reason we introduce "reasonable" classes of input and output mappings.

An input mapping I is *Presburger* if the set of pairs $(X, C) \in \mathrm{Pop}(\Sigma) \times \mathrm{Pop}(Q)$ such that $C = I(X)$ is definable in Presburger arithmetic. An output mapping O is *Presburger* if the same holds for the set of pairs $(C, b) \in \mathrm{Pop}(Q) \times \{0, \perp, 1\}$ such that $O(C) = b$.

A *population protocol* is a triple $(\mathcal{A}, \mathtt{I}, \mathtt{O})$, where $\mathcal{A}$ is a protocol scheme, and $\mathtt{I}(X, C)$ and $\mathtt{O}(C, b)$ are formulas of Presburger arithmetic denoting a Presburger input mapping I and a Presburger output mapping O, respectively.

Presburger input and output mappings are still quite general. Many papers onlyconsider a more restricted class. An input mapping I is *simple* if for every input variable $\sigma \in \Sigma$ there exists a state $q_\sigma \in Q$ such that

$$I(X) = \sum_{\sigma \in \Sigma} X(\sigma)\, \mathbf{q}_\sigma$$

for every input population X on Σ. Intuitively, if I is simple then each input variable is assigned a state, and the operator can prepare the initial configuration for the input X by going through all input variables σ, and putting $X(\sigma)$ agents in the state corresponding to σ.

Similarly, an output mapping O is *simple* if there exists a partition (Q_0, Q_1) of Q such that

$$O(C) \;=\; \begin{cases} 0 & \text{if } \mathrm{Sup}(C) \subseteq Q_0 \\ 1 & \text{if } \mathrm{Sup}(C) \subseteq Q_1 \\ \perp & \text{otherwise} \end{cases}$$

for every configuration C. Notice that O is well defined because $\mathrm{Sup}(C) \neq \emptyset$.

Example 6 (Debating philosophers.) The input mapping given in Example 5 is Presburger. Indeed, if we represent the input X satisfying $X(\mathtt{F}) = f$ and $X(\mathtt{A}) = a$ by the pair (f, a), and the configuration C satisfying $C(\mathtt{RF}) = rf$, $C(\mathtt{RA}) = ra$, $C(\mathtt{TF}) = tf$, and $C(\mathtt{TA}) = ta$ by the vector (rf, ra, tf, ta), then the set of pairs (X, C) such that $I(X) = C$ is defined by the Presburger formula

$$\mathtt{I}(f, a, rf, ra, tf, ta) := (rf = f \wedge ra = a \wedge tf = 0 \wedge ta = 0)$$

The output mapping is also Presburger. Moreover, both mappings are simple. Indeed, for I we have

$$I(f, a) = f\,\mathtt{RF} + a\,\mathtt{RA}$$

For the output mapping simply consider $Q_0 = \{\mathtt{RA}, \mathtt{TA}\}$ and $Q_0 = \{\mathtt{RF}, \mathtt{TF}\}$. □

Remark 2 A particular case of protocols with Presburger input and output mappings are population protocols with leader [3]. In these protocols the initial configuration contains one agent, called the *leader*, occupying a distinguished initial state q_l not initially occupied by any other agent. This corresponds to the input mapping $I(X) = \mathbf{q}_l + \sum_{\sigma \in \Sigma} X(\sigma)\,\mathbf{q}_\sigma$ which is obviously Presburger. □

Stabilization and well-specified protocols. An execution $C_0, C_1, \ldots$ *stabilizes to b* for a given $b \in \{0, \perp, 1\}$ if there exists $n \in \mathbb{N}$ such that $O(C_m) = b$ for every $m \geq n$ (if the execution is finite, then this means for every m between n and the length of the execution). Notice that there may be many different executions from a given configuration C_0, each of which may stabilize to 0, 1, or $\perp$, or not stabilize at all.

A population protocol $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ is *well specified* if for every input population $X \in \mathrm{Pop}(\Sigma)$, every fair execution of $\mathcal{A}$ starting at $I(X)$ stabilizes to the same value $b \in \{0, 1\}$. Otherwise, the protocol is *ill specified*. Finally, population protocol *computes* a predicate Π if for every $X \in \mathrm{Pop}(\Sigma)$, every fair execution of $\mathcal{A}$ starting at $I(X)$ stabilizes to $\Pi(X)$. It follows easily from the definitions that a protocol computes a predicate iff it is well specified.

Example 7 (Debating philosophers.) Let us show that the population protocol $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ of the debating philosophers is well specified. By Lemma 2, every fair execution eventually gets trapped in bottom configurations, that is, in configurations of $\mathcal{B} = \mathcal{B}_{\mathtt{F}} \cup \mathcal{B}_{\mathtt{A}}$, where $\mathcal{B}_{\mathtt{F}}$ and $\mathcal{B}_{\mathtt{A}}$ were computed in Example 3:

$$\mathcal{B}_{\mathtt{F}} = \{\, (rf, 0, tf, 0) \mid rf + tf > 0 \,\} \quad \text{and} \quad \mathcal{B}_{\mathtt{A}} = \{\, (0, ra, 0, ta) \mid ra + ta > 0 \,\}$$

Since in the configurations of $\mathcal{B}$ all philosophers have the same opinion, it is not possible to move from $\mathcal{B}_{\mathtt{F}}$ to $\mathcal{B}_{\mathtt{A}}$, or vice versa. So a fair execution gets trapped either in $\mathcal{B}_{\mathtt{F}}$ or in $\mathcal{B}_{\mathtt{A}}$, and therefore every fair execution stabilizes to 0 or 1.

It remains to show that for every fixed initial configuration $C_0 = (rf_0, ra_0, 0, 0)$, either all fair executions starting at C_0 get trapped in $\mathcal{B}_{\mathtt{F}}$, or they all get trapped in $\mathcal{B}_{\mathtt{A}}$. We prove that they get trapped in $\mathcal{B}_{\mathtt{A}}$ if $rf_0 \geq ra_0$, and in $\mathcal{B}_{\mathtt{F}}$ otherwise.

Let $\mathcal{C}_1 = \{(tf, ra, rf, ta) \mid rf < ra\}$. By direct inspection of the transitions, if $C \in \mathcal{C}_1$ and $C \to C'$, then $C' \in \mathcal{C}_1$. Therefore, if $rf_0 \geq ra_0$ then a fair execution starting at C_0 gets trapped in configurations of $\mathcal{B} \cap \mathcal{C}_1$, and so only in configurations of $\mathcal{B}_{\mathtt{A}}$.

Let $\mathcal{C}_2 = \{(tf, ra, rf, ta) \mid tf \geq ra \wedge tf + rf > 0\}$ By direct inspection of the transitions, if $C \in \mathcal{C}_2$ and $C \to C'$, then $C' \in \mathcal{C}_2$. (For the transition $(\mathrm{R}\alpha, \mathrm{T}\beta) \mapsto (\mathrm{R}\alpha, \mathrm{T}\alpha)$, observe that if the transition is enabled then $rf > 0$.) Assume $C_0 = (tf_0, ra_0, 0, 0)$ satisfies $tf_0 \geq ra_0$. Since configurations contain at least one agent, we have $tf_0 > 0$ and so $C_0 \in \mathcal{C}_2$. Therefore, a fair execution starting at C_0 gets trapped in configurations of $\mathcal{B} \cap C_2$, and so only in configurations of $\mathcal{B}_{\mathtt{F}}$.

So the protocol of the debating philosophers is well specified, hence it computes a predicate $\Pi\colon \text{Pop}(\{\mathtt{F}, \mathtt{A}\}) \to \{0, 1\}$. This predicate is just the *majority* predicate: $\Pi(f, a) = 1$ iff $f \geq a$. □

3.5 Verification Problems

Angluin et al. [2] showed that well-specified population protocols can compute all Presburger predicates. Later, Angluin, Aspnes and Eisenstat [4] proved by means of an involved argument that they can only compute Presburger predicates. However, a protocol can be ill specified, or be well specified but compute a predicate different from the one intended. Finally, given a protocol we would like to obtain a Presburger formula for the predicate it computes. So we study the following three problems.

- The *well-specification problem*: given a population protocol $(\mathcal{A}, \mathtt{I}, \mathtt{O})$, is it well specified?
- The *fitting problem*: given a population protocol $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ and a Presburger predicate Π, does $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ compute Π?
- The *tailor problem*: given a well-specified population protocol $(\mathcal{A}, \mathtt{I}, \mathtt{O})$, compute (in the standard sense) a Presburger formula for the predicate computed (in the population protocol sense) by $(\mathcal{A}, \mathtt{I}, \mathtt{O})$.

Note that the fitting problem does not assume $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ to be well specified. Consequently, if $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ does not compute Π then either the population protocol is ill specified, or it stabilizes to $b \in \{0, 1\}$ for some input $X \in \text{Pop}(\Sigma)$ such that $\Pi(X) = 1 - b$.

In the rest of the paper we obtain the following results:

- The well-specification and fitting problems are Turing-reducible in elementary time to the reachability problem for Petri nets.
  In other words, we show that both problems can be solved in elementary time with the help of an oracle for the reachability problem for Petri nets. In particular, this proves that both problems are decidable.
- The reachability problem for Petri nets can be reduced in polynomial time to the (complements of the) well-specification or the fitting problems.
  Together with the previous result, this shows that the well-specification and fitting problems can be solved in elementary time if and only if the reachability problem for Petri nets can be solved in elementary time.
- There is an algorithm for the tailor problem.
  This algorithm can also be used to solve the well-specification and fitting problems. However, it consists of two semi-decision algorithms, and currently we do not know of any elementary time reduction to the reachability problem. As a corollary of this algorithm we obtain an alternative proof to the result of Angluin et al. [4].

4 Population Protocols as Petri Nets

The computation of a population protocol can be simulated by an associated Petri net. This allows us to apply results on Petri nets to population protocols.

A Petri net $N = (P, T, F)$ consists of a finite set P of *places*, a finite set T of *transitions*, and a *flow function* $F\colon (P \times T) \cup (T \times P) \to \mathbb{N}$. The *preset* of a transition t is the multiset ${}^\bullet t$

of places given by ${}^\bullet t(p) = F(p,t)$ and its *postset* the multiset ${}^\bullet t$ given by $t^\bullet(p) = F(t,p)$. A *marking* $M \in \mathbb{N}^P$ is a multiset on the set P of places and we say that M puts $M(p)$ *tokens* in place p. A transition $t \in T$ is *enabled at* marking M, written $M\,[t\rangle$, if ${}^\bullet t \leq M$. A transition t that is enabled at M can *fire*, yielding the marking $M' = M - {}^\bullet t + t^\bullet$. We denote this fact as $M\,[t\rangle\,M'$. We extend enabledness and firing inductively to words of transitions as follows. Let $w = t_1 \dots t_k$ be a finite word of transitions $t_j \in T$. We write $M\,[w\rangle\,M'$ if there exists a sequence $M_0, \dots, M_k$ of markings such that $M = M_0\,[t_1\rangle\,M_1 \cdots [t_k\rangle\,M_k = M'$, and say that M' is *reachable from* M.

Given a Petri net $N = (P,T,F)$, a set $\mathcal{M}$ of markings, and a language $W \subseteq T^*$, we introduce the sets:

$$post_N(\mathcal{M}, W) = \{M' \in \mathbb{N}^P \mid \exists M \in \mathcal{M}\, \exists w \in W\colon\ M\,[w\rangle\,M'\}$$
$$pre_N(\mathcal{M}, W) = \{M \in \mathbb{N}^P \mid \exists M' \in \mathcal{M}\, \exists w \in W\colon\ M\,[w\rangle\,M'\}\ .$$

When $W = T^*$ these sets are denoted by $post^*_N(\mathcal{M})$ and $pre^*_N(\mathcal{M})$, respectively.

The *reachability problem* for Petri nets asks, given a Petri net N and two markings M, M' of N, whether M' is reachable from M, or equivalenty whether $M' \in post_N(\{M\})$. The problem is known to be decidable in non-primitive recursive time, and EXPSPACE-hard. It is open whether the problem has an algorithm that runs in elementary time, i.e., in k-EXPTIME for some number k independent of the input.

Given two sets $\mathcal{M}, \mathcal{M}'$ of markings, we say that $\mathcal{M}'$ is reachable from $\mathcal{M}, \mathcal{M}'$ if there are $M \in \mathcal{M}$ and $M' \in \mathcal{M}'$ such that M' is reachable from M. The reachability problem for Presburger-definable sets of markings is also decidable:

Theorem 1 *Let N be a Petri net, and let ϕ, ϕ' be two Presburger formulas denoting sets $\mathcal{M}, \mathcal{M}'$ of markings of N. The problem whether $\mathcal{M}'$ is reachable from $\mathcal{M}$ can be reduced in elementary time to the reachability problem for Petri nets, and is thus decidable.*

Proof Since similar reductions are well known (see e.g. [14]), we only sketch the argument. Let d be the number of places of N. Markings of N can then be represented as vectors of $\mathbb{N}^d$. Since $\mathcal{M}$ and $\mathcal{M}'$ are Presburger definable, they are semi-linear [13], and we can compute in elementary time (actually, in triple exponential time in N), semi-linear representations for $\mathcal{M}$ and $\mathcal{M}'$.

Let $\{(r_1; P_1), \dots, (r_n; P_n)\}$ and $\{(r'_1, P'_1), \dots, (r'_m, P'_m)\}$ be semi-linear representations of $\mathcal{M}$ and $\mathcal{M}'$. We sketch the behavior of a Petri net $\widehat{N}$ with an initial marking $\widehat{M}$ that, loosely speaking, nondeterministically generates an initial marking M_0 of N, simulates N on this marking, nondeterministically stops the simulation at some point in time, and nondeterministically checks if the marking M reached by N when the simulation is stopped belongs to $\mathcal{M}'$.

The marking M_0 is generated as follows. Initially $\widehat{N}$ nondeterministically fires a transition from a set $\{t_1, \dots, t_n\}$, containing a transition for each linear set in the representation of $\mathcal{M}$. After firing, say, transition t_i, the net proceeds to nondeterministically generate a marking of (r_i, P_i) where, say, $P_i = \{p_{i1}, \dots, p_{ik}\}$. For this it first fires a transition that puts r_i tokens in the places of N, and then it proceeds to repeatedly fire transitions $t_{i1}, \dots, t_{ik}$ such that the firing of t_{ij} adds p_{ij} tokens to the places of N. The net can stop these firings at any time by nondeterministically choosing to fire a transition *start*, after which it starts simulating N.

The simulation is stopped nondeterministically by firing a transition *stop*. Let M be the marking of N after the simulation stops. The net nondeterministically guesses that M

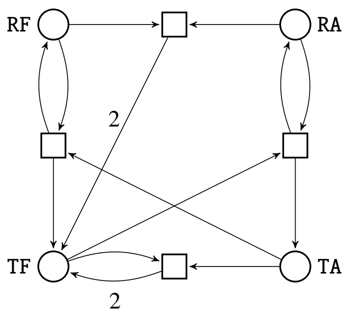

Fig. 1 Petri net for the protocol scheme of the debating philosophers. Transitions t such that ${}^\bullet t = t^\bullet$ are not shown.

belongs to the linear set (r'_j, P'_j) of the representation of $\mathcal{M}'$ by firing a transition t'_i for some $1 \le i \le m$. Assume $P'_i = \{p'_{i1}, \ldots, p'_{ik'}\}$. The net proceeds to nondeterministically check the guess by first firing a transition that removes r'_i tokens from the places of N, and then repeatedly firing transitions $t'_{i1}, \ldots, t'_{ik'}$, where the firing of t'_{il} removes p'_{ij} tokens from the places of N. If the guess is correct, i.e., if M belongs to the linear set (r'_j, P'_j), then the net can reach the empty marking; otherwise, the nondeterministic check gets stuck at some marking different from the empty marking. Therefore, the empty marking can be reached from $\widehat{M}$ iff some marking of $\mathcal{M}'$ is reachable from some marking of $\mathcal{M}$. □

Given a protocol scheme $\mathcal{A} = (Q, \Delta)$, we define the Petri net $N(\mathcal{A}) = (Q, \Delta, F)$, whose places and transitions are the states and transitions of the protocol, respectively, and ${}^\bullet\delta = \{q_1, q_2\}, \delta^\bullet = \{q'_1, q'_2\}$ for every $\delta = (q_1, q_2) \mapsto (q'_1, q'_2)$ in Δ. Note that a configuration of the protocol scheme $\mathcal{A}$ is a marking of the Petri net $N(\mathcal{A})$. Further, whenever $C \xrightarrow{\delta} C'$ for configurations C and C', we have $C\,[\delta\rangle\,C'$ in the Petri net, and vice versa. Figure 1 shows the Petri net for the protocol scheme of the debating philosophers. Transitions t such that ${}^\bullet t = t^\bullet$ (whose firing does not change the current marking) have been omitted.

5 The Well-Specification Problem is Decidable

We first characterize the ill specified population protocols in terms of the bottom configurations of their configuration graphs.

Definition 1 Let $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ be a population protocol, and let $\mathcal{B}$ be the set of bottom configurations of its configuration graph. We define $\mathcal{B}_0$ as the set of configurations $C \in \mathcal{B}$ such that every configuration C' in the same SCC as C satisfies $O(C) = 0$[2]. The set $\mathcal{B}_1$ is defined analogously.

Lemma 3 *A population protocol* $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ *is ill specified iff*
(1) $\mathcal{B} \setminus (\mathcal{B}_0 \cup \mathcal{B}_1)$ *is reachable from* $\mathcal{I}$*, or*
(2) $\mathcal{I}$ *contains a configuration* $C \in \mathcal{I}$ *such that both* $\mathcal{B}_0$ *and* $\mathcal{B}_1$ *are reachable from* C.

Proof By definition, a protocol is ill specified iff
(a) some fair execution starting at a configuration of $\mathcal{I}$ does not stabilize to either 0 or 1; or

[2] Observe that we do *not* define $\mathcal{B}_0$ as the set of configurations $C \in \mathcal{B}$ such that $O(C) = 0$.

(b) two fair executions starting at the same configuration of $\mathcal{I}$ stabilize to 0 and 1, respectively.

We prove that (a) holds iff (1) holds, and (b) holds iff (2) holds.

(a) $\Leftrightarrow$ (1). By Lemma 2, a fair execution eventually gets trapped in a bottom SCC $\mathcal{S}$ of the configuration graph, and visits infinitely often every configuration of $\mathcal{S}$. Therefore, the execution does not stabilize to either 0 or 1 iff either $O(C) = \bot$ for some $C \in \mathcal{S}$, or $\mathcal{S}$ contains two configurations C_1, C_2 such that $O(C_1) \neq O(C_2)$. In both cases we have $\mathcal{S} \cap (\mathcal{B}_0 \cup \mathcal{B}_1) = \emptyset$, and so $\mathcal{S} \subseteq \mathcal{B} \setminus (\mathcal{B}_0 \cup \mathcal{B}_1)$.

(b) $\Leftrightarrow$ (2). By Lemma 2, two executions that stabilize to 0 and 1 get trapped in two bottom SCCs $\mathcal{S}_0$ and $\mathcal{S}_1$, and visit all configurations of these SCCs infinitely often. So, we have $O(C_0) = 0$ for every $C_0 \in \mathcal{S}_0$, and $O(C_1) = 1$ for every $C_1 \in \mathcal{S}_1$. It follows $\mathcal{S}_0 \subseteq \mathcal{B}_0$ and $\mathcal{S}_1 \subseteq \mathcal{B}_1$, and so both $\mathcal{B}_0$ and $\mathcal{B}_1$ are reachable from some $C \in \mathcal{I}$. □

This lemma reduces the ill specification problem to reachability questions for the sets $\mathcal{I}$, $\mathcal{B}$, $\mathcal{B}_0$, and $\mathcal{B}_1$. We use some results of Petri net theory to prove that all these sets are effectively Presburger, which allows us to apply Theorem 1.

The *mutual reachability relation* of a Petri net N is the binary relation over the markings of N that containing the pairs (M, M') such that M' is reachable from M and M is reachable from M' (equivalently, the pairs (M, M') such that M and M' belong to the same SCC of the reachability graph). It is easy to see that the mutual reachability relation is an equivalence relation, and it is closed under addition: if (M_1, M_1') and (M_2, M_2') belong to the relation, then so does $(M_1 + M_2, M_1' + M_2')$. Using a result of Eilenberg and Schützenberger about rational sets in commutative monoids [10] one can then prove that the mutual reachability relation is Presburger-definable. However, the proof of this result is non-constructive. This problem was overcome by Jérôme Leroux [16]:

Theorem 2 **([16])** *There is an algorithm that takes as input a Petri net N and returns a Presburger formula denoting the mutual reachability relation of N. Moreover, the algorithm runs is elementary time.*

Using this theorem, we can easily derive an algorithm to construct Presburger formulas for $\mathcal{B}$, $\mathcal{B}_0$, and $\mathcal{B}_1$.

Proposition 1 *There is an elementary-time algorithm that takes as input a protocol scheme and returns Presburger formulas denoting the sets $\mathcal{B}$, $\mathcal{B}_0$, and $\mathcal{B}_1$.*

Proof We show that the predicate $\mathtt{B}(C)$ associated to the set of bottom configurations is definable in Presburger arithmetic. Let us introduce the predicate $\mathtt{MR}(C, C')$ associated to the mutual reachability relation. Theorem 2 shows that $\mathtt{MR}(C, C')$ is effectively Presburger. Now, we just observe that C is a bottom configuration iff for every configuration C' such that C and C' are mutually reachable and for every C'' such that $C' \to C''$, we have C and C'' are also mutually reachable:

$$\mathtt{B}(C) = \forall C' \, \forall C'' : (\mathtt{MR}(C, C') \wedge C' \to C'') \Rightarrow \mathtt{MR}(C, C'') \ .$$

We claim that $\mathcal{B}_b$ is a Presburger set of configurations. To prove this, we just notice that $\mathcal{B}_b$ is denoted by the following formula:

$$\mathtt{B}_b(C) = \mathtt{B}(C) \wedge \forall C' : \mathtt{MR}(C, C') \Rightarrow \mathtt{O}(C', b) \ .$$

□

Together with Theorem 1, Proposition 1 shows that we decide reachability questions between $\mathcal{I}$ (which is a Presburger set by definition), and the sets of bottom configurations. The next theorem reduces conditions (1) and (2) of Lemma 3 to such questions.

Theorem 3 *The ill specification problem is Turing-reducible in elementary time to the reachability problem for Petri nets, and thus decidable.*

Proof Given a population protocol $(\mathcal{A}, \mathtt{I}, \mathtt{O})$, we show that conditions (1) and (2) of Lemma 3 are reducible to the reachability problem for Petri nets in elementary time. To check condition (1) we proceed as follows:

- Using Proposition 1, compute a Presburger formula $\phi_\perp$ denoting the set $\mathcal{B} \setminus (\mathcal{B}_0 \cup \mathcal{B}_1)$.
- Apply Theorem 1 to the net $N(\mathcal{A})$ and the formulas $\mathtt{I}$ and $\phi_\perp$.

Checking condition (2) requires some more work. Consider the net $(N(\mathcal{A}) \parallel N(\mathcal{A}))$ obtained by putting two disjoint copies of $N(\mathcal{A})$ side by side. (Formally, if $N(\mathcal{A}) = (P, T, F)$, then we take a net (P', T', F') isomorphic to (P, T, F) and satisfying $(P' \cup T') \cap (P \cup T) = \emptyset$, and let $(N(\mathcal{A}) \parallel N(\mathcal{A})) = (P \cup P', T \cup T', F \cup F')$.) We denote a marking of $(N(\mathcal{A}) \parallel N(\mathcal{A}))$ as (M, M'), meaning that its projections onto P and P' are M and M', respectively. It follows easily from this definition that (M, M') is reachable from (M_0, M_0') in $(N(\mathcal{A}) \parallel N(\mathcal{A}))$ iff M is reachable from M_0 and M' is reachable from M_0' in $N(\mathcal{A})$. In particular, since the non-empty markings of $N(\mathcal{A})$ are the configurations of $\mathcal{A}$, condition (2) of Lemma 3 holds iff there are non-empty markings M_I, M_0, M_1 of $N(\mathcal{A})$ such that

- $M_I \in \mathcal{I}$, $M_0 \in \mathcal{B}_0$, $M_1 \in \mathcal{B}_1$, and
- (M_0, M_1) is reachable from (M_I, M_I) in $(N(\mathcal{A}) \parallel N(\mathcal{A}))$.

So to check condition (2) we proceed as follows:

- Construct a Presburger formula ϕ_{II} denoting the set of markings of $(N(\mathcal{A}) \parallel N(\mathcal{A}))$ of the form $\{(M, M) \mid M \in \mathcal{I}\}$.
  This is possible because the set $\mathcal{I}$ is Presburger.
- Construct a Presburger formula ϕ_{01} denoting the set of markings of $(N(\mathcal{A}) \parallel N(\mathcal{A}))$ of the form $\{(M_0, M_1) \mid M_0 \in \mathcal{B}_0, M_1 \in \mathcal{B}_1\}$.
- Apply Theorem 1 to the net $(N(\mathcal{A}) \parallel N(\mathcal{A}))$ and the formulas ϕ_{II} and ϕ_{01}. □

6 The Fitting Problem is Decidable

We show that the fitting problem is Turing-reducible in elementary time to the reachability problem for Petri nets.

Theorem 4 *The fitting problem is Turing-reducible in elementary time to the reachability problem for Petri nets, and thus decidable.*

Proof Let $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ be a population protocol and let $\Pi\colon \mathrm{Pop}(\Sigma) \to \{0, 1\}$ be a Presburger predicate. We reduce the fitting problem to the (complement of the) reachability problem for Presburger sets of markings, and apply Theorem 1.

Recall that $\mathtt{I}(X, C)$ is a formula of Presburger arithmetic that holds iff $I(X) = C$, that is, if $C \in \mathrm{Pop}(Q)$ is the initial configuration for the input $X \in \mathrm{Pop}(\Sigma)$. We define the formulas

$$\mathtt{I}_1(C) = \exists X\colon \mathtt{I}(X, C) \wedge \Pi(X) \qquad \mathtt{I}_0(C) = \exists X\colon \mathtt{I}(X, C) \wedge \neg\Pi(X)$$

and the sets $\mathcal{I}_1, \mathcal{I}_0$ of configurations satisfying $\mathtt{I}_1, \mathtt{I}_0$. So $\mathcal{I}_1$ (resp. $\mathcal{I}_0$) is the set of initial configurations of $\mathcal{A}$ corresponding to the inputs that satisfy Π (resp. do not satisfy Π). Clearly, both sets are Presburger-definable.

Let $\mathcal{B}$, $\mathcal{B}_0$, and $\mathcal{B}_1$ as in Definition 1. We claim that $(\mathcal{A}, \mathtt{I}, 0)$ computes Π iff $\mathcal{B} \setminus \mathcal{B}_0$ is not reachable from $\mathcal{I}_0$, and $\mathcal{B} \setminus \mathcal{B}_1$ is not reachable from $\mathcal{I}_1$. By Lemma 2 and the definition of $\mathcal{B}_0$ and $\mathcal{B}_1$, a fair computation stabilizes to $b \in \{0, 1\}$ iff it gets trapped in a bottom SCC contained in $\mathcal{B}_b$. Therefore, $\mathcal{B} \setminus \mathcal{B}_b$ is not reachable from $\mathcal{I}_b$ iff every fair computation from $\mathcal{I}_b$ stabilizes to b. This proves the claim.

Let $N(\mathcal{A})$ be the Petri net associated to $\mathcal{A}$. By the claim, $(\mathcal{A}, \mathtt{I}, 0)$ does not compute Π iff some marking of $\mathcal{B} \setminus \mathcal{B}_0$ is reachable in $N(\mathcal{A})$ from some marking of $\mathcal{I}_0$, or some marking of $\mathcal{B} \setminus \mathcal{B}_1$ is reachable in $N(\mathcal{A})$ from some marking of $\mathcal{I}_1$. Since, by Proposition 1, $\mathcal{B}$, $\mathcal{B}_0$ and $\mathcal{B}_1$ are Presburger sets, and computable in elementary time, so are $\mathcal{B} \setminus \mathcal{B}_0$ and $\mathcal{B} \setminus \mathcal{B}_1$. So the fitting problem reduces to in elementary time to (the complements of) two instances of the reachability problem for Presburger sets. □

7 Lower Bounds for the Well-Specification and Fitting Problems

We show that the reachability problem for Petri nets can be reduced to the complements of the well-specification and fitting problems.

Theorem 5 *The reachability problem for Petri nets is polynomially reducible to ill-specification problem and to the complement of the fitting problem for population protocols (in both cases even with simple output mappings).*

Proof We proceed by means of a sequence of reductions. First, the reachability problem for Petri nets can be reduced in polynomial time to the *single-place zero-reachability problem* [14]:

Given: a Petri net $N = (P, T, F)$, a marking $M_0 \in \mathbb{N}^P$, and a distinguished place $z \in P$.
Decide: Is there a marking M reachable from M_0 such that $M(z) = 0$?

We claim that this problem can in turn be reduced in polynomial time to the problem for the special case in which, additionally,
(a) $M_0(z) > 0$,
(b) N with M_0 as initial marking is deadlock-free,
(c) no two transitions of N have the same input and output places (i.e., if ${}^\bullet t_1 = {}^\bullet t_2$ and $t_1{}^\bullet = t_2{}^\bullet$ then $t_1 = t_2$),
(d) the range of the flow function F is $\{0, 1\}$, and
(e) every transition t of N satisfies $1 \le |{}^\bullet t| \le 2$ and $1 \le |t^\bullet| \le 2$,

Proof of the claim. For (a): clear, because problem instances with $M_0(z) = 0$ are trivial (take $M = M_0$). For (b), if N is not deadlock-free we just add a new place p_0, initially marked, and a new transition t_0 with ${}^\bullet t_0 = t_0{}^\bullet = \mathbf{p}_0$. For (c), if there are several transitions with the same input and output places, we can safely remove all but one, without changing the set of reachable markings. For (d) and (e), it suffices to replace every transition of N non-complying with the conditions by an adequate "gadget" that simulates the firing of a transition t one place at a time: first the places of ${}^\bullet t$ then those of $t^\bullet$. Within ${}^\bullet t$ and $t^\bullet$, the ordering is given by a total order $\prec$ on the places of P such that z is the largest place in this ordering. Figure 2 illustrates the gadget construction (bottom) given the original transition (top); the reader can easily guess the general definition. It is important that the "gadget" simulates the firing of a transition t going through the places of ${}^\bullet t \cup t^\bullet$ in the order given by

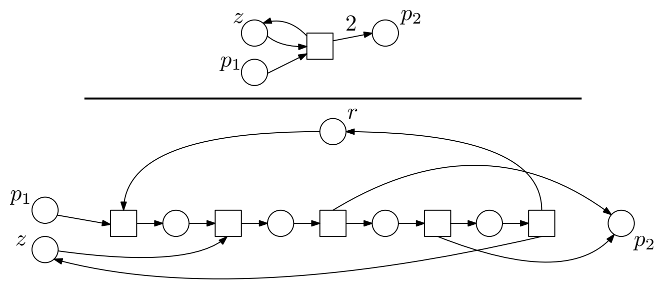

Fig. 2 Above the original transition, below its gadget. We assume $p_1 \prec p_2 \prec z$.

$\prec$. The place r is shared by all the gadgets of all transitions. Loosely speaking, r guarantees that at most one gadget is active at a time.

Let $N' = (P', T', F')$ be the result of performing this transformation. We have $P' = P \cup P_{aux}$, where P_{aux} are the auxiliary places used in the gadgets. Let $M'_0 \in \mathbb{N}^{P'}$ be such that $M'_0 = M_0 + \mathbf{r}$. The following two properties are easy to prove:

(1) The reachable markings of N and the projections onto P of the reachable markings of N' that put one token in the special place r coincide.
Loosely speaking, the net N' simulates the firings of transitions of N by executing the corresponding gadget. If N' tries to simulate the firing of a transition of N that is not currently enabled, then the gadget cannot execute and N' reaches a deadlock. The markings of N' with one token in r are those in which every execution of a gadget could be successfully completed.

(2) If some reachable marking M' of N' satisfies $M'(z) = 0$ then some reachable marking M'' satisfies $M''(z) = 0$ and $M''(r) = 1$.
Indeed, since z is the largest place among the input places of any transition, any reachable marking M' with $M'(z) = 0$ is a marking in which the first part of the gadget for some output transition of z has been successfully executed. The marking M'' is reached by completing the execution of the gadget.

It follows easily from (1) and (2) that N has a reachable marking M such that $M(z) = 0$ iff N' has a reachable marking M' such that $M'(z) = 0$.

Proof of the theorem. Given an instance of the single-place zero-reachability problem satisfying the additional constraints above, we construct a population protocol $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ with Presburger input mapping. We first describe the protocol scheme $\mathcal{A} = (Q, \Delta)$. The set Q of states of the protocol contains

– a state q_p for every place $p \in P$;
– a state q_t for every transition $t \in T$; and
– two states *Source* and *Sink*.

Let $Q_z = \{q_z\} \cup \{q_t \mid F(z,t) = 1\}$ The output mapping O, which is simple, is given by the partition Q_0, Q_1 with $Q_0 = Q_z \cup \{Sink\}$. The input mapping $I\colon \text{Pop}(\Sigma) \to \text{Pop}(Q)$ is defined as follows. The set Σ is a singleton $\{\sigma\}$, and I assigns to the number n—a population of Pop($\{\sigma\}$)— the configuration that puts

– n agents in *Source*;
– $M_0(p)$ agents in q_p for every place p; and
– 0 agents elsewhere.

Observe that I is a Presburger mapping.

We now describe the transitions of the protocol. By condition (c), we can identify a Petri net transition t and the pair $({}^\bullet t, t^\bullet)$, and so we use the notation $t = ({}^\bullet t, t^\bullet)$. Following (e), we define the set Δ of protocol transitions as follows:

(1) for every Petri net transition $t = (\{p_1, p_2\}, \{p_3, p_4\})$, two protocol transitions

$$(q_{p_1}, q_{p_2}) \;\mapsto\; (q_t, \mathit{Sink}) \quad \text{and} \quad \delta_t := (q_t, \mathit{Source}) \;\mapsto\; (q_{p_3}, q_{p_4})$$

(we call the second transition δ_t, for future reference).

(2) for every Petri net transition $t = (\{p_1, p_2\}, \{p_3\})$, two protocol transitions

$$(q_{p_1}, q_{p_2}) \;\mapsto\; (q_t, \mathit{Sink}) \quad \text{and} \quad \delta_t := (q_t, \mathit{Source}) \;\mapsto\; (q_{p_3}, \mathit{Sink})$$

(3) for every Petri net transition $t = (\{p_1\}, \{p_2, p_3\})$, one protocol transition

$$\delta_t := (q_{p_1}, \mathit{Source}) \;\mapsto\; (q_{p_2}, q_{p_3})$$

(4) for every Petri net transition $t = (\{p_1\}, \{p_2\})$, one protocol transition

$$\delta_t := (q_{p_1}, \mathit{Source}) \;\mapsto\; (q_{p_2}, \mathit{Sink})$$

(5) for every $q' \in Q_1$, and every $q \in Q_z$, a protocol transition

$$(q', q) \;\mapsto\; (\mathit{Sink}, q) \ .$$

The transitions of (1)–(4) simulate the firing of the Petri net transition t (in the case of transitions in (1)–(2), firing t is simulated by the occurrence, one after the other, of two protocol transitions). Observe that the simulation of a transition t of type (1) can "get stuck": after the occurrence of $(q_{p_1}, q_{p_2}) \;\mapsto\; (q_t, \mathit{Sink})$ there may be no agent in *Source*, and then $(q_t, \mathit{Source}) \;\mapsto\; (q_{p_3}, q_{p_4})$ cannot occur. This is also true for the transitions of type (2).

Intuitively, the transitions of (5) can move agents from any state in Q_1 to *Sink* ($\in Q_0$) "as long as there is at least one token in z". This is the case if there is at least one agent in q_z, or if the simulation of a transition $\{q_t \mid F(z,t) = 1\}$ has got stuck, and so has not been completed. Observe that this is the case if there is at least one agent in Q_z.

In all cases, simulating the firing of t requires one agent to leave the *Source* state. Since, moreover, no agents ever enter *Source*, each execution of $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ contains only finitely many occurrences of transitions of (1)–(4). Further, since every transition of (5) moves an agent to *Sink*, and no agents ever leave *Sink*, the transitions of (5) also occur only finitely often. Therefore all fair executions of $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ are finite.

Assume that some reachable marking M of N satisfies $M(z) = 0$. Let $\tau \in T^*$ be such that $M_0\,[\tau\rangle\, M$, and let k be the length of τ. Since $M_0(z) > 0$, we have $k > 0$. We claim that $\mathcal{A}$ has a fair (finite) execution from $I(k\,\sigma)$ that does not stabilize. Consider the execution that starts by simulating τ through transitions (1)–(4). At the end of this simulation the protocol reaches a configuration C such that $C(\mathit{Source}) = 0 = C(Q_z)$, and $C(\mathit{Sink}) > 0$ (this follows from $k > 0$). Since $C(\mathit{Source}) = 0$, no δ_t transition can occur from C, and so, by exhaustively executing transitions from (1)–(2), we reach a configuration C' that that does not enable any transition of (1)–(4), still satisfies $C'(\mathit{Source}) = 0 = C'(Q_z)$ and $C'(\mathit{Sink}) > 0$, and further satisfies $C'(q_p) + \sum_{t \mid F(p,t)=1} C'(q_t) > 0$ (because transitions of (1)–(2) may have moved the agents in q_p to states q_t with $F(p,t) = 1$). Since $C'(Q_z) = 0$, the configuration C' does not enable any transition of (5) either. So the execution cannot be extended, hence it is fair. Since $\mathit{Sink} \in Q_0$ and $\{q_p\} \cup \{q_t \mid F(p,t) = 1\} \subseteq Q_1$, and both are occupied by agents in C', we have $O(C') = \bot$. So $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ is ill specified.

Assume now that every reachable marking M of N satisfies $M(z) > 0$. We prove that every fair execution stabilizes to 0. Since all fair executions of $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ are finite, given a fair execution $C_0 C_1 \dots C_n$ we have to prove $O(C_n) = 0$ or, equivalently, $C_n(q) = 0$ for every $q \in Q_1$. Let $\sigma = \delta_1, \dots, \delta_n$ be the sequence of protocol transitions such that $C_0 \xrightarrow{\delta_1} C_1 \cdots C_{n-1} \xrightarrow{\delta_n} C_n$, and let $\delta_{t_1} \cdots \delta_{t_j}$ be the projection of σ onto the set of transitions $\{\delta_t \mid t \in T\}$. It follows from the definition of $\mathcal{A}$ that $t_1 \dots t_j$ is a firing sequence of N from the marking M_0. Let M be the marking of N reached by firing this sequence. By hypothesis we have $M(z) > 0$. Let C_j be the configuration reached after executing the protocol transition δ_{t_j} in σ. Since C_j corresponds to the marking M, we have $C_j(q_z) > 0$. Since no transition of $\{\delta_t \mid t \in T\}$ occurs between C_j and C_n, we have $C_n(Q_z) > 0$. Finally, since by hypothesis the execution $C_0 C_1 \dots C_n$ cannot be extended, C_n does not enable any transition of (5), and so $C_n(Q_1) = 0$.

The same reduction shows hardness for the complement of the fitting problem for the predicate *false*. □

8 An Algorithm for the Tailor Problem

We present an algorithm for the tailor problem (given a well-specified population protocol obtain a Presburger formula for the predicate it computes), based on the notion of *certificates*. A certificate of a protocol $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ is a string x satisfying certain conditions, specified in Section 8.1 below. After defining certificates, we prove the following properties:

(1) If a protocol has a certificate, then it is well specified. Moreover, there is an algorithm that, given a protocol and a certificate, returns a Presburger formula for the predicate computed by the protocol.

(2) There is an algorithm that, given a protocol and a string x, decides if x is a certificate of the protocol.

(3) If a protocol is well specified, then it has a certificate.

These properties immediately lead to an algorithm for the tailor problem: enumerate all strings x; check if x is a certificate using property (2); if x is a certificate, compute a formula for the predicate computed by the protocol using property (1). The termination of the algorithm is ensured by property (3).

After defining certificates in Section 8.1, properties (1)–(3) are proved in in three different sections. Property (3) has the most involved proof, and requires to introduce some further results from the theory of Petri nets.

8.1 Certificates

We need some definitions and notations. Let $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ be a population protocol.

- A configuration C of $\mathcal{A}$ is a 0-*configuration* (resp. 1-*configuration*) if $O(C) = 0$ (resp. $O(C) = 1$).
- A set $\mathcal{C}$ of configurations of $\mathcal{A}$ is *inductive* if $C \in \mathcal{C}$ and $C \rightarrow C'$ implies $C' \in \mathcal{C}$.
- Given a language $W \subseteq \Delta^*$ and a set $\mathcal{C}$ of configurations, $pre_{\mathcal{A}}(\mathcal{C}, W)$ denotes the set of configurations C such that $C \xrightarrow{w} C'$ for some word $w \in W$ and some $C' \in \mathcal{C}$. We write $pre^*_{\mathcal{A}}(\mathcal{C})$ to denote $pre_{\mathcal{A}}(\mathcal{C}, \Delta^*)$. The definitions for $post_{\mathcal{A}}(\mathcal{C}, W)$ and $post^*_{\mathcal{A}}(\mathcal{C})$ are as expected.

Definition 2 Let $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ be a population protocol such that $\mathcal{A} = (Q, \Delta)$. A *certificate* for the population protocol $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ is a tuple $(\mathtt{S}_0, \mathtt{S}_1, \mathtt{D}_0, \mathtt{D}_1, w_1, \ldots, w_k)$, where $\mathtt{S}_0, \mathtt{S}_1, \mathtt{D}_0, \mathtt{D}_1$ are predicates in Presburger arithmetic denoting Presburger sets of configurations $\mathcal{S}_0, \mathcal{S}_1, \mathcal{D}_0, \mathcal{D}_1$, and $w_1, \ldots, w_k$ are words in Δ^* denoting the language $W = w_1^* \ldots w_k^*$, such that:
(1) $\mathcal{S}_0, \mathcal{S}_1, \mathcal{D}_0, \mathcal{D}_1$ are inductive.
(2) The pair $(\mathcal{I}_0, \mathcal{I}_1)$, where $\mathcal{I}_0 = \mathcal{S}_0 \cap \mathcal{I}$ and $\mathcal{I}_1 = \mathcal{S}_1 \cap \mathcal{I}$, is a partition of $\mathcal{I}$.
(3) $\mathcal{D}_0$ is a set of 0-configurations such that $\mathcal{S}_0 \subseteq pre_{\mathcal{A}}(\mathcal{D}_0, W)$.
(4) $\mathcal{D}_1$ is a set of 1-configurations such that $\mathcal{S}_1 \subseteq pre_{\mathcal{A}}(\mathcal{D}_1, W)$.

Observe that, by condition (2), all initial configurations belong to $\mathcal{S}_0 \cup \mathcal{S}_1$. So, by condition (1), $\mathcal{S}_0 \cup \mathcal{S}_1$ contains all configurations reachable from initial configurations. Condition (3) ensures that from every configuration of $\mathcal{S}_0$ one can reach and get trapped in a set of 0-configurations (because $\mathcal{D}_0$ is inductive); condition (4) is a similar property for 1-configurations.

8.2 Certificates Ensure Well-Specification

We show that if a protocol has a certificate, then it is well specified. Moreover, a Presburger formula for the predicate computed by the protocol can be easily extracted from the certificate.

Lemma 4 *If a population protocol* $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ *has a certificate* $(\mathtt{S}_0, \mathtt{S}_1, \mathtt{D}_0, \mathtt{D}_1, w_1, \ldots, w_k)$, *then the protocol is well specified and computes the predicate* $\Pi\colon \mathrm{Pop}(\Sigma) \to \{0, 1\}$ *defined as follows:*

$$\Pi(X) = \begin{cases} 0 & \textit{if } \exists C\colon \mathtt{I}(X, C) \wedge \mathtt{S}_0(C) \\ 1 & \textit{if } \exists C\colon \mathtt{I}(X, C) \wedge \mathtt{S}_1(C)\ . \end{cases}$$

In particular, the algorithm that given a protocol and a certificate outputs the formula $\exists C\colon \mathtt{I}(X, C) \wedge \mathtt{S}_0(C)$ *yields a correct solution to the tailor problem.*

Proof Let $\mathcal{S}_0, \mathcal{S}_1, \mathcal{D}_0, \mathcal{D}_1$ be the Presburger sets of configurations denoted by $\mathtt{S}_0, \mathtt{S}_1, \mathtt{D}_0, \mathtt{D}_1$, respectively. Let $W = w_1^* \ldots w_k^*$. Since $\mathcal{I}_0$ and $\mathcal{I}_1$ form a partition of $\mathcal{I}$, it suffices to prove that every fair execution starting at $\mathcal{I}_b$ stabilizes to b. Let $C \in \mathcal{I}_b$ and let $C_0, C_1, \ldots$ be a fair execution starting at C. By Lemma 2 the execution gets trapped in a bottom SCC. Hence, there exists $n \in \mathbb{N}$ such that C_n is a bottom configuration. As $\mathcal{S}_b$ is inductive, it follows that $C_n \in \mathcal{S}_b$. Moreover, as $\mathcal{S}_b \subseteq pre_{\mathcal{A}}(\mathcal{D}_b, W)$, there exists a word $w \in W$ and a configuration $C' \in \mathcal{D}_b$ such that $C_n \xrightarrow{w} C'$. Since C_n is a bottom configuration, there exists a word $w' \in \Delta^*$ such that $C' \xrightarrow{w'} C_n$. Now, let $m \geq n$. Since C_m is reachable from C_n, it follows that C_m is reachable from C'. As $C' \in \mathcal{D}_b$ and $\mathcal{D}_b$ is inductive, it follows that $C_m \in \mathcal{D}_b$. As $\mathcal{D}_b$ is a set of b-configurations, it follows that $O(C_m) = b$; thus, the execution stabilizes to b. □

Example 8 (Certificate for the parity predicate.) We describe a population protocol and show with the help of a certificate that it computes a given predicate. In the following $b \in \{0, 1\}$.

Let $\Sigma = \{\sigma\}$. Abusing language, we identify the mapping $X\colon \mathrm{Pop}(\Sigma) \to \mathbb{N}$ given by $X(\sigma) = n$ with the number n. The *parity predicate* $\Pi\ :\ \mathrm{Pop}(\Sigma) \to \{0, 1\}$ is given by $\Pi(n) = 0$ if n is even, and $\Pi(n) = 1$ otherwise.

The protocol $(\mathcal{A}, \mathtt{I}, \mathtt{O})$, where $\mathcal{A} = (Q, \Delta)$, is defined as follows:

- $Q = \{A_0, A_1, P_0, P_1\}$. Agents in $\{A_0, A_1\}$ are *active*, and those in $\{P_0, P_1\}$ are *passive*. Further, agents in $\{A_b, P_b\}$ *carry (the value) b*.
- $\Delta = \{\delta_{x,y}, \delta_x \mid x, y \in \{0, 1\}\}$, where

$$\delta_{x,y} \;=\; (A_x, A_y) \mapsto (A_{x+y}, P_{x+y}) \quad \text{and} \quad \delta_x \;=\; (A_x, P_{1-x}) \mapsto (A_x, P_x)\ .$$

  Intuitively, in $\delta_{x,y}$ two active agents add their values modulo 2, and one of them becomes passive; in δ_x an active agent changes the value of a passive agent.
- $I(n) \;=\; n\mathbf{A}_1$ for every $n \in \mathbb{N}$. That is, to compute the parity of n the protocol starts with n active agents carrying 1 (n agents in state A_1, and no agents elsewhere).
- $O(C) \;=\; b$ if $Sup(C) \subseteq \{A_b, P_b\}$, and $O(C) \;=\; \bot$ otherwise. That is, a configuration has output $b \in \{0, 1\}$ if currently all agents carry b, otherwise it has output $\bot$.
  We provide a certificate of the fact that the protocol computes Π.
- $\mathtt{D}_b(C) := (C(A_b) = 1 \land C(A_{1-b}) = 0 \land C(P_{1-b}) = 0)$.
  Notice that the set of configurations $\mathcal{D}_b$ denoted by $\mathtt{D}_b$ is inductive. In fact, since configurations of $\mathcal{D}_b$ only have one active agent, and all their agents carry the same value b, they enable no transitions.
- $\mathtt{S}_0(C)$ and $\mathtt{S}_1(C)$ are Presburger formulas for "$C(A_1)$ is even" and "$C(A_1)$ is odd". Inspection of Δ immediately shows that the sets $\mathcal{S}_0$ and $\mathcal{S}_1$ denoted by $\mathtt{S}_0(C)$ and $\mathtt{S}_1(C)$ are inductive. Notice that $\mathcal{I} \cap \mathcal{S}_0$ and $\mathcal{I} \cap \mathcal{S}_1$ is a partition of $\mathcal{I}$.
- $W = \delta_{1,1}^* \; \delta_{0,0}^* \; \delta_{1,0}^* \; \delta_0^* \; \delta_1^*$.
  W models a strategy to reach $\mathcal{D}_0 \cup \mathcal{D}_1$ from any configuration. First execute the transition $\delta_{1,1}$ as long as possible, until there is at most one active agent carrying a 1. Then execute $\delta_{0,0}$ as long as possible, until there is at most one active agent carrying a 0. Then execute $\delta_{1,0}$ if possible, reaching a configuration with exactly one active agent carrying a value b. Finally, execute δ_0 as long as possible, followed by δ_1 as long as possible, leading to a configuration in which every passive agent also carries the value b. □

8.3 Checking Certificates

Using a result of one of the authors [18], we show that the problem of checking if a given tuple is a certificate reduces to the problem of checking if a closed formula of Presburger arithmetic is true, and so decidable.

Lemma 5 *Given a protocol* $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ *and a tuple* $(\mathtt{S}_0, \mathtt{S}_1, \mathtt{D}_0, \mathtt{D}_1, w_1, \ldots, w_k)$*, it is decidable whether the tuple is a certificate of the protocol.*

Proof We show that conditions (1)–(4) of Definition 2 can be effectively expressed in Presburger arithmetic. For (1), a set $\mathcal{M}$ of configurations denoted by a predicate $\mathtt{M}(C)$ in Presburger arithmetic is inductive iff the following Presburger formula is valid:

$$\forall C, C' \colon (\mathtt{M}(C) \land C \to C') \Rightarrow \mathtt{M}(C')\ .$$

So the inductiveness of $\mathcal{S}_0, \mathcal{S}_1, \mathcal{D}_0, \mathcal{D}_1$ is expressible. For (2), $(\mathcal{I}_0, \mathcal{I}_1)$ is a partition of $\mathcal{I}$ iff

$$\forall C \colon (\exists X \colon \mathtt{I}(X, C)) \Leftrightarrow ((\mathtt{I}_0(C) \land \neg \mathtt{I}_1(C)) \lor (\neg \mathtt{I}_0(C) \land \mathtt{I}_1(C)))$$

is valid, where $\mathtt{I}_b(C) = (\exists X \colon \mathtt{I}(X, C)) \land \mathtt{S}_b(C)$. For (3-4), $\mathcal{D}_b$ is a set of b-configurations iff

$$\forall C \colon \mathtt{D}_b(C) \Rightarrow \mathtt{O}(C, b)$$

is valid. It remains to express $\mathcal{S}_b \subseteq pre_{\mathcal{A}}(\mathcal{D}_b, W)$. Observe that for every word $w \in \Delta^*$, the relation $\xrightarrow{w^*}$ defined by $C \xrightarrow{w^*} C'$ if $C \xrightarrow{w^n} C'$ for some $n \in \mathbb{N}$ is effectively definable in Presburger arithmetic. (For $w = \delta$, where $\delta = (q_1, q_2) \mapsto (q_1', q_2')$, this follows easily from $C' = C - (\mathbf{q}_1 + \mathbf{q}_2) + (\mathbf{q}_1' + \mathbf{q}_2')$. For the general case, see the reference [18].) So the inclusion holds iff

$$\forall C_0\colon (\mathtt{S}_b(C_0) \Rightarrow \exists C_1, \ldots, C_k\colon C_0 \xrightarrow{w_1^*} C_1 \cdots \xrightarrow{w_k^*} C_k \ \wedge \mathtt{D}_b(C_k))$$

is valid. □

8.4 Every Well-Specified Protocol has a Certificate

We prove that every well-specified protocol has a certificate.

Let $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ be a well-specified protocol. Let $\mathcal{I}_0$ and $\mathcal{I}_1$ be the subsets of initial configurations for which the protocol computes 0 and 1, respectively. Since the protocol is well specified, the pair $(\mathcal{I}_0, \mathcal{I}_1)$ is a partition of $\mathcal{I}$.

We choose $\mathtt{D}_0$ and $\mathtt{D}_1$ as Presburger formulas denoting the sets $\mathcal{B}_0$ and $\mathcal{B}_1$ of $\mathcal{A}$, as defined in Definition 1. These formulas exist and can be computed by Proposition 1, which shows that $\mathcal{B}_0$ and $\mathcal{B}_1$ are effectively Presburger. Observe that, with this choice, $\mathcal{D}_b$ is a set of b-configurations. Moreover, since any configuration reachable from a bottom configuration is also a bottom configuration, $\mathcal{D}_b$ is inductive.

Before choosing the sets $\mathcal{S}_0$ and $\mathcal{S}_1$, let us consider the tentative choice $\mathcal{S}_0' = post^*_{\mathcal{A}}(\mathcal{I}_0)$, and $\mathcal{S}_1' = post^*_{\mathcal{A}}(\mathcal{I}_1)$. The sets $\mathcal{S}_0'$ and $\mathcal{S}_1'$ are clearly inductive. Moreover, since the protocol is well specified, we have $\mathcal{S}_0' \cap \mathcal{I} = \mathcal{I}_0$ and $\mathcal{S}_1' \cap \mathcal{I} = \mathcal{I}_1$. Indeed, since $(\mathcal{I}_0, \mathcal{I}_1)$ is a partition of $\mathcal{I}$, if $\mathcal{S}_0' \cap \mathcal{I} \supsetneq \mathcal{I}_0$ then $\mathcal{S}_0' \cap \mathcal{I}_1 \neq \emptyset$, and so there is a configuration with two fair computations stabilizing to 0 and to 1, contradicting the assumption that the protocol is well specified.

However, we still miss two important properties: $\mathcal{S}_0'$ and $\mathcal{S}_1'$ may not be Presburger sets, and there may be no language W satisfying conditions (3) and (4). At this point we get help from the following two results:

Theorem 6 **([4])** *If* $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ *is well specified, then* $\mathcal{I}_0$ *and* $\mathcal{I}_1$ *are Presburger sets.*

Theorem 7 **([17, Lemma 9.1])** *Let N be a Petri net, and let $\mathcal{M}$ and $\mathcal{M}'$ be Presburger sets of markings of N such that $post^*_N(\mathcal{M}) \cap \mathcal{M}' = \emptyset$. There exists a Presburger inductive set of markings $\mathcal{S}$ such that $\mathcal{M} \subseteq \mathcal{S}$ and $\mathcal{S} \cap \mathcal{M}' = \emptyset$.*

Applying Theorem 7 to $\mathcal{M} = \mathcal{I}_0$ and $\mathcal{M}' = \mathcal{I}_1$ (which are Presburger by Theorem 6) yields an inductive *and* Presburger set $\mathcal{S}_0 \supseteq \mathcal{S}_0'$ such that $\mathcal{S}_0 \cap \mathcal{I}_1 = \emptyset$, and therefore $\mathcal{S}_0 \cap \mathcal{I}_1 = \mathcal{I}_0$. Similarly, applying the theorem to $\mathcal{M} = \mathcal{I}_1$ and $\mathcal{M}' = \mathcal{I}_0$, we obtain a corresponding set $\mathcal{S}_1$.

The existence of the bounded language W follows directly from another result of net theory:

Theorem 8 **([18, Corollary XI.3])** *For every Petri net $N = (P, T, F)$ and for every Presburger sets of markings $\mathcal{S}$ and $\mathcal{D}$ such that $\mathcal{S} \subseteq pre^*_N(\mathcal{D})$, there exists a sequence $w_1, \ldots, w_k$ of words in T^* such that the bounded language $W \subseteq w_1^* \ldots w_k^*$ satisfies $\mathcal{S} \subseteq pre_N(\mathcal{D}, W)$.*

Applying the theorem to $\mathcal{S}_0$ and $\mathcal{D}_0$ and to $\mathcal{S}_1$ and $\mathcal{D}_1$, we obtain two languages W_0, W_1. It then suffices to take $W = W_0 W_1$ since $W \supseteq W_0 \cup W_1$.

8.5 Well-Specified Protocols Compute Presburger Predicates: A New Proof

Angluin et al. have shown—a celebrated result—that well-specified population protocols compute exactly the Presburger-definable predicates [4]. The proof that every Presburger definable predicate is computed by some protocol profits from the fact that every formula of Presburger arithmetic is equivalent to a quantifier-free formula with divisibility predicates [11]. Using this result, it suffices to exhibit protocols computing some simple predicates, and prove that predicates computed by population protocols are closed under conjunction and disjunction, which is achieved by a rather straightforward product construction. The other direction, showing that population protocols can only compute Presburger predicates, is far more involved. We show that this direction follows from recent results of Petri net theory obtained by one of the authors. In fact, we slightly generalize the results of Angluin et al. [4], which hold for simple input and output mappings, to the more general Presburger mappings.

Let us first introduce some notations. The set of non-negative rational numbers is denoted by $\mathbb{Q}_{\geq 0}$. Vectors in $\mathbb{Q}^d_{\geq 0}$ and subsets of $\mathbb{Q}^d_{\geq 0}$ are denoted in bold face. Given two subsets $\mathbf{X}$ and $\mathbf{Y}$ of $\mathbb{Q}^d_{\geq 0}$, we write $\mathbf{X} + \mathbf{Y}$ for the set $\{\mathbf{x} + \mathbf{y} \mid (\mathbf{x}, \mathbf{y}) \in \mathbf{X} \times \mathbf{Y}\}$. Symmetrically, for a set $R \subseteq \mathbb{Q}_{\geq 0}$ and a set $\mathbf{X} \subseteq \mathbb{Q}^d_{\geq 0}$, we write $R\mathbf{X}$ for the set $\{r\mathbf{x} \mid (r, \mathbf{x}) \in R \times \mathbf{X}\}$. When R, $\mathbf{X}$, or $\mathbf{Y}$ are reduced to a singleton set $\{r\}$, $\{\mathbf{x}\}$ or $\{\mathbf{y}\}$, we simply denote $\mathbf{X} + \mathbf{Y}$ by $\mathbf{x} + \mathbf{Y}$ or $\mathbf{X} + \mathbf{y}$, and $R\mathbf{X}$ by $r\mathbf{X}$ or $R\mathbf{x}$.

A *conic set* is a subset $\mathbf{C}$ of $\mathbb{Q}^d_{\geq 0}$ containing $\mathbf{0}$ and satisfying $\mathbf{C} + \mathbf{C} \subseteq \mathbf{C}$ and $\mathbb{Q}_{\geq 0}\mathbf{C} \subseteq \mathbf{C}$. A subset $\mathbf{P}$ of $\mathbb{N}^d$ is *periodic* if $\mathbf{0} \in \mathbf{P}$ and $\mathbf{P} + \mathbf{P} \subseteq \mathbf{P}$. The periodic set $\mathbf{P}$ generated by a subset $\mathbf{S}$ of $\mathbb{N}^d$ is the set $\{\mathbf{s}_1 + \cdots + \mathbf{s}_k \mid k \in \mathbb{N},\ \mathbf{s}_1, \ldots, \mathbf{s}_k \in \mathbf{S}\}$. Notice that $\mathbb{Q}_{\geq 0}\mathbf{P}$ is a conic set for every periodic set $\mathbf{P}$. A periodic set $\mathbf{P}$ is said to be *asymptotically-definable* if the conic set $\mathbb{Q}_{\geq 0}\mathbf{P}$ is definable in $FO(\mathbb{Q}_{\geq 0}, +)$.

With these definitions, a *linear set* is a subset of $\mathbb{N}^d$ of the form $\mathbf{b} + \mathbf{P}$ where $\mathbf{b} \in \mathbb{N}^d$ and $\mathbf{P}$ is a periodic set generated by a finite set. An *almost-linear set* is a set of the form $\mathbf{b} + \mathbf{P}$ where $\mathbf{b} \in \mathbb{N}^d$ and $\mathbf{P}$ is an asymptotically-definable periodic set. Just like a *semi-linear set* is a finite union of linear sets, an *almost-semi-linear set* is a finite union of almost linear sets. The reachability sets of Petri nets are almost-semi-linear, as shown by the following theorem.

Theorem 9 ([17, Corollary 6.3]) *The sets $post^*_N(\mathcal{X}) \cap \mathcal{Y}$ and $pre^*_N(\mathcal{Y}) \cap \mathcal{X}$ are almost semi-linear for every Petri net N and for every Presburger sets of markings $\mathcal{X}, \mathcal{Y}$.*

Almost semi-linear sets can be approximated by Presburger sets as follows. The *linearization* of an almost-linear set $\mathbf{b} + \mathbf{P}$ is the semi-linear set $\mathbf{b} + [(\mathbf{P} - \mathbf{P}) \cap \mathbb{Q}_{\geq 0}\mathbf{P}]$ where $\mathbf{P} - \mathbf{P} = \{\mathbf{p} - \mathbf{q} \mid \mathbf{p}, \mathbf{q} \in \mathbf{P}\}$. Notice that a linearization of an almost semi-linear set is an over-approximation. Assume that $\mathbf{X}$ is an almost-semi-linear set of the form $\bigcup_{j=1}^{k} \mathbf{X}_j$ where $\mathbf{X}_j$ is an almost linear set. The semi-linear set $\bigcup_{j=1}^{k} \mathbf{S}_j$, where $\mathbf{S}_j$ is the linearization of $\mathbf{X}_j$, is called a *linearization* of $\mathbf{X}$. Since the decomposition of an almost semi-linear set into a finite union of almost-linear sets is not unique, an almost-semi-linear set may have multiple linearizations.

Linearizations are tight over-approximations of almost semi-linear sets. The tightness is captured in Lemma 6 below, which uses the notion of dimension. The *dimension* of a non-empty subset $\mathbf{S}$ of $\mathbb{N}^d$ is the minimal $r \in \mathbb{N}$ such that $\mathbf{S}$ is included in a semi-linear set $\bigcup_{j=1}^{k} \mathbf{b}_j + \mathbf{P}_j$ such that every $\mathbf{P}_j$ is a periodic set generated by at most r vectors. The dimension of the empty set is defined to be equal to -1. Intuitively, the lemma states that if two almost-semi-linear sets are disjoint, then their linearizations cannot have a “large” intersection: the dimension of the intersection must be strictly smaller than the dimension of at least one of the sets.

Lemma 6 ([17, Corollary 8.4]) *Let* $\mathbf{S}$ *and* $\mathbf{T}$ *be linearizations of non-empty, almost-semi-linear sets* $\mathbf{X}$ *and* $\mathbf{Y}$ *such that* $\mathbf{X} \cap \mathbf{Y} = \emptyset$*. The following relation holds:*

$$\dim(\mathbf{S} \cap \mathbf{T}) < \max\{\dim(\mathbf{X}), \dim(\mathbf{Y})\} \ .$$

A subset $\mathbf{X}$ of $\mathbb{N}^d$ is said to be *decomposable* if $\mathbf{X} \cap \mathbf{S}$ is almost semi-linear for every semi-linear set $\mathbf{S}$. It follows form Theorem 9 that reachability sets of Petri nets are decomposable. The following lemma shows that if the complement of a decomposable set is decomposable, then the set is semi-linear.

Lemma 7 *Disjoint decomposable sets* $\mathbf{X}, \mathbf{Y}$ *such that* $\mathbf{X} \cup \mathbf{Y}$ *is semi-linear are semi-linear.*

Proof Let us prove by induction on $r \in \mathbb{N}$ that for every semi-linear set $\mathbf{A}$ such that $\dim(\mathbf{A}) < r$ and for every partition $\mathbf{X}, \mathbf{Y}$ of $\mathbf{A}$ into decomposable sets, the sets $\mathbf{X}$ and $\mathbf{Y}$ are semi-linear. The case $r = 0$ is immediate since in this case $\mathbf{A}$ is empty. Assuming that the statement is true for r, let us prove it for $r + 1$. Consider a semi-linear set $\mathbf{A}$ such that $\dim(\mathbf{A}) = r$, and a partition $\mathbf{X}, \mathbf{Y}$ of $\mathbf{A}$ into decomposable sets. In particular $\mathbf{X}$ and $\mathbf{Y}$ are almost semi-linear. If $\mathbf{X}$ or $\mathbf{Y}$ is empty, then $\mathbf{X}$ and $\mathbf{Y}$ are semi-linear, and so we can assume that these two sets are non-empty. Let $\mathbf{S}$ and $\mathbf{T}$ be linearizations of $\mathbf{X}$ and $\mathbf{Y}$, respectively. Lemma 6 shows that $\dim(\mathbf{A}') < r$ where $\mathbf{A}'$ is the semi-linear set defined as the intersection $\mathbf{S} \cap \mathbf{T}$. We introduce the decomposable set $\mathbf{X}'$ and $\mathbf{Y}'$ defined as $\mathbf{X} \cap \mathbf{A}'$ and $\mathbf{Y} \cap \mathbf{A}'$. Notice that $\mathbf{X}', \mathbf{Y}'$ is a partition of $\mathbf{A}'$. By induction, it follows that $\mathbf{X}'$ and $\mathbf{Y}'$ are semi-linear. Now, just notice that $\mathbf{X} = (\mathbf{S} \backslash \mathbf{A}') \cup \mathbf{X}'$ and $\mathbf{Y} = (\mathbf{T} \backslash \mathbf{A}') \cup \mathbf{Y}'$. We derive that $\mathbf{X}$ and $\mathbf{Y}$ are semi-linear, and the induction is proved. □

We are now ready to prove our result:

Theorem 10 *For every Petri net N, and for every Presburger sets of markings $\mathcal{M}$, $\mathcal{F}_0, \mathcal{F}_1$: if $\mathcal{M}_0 = \mathcal{M} \cap pre_N^*(\mathcal{F}_0)$ and $\mathcal{M}_1 = \mathcal{M} \cap pre_N^*(\mathcal{F}_1)$ is a partition of $\mathcal{M}$, then $\mathcal{M}_0$ and $\mathcal{M}_1$ are Presburger sets.*

Proof From Theorem 9, it follows that $pre_N^*(\mathcal{F}_0)$ and $pre_N^*(\mathcal{F}_1)$ are decomposable sets. Thus $\mathcal{M}_0$ and $\mathcal{M}_1$ defined as $\mathcal{M} \cap pre_N^*(\mathcal{F}_0)$ and $\mathcal{M} \cap pre_N^*(\mathcal{F}_1)$ are decomposable. Since $\mathcal{M}_0 \cap \mathcal{M}_1 = \emptyset$ and $\mathcal{M}_0 \cup \mathcal{M}_1 = \mathcal{M}$, it follows from Lemma 7 that $\mathcal{M}_0$ and $\mathcal{M}_1$ are Presburger. □

Corollary 1 *Well-specified population protocols only compute Presburger predicates.*

Proof Let $(\mathcal{A}, \mathtt{I}, \mathtt{O})$ be a well-specified protocol. Then $\mathcal{I}$, $\mathcal{B}_0$, and $\mathcal{B}_1$ are Presburger sets. Applying Theorem 10 to $\mathcal{M} := \mathcal{I}, \mathcal{F}_0 := \mathcal{B}_0$, and $\mathcal{F}_1 := \mathcal{B}_1$, we obtain that $\mathcal{I} \cap pre_N^*(\mathcal{B}_0)$ and $\mathcal{I} \cap pre_N^*(\mathcal{B}_1)$ are Presburger sets. Since the protocol is well specified, each configuration of $\mathcal{I}$ can reach exactly one of $\mathcal{B}_0$ and $\mathcal{B}_1$, these two sets are equal to $\mathcal{I}_0$ and $\mathcal{I}_1$, respectively. So $\mathcal{I}_0$ and $\mathcal{I}_1$ are Presburger sets. □

9 Certificate-Based Algorithms for Well-Specification and Correctness

Certificates provide an alternative algorithm to decide the well-specification and fitting problems. If we apply our algorithm for the tailor problem to an arbitrary protocol, then two cases are possible: if the protocol is well specified, then the algorithm terminates and returns a

Presburger formula for the computed predicate. If the protocol is ill specified, then it has no certificate, and the algorithm does not terminate. In other words, our algorithm for the tailor problem is at the same time a semi-decision procedure for the well-specification problem.

In order to obtain a decision procedure, it suffices to run this semi-decision procedure for well-specification in parallel with a semi-decision procedure for ill-specification. But this second semi-decision procedure is easy to find. Recall that a protocol is ill specified if there is an input X and either
(1) a fair computation starting at the configuration $I(X)$ that does not stabilize, or
(2) two fair computations starting at$I(X)$, and stabilizing to opposite values.
The semi-decision procedure for ill-specification enumerates all inputs X, constructs for each of them the fragment of the reachability graph with root $I(X)$, which is finite by Lemma 1, and examines the bottom SCCs of this graph to decide if conditions (1) or (2) hold.

Since the semi-decision procedure for the well-specification problem returns a Presburger formula for the computed predicate, we can also use this combination of semi-decision procedures to solve the fitting problem: if the protocol is ill specified, then it does not fit any predicate; if the protocol is well specified, then we check whether the Presburger formulas for the intended predicate and the computed predicate are equivalent, which is a decidable problem.

Acknowledgements We thank the CONCUR reviewers for their insightful feedback.

References

1. P. A. Abdulla, K. Cerans, B. Jonsson, and Y.-K. Tsay. General decidability theorems for infinite-state systems. In *LICS '96: Proc. 11th Annual IEEE Symp. on Logic in Computer Science*, pages 313–321. IEEE Computer Society, 1996.
2. D. Angluin, J. Aspnes, Z. Diamadi, M. J. Fischer, and R. Peralta. Computation in networks of passively mobile finite-state sensors. In *PODC '04*, pages 290–299. ACM, 2004.
3. D. Angluin, J. Aspnes, and D. Eisenstat. Fast computation by population protocols with a leader. In *DISC '06*, volume 4167 of *LNCS*, pages 61–75. Springer, 2006.
4. D. Angluin, J. Aspnes, and D. Eisenstat. Stably computable predicates are semilinear. In *PODC '06*, pages 292–299. ACM, 2006.
5. K. R. Apt and D. C. Kozen. Limits for automatic verification of finite-state concurrent systems. *Information Processing Letters*, 22(6):307 – 309, 1986.
6. I. Chatzigiannakis, O. Michail, and P. G. Spirakis. Algorithmic verification of population protocols. In S. Dolev, J. A. Cobb, M. J. Fischer, and M. Yung, editors, *Stabilization, Safety, and Security of Distributed Systems - 12th International Symposium, SSS 2010, New York, NY, USA, September 20-22, 2010. Proceedings*, volume 6366 of *Lecture Notes in Computer Science*, pages 221–235. Springer, 2010.
7. J. Clement, C. Delporte-Gallet, H. Fauconnier, and M. Sighireanu. Guidelines for the verification of population protocols. In *ICDCS '11*, pages 215–224, 2011.
8. Y. Deng and J. Monin. Verifying self-stabilizing population protocols with coq. In W. Chin and S. Qin, editors, *TASE 2009, Third IEEE International Symposium on Theoretical Aspects of Software Engineering, 29-31 July 2009, Tianjin, China*, pages 201–208. IEEE Computer Society, 2009.
9. Z. Diamadi and M. J. Fischer. A simple game for the study of trust in distributed systems. *Wuhan University Journal of Natural Sciences*, 6(1–2):72–82, 2001.
10. S. Eilenberg and M. P. Schützenberger. Rational sets in commutative monoids. *Journal of Algebra*, 13(2):173–191, 1969.
11. H. B. Enderton. *A Mathematical introduction to logic*. Academic Press, 2001 San Diego London Toronto, 2001.
12. A. Finkel and P. Schnoebelen. Well-structured transition systems everywhere! *Theoretical Computer Science*, 256(1-2):63–92, 2001.
13. S. Ginsburg and E. H. Spanier. Semigroups, presburger formulas, and languages. *Pacific Journal of Mathematics*, 16(2):285–296, 1966.

14. M. H. T. Hack. Decidability questions for Petri nets. Technical Report 161, MIT, 1976.
15. J. Leroux. The general vector addition system reachability problem by presburger inductive invariants. In *LICS '09*, pages 4–13. IEEE Computer Society, 2009.
16. J. Leroux. Vector addition system reversible reachability problem. In *CONCUR '11*, volume 6901 of *LNCS*, pages 327–341. Springer, 2011.
17. J. Leroux. Vector addition systems reachability problem (a simpler solution). In *Turing-100: The Alan Turing Centenary Conference*, volume 10 of *EPiC Series*, pages 214–228. EasyChair, 2012.
18. J. Leroux. Presburger vector addition systems. In *LICS '13*, pages 23–32. IEEE Computer Society, 2013.
19. S. Navlakha and Z. Bar-Joseph. Distributed information processing in biological and computational systems. *Commun. ACM*, 58(1):94–102, Dec. 2014.
20. J. Pang, Z. Luo, and Y. Deng. On automatic verification of self-stabilizing population protocols. In *Second IEEE/IFIP International Symposium on Theoretical Aspects of Software Engineering, TASE 2008, June 17-19, 2008, Nanjing, China*, pages 185–192. IEEE Computer Society, 2008.
21. J. Sun, Y. Liu, J. S. Dong, and J. Pang. PAT: towards flexible verification under fairness. In A. Bouajjani and O. Maler, editors, *Computer Aided Verification, 21st International Conference, CAV 2009, Grenoble, France, June 26 - July 2, 2009. Proceedings*, volume 5643 of *Lecture Notes in Computer Science*, pages 709–714. Springer, 2009.