Abstract
Population protocols 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 in which the interacting devices are chosen uniformly at random.

In well designed population protocols, for every initial configuration of devices, and for every computation starting at this configuration, all devices eventually agree on a consensus value. We address the problem of automatically computing a parametric bound on the expected time the protocol needs to reach this consensus. We present the first algorithm that, when successful, outputs a function $f(n)$ such that the expected time to consensus is bound by $O(f(n))$, where $n$ is the number of devices executing the protocol. We experimentally show that our algorithm terminates and provides good bounds for many of the protocols found in the literature.

1 Introduction
Population protocols are a model of distributed computation in which agents with very limited computational resources randomly interact in pairs to perform computational tasks \[3, 4\]. They have been used as an abstract model of wireless networks, chemical reactions, and gene regulatory networks, and it has been shown that they can be implemented at molecular level (see, e.g., \[22, 20, 10, 19\]).
Population protocols compute by reaching a stable consensus in which all agents agree on a common output (typically a Boolean value). The output depends on the distribution of the initial states of the agents, called the initial configuration, and so a protocol computes a predicate that assigns a Boolean value to each initial configuration. For example, a protocol in which all agents start in the same state computes the predicate $x \geq c$ if the agents agree to output 1 when there are at least $c$ of them, and otherwise agree to output 0. A protocol with two initial states computes the majority predicate $x \geq y$ if the agents agree to output 1 exactly when the initial number of agents in the first state is greater than or equal to the initial number of agents in the second state.

In previous work, some authors have studied the automatic verification of population protocols. Since a protocol has a finite state space for each initial configuration, model checking algorithms can be used to verify that the protocol behaves correctly for a finite number of initial configurations. However, this technique cannot prove that the protocol is correct for every configuration. In [15], it was shown that the problem of deciding whether a protocol computes some predicate, and the problem of deciding whether it computes a given predicate, are both decidable and at least as hard as the reachability problem for Petri nets.

In practice, protocols should not only correctly compute a predicate, but also do it fast. The most studied quantitative measure is the expected number of pairwise interactions needed to reach a stable consensus. The measure is defined for the stoichiometric model in which the pair of agents of the next interaction are picked uniformly at random. A derived measure is the parallel time, defined as the number of interactions divided by the number of agents. The first paper on population protocols already showed that every predicate can be computed by a protocol with expected total number of interactions $O(n^2 \log n)$, where $n$ is the number of agents [3, 4]. Since then, there has been considerable interest in obtaining upper and lower bounds on the number of interactions for some fundamental tasks, like leader election and majority, and there is also much work on finding trade-offs between the speed of a protocol and its number of states (see, e.g., [13, 1, 6] and the references therein). However, none of these works addresses the verification problem: given a protocol, determine its expected number of interactions.

As in the qualitative case, probabilistic model checkers can be used to compute the expected number of interactions for a given configuration. Indeed, in this case the behaviour of the protocol is captured by a finite-state Markov chain, and the expected number of interactions can be computed as the expected number of steps until a bottom strongly connected component of the chain is reached. This was the path followed in [11], using the PRISM probabilistic model checker. However, as in the functional case, this technique cannot give a bound valid for every configuration.

This paper presents the first algorithm for the automatic computation of an upper bound on the expected number of interactions. The algorithm takes advantage of the hierarchical structure of population protocols where an initial configuration reaches a stable consensus by passing through finitely many “stages”. Entering a next stage corresponds to entering a configuration where some behavioral restrictions become permanent (for example, some interactions become permanently disabled, certain states will never be populated again, etc.). The algorithm automatically identifies such stages and computes a finite acyclic stage graph representing the protocol evolution. If all bottom stages of the graph correspond to stabilized configurations, the algorithm proceeds by deriving bounds for the expected number of interactions to move from one stage to the next, and computes a bound for the expected number of interactions by taking an “asymptotic maximum” of these bounds. In unsuitable cases, the resulting upper bound can be higher than the actual expected number.
of interactions. We report on an implementation of the algorithm and its application to case studies.

Related work. To the best of our knowledge, we present the first algorithm for the automatic quantitative verification of population protocols. In fact, even for sequential randomized programs, the automatic computation of the expected time is little studied. After the seminal work of Flajolet et al. in [16], there is recent work by Kaminski et al. [18] on the computation of expected runtimes using weakest preconditions, by Chatterje et al. on the automated analysis of recurrence relations for expected time [9], by Van Chan Ngo et al. [21] on the automated computation of bounded expectations using amortized resource analysis, and by Batz et al. [5, 21] on the computation of sampling times for Bayesian networks. These works are either not targeted to distributed systems like population protocols, or do not provide the same degree of automation as ours.

Structure of the paper. In Section 2 we introduce population protocols and a simple modal logic to reason about their behaviours. In Section 3, we introduce stage graphs and explain how they allow to prove upper bounds on the expected number of interactions of population protocols. We then give a dedicated algorithm for the computation of stage graphs in Section 4, analyze the bounds derived by this algorithm in Section 5, and report on experimental results in Section 6. Finally, we conclude in Section 7.

2 Population protocols

In this section, we introduce population protocols and their semantics. We assume familiarity with basic notions of probability theory, such as probability space, random variables, expected value, etc. When we say that some event happens almost surely, we mean that the probability of the event is equal to one. We use \( \mathbb{N} \) to denote the set of non-negative integers.

A population consists of \( n \) agents with states from a finite set \( Q = \{A, B, \ldots\} \) interacting according to a directed interaction graph \( G \) (without self-loops) over the agents. The interaction proceeds in a sequence of steps, where in each step an edge of the interaction graph is selected uniformly at random, and the states \( (A, B) \) of the two chosen agents are updated according to a transition function containing rules of the form \((C, D) \rightarrow (E, F)\). We assume that for each pair of states \((C, D)\), there is at least one rule \((C, D) \rightarrow (E, F)\). If there are several rules with the same left-hand side, then one is selected uniformly at random. The unique agent identifiers are not known to the agents and not used by the protocol.

Most of the population protocols studied for complete interaction graphs have a symmetric transition function where pairs \((A, B)\) and \((B, A)\) are updated in the same way. For the sake of simplicity, we restrict our attention to symmetric protocols. Then, the transitions can be

\footnote{All of the presented results can easily be extended to non-symmetric population protocols. The only technical difference is the way of evaluating/estimating the probability of executing a given transition}
written simply as $AB \rightarrow CD$, because the ordering of states before/after the $\rightarrow$ symbol is irrelevant. Formally, $AB$ and $CD$ are understood as elements of $Q^{(2)}$, i.e., multisets over $Q$ with precisely two elements.

**Definition 1.** A population protocol is a tuple $\mathcal{P} = (Q, T, \Sigma, I, O)$ where

- $Q$ is a non-empty finite set of states;
- $T : Q^{(2)} \times Q^{(2)}$ is a total transition relation;
- $\Sigma$ is a non-empty finite input alphabet;
- $I : \Sigma \rightarrow Q$ is the input function mapping input symbols to states,
- $O : Q \rightarrow \{0, 1\}$ is the output function.

We write $AB \rightarrow CD$ to indicate that $(AB, CD) \in T$. When defining the set $T$, we usually specify the outgoing transitions only for some subset of $Q^{(2)}$. For the other pairs $AB$, there (implicitly) exists a single idle transition $AB \rightarrow AB$. We also write $I(\Sigma)$ to denote the set $\{q \in Q \mid q = I(\sigma) \text{ for some } \sigma \in \Sigma\}$.

### 2.1 Executing population protocols

A transition $AB \rightarrow CD$ is enabled in a configuration $C$ if $C - 1_A - 1_B \geq 0$. A transition $AB \rightarrow CD$ enabled in $C$ can fire and thus produce a configuration $C' = C - 1_A - 1_B + 1_C + 1_D$.

The probability of executing a transition $AB \rightarrow CD$ enabled in $C$ is defined by

$$
\mathbb{P}[C, AB \rightarrow CD] = \begin{cases} 
\frac{C(A) \cdot (C(A) - 1)}{n^2 - n} \cdot |\{EF \in Q^{(2)} : AA \rightarrow EF\}| & \text{if } A = B, \\
\frac{2 \cdot C(A) \cdot C(B)}{n^2 - n} \cdot |\{EF \in Q^{(2)} : AB \rightarrow EF\}| & \text{if } A \neq B.
\end{cases}
$$

where $n$ is the size of $C$. Note that $2 \cdot C(A) \cdot C(B)$ is the number of directed edges connecting agents in states $A$ and $B$ (when $A \neq B$), and $n^2 - n$ is the total number of directed edges in a complete directed graph without self-loops with $n$ vertices. If a pair of agents in states $A$ and $B$ is selected, one of the outgoing transitions of $AB$ is chosen uniformly at random.

We write $C \rightarrow C'$ to indicate that $C'$ is obtained from $C$ by firing some transition, and we use $\mathbb{P}[C \rightarrow C']$ to denote the probability of executing a transition enabled in $C$ producing $C'$. Note that there can be several transitions enabled in $C$ producing $C'$, and $\mathbb{P}[C \rightarrow C']$ is the total probability of executing some of them.

An execution initiated in a given configuration $C$ is a finite sequence $C_0, \ldots, C_\ell$ of configurations such that $\ell \in \mathbb{N}$, $C_0 = C$, and $C_i \rightarrow C_{i+1}$ for all $i < \ell$. A configuration $C'$ is reachable from a configuration $C$ if there is an execution initiated in $C$ ending in $C'$. A run is an infinite sequence of configurations $\omega = C_0, C_1, \ldots$ such that every finite prefix of $\omega$ is an execution. The configuration $C_0$ of a run $\omega$ is also denoted by $\omega_i$. For a given execution $C_0, \ldots, C_\ell$, we use $\text{Run}(C_0, \ldots, C_\ell)$ to denote the set of all runs starting with $C_0, \ldots, C_\ell$.

For every configuration $C$, we define the probability space $(\text{Run}(C), \mathcal{F}, \mathbb{P}_C)$, where $\mathcal{F}$ is the $\sigma$-algebra generated by all $\text{Run}(C_0, \ldots, C_\ell)$ such that $C_0, \ldots, C_\ell$ is an execution initiated in $C$, and $\mathbb{P}_C$ is the unique probability measure satisfying $\mathbb{P}_C(\text{Run}(C_0, \ldots, C_\ell)) = \prod_{i=0}^{\ell} \mathbb{P}[C_i \rightarrow C_{i+1}]$.

### 2.2 A simple modal logic for population protocols

To specify properties of configurations, we use a qualitative variant of the branching-time logic EF. Let $AP = Q \cup \{A! \mid A \in Q\}$ such that there is a non-idle transition $AA \rightarrow BC$. in a given configuration.
The formulae of our qualitative logic are constructed in the following way, where \( a \) ranges over \( AP \cup \{ Out_0, Out_1 \} \):
\[
\varphi ::= a \mid \neg \varphi \mid \varphi_0 \land \varphi_1 \mid \Box \varphi \mid \Diamond \varphi.
\]

The semantics is defined inductively:
\[
\begin{align*}
C & \models A & \text{iff} & C(A) > 0, \\
C & \models A! & \text{iff} & C(A) = 1, \\
C & \models Out_0 & \text{iff} & O(A) = 0 \text{ for all } A \in Q \text{ such that } C(A) > 0, \\
C & \models Out_1 & \text{iff} & O(A) = 1 \text{ for all } A \in Q \text{ such that } C(A) > 0, \\
C & \models \neg \varphi & \text{iff} & C \not\models \varphi, \\
C & \models \varphi_0 \land \varphi_1 & \text{iff} & C \models \varphi_0 \text{ and } C \models \varphi_1, \\
C & \models \Box \varphi & \text{iff} & \mathbb{P}_C(\{ \omega \in Run(C) \mid \omega_i \models \varphi \text{ for all } i \in \mathbb{N} \}) = 1, \\
C & \models \Diamond \varphi & \text{iff} & \mathbb{P}_C(\{ \omega \in Run(C) \mid \omega_i \models \varphi \text{ for some } i \in \mathbb{N} \}) = 1.
\end{align*}
\]

Note that \( C \models \Box \varphi \) iff all configurations reachable from \( C \) satisfy \( \varphi \), and \( C \models \Diamond \varphi \) iff a run initiated in \( C \) visits a configuration satisfying \( \varphi \) almost surely (i.e., with probability one). We also use \( \top, \bot \), and other propositional connectives whose semantics is defined in the standard way. Furthermore, we occasionally interpret a given set of configurations \( B \) as a formula where \( C \models B \) iff \( C \in B \).

For every formula \( \varphi \), we define a random variable \( \text{Steps}_\varphi \), assigning to every run \( C_0, C_1, \ldots \) either the least \( \ell \in \mathbb{N} \) such that \( C_\ell \models \varphi \), or \( \infty \) if there is no such \( \ell \). For a given configuration \( C \), we use \( \mathbb{E}_C[\text{Steps}_\varphi] \) to denote the expected value of \( \text{Steps}_\varphi \) in the probability space \( (\text{Run}(C), \mathcal{F}, \mathbb{P}_C) \).

### 2.3 Computable predicates, interaction complexity

Every input \( X \in \mathbb{N}^\Sigma \) is mapped to the configuration \( C_X \) such that
\[
C_X(q) = \sum_{\sigma \in \Sigma \text{ s.t. } I(\sigma) = q} X(\sigma) \quad \text{for every } q \in Q.
\]

An initial configuration is a configuration of the form \( C_X \) where \( X \) is an input. A configuration \( C \) is stable if \( C \models \text{Stable} \), where \( \text{Stable} \equiv (\Box \text{Out}_0) \lor (\Box \text{Out}_1) \). We say that a protocol \( P \) terminates if \( C \models \Diamond \text{Stable} \) for every initial configuration \( C \). A protocol \( P \) computes a unary predicate \( A \) on inputs if it terminates and every stable configuration \( C' \) reachable from an initial configuration \( C_X \) satisfies \( C' \models \text{Out}_x \), where \( x \) is either 1 or 0 depending on whether \( X \) satisfies \( A \) or not, respectively.

The interaction complexity of \( P \) is a function \( \text{InterComplexity}_P \) assigning to every \( n \geq 1 \) the maximal \( \mathbb{E}_C[\text{Steps}_\text{Stable}] \), where \( C \) ranges over all initial configurations of size \( n \). Since several interactions may be running in parallel, the time complexity of \( P \) is defined as \( \text{InterComplexity}_P(n) \) divided by \( n \). Hence, asymptotic bounds on interaction complexity immediately induce the corresponding bounds on time complexity.

### 2.4 Running examples

A well-studied predicate for population protocols is \( \text{majority} \). Here, \( \Sigma = \{ A, B \} \), \( I(A) = A \), \( I(B) = B \), and the protocol computes whether there are at least as many agents in state \( B \) as there are in state \( A \). As running examples, we use two different protocols for computing majority, taken from [14] and [17].
Automatic Analysis of Expected Termination Time for Population Protocols

Example 2 (majority protocol of [14]). We have that $Q = \{A, B, a, b\}$, $O(A) = O(a) = 0$, $O(B) = O(b) = 1$, and the transitions are the following: $AB \rightarrow ab$, $Ab \rightarrow Aa$, $Ba \rightarrow Bb$ and $ba \rightarrow bb$.

Example 3 (majority protocol of [17]). Here, $Q = \{A, B, C, a, b\}$, $O(A) = O(a) = 0$, $O(B) = O(b) = O(C) = 1$, and the transitions are the following: $AB \rightarrow bC$, $AC \rightarrow Aa$, $BC \rightarrow Bb$, $Ba \rightarrow Bb$, $Ab \rightarrow Aa$ and $Ca \rightarrow Cb$.

Stages of population protocols

Most of the existing population protocols are designed so that each initial configuration passes through finitely many “stages” before reaching a stable configuration. Entering a next stage corresponds to performing some additional non-reversible changes in the structure of configurations. Hence, the transition relation between stages is acyclic, and each configuration in a non-terminal stage eventually enters one of the successor stages with probability one. This intuition is formalized in our next definition.

Definition 4. Let $P = (Q, T, \Sigma, I, O)$ be a population protocol. A stage graph for $P$ is a triple $G = (\mathcal{S}, \rightarrow, [\cdot])$ where $\mathcal{S}$ is a finite set of stages, $\rightarrow \subseteq \mathcal{S} \times \mathcal{S}$ is an acyclic transition relation, and $[\cdot]$ is a function assigning to each $S \in \mathcal{S}$ a set of configurations $[S]$ such that the following conditions are satisfied:

(a) For every initial configuration $C$ there is some $S \in \mathcal{S}$ such that $C \in [S]$.
(b) For every $S \in \mathcal{S}$ with at least one successor under $\rightarrow$, and for every $C \in [S]$, we have that $C \models \Diamond \text{Term}(S)$, where $\text{Term}(S) \equiv \bigvee_{S \rightarrow S'} [S']$.

Note that a stage graph for $P$ is not determined uniquely. Even a trivial graph with one stage $S$ and no transitions such that $[S]$ is the set of all configurations is a valid stage graph by Definition 4. To analyze the interaction complexity of $P$, we need to construct a stage graph so that the expected number of transitions needed to move from stage to stage can be determined easily, and all terminal stages consist only of stable configurations (see Lemma 5 below).

Formally, a stage $S$ is terminal if it does not have any successors, i.e., there is no $S'$ satisfying $S \rightarrow S'$. Let $T$ be the set of all terminal stages, and let $\text{Term} \equiv \bigvee_{S \in T} [S]$. It follows directly from Definition 4 (b) that $C \models \Diamond \text{Term}$ for every initial configuration $C$. Let $\text{ReachTerminal}_G$ be a function assigning to every $n \geq 1$ the maximal $E_C[\text{Steps}_{\text{Term}}]$, where $C$ ranges over all initial configurations of size $n$. Furthermore, for every $S \in \mathcal{S}$, we define a function $\text{ReachNext}_S$ assigning to every $n \geq 1$ the maximal $E_C[\text{Steps}_{\text{Succ}(S)}]$, where $C$ ranges over all configurations of $[S]$ of size $n$ (if $[S]$ does not contain any configuration of size $n$, we put $\text{ReachNext}_S(n) = 0$).

An asymptotic upper bound for $\text{ReachTerminal}_G$ can be obtained by developing an asymptotic upper bound for all $\text{ReachNext}_S$, where $S \in \mathcal{S}$. Even though such a bound on $\text{ReachTerminal}_G$ depends on $[S]$, the latter is a constant since it is independent from the number of agents. Therefore, the following holds:

Lemma 5. Let $P = (Q, T, \Sigma, I, O)$ be a population protocol and $G = (\mathcal{S}, \rightarrow, [\cdot])$ a stage graph for $P$. Let $f : N \rightarrow N$ be a function such that $\text{ReachNext}_S \in O(f)$ for all $S \in \mathcal{S}$. Then $\text{ReachTerminal}_G \in O(f)$.

---

5 Recall that sets of configurations can be interpreted as formulae of the modal logic introduced in Section 2.2.
Observe that if every terminal stage \( S \) satisfies \([ S ] \subseteq \text{Stable}\), then \( \text{InterComplexity}_\mathcal{P} \leq \text{ReachTerminal}_\mathcal{G} \) (pointwise). Thus, we obtain the following:

**Lemma 6.** Let \( \mathcal{P} = (Q, T, \Sigma, I, O) \) be a population protocol and \( \mathcal{G} = (S, \rightarrow, [\cdot]) \) a stage graph for \( \mathcal{P} \) such that \([ S ] \subseteq \text{Stable}\) for every terminal stage \( S \). Let \( f : \mathbb{N} \rightarrow \mathbb{N} \) be a function such that \( \text{ReachNext}_S \in \mathcal{O}(f) \) for all \( S \in S \). Then \( \text{InterComplexity}_\mathcal{P} \in \mathcal{O}(f) \).

### 3.1 An example of a stage graph

In this section, we give an example of a stage graph \( \mathcal{G} \) for the majority protocol \( \mathcal{P} \) of Example 3 and we show how to analyze the interaction complexity of \( \mathcal{P} \) using \( \mathcal{G} \).

The stage graph \( \mathcal{G} \) of Fig. 1 is a simplified version of the stage graph computed by the algorithm of the forthcoming Section 4. Intuitively, the hierarchy of stages corresponds to “disabling more and more states” along runs initiated in initial configurations. For each stage \( S_i \) of \( \mathcal{G} \), the set \([ S_i ]\) consists of all configurations satisfying the associated formula shown in Fig. 1. Since \([ S_0 ]\) is precisely the set of all initial configurations, Condition (a) of Definition 4 is satisfied. For every \( C_0 \in [ S_0 ]\), transition \( AB \rightarrow bC \) can be executed in all configurations reachable from \( C_0 \) until \( A \) or \( B \) disappears. Furthermore, the number of \( A \)’s and \( B \)’s can only decrease along every run initiated in \( C_0 \). Hence, \( C_0 \) almost surely reaches a configuration \( C \) where \( A \) or \( B \) (or both of them) disappear. Note that if, e.g., \( C(A) = 0 \) and \( C(B) > 0 \), then this property is “permanent”, i.e., every successor \( C' \) of \( C \) also satisfies \( C'(A) = 0 \) and \( C'(B) > 0 \). Thus, we obtain the stages \( S_1 \), \( S_2 \), and \( S_3 \). Observe that if \( A \) and \( B \) disappear simultaneously (which happens iff the initial configuration \( C_0 \) satisfies \( C_0(A) = C_0(B) \)), then the configuration \( C \) will contain at least one copy of \( C \) which cannot be removed.

In all configurations of \([ S_1 ]\), the only potentially executable transitions are the following: \( AC \rightarrow Ac \), \( Ab \rightarrow Aa \), \( Ca \rightarrow Cb \). Since \( A \) appears in all configurations reachable from configurations of \([ S_1 ]\), the transition \( AC \rightarrow Aa \) stays enabled in all of these configurations until \( C \) disappears. Hence, every configuration of \([ S_1 ]\) almost surely reaches a configuration of \([ S_2 ]\). Similarly, we can argue that all configurations of \([ S_2 ]\) almost surely reach a configuration of \([ S_3 ]\), etc. Hence, Condition (b) of Definition 4 is also satisfied.

Let \( C_0 \in [ S_0 ]\) be an initial configuration of size \( n \), and let \( C \) be a configuration reachable from \( C_0 \) such that \( m = \min\{C(A), C(B)\} > 0 \). The probability of firing \( AB \rightarrow bC \) stays larger than \( m^2/n^2 \) in all configurations reached from \( C \) by executing a finite sequence of transitions different from \( AB \rightarrow bC \). This means that \( AB \rightarrow bC \) is fired after at most \( n^2/m^2 \).
trials on average. Since $\min \{C_0(A), C_0(B)\} \leq n/2$, we obtain
\[
ReachNext_{S_i}(n) \leq \sum_{i=1}^{\lfloor n/2 \rfloor} \frac{n^2}{i^2} \leq n^2 \cdot \sum_{i=1}^{\lfloor n/2 \rfloor} \frac{1}{i^2} \leq n^2 \cdot H_{n,2} \in O(n^2).
\]
Here, $H_{n,2}$ is the $n$-th Harmonic number of order 2. As $\lim_{n \to \infty} H_{n,2} = C < \infty$, we have that
\[
n^2 \cdot H_{n,2} \in O(n^2).
\]
Now, let us analyze $ReachNext_{S_1}(n)$. Let $C \in [S_1]$ be a configuration of size $n$. We need to fire the transition $AC \rightarrow Aa$ repeatedly until all $C$’s disappear. Let $C'$ be a configuration reachable from $C$ such that $C'(C) = m$. Since $C \models \Box(A \land \neg B)$, we have that $C'(A) > 0$, and hence the probability of firing $AC \rightarrow Aa$ in $C'$ is at least $m/n^2$. Thus, we obtain
\[
ReachNext_{S_1}(n) \leq \sum_{i=1}^{n} \frac{n^2}{i} \leq n^2 \cdot \sum_{i=1}^{n} \frac{1}{i} \leq n^2 \cdot H_n \in O(n^2 \log(n)).
\]
Here $H_n$ denotes the $n$-th Harmonic number (of order 1). Since $\lim_{n \to \infty} H_n = c \cdot \log(n)$ where $c$ is a constant, we get $n^2 \cdot H_n \in O(n^2 \log(n))$.

Similarly, we can show that $ReachNext_{S_i}(n) \in O(n^2 \log(n))$ for every stage $S_i$ of the considered stage graph. Since all configurations associated to terminal stages are stable, we can apply Lemma 6 and conclude that $InterComplexity_{AP} \in O(n^2 \log(n))$. Let us note that the algorithm of the forthcoming Section 4 can derive this result fully automatically in less than a second.

4 Computing a stage graph

In this section, we give an algorithm computing a stage graph for a given population protocol. Intuitively, the algorithm tries to identify a subset of transitions which will be simultaneously disabled and permanently disabled in the future with probability one, and also performs a kind of “case analysis” how this can happen. The resulting stage graph admits computing an upper asymptotic bounds on $ReachNext_{S}$ for every stage $S$, which allows to compute an asymptotic upper bound on the interaction complexity of the protocol by applying Lemma 6.

For the rest of this section, we fix a population protocol $\mathcal{P} = (Q, T, \Sigma, I, O)$. A valuation is a partial function $\nu : AP_{\mathcal{P}} \rightarrow \{\text{tt, ff}\}$ such that $\nu(AI) = \text{tt}$ implies $\nu(A) = \text{tt}$ whenever $AI, A \in \text{Dom}(\nu)$, where $\text{Dom}(\nu)$ is the domain of $\nu$. Slightly abusing our notation, we also denote by $\nu$ the propositional formula
\[
\bigwedge_{p \in \text{Dom}(\nu)} p \land \bigwedge_{p \in \text{Dom}(\nu)} \lnot p
\]

Hence, by writing $C \models \nu$ we mean that $C$ satisfies the above formula.

For every transition head $AB \in Q^{(2)}$, let $\xi_{AB}$ be either the formula $\neg A \lor \neg B$ or the formula $\neg A \lor AI$, depending on whether $A \neq B$ or $A = B$, respectively. Hence, the formulae $\xi_{AB}$ and $\lnot \xi_{AB}$ say that all transitions of the form $AB \rightarrow CD$ are disabled and enabled, respectively. For a given set $\mathcal{T} \subseteq Q^{(2)}$, consider the propositional formula $\Psi_{\mathcal{T}} \equiv \bigwedge_{AB \in \mathcal{T}} \xi_{AB}$. To simplify our notation, we write just $\mathcal{T}$ instead of $\Psi_{\mathcal{T}}$, i.e., $C \models \mathcal{T}$ iff all transitions specified by $\mathcal{T}$ are disabled in $C$.

**Definition 7.** Let $\mathcal{P} = (Q, T, \Sigma, I, O)$ be a population protocol. A $\mathcal{P}$-stage is a triple $S = (\Phi, \pi, \mathcal{T})$ where
- $\Phi$ is a propositional formula over $AP_{\mathcal{P}}$, 
- $\pi$ is a mapping from $AP_{\mathcal{P}}$ to $\{\text{true, false}\}$,
- $\mathcal{T}$ is a set of transition pairs.

We say that $\Psi_{\mathcal{T}}$ is a $\Phi$-based transition system for $\mathcal{T}$, and denote it by $\text{TransSys}_{\Phi}(\mathcal{T})$. The set of all $\Phi$-based transition systems is denoted by $\text{TransSys}_{\Phi}$. The $\Phi$-based transition system $\text{TransSys}_{\Phi}(\mathcal{T})$ is a transition system if and only if $\Phi$ is a tautology. If $\Phi$ is a tautology, then $\text{TransSys}_{\Phi}(\mathcal{T})$ is a transition system if and only if $\mathcal{T}$ is a set of transition pairs.

**Theorem 8.** Let $\mathcal{P} = (Q, T, \Sigma, I, O)$ be a population protocol. Then, the $\Phi$-based transition system $\text{TransSys}_{\Phi}(\mathcal{T})$ is a transition system if and only if $\Phi$ is a tautology. If $\Phi$ is a tautology, then $\text{TransSys}_{\Phi}(\mathcal{T})$ is a transition system if and only if $\mathcal{T}$ is a set of transition pairs.


- $\pi$ is a valuation, called the persistent valuation,
- $\mathcal{T} \subseteq Q^{(2)}$ is a set of transition heads, called the permanently disabled transition heads.

For every $P$-stage $S = (\Phi, \pi, T)$, we put $[[S]] = \{ C \mid C \models \Phi \land \Box \pi \land \Box T \}$.

Our algorithm computes a stage graph for $P$ gradually by adding more and more $P$-stages. It starts by inserting the initial $P$-stage $S_0 = (\Phi, \emptyset, \emptyset)$, where

$$\Phi \equiv \left( \bigvee_{A \in I(\Sigma)} A \right) \land \bigwedge_{A \in Q - I(\Sigma)} \neg A .$$

Note that $[[S_0]]$ is precisely the set of all initial configurations (the empty conjunction is interpreted as true). Then, the algorithm picks an unprocessed $P$-stage in the part of the stage graph constructed so far, and computes its immediate successors. This goes on until all $P$-stages become either internal or terminal. Since the total number of constructed $P$-stages can be exponential in the size of $P$, the worst-case complexity of our algorithm is exponential. However, as we shall see in Section 5, protocols with hundreds of states and transitions can be successfully analyzed even by our prototype implementation.

Let $S = (\Phi, \pi, T)$ be a non-terminal $P$-stage, and let $AP_S \subseteq AP_P$ be the set of all atomic propositions appearing in the formula $\Phi$. The successor $P$-stages of $S$ are constructed as follows. First, the algorithm computes the set $Val_S$ consisting of all valuations $\nu$ with domain $AP_S$ such that $\nu$ satisfies $\Phi$ when the latter is interpreted over $AP_S$. Intuitively, this corresponds to dividing $[[S]]$ into disjoint “subcases” determined by different $\nu$’s (as we shall see, $\Phi$ always implies the formula $\pi \land \mathcal{T}$, so $\nu$ cannot be in conflict with the information represented by $\pi$ and $T$; furthermore, we have $Dom(\pi) \subseteq Dom(\nu)$). Then, for each $\nu \in Val_S$, a $P$-stage $S_\nu$ is constructed, and $S_\nu$ may or may not become a successor of $S$. If none of these $S_\nu$ becomes a successor of $S$, then $S$ is declared as terminal.

Let us fix some $\nu \in Val_S$. In the rest of this section, we show how to compute the $P$-stage $S_\nu = (\Phi_{\nu}, \pi_{\nu}, T_\nu)$, and how to determine whether or not $S_\nu$ becomes a successor of $S$. An explicit pseudocode for constructing $S_\nu$ is given in in the appendix.

### 4.1 Computing the valuation $\pi_\nu$

The valuation $\pi_\nu$ is obtained by extending $\pi$ with the “permanent part” of $\nu$. Intuitively, we try to identify $A \in Q$ such that $\nu(A) = \text{tt}$ (or $\nu(A) = \text{ff}$) and all transitions containing $A$ on the left-hand (or the right-hand) side are permanently disabled. Furthermore, we also try to identify $A \in Q$ such that $\nu(A!) = \text{tt}$ and the number of $A$’s cannot change by firing transitions which are not permanently disabled. Technically, this is achieved by a simple fixed-point computation guaranteed to terminate quickly. The details are given in the appendix.

### 4.2 Computing the set $T_\nu$ and the formula $\Phi_\nu$

In some cases, the constructed persistent valuation $\pi_\nu$ already guarantees that a configuration satisfying $\pi_\nu \land \mathcal{T}$ is stable or cannot evolve (fire non-idle transitions) any further. Then, we in fact identified a subset of configurations belonging to $[[S]]$ which does not require any further analysis. Hence, we put $T_\nu = \mathcal{T}$, $\Phi_\nu = \pi_\nu$, and the configuration $S_\nu$ becomes a successor $P$-stage of $S$ declared as terminal.

Formally, we say that $(\pi_\nu, \mathcal{T})$ is stable if there is $x \in \{0, 1\}$ such that for all states $A \in Q$ where $\pi_\nu(A) = \text{tt}$ or $A \notin Dom(\pi_\nu)$ we have that $Out(A) = x$, and for every transition $CD \rightarrow EF$ where $Out(E) \neq x$ or $Out(F) \neq x$, the formula $(\pi_\nu \land \mathcal{T}) \Rightarrow \xi_{CD}$ is a propositional
tautology. Furthermore, we say that \((\pi_\nu, T)\) is dead if it is not stable and for every non-idle transition \(CD \rightarrow EF\) we have that the formula \((\pi_\nu \land T) \Rightarrow \xi_{CD}\) is a propositional tautology.

If \(S_\nu\) is not stable or dead, we use \(\pi_\nu\) and \(T\) to compute the transformation graph \(G_\nu\), and then analyze \(G_\nu\) to determine \(T_\nu\) and \(\Phi_\nu\).

### 4.2.1 The transformation graph

The vertices of the transformation graph \(G_\nu\) are the states which have not yet been permanently disabled according to \(\pi_\nu\), and the edges are determined by a set of transitions whose heads have not yet been permanently disabled according to \(\pi_\nu\) and \(T\). Formally, we put \(G_\nu = (V, \rightarrow)\) where the set of vertices \(V\) consists of all \(A \in Q\) such that either \(A \not\in \text{Dom}(\pi_\nu)\) or \(\pi_\nu(A) = tt\), and the set of edges is determined as follows: Let \(AB \rightarrow CD\) be a non-idle transition such that \((\pi_\nu \land T) \Rightarrow \xi_{AB}\) is not a tautology.

- If the sets \(\{A, B\}\) and \(\{C, D\}\) are disjoint, then the transition generates the edges \(A \rightarrow C\), \(A \rightarrow D\), \(B \rightarrow C\), \(B \rightarrow D\). Intuitively, both \(A\) and \(B\) can be “transformed” into \(C\) or \(D\).
- Otherwise, the transition has the form \(AB \rightarrow AD\) for \(B \neq D\). In this case it generates the edge \(B \rightarrow D\). Intuitively, \(B\) can be “transformed” into \(D\) in the context of \(A\).

#### Example 8.

Consider the protocol of Example 2 and its initial stage \(S = (\Phi, \pi, T)\) where \(\Phi = (A \lor B) \land \neg a \land \neg b\) and \(\pi = T = \emptyset\). Three valuations satisfy \(\Phi\); in particular the valuation \(\nu\) which sets to \(tt\) precisely the variables \(A\) and \(B\). Since both \(A\) and \(B\) can disappear in the future, and both \(a\) and \(b\) can become populated, the “permanent part” of \(\nu\), i.e., the valuation \(\pi_\nu\), has the empty domain. The transformation graph \(G_\nu\) is shown in Fig. 2 (left).

Consider now the majority protocol of Example 3 with initial stage \((\Phi, \emptyset, \emptyset)\) (where \(\Phi\) says there are only \(A\)’s and \(B\)’s), and a valuation \(\nu\) which sets to \(tt\) precisely the variables \(A\) and \(B\). The domain of \(\pi_\nu\) is again the empty set, and the transformation graph \(G_\nu\) is shown in Fig. 2 (right).

A key observation about transformation graphs is that all transitions generating edges connecting two different strongly connected components (SCCs) of \(G_\nu\) become simultaneously disabled in the future almost surely. More precisely, let \(\text{Exp}_\nu\) be the set of all \(AB \in Q^{(2)}\) such that there exists a transition \(AB \rightarrow CD\) generating an edge of \(G_\nu\) connecting two different SCCs of \(G_\nu\). We have the following:

#### Lemma 9.

Let \(G_\nu\) be a transformation graph, and let \(C\) be a configuration such that \(C \models \Box \pi_\nu \land \Box T\). Then \(C \models \Diamond \text{Exp}_\nu\). Furthermore, \(C \models \Diamond \Box \text{Exp}_\nu\).

However, there is a subtle problem. When the transitions specified by \(\text{Exp}_\nu\) become simultaneously disabled for the first time, they may be disabled only temporarily, i.e., \(C\) does not have to satisfy the formula \(\Box (\text{Exp}_\nu \Rightarrow \Box \text{Exp}_\nu)\). As we shall see in Section 5, it is relatively easy to obtain an upper bound on the expected number of transitions needed to visit a configuration satisfying \(\text{Exp}_\nu\). However, it is harder to give an upper bound on the expected
number of transitions needed to reach a configuration satisfying $\square Exp_\nu$ (i.e., entering the next stage) unless $\mathcal{C} \models \square(Exp_\nu \Rightarrow \square Exp_\nu)$. This difficulty is addressed in the next section.

Example 10. We continue with Example 8. For the transformation graph of Fig. 2 (left), we have $Exp_\nu = \{AB\}$. For the transformation graph of Fig. 2 (right), we have $Exp_\nu = \{AB,AC,BC\}$. Hence, according to Lemma 9, every initial configuration of the majority protocol of Example 2 almost surely reaches a configuration satisfying $\neg A \lor \neg B$, and every initial configuration of the majority protocol of Example 4 almost surely reaches a configuration satisfying $(\neg A \lor \neg B) \land (\neg A \lor \neg C) \land (\neg B \lor \neg C)$. Furthermore, in both cases $\mathcal{C} \models \square(Exp_\nu \Rightarrow \square Exp_\nu)$ for every initial configuration $\mathcal{C}$.

4.2.2 Computing $T_\nu$ and $\Phi_\nu$: Case $Exp_\nu \neq \emptyset$

Let $\Gamma_\nu \equiv \nu \land \square \pi_\nu \land \square T$, and let $\mathcal{C}$ be a configuration satisfying $\Gamma_\nu$. A natural idea to construct $T_\nu$ is to enrich $T$ by $Exp_\nu$. However, $Exp_\nu$ can be empty, i.e., the transformation graph $G_\nu$ may consist just of disconnected SCCs. For this reason we first consider the case where $Exp_\nu$ is nonempty.

Computing $T_\nu$. As discussed in Section 4.2.1, the fact that $\mathcal{C} \models \square \square Exp_\nu$ does not necessarily imply $\mathcal{C} \models \square(Exp_\nu \Rightarrow \square Exp_\nu)$ complicates the interaction complexity analysis. Therefore, after computing $Exp_\nu$, we try to compute a non-empty subset $J_\nu \subseteq Exp_\nu$ such that $\mathcal{C} \models \square(\nu \Rightarrow \square J_\nu)$ for all configurations $\mathcal{C}$ satisfying $\Gamma_\nu$. If we succeed, we put $T_\nu = T \cup J_\nu$. Otherwise, $T_\nu = T \cup Exp_\nu$. Intuitively, the set $J_\nu$ is the largest subset of $Exp_\nu$ such that every element of $M$ can be re-enabled only by firing a transition which has been identified as permanently disabled. This again leads to a simple fixed-point computation, which is detailed in the appendix.

A proof of the next lemma is straightforward.

Lemma 11. For every configuration $\mathcal{C}$ such that $\mathcal{C} \models \Gamma_\nu$ we have that
(a) $\mathcal{C} \models \square(\pi_\nu \land T \land Exp_\nu)$
(b) $\mathcal{C} \models \square(J_\nu \Rightarrow \square J_\nu)$

If $J_\nu \neq \emptyset$, we put $T_\nu = T \cup J_\nu$. Otherwise, we put $T_\nu = T \cup Exp_\nu$.

Computing $\Phi_\nu$. We say that a configuration $\mathcal{C}$ is $S_\nu$-entering if $\mathcal{C} \models \square \pi_\nu \land \square T_\nu$ and there is an execution $\mathcal{C}_0,\ldots,\mathcal{C}_\ell$ such that $\mathcal{C}_0 \models \Gamma_\nu$, $\mathcal{C}_\ell = \mathcal{C}$, and $\mathcal{C}_j \models \square \pi_\nu \land \square T_\nu$ for all $j < \ell$. An immediate consequence of Lemma 11 is the following:

Lemma 12. Almost all runs initiated in a configuration satisfying $\Gamma_\nu$ visit an $S_\nu$-entering configuration.
is a transition specified by $J_\nu$ enabled in $C$, the last transition executed before visiting an $S_\nu$-entering configuration must be a transition “transforming” some $A \in Q_\nu$, i.e., a transition of the form $AB \mapsto CD$ generating an edge $A \mapsto C$ of $G_\nu$. Let $K_\nu$ be the set of all right-hand sides of all such transitions. The formula $\Phi_\nu$ is defined as follows:

$$\Phi_\nu \equiv \begin{cases} 
\pi_\nu \land \mathcal{T}_\nu \land \nu & \text{if } J_\nu \text{ is } \nu\text{-disabled}, \\
\pi_\nu \land \mathcal{T}_\nu \land \left( \bigvee_{C \in E_\nu} \neg \xi_{CD} \right) & \text{if } J_\nu \text{ is } \nu\text{-enabled}, \\
\pi_\nu \land \mathcal{T}_\nu & \text{otherwise}.
\end{cases}$$

It is easy to check that every $S_\nu$-entering configuration satisfies the formula $\Phi_\nu$. The constructed $P$-stage $S_\nu = (\Phi_\nu, \pi_\nu, \mathcal{T}_\nu)$ becomes a successor of the $P$-stage $S$.

4.2.3 Computing $\mathcal{T}_\nu$ and $\Phi_\nu$: Case $\text{Exp}_\nu = \emptyset$.

In this case $G_\nu$ is a collection of disconnected SCCs. We put $\mathcal{T}_\nu = \mathcal{T}$. In the rest of the section we show how to construct the formula $\Phi_\nu$.

We say that an edge $A \mapsto B$ of $G_\nu$ is stable if there is a transition $AC \mapsto BD$ generating $A \mapsto B$ such that $\pi_\nu(C) = \text{tt}$. Let $I_\nu$ be the union of all non-bottom SCCs of the directed graph obtained from $G_\nu$ by considering only the stable edges of $G_\nu$.

\textbf{Lemma 13.} For every configuration $C$ such that $C \models \Gamma_\nu$ we have that $C \models \Diamond (\bigwedge_{A \in I_\nu} \neg A)$.

Similarly as above, we say that $C$ is $S_\nu$-entering if $C \models \Box \pi_\nu \land \Box \mathcal{T}_\nu \land \bigwedge_{A \in I_\nu} \neg A$ and there is an execution $C_0, \ldots, C_\ell$ such that $C_0 \models \Gamma_\nu$, $C_\ell = C$, and $C_j$ does not satisfy the above formula for all $j < \ell$.

Observe that if $\nu(A) = \text{ff}$ for all $A \in I_\nu$, then $\nu$ implies $\bigwedge_{A \in I_\nu} \neg A$ and hence every configuration $C$ satisfying $\Gamma_\nu$ is $S_\nu$-entering. Further, if $\nu(A) = \text{tt}$ for some $A \in I_\nu$, then the last transition executed before visiting an $S_\nu$-entering configuration is a transition $EF \mapsto CD$ generating a stable edge $E \mapsto C$ of $G_\nu$ where $E \in I_\nu$ and $C \notin I_\nu$. Let $L_\nu$ be the set of all right-hand sides of all such transitions. We put

$$\Phi_\nu \equiv \begin{cases} 
\pi_\nu \land \mathcal{T}_\nu \land \left( \bigwedge_{A \in I_\nu} \neg A \right) \land \left( \bigvee_{C \in E_\nu} \neg \xi_{CD} \right) & \text{if } \nu(A) = \text{tt} \text{ for some } A \in I_\nu, \\
\pi_\nu \land \mathcal{T}_\nu \land \nu & \text{if } \nu(A) = \text{ff} \text{ for all } A \in I_\nu, \\
\pi_\nu \land \mathcal{T}_\nu & \text{otherwise}.
\end{cases}$$

We say that the constructed $P$-stage $S_\nu = (\Phi_\nu, \pi_\nu, \mathcal{T}_\nu)$ is redundant if there is a $P$-stage $S' = (\Phi', \pi', \mathcal{T}')$ on the path from the initial stage $S_0$ to $S$ such that $\pi_\nu = \pi'$, $\mathcal{T}_\nu = \mathcal{T}'$, and $\Phi'$ implies $\Phi_\nu$. The $P$-stage $S_\nu$ becomes a successor of $S$ if $S_\nu$ is not redundant. This ensures termination of the algorithm even for poorly designed population protocols.

5 Computing the interaction complexity

We show how to compute an upper asymptotic bounds on $\text{ReachNext}_S$ for every stage $S$ in the stage graph constructed in Section 4.

For the rest of this section, we fix a population protocol $P = (Q, T, \Sigma, I, O)$, a $P$-stage $S = (\Phi, \pi, \mathcal{T})$, and its successor $S_\nu = (\Phi_\nu, \pi_\nu, \mathcal{T}_\nu)$. Recall the formula $\Gamma_\nu$, the graph
$G_\nu = (V, \to)$, and the sets $\text{Exp}_\nu$, $\mathcal{J}_\nu$ defined in Section 4. We show how to compute an asymptotic upper bound on the function $\text{Reach}_{S,S_\nu}$ that assigns to every $n \geq 1$ the maximal $E_C[\text{Steps}_{\text{Enter}(S_\nu)}]$, where $\text{Enter}(S_\nu)$ is a fresh atomic proposition satisfied precisely by all $S_\nu$-entering configurations, and $C$ ranges over all configurations of size $n$ satisfying $\Gamma_\nu$ (if there is no such configuration of size $n$, we put $\text{Reach}_{S,S_\nu}(n) = 0$). Observe that $\max_{S_\nu}\{\text{Reach}_{S,S_\nu}\}$, where $S_\nu$ ranges over all successor stages of $S$, is then an asymptotic upper bound on $\text{Reach}_{\text{Next}_S}$. 

Let us note that if $P$ terminates, then $\text{InterComplexity}_P \in 2^{2^{O(n)}}$. This trivial bound follows by observing that the number of all configurations of size $n$ is $2^{O(n)}$, and the probability of reaching a stable configuration in $2^{O(n)}$ transitions is $2^{-2^{O(n)}}$; this immediately implies the mentioned upper bound on $\text{InterComplexity}_P$. As we shall see, the worst asymptotic bound on $\text{Reach}_{S,S_\nu}$ is $2^{O(n)}$, and in many cases, our results allow to derive even a polynomial upper bound on $\text{Reach}_{S,S_\nu}$.

Recall that if $(\pi_\nu, T)$ is stable or dead, we have that $\text{Reach}_{S,S_\nu}(n) = 0$ for all $n \in \mathbb{N}$ (in this case, we define $S_\nu$-entering configurations are the configurations satisfying $\square(\pi_\nu \land T)$). Now suppose $(\pi_\nu, T)$ is not stable or dead. Furthermore, let us first assume $\text{Exp}_\nu = \emptyset$. Then, the upper bound on $\text{Reach}_{S,S_\nu}$ is singly exponential in $n$.

**Theorem 14.** If $\text{Exp}_\nu = \emptyset$, then $\text{Reach}_{S,S_\nu} \in 2^{O(n)}$.

Now assume $\text{Exp}_\nu \neq \emptyset$. Let $U \subseteq Q$ be the set of all states appearing in some non-bottom SCC of $G_\nu$. We start with some auxiliary definitions.

**Definition 15.** For every $A \in U$, let $\text{Exp}_\nu[A]$ be the set of all $B \in Q$ such that $AB \in \text{Exp}_\nu$. We say that $S_\nu$ is fast if, for every $A \in U$, the formula $(\pi_\nu \land T \land \neg \text{Exp}_\nu \land A) \Rightarrow (\bigvee_{B \in \text{Exp}_\nu[A]} \neg \xi_{AB})$ is a propositional tautology.

**Definition 16.** For every $A \in V$, let $[A]$ be the SCC of $G_\nu$ containing $A$. We say that $S_\nu$ is very fast if every transition $AB \to CD$ such that $AB, CD \in V^{(2)}$ and $\{A, B, C, D\} \cap U \neq \emptyset$ satisfies one of the following conditions:
- The formula $(\pi_\nu \land T) \Rightarrow \xi_{AB}$ is a propositional tautology.
- $[C] \neq [A] \neq [D]$ and $|C| \neq |B| \neq |D|$.

**Theorem 17.** If $\text{Exp}_\nu \neq \emptyset$ and $\mathcal{J}_\nu \neq \emptyset$, then
- $\text{Reach}_{S,S_\nu} \in \mathcal{O}(n^3)$.
- If $S_\nu$ is fast, then $\text{Reach}_{S,S_\nu} \in \mathcal{O}(n^2 \cdot \log(n))$.
- If $S_\nu$ is very fast, then $\text{Reach}_{S,S_\nu} \in \mathcal{O}(n^2)$.

Computing an asymptotic upper bound on $\text{Reach}_{S,S_\nu}$ when $\text{Exp}_\nu \neq \emptyset$ and $\mathcal{J}_\nu = \emptyset$ is more complicated. We show that a polynomial upper bound always exists, and that the degree of the polynomial is computable. However, our proof does not yield an efficient algorithm for computing/estimating the degree.

**Theorem 18.** If $\text{Exp}_\nu \neq \emptyset$ and $\mathcal{J}_\nu = \emptyset$, then $\text{Reach}_{S,S_\nu} \in \mathcal{O}(n^c)$ for some computable constant $c$.

## 6 Experimental results

We have implemented our approach as a tool\footnote{The tool and its benchmarks are available at \url{https://github.com/blondini/pp-time-analysis}} that takes a population protocol as input and follows the procedure of Section 4 to construct a stage graph together with an upper
Automatic Analysis of Expected Termination Time for Population Protocols

bound on $\text{InterComplexity}_p$. Our tool is implemented in Python 3, and uses the SMT solver Microsoft Z3\(^7\) to test for tautologies and to obtain valid valuations.

We tested our implementation on multiple protocols drawn from the literature: a simple broadcast protocol\(^8\), the majority protocols of Example 2, Example 3 and 2, various flock-of-birds protocols\(^1\)\(^1\)\(^1\),\(^7\)\(^1\), a remainder protocol\(^8\) and a threshold protocol\(^8\). Most of these protocols are parametric, i.e. they are a family of protocols depending on some parameters. For these protocols, we increased their parameters until reaching a timeout. In particular, for the logarithmic flock-of-birds protocol computing $x \geq c$, we used thresholds of the form $c = 2^i - 1$ as they essentially consist the most complicated case of the protocol.

All tests were performed on the same computer equipped with eight Intel® Core™ i5-8250U 1.60 GHz CPUs, 8 GB of memory and Ubuntu Linux 17.10 (64 bits). Each test had a timeout of 1000 seconds (~16.67 minutes). The duration of each test was evaluated as the sum of the user time and sys time reported by the Python time library.

The results of the benchmarks are depicted in Table 1 where the bound column refers to the derived upper bound on $\text{InterComplexity}_p$. In particular, the tool derived exponential and $n^2 \cdot \log n$ bounds for the protocols of Example 2 and Example 3 respectively. The generated trees across all instances grow in width but not much in height: the maximum height between the roots and the leaves varies between 2 and 5, and most nodes are leaves.

It is worth noting that the $n^2 \log n$ bounds obtained in Table 1 for the average-and-conquer and remainder protocols are tight with respect to the best known bounds\(^2\)\(^1\)\(^4\). However, some of the obtained bounds are not tight, e.g. we report $n^3$ for the threshold protocol but

\(^7\) Protocol of Example 2 without the tie-breaking rule $aa \rightarrow bb$ (only correct if $x \neq y$).

\(^8\) An adapted version of the protocol of\(^9\) Sect. 3 without so-called k-way transitions.

\(^9\) The protocol is only correct assuming $x \neq y$. 

\[\text{Table 1 Results of the experimental evaluation where } |Q|, |T| \text{ and } |S| \text{ correspond respectively to the number of states and transitions of the protocol, and the number of nodes of its stage graph.}\]

| Protocol | $|S|$ | Bound | Time |
| --- | --- | --- | --- |
| Flocks-of-bird protocol\(^1\)\(^1\)\(^1\) | $x \geq c$ | $c = 5$ | $c = 10$ | $c = 15$ | $c = 20$ | $c = 25$ | $c = 30$ | $c = 35$ | $c = 40$ | $c = 45$ | $c = 50$ | $c = 55$ |
| | | 6 | 11 | 16 | 21 | 26 | 30 | 35 | 40 | 45 | 50 | 55 |
| | | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| | | $n^3$ | $n^3$ | $n^3$ | $n^3$ | $n^3$ | $n^3$ | $n^3$ | $n^3$ | $n^3$ | $n^3$ | $n^3$ |
| | | 0.8 | 4.0 | 12.1 | 28.9 | 58.0 | 118.9 | 222.3 | 366.2 | 495.3 | 952.8 | 2200 |
| | | Average-and-conquer protocol\(^8\) | $x \geq y$ with params. $m$ and $d$ | $m = 3, d = 1$ | $m = 3, d = 2$ | $m = 5, d = 1$ | $m = 5, d = 2$ | $m = 7, d = 1$ | $m = 7, d = 2$ | $m = 7, d = 3$ | $m = 7, d = 4$ | $m = 7, d = 5$ | $m = 7, d = 6$ |
| | | 6 | 23 | 36 | 40 | 46 | 55 | 63 | 77 | 94 | 122 | 169 |
| | | $n^3 \cdot \log n$ | $n^3 \cdot \log n$ | $n^3 \cdot \log n$ | $n^3 \cdot \log n$ | $n^3 \cdot \log n$ | $n^3 \cdot \log n$ | $n^3 \cdot \log n$ | $n^3 \cdot \log n$ | $n^3 \cdot \log n$ | $n^3 \cdot \log n$ | $n^3 \cdot \log n$ |
| | | 2.0 | 8.0 | 40.1 | 40.1 | 70.0 | 70.0 | 70.0 | 70.0 | 70.0 | 70.0 | 70.0 |
| | | Threshold protocol\(^8\) | $\sum_{i=0}^{\log n} x_i = 0$ (mod $m$) | $m = 3$ | $m = 5$ | $m = 7$ | $m = 9$ | $m = 10$ | $m = 12$ | $m = 15$ | $m = 18$ | $m = 20$ | $m = 24$ |
| | | 5 | 7 | 9 | 11 | 12 | 15 | 21 | 27 | 33 | 45 | 63 |
| | | $n^2 \cdot \log n$ | $n^2 \cdot \log n$ | $n^2 \cdot \log n$ | $n^2 \cdot \log n$ | $n^2 \cdot \log n$ | $n^2 \cdot \log n$ | $n^2 \cdot \log n$ | $n^2 \cdot \log n$ | $n^2 \cdot \log n$ | $n^2 \cdot \log n$ | $n^2 \cdot \log n$ |
| | | 0.8 | 12.5 | 88.9 | 544.0 | T/O | T/O | T/O | T/O | T/O | T/O | T/O |

\[\text{Table 1 Results of the experimental evaluation where } |Q|, |T| \text{ and } |S| \text{ correspond respectively to the number of states and transitions of the protocol, and the number of nodes of its stage graph.}\]
an $n^2 \log n$ upper bound was shown in [4]. Moreover, it seems possible to decrease the $n^3$
bound to $n^2$ for the flocks-of-bird protocol of [4]. We are unsure of the precise bounds for the
remaining protocols.

7 Conclusion

We have presented the first algorithm for quantitative verification of population protocols
able to provide asymptotic bounds valid for any number of agents. The algorithm is able to
compute good bounds for many of the protocols described in the literature.

The algorithm is based on the notion of stage graph, a concept that can be of independent
value. In particular, we think that stage graphs can be valuable for fault localization and
perhaps even automatic repair of ill designed protocols.

An interesting question is whether our algorithm is “weakly complete”, meaning that for
every predicate there exists a protocol for which our algorithm can compute the exact time
bound. We know that this is the case for protocols with leaders, and conjecture that the
result extends to all protocols, but currently we do not have a proof.

Another venue for future research is the automatic computation of lower bounds. Here,
while stage graphs will certainly be useful, they do not seem to be enough, and will have to
be complemented with other techniques.

References

1 Dan Alistarh, James Aspnes, David Eisenstat, Rati Gelashvili, and Ronald L. Rivest. Time-
space trade-offs in population protocols. In Proc. 28th Annual ACM-SIAM Symposium on
Discrete Algorithms (SODA), pages 2560–2579, 2017. doi:10.1137/1.9781611974782

2 Dan Alistarh, Rati Gelashvili, and Milan Vojnović. Fast and exact majority in population
protocols. In Proc. ACM Symposium on Principles of Distributed Computing (PODC),
pages 47–56, 2015. doi:10.1145/2767386.2767429

3 Dana Angluin, James Aspnes, Zoë Diamadi, Michael J. Fischer, and René Peralta. Com-
putation in networks of passively mobile finite-state sensors. In Proc. 23rd Annual ACM
Symposium on Principles of Distributed Computing (PODC), pages 290–299, 2004.
doi:10.1145/1011767.1011810

4 Dana Angluin, James Aspnes, Zoë Diamadi, Michael J. Fischer, and René Peralta. Com-
putation in networks of passively mobile finite-state sensors. Distributed Computing,
18(4):235–253, Jan 2006. doi:10.1007/s00446-005-0138-3

5 Kevin Batz, Benjamin Lucien Kaminski, Joost-Pieter Katoen, and Christoph Matheja.
How long, O bayesian network, will I sample thee? - A program analysis perspective on
expected sampling times. In Proc. 27th European Symposium on Programming (ESOP),
pages 186–213. Springer, 2018. doi:10.1007/978-3-319-89884-1_7

6 Amanda Belleville, David Doty, and David Soloveichik. Hardness of computing and approx-
imating predicates and functions with leaderless population protocols. In Proc. 44th Inter-
national Colloquium on Automata, Languages, and Programming (ICALP), pages 141:1–
141:14, 2017. doi:10.4230/LIPIcs.ICALP.2017.141

7 Michael Blondin, Javier Esparza, and Stefan Jaax. Large flocks of small birds: on the
minimal size of population protocols. In Proc. 35th Symposium on Theoretical Aspects of
Computer Science (STACS), pages 16:1–16:14, 2018. doi:10.4230/LIPIcs.STACS.2018
8 Michael Blondin, Javier Esparza, Stefan Jaax, and Philipp J. Meyer. Towards efficient verification of population protocols. In Proc. 36th ACM Symposium on Principles of Distributed Computing (PODC), pages 423–430, 2017. doi:10.1145/3087801.3087816
9 Krishnendu Chatterjee, Hongfei Fu, and Aniket Murhekar. Automated recurrence analysis for almost-linear expected-runtime bounds. In Proc. 29th International Conference on Computer Aided Verification (CAV), pages 118–139, 2017. doi:10.1007/978-3-319-63387-9_6
10 Ho-Lin Chen, Rachel Cummings, David Doty, and David Soloveichik. Speed faults in computation by chemical reaction networks. Distributed Computing, 30(5):373–390, 2017. doi:10.1007/s00446-015-0255-6
11 Julien Clément, Carole Delporte-Gallet, Hugues Fauconnier, and Mihaela Sighireanu. Guidelines for the verification of population protocols. In Proc. 31st International Conference on Distributed Computing Systems (ICDCS), pages 215–224, 2011. doi:10.1109/ICDCS.2011.36
12 Leonardo Mendonça de Moura and Nikolaj Bjørner. Z3: an efficient SMT solver. In Proc. 14th International Conference Tools and Algorithms for the Construction and Analysis of Systems (TACAS), pages 337–340, 2008. doi:10.1007/978-3-540-78800-3_24
13 David Doty and David Soloveichik. Stable leader election in population protocols requires linear time. In Proc. 29th International Symposium on Distributed Computing (DISC), pages 602–616, 2015. doi:10.1007/978-3-662-48653-5_40
14 Moez Draief and Milan Vojnović. Convergence speed of binary interval consensus. In Proc. 29th IEEE International Conference on Computer Communications (INFOCOM), pages 1792–1800, 2010. doi:10.1109/INFCOM.2010.5461999
15 Javier Esparza, Pierre Ganty, Jérôme Leroux, and Rupak Majumdar. Verification of population protocols. Acta Informatica, 54(2):191–215, 2017. doi:10.1007/s00236-016-0272-3
16 Philippe Flajolet, Bruno Salvy, and Paul Zimmermann. Automatic average-case analysis of algorithm. Theoretical Computer Science, 79(1):37–109, 1991. doi:10.1016/0304-3975(91)90145-R
17 Stefan Jaax. Personal communication, April 2018.
18 Benjamin Lucien Kaminski, Joost-Pieter Katoen, Christoph Matheja, and Federico Olmedo. Weakest precondition reasoning for expected run-times of probabilistic programs. In Proc. 25th European Symposium on Programming (ESOP), pages 364–389. Springer, 2016. doi:10.1007/978-3-662-49498-1_15
19 Othon Michail and Paul G. Spirakis. Elements of the theory of dynamic networks. Communications of the ACM, 61(2):72, 2018. doi:10.1145/3156693
20 Saket Navlakha and Ziv Bar-Joseph. Distributed information processing in biological and computational systems. Communications of the ACM, 58(1):94–102, 2015. doi:10.1145/2678280
21 Van Chan Ngo, Quentin Carbonneaux, and Jan Hoffmann. Bounded expectations: resource analysis for probabilistic programs. In Proc. 39th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI), pages 496–512, 2018. doi:10.1145/3192366.3192394
22 Etienne Perron, Dinkar Vasudevan, and Milan Vojnović. Using three states for binary consensus on complete graphs. In Proc. 28th IEEE International Conference on Computer Communications (INFOCOM), pages 2527–2535, 2009. doi:10.1109/INFCOM.2009.5062181
A Section 3 (Stages of population protocols)

Lemma 5. Let \( \mathcal{P} = (Q, T, \Sigma, I, O) \) be a population protocol and \( \mathcal{G} = (S, \rightarrow, [\cdot]) \) a stage graph for \( \mathcal{P} \). Let \( f : \mathbb{N} \to \mathbb{N} \) be a function such that \( \text{ReachNext}_S \in \mathcal{O}(f) \) for all \( S \in \mathcal{S} \). Then \( \text{ReachTerminal}_{\mathcal{G}} \in \mathcal{O}(f) \).

Proof. Let \( C_0 \) be an initial configuration. For every \( i \in \mathbb{N} \), we define random variables \( \text{Move}_i \) and \( \text{Stages}_i \) over the runs initiated in \( C_0 \) inductively as follows. Let \( \omega = C_0, C_1, \ldots \) be a run initiated in \( C_0 \). Then

- \( \text{Move}_0(\omega) = 0 \)
- \( \text{Stages}_0(\omega) = \{ S \in \mathcal{S} \mid C_0 \in [S] \} \).
- Let \( M = \{ S' \in \mathcal{S} \mid S \mapsto S' \text{ for some } S \in \text{Stages}_i(\omega) \} \). If \( M = \emptyset \), we put \( \text{Move}_{i+1}(\omega) = 0 \) and \( \text{Stages}_{i+1}(\omega) = \text{Stages}_i(\omega) \). Otherwise, let \( k = \sum_{j=0}^{\infty} \text{Move}_j(\omega) \). We define \( \text{Move}_{i+1}(\omega) \) as the least \( \ell \in \mathbb{N} \) such that \( C_{k+\ell} \in \bigcup_{S \in M} [S] \), or \( \infty \) if no such \( \ell \in \mathbb{N} \) exists (this includes the case when \( k = \infty \)). Furthermore, if \( \text{Move}_{i+1}(\omega) < \infty \), we put \( \text{Stages}_{i+1}(\omega) = \{ S \in M \mid C_{k+\text{Move}_{i+1}}(\omega) \in [S] \} \);

Condition (b) of Definition 4 immediately implies \( \mathbb{P}_{C_0}[\text{Move}_i = \infty] = 0 \) for all \( i \in \mathbb{N} \). Since \( \mapsto \) is acyclic, for almost all runs \( \omega \) initiated in \( C_0 \) we have that \( \text{Move}_i(\omega) = 0 \) for every \( i \geq |\mathcal{S}| \).

Thus, we obtain

\[
\mathbb{E}_{C_0}[\text{Steps}_{\text{Term}}] \leq \mathbb{E}_{C_0} \left[ \sum_{i=0}^{\infty} \text{Move}_i \right] = \mathbb{E}_{C_0} \left[ \sum_{i=0}^{|\mathcal{S}|} \text{Move}_i \right] = \sum_{i=0}^{|\mathcal{S}|} \mathbb{E}_{C_0}[\text{Move}_i].
\]

Clearly, \( \mathbb{E}_{C_0}[\text{Move}_i] \leq \max_{S \in \mathcal{S}} \mathbb{E}_C[\text{Steps}_{\text{Suc}(S)}] \) where \( C \) ranges over all configurations of \( [\mathcal{S}] \) whose size is equal to the size of \( C_0 \). In other words, \( \mathbb{E}_{C_0}[\text{Move}_i] \leq \max_{S \in \mathcal{S}} \text{ReachNext}_S(n) \), where \( n \) is the size of \( C_0 \). Since \( \text{ReachNext}_S \in \mathcal{O}(f) \) for all \( S \in \mathcal{S} \) and \( |\mathcal{S}| \) is a constant, we obtain \( \text{ReachTerminal}_{\mathcal{G}} \in \mathcal{O}(f) \).}

B Section 4 (Computing a stage graph)

B.1 The procedure for computing the valuation \( \pi_v \).

First, we show how to compute two sets \( \mathcal{M} \subseteq Q \) and \( \mathcal{N} \subseteq Q \) satisfying the following properties:

1. \( \nu(A) = \text{ff} \) for every \( A \in \mathcal{M} \), and \( \nu(A!) = \text{tt} \) for every \( A \in \mathcal{N} \).
   (Every configuration satisfying \( \nu \) puts no agents in states of \( \mathcal{M} \), and exactly one agent in each state of \( \mathcal{N} \).)

2. For every configuration \( C \in [\mathcal{S}] \) such that \( C \models \nu \) and for every configuration \( C' \) reachable from \( C \), \( C' \models \neg A \) for every \( A \in \mathcal{M} \), and \( C' \models A! \) for every \( A \in \mathcal{N} \).
   (Every configuration reachable from a configuration satisfying \( \nu \) puts no agents in states of \( \mathcal{M} \), and exactly one agent in each state of \( \mathcal{N} \).)

The pair \( (\mathcal{M}, \mathcal{N}) \) is computed as the greatest fixed-point of a function \( f : 2^Q \times 2^Q \to 2^Q \times 2^Q \).

Intuitively, we start with the pair of sets \( (M_0, N_0) \) such that \( A \in M_0 \)iff \( \nu(A) = \text{ff} \) and \( A \in N_0 \) iff \( \nu(A!) = \text{tt} \), i.e., with largest pair of sets satisfying (1). Then we repeatedly remove states for which we can determine that (2) does not hold. For example, if \( M_0 = \{ A, B \} \) and the protocol has a transition \( CD \mapsto AD \), then we can remove \( A \) from \( M_0 \), because there exists a configuration \( C \) satisfying \( \neg A \wedge \neg B \), from which we can reach a configuration satisfying \( A \).

Formally, for a given pair \( (M, N) \in 2^Q \times 2^Q \), let \( f \) be the function that returns the pair \( (M', N') \) given by:
the set $M'$ consists of all $A \in Q$ where $\nu(A) = \texttt{tt}$ and every transition of the form $CD \mapsto AB$ satisfies either $(C,D) \cap M \neq \emptyset$, or $CD \in T$, or $C = D$ and $C \in N$;

- the set $N'$ consists of all $A \in Q$ where $\nu(A) = \texttt{tt}$, and the following conditions are satisfied:
  - Let $AB \mapsto CD$ be a transition such that $A \neq B$ and $C \neq A \neq D$. Then $B \in M$ or $AB \in T$.
  - Let $AB \mapsto AA$ be a transition such that $A \neq B$. Then $B \in M$ or $AB \in T$.
  - Let $CD \mapsto AB$ be a transition such that $C \neq A \neq D$. Then either $(C,D) \cap M \neq \emptyset$, or $CD \in T$, or $C = D$ and $C \in N$.

Observe that $f$ is monotone, hence the greatest fixed-point $(M,N)$ of $f$ exists and can be computed in polynomial time. Further, let $E$ be the set of all $A \in Q$ such that $\nu(A) = \texttt{tt}$ and every transition of the form $AB \mapsto CD$, where $C \neq A \neq D$, satisfies either $B \in M$, or $AB \in T$, or $A = B$ and $A \in N$. Intuitively, this is the set of states that must necessarily contain exactly one agent, and so we put $\pi_v(A) = \texttt{ff}$ for all $A \in M$, $\pi_v(A) = \texttt{tt}$ for all $A \in E$, and $\pi_v(A) = \texttt{tt}$ for all $A \in N$.

**Example 19.** Let $P$ be the protocol of Example 2 and let $S = (\Phi, \emptyset, \emptyset)$ be the initial $P$-stage, where $\Phi \equiv (A \lor B) \land \neg a \land \neg b$. There are three valuations $\nu_A, \nu_B, \nu_{AB}$ satisfying $\Phi$, which set to $\texttt{tt}$ precisely the variable $A$, or $B$, or both $A$ and $B$, respectively. The fixed-point computation starts from the sets $\{(B,a,b),\emptyset\}$, $\{(A,a,b),\emptyset\}$, and $\{(a,b),\emptyset\}$, respectively. The greatest fixed-point $(M,N)$ is $\{(B,a,b),\emptyset\}$, $\{(A,a,b),\emptyset\}$, and $\{(\emptyset,\emptyset)\}$, respectively. We have $\text{Dom}(\pi_{\nu_A}) = \text{Dom}(\pi_{\nu_B}) = \{A,B,a,b\}$ and $\text{Dom}(\pi_{\nu_{AB}}) = \emptyset$.

**B.2 The procedure for computing $J_v$.**

Consider a subset $M \subseteq \text{Exp}_v$ and a configuration $C'$ reachable from a configuration satisfying $\Gamma_v$ and such that $C' \models M$. Let $C_0, \ldots, C_\ell$ be an execution initiated in $C'$ such that $C_i \models M$ for all $i < \ell$, and some transition specified by $M$ is re-enabled in $C_\ell$. Let $AB \mapsto CD$ be the transition fired when moving from $C_{\ell-1}$ to $C_\ell$. Since $C_{\ell-1} \models M$ and firing $AB \mapsto CD$ enables some transition specified by $M$ in $C_\ell$, one of the following conditions holds:

- $CD \in M$,
- there is $E \in V$ such that $CE \in M$, $E \neq D$, $C_{\ell-1}(E) > 0$,
- there is $E \in V$ such that $DE \in M$, $E \neq C$, $C_{\ell-1}(E) > 0$.

The set $J_v$ is the largest $M \subseteq \text{Exp}_v$ such that, for every $EF \in M$, the following holds:

- For every transition of the form $AB \mapsto EF$ the formula $(\pi_v \land T \land M) \Rightarrow \xi_{AB}$ is a propositional tautology.
- For every transition of the form $AB \mapsto EG$ where $G \neq F$ we have that
  - if $E \neq F$, then the formula $(-E \land F \land \pi_v \land T \land M) \Rightarrow \xi_{AB}$ is a tautology;
  - if $E = F$, then $A = E$, or $B = E$, or $(E! \land \pi_v \land T \land M) \Rightarrow \xi_{AB}$ is a tautology.
- For every transition of the form $AB \mapsto FG$ where $G \neq E$ we have that
  - if $E \neq F$, then the formula $(-F \land E \land \pi_v \land T \land M) \Rightarrow \xi_{AB}$ is a tautology;
  - if $E = F$, then $A = F$, or $B = F$, or $(F! \land \pi_v \land T \land M) \Rightarrow \xi_{AB}$ is a tautology.

Observe that $J_v$ is computable by a simple fixed-point algorithm.

**B.3 A pseudocode for computing the stage $S_v = (\Phi_v, \pi_v, T_v)$**

An explicit pseudocode for constructing the stage $S_v = (\Phi_v, \pi_v, T_v)$ is given in Algorithm [1].
Algorithm 1: Computing the $P$-stage $S_{\nu}$.

**Input:** $S = (\Phi, \pi, T)$, an assignment $\nu \in \text{Val}_S$

**Output:** $S_{\nu} = (\Phi_{\nu}, \pi_{\nu}, T_{\nu})$.

1. compute $\pi_{\nu}$
2. if $(\pi_{\nu}, T)$ is stable or dead then
   3. return $(\pi_{\nu}, \pi_{\nu}, T)$
3. end
4. compute $G_{\nu}$
5. compute $\text{Exp}_{\nu}$
6. compute $J_{\nu}$
7. if $\text{Exp}_{\nu} \neq \emptyset$ then
   8. if $J_{\nu} \neq \emptyset$ then
      9. $T_{\nu} := T \cup J_{\nu}$
     10. if $J_{\nu}$ is $\nu$-disabled then
        11. $\Phi_{\nu} := \pi_{\nu} \land T_{\nu} \land \nu$
     12. else if $J_{\nu}$ is $\nu$-enabled then
        13. compute the set $K_{\nu}$
        14. $\Phi_{\nu} := \pi_{\nu} \land T_{\nu} \land \left( \bigvee_{CD \in K_{\nu}} \eta(CD) \right)$
     15. else
        16. $\Phi_{\nu} := \pi_{\nu} \land T_{\nu}$
     17. end
   18. end
   19. else
   20. $T_{\nu} := T \cup \text{Exp}_{\nu}$
   21. $\Phi_{\nu} := \pi_{\nu} \land T_{\nu}$
   22. end
23. else
24. $T_{\nu} := T$
25. compute the set $I_{\nu}$
26. if $\nu(A) = tt$ for some $A \in I_{\nu}$ then
27. compute the set $L_{\nu}$
28. $\Phi_{\nu} := \pi_{\nu} \land T_{\nu} \land \left( \bigwedge_{A \in I_{\nu}} \neg A \right) \land \left( \bigvee_{CD \in \mathcal{L}_{\nu}} \eta(CD) \right)$
29. else if $\nu(A) = ff$ for all $A \in I_{\nu}$ then
30. $\Phi_{\nu} := \pi_{\nu} \land T_{\nu} \land \nu$
31. else
32. $\Phi_{\nu} := \pi_{\nu} \land T_{\nu} \land \left( \bigwedge_{A \in I_{\nu}} \neg A \right)$
33. end
34. end
35. return $(\Phi_{\nu}, \pi_{\nu}, T_{\nu})$
Section 5 (Computing the interaction complexity)

Theorem 14. If \( \exp_\nu = \emptyset \), then \( \reach_{S,S_\nu} \in 2^{O(n)} \).

Proof. Recall the definition of \( \mathcal{I}_\nu \) given in Section 4.2.3. Let \( C \) be a configuration of size \( n \) reachable from a configuration satisfying the formula \( \Gamma_\nu \). Then there is a configuration \( C' \) reachable from \( C \) in at most \( |Q| \cdot n \) transitions such that \( C' \models \bigwedge_{A \in \mathcal{I}_\nu} \lnot A \). The probability of firing a given transition in a given configuration is at least \( 1/n^2 \), hence the probability of reaching such a \( C' \) from \( C \) in at most \( |Q| \cdot n \) transitions is \( 2^{-O(n)} \). On average, we need to perform such an execution at most \( 2^{O(n)} \) times, which yields the \( 2^{O(n)} \) bound.

Theorem 17. If \( \exp_\nu \neq \emptyset \) and \( J_\nu \neq \emptyset \), then
- \( \reach_{S,S_\nu} \in O(n^3) \).
- If \( S_\nu \) is fast, then \( \reach_{S,S_\nu} \in O(n^2 \cdot \log(n)) \).
- If \( S_\nu \) is very fast, then \( \reach_{S,S_\nu} \in O(n^2) \).

Proof. For every SCC of \( G_\nu \), we define its distance inductively as follows: the distance of every bottom SCC is 0, and the distance of a non-bottom SCC is the maximal distance of its immediate successors plus 1. For all \( A \in Q \), let \( w(A) \) be a non-negative integer defined by
\[
\begin{cases}
  d & \text{if } A \text{ appears in a non-bottom SCC of } G \text{ with distance } d, \\
  0 & \text{otherwise}.
\end{cases}
\]

Since \( J_\nu \subseteq \exp_\nu \), for every \( n \in \mathbb{N} \) we have that \( \reach_{S,S_\nu}(n) \leq \max_C \mathbb{E}_C[\text{Steps}_{\exp_\nu}] \), where \( C \) ranges over all configurations of size \( n \) satisfying the formula \( \Gamma_\nu \). Hence, it suffices to give an appropriate upper bound on \( \mathbb{E}_C[\text{Steps}_{\exp_\nu}] \).

Let \( C \) be a configuration of size \( n \) reachable from a configuration satisfying the formula \( \Gamma_\nu \). The potential of \( C \) is defined by \( \alpha_C = \sum_{A \in Q} w(A) \cdot \mathcal{C}(A) \). Clearly, \( 0 \leq \alpha_C \leq |Q| \cdot n \). Suppose that \( C \) fires a transition \( AC \rightarrow BD \) and enters a configuration \( C' \). It follows immediately from the definition of \( G_\nu \) that \( \alpha_{C'} \leq \alpha_C \). Further, \( \alpha_{C'} < \alpha_C \) iff \( AC \rightarrow BD \) generates an edge \( A \rightarrow B \) of \( G \) such that \( A \) and \( B \) belong to different SCC’s of \( G_\nu \). Consequently, if \( \alpha_C = 0 \), then \( C \models \exp_\nu \). We show that if \( C \not\models \exp_\nu \), then the expected number of transitions fired before reaching a configuration \( C' \) such that \( C' \models \exp_\nu \) or \( \alpha_{C'} < \alpha_C \) is bounded by \( c \cdot n^2 \), where \( c \) is a positive constant depending only of \( \mathcal{P} \). If \( S_\nu \) is fast, then this bound can be improved to \( c' \cdot (n^2/\alpha_C) \). Since \( \alpha_C \leq |Q| \cdot n \), we immediately obtain that \( \mathbb{E}_C[\text{Steps}_{\exp_\nu}] \) is \( O(n^3) \). If \( S_\nu \) is fast, this bound can be improved to \( \sum_{k=1}^{|Q| n} c' \cdot (n^2/k) = c' \cdot n^2 \cdot \mathcal{H}_i(n) \), where \( \mathcal{H}_i \) is the \( i \)-th Harmonic number. Since \( \mathcal{H}_i \) is \( \Theta(\log i) \), we obtain that \( \mathbb{E}_C[\text{Steps}_{\exp_\nu}] \) is \( O(n^2 \cdot \log(n)) \).

So, let \( C \) be a configuration reachable from a configuration satisfying the formula \( \Gamma_\nu \) such that \( C \not\models \exp_\nu \). The probability of firing a transition leading to a \( C' \) such that either \( \alpha_{C'} < \alpha_C \) or \( C' \models \exp_\nu \) is at least \( c/n^2 \), where \( c \) is a constant depending only on \( \mathcal{P} \). If \( S_\nu \) is fast, then this bound can be improved to \( c' \cdot (\alpha_C/n^2) \) where \( c' \) is another constant depending only on \( \mathcal{P} \) (this follows by observing that there is \( A \in \mathcal{U} \) such that \( \mathcal{C}(A) \geq \alpha_C/|Q| \)). If this trial is unsuccessful, i.e., \( C \) executes a transition leading to a \( C' \) such that \( \alpha_{C'} = \alpha_C \) and \( C' \not\models \exp_\nu \), another independent trial is performed in \( C' \) (the success probability is again at least \( c/n^2 \), or at least \( c' \cdot (\alpha_C/n^2) \) if \( S_\nu \) is fast). Hence, on average, at most \( n^2/c' \) trials are needed to enter a configuration \( C'' \) such that \( \alpha_{C''} < \alpha_C \) or \( C'' \models \mathcal{F}_\nu \). If \( S_\nu \) is fast, this bound can be improved to \( c'' \cdot (n^2/\alpha_C) \), where \( c'' = 1/c' \).

The case when \( S_\nu \) is very fast is handled similarly. For every configuration \( C \) reachable form a configuration satisfying \( \Gamma_\nu \), we say that a given \( A \in V \) is active in \( C \) if there is a
transition of the form $AB \rightarrow CD$ enabled in $C$. Let $\text{Act}_C$ be the set of all active $A$’s for which there is no active $B$ such that $[A] \neq [B]$ and $[A]$ is reachable from $[B]$ in the graph of SCC’s determined by $G_\nu$. Let $\beta_C = \sum_{A \in \text{Act}_C} d([A])$, where $d([A])$ is the distance of $[A]$. Note that if $\beta_C = 0$, then $C \models \text{Exp}_\nu$. We show that if $\beta_C > 0$, then the expected number of transition fired before reaching a configuration $C'$ such that $\beta_C < \beta_C$ is $O(n^2)$. This clearly suffices, because $\beta_C < |Q|^2$. First, observe that there must be $A, B \in \text{Act}_C$ and a transition of the form $AB \rightarrow CD$ enabled in $C$. Since $A, B \in \text{Act}_C$, the number of $A$’s and $B$’s can only decrease along all executions initiated in $C$ (see Definition 16), and the transition $AB \rightarrow CD$ can be fired at most $\min\{C(A), C(B)\}$ times. If $C'$ is a successor of $C$ such that $\min\{C'(A), C'(B)\} = i$, the probability of firing $AB \rightarrow CD$ in $C'$ is at least $i^2/(c \cdot n^2)$ where $c$ is the total number of transitions with the head $AB$. Since $\min\{C(A), C(B)\} \leq n$, the expected number of transition needed to enter a configuration $C''$ from $C$ such that $C''(A) = 0$ or $C''(B) = 0$ is bounded by $\sum_{i=1}^n (c \cdot n^2)/i^2 = c \cdot n^2 \cdot \mathcal{H}_{n,2} \in \mathcal{O}(n^2)$. It is easy to check that $\beta_{C''} < \beta_C$, and we are done. ▲

**Theorem 18.** If $\text{Exp}_\nu \neq \emptyset$ and $\mathcal{J}_\nu = \emptyset$, then $\text{Reach}_{S, S'} \in \mathcal{O}(n^c)$ for some computable constant $c$.

**Proof.** Let $C$ be a configuration of size $n$ reachable from a configuration satisfying the formula $\Gamma_\nu$. By using the arguments of Theorem 17, we obtain that $E_C[\text{Steps}_{\text{Exp}_\nu}] = \mathcal{O}(n^3)$. However, after reaching a configuration $C'$ such that $C' \models \text{Exp}_\nu$, it may happen that the transitions specified by $\text{Exp}_\nu$ are disabled only temporarily, i.e., it is still possible to reach a configuration $C''$ from $C'$ such that $C'' \not\models \text{Exp}_\nu$. First, we show that if such a $C''$ is reachable from $C'$, then it is reachable in at most $d$ transitions, where $d$ is a constant depending only on $\mathcal{P}$. This follows by observing that the set of configurations which can reach such a $C''$ is upward closed w.r.t. point-wise ordering, and hence there (effectively) exist finitely many minimal configurations with this property. Each of these minimal configurations can reach a configuration violating $\text{Exp}_\nu$ in a constant number of transitions, and all larger configurations can perform the same sequence and thus reach a configuration violating $\text{Exp}_\nu$. Hence, the $d$ can be chosen as the maximum of these finitely many (computable) constants.

A progress transition is a transition of the form $AB \rightarrow CD$ where $AB \in \text{Exp}_\nu$ and $AB \rightarrow CD$ generates an edge of $G_\nu$ connecting two different SCC’s of $G_\nu$. Note that the total number of progress transitions fired along a run initiated in $C'$ is bounded by $|Q| \cdot n$. Furthermore, the probability of executing a progress transition in at most $d + 1$ steps is bounded from below by $n^{-2(d+1)}$. Hence, on average, we need to perform at most $n^{2(d+1)}$ executions of length $d + 1$ to fire a progress transition or reach a configuration satisfying $\Box \text{Exp}_\nu$. This implies $\text{Reach}_{S, S'}(n)$ is bounded by $|Q| \cdot n \cdot n^{2(d+1)} \cdot (d + 1)$, which is $\mathcal{O}(n^c)$ where $c = 2d + 3$. ▲

### D Section 6 (Experimental results)

#### Detailed experimental results

| Predicate and parameters | $|Q|$ | $|T|$ | # stages | # leaves | Depth | Bound | Time (secs.) |
|--------------------------|------|------|---------|---------|------|-------|-------------|
| $x_1 \lor \ldots \lor x_n$ | 11   | 2    | 5       | 3       | 2    | $n^2 \cdot \log n$ | 0.103 |
| $x \geq y$               | 5    | 6    | 13      | 8       | 3    | $n^2 \cdot \log n$ | 0.375 |
| $x \geq y$               | 4    | 3    | 9       | 5       | 3    | $n^2 \cdot \log n$ | 0.221 |
| $x \geq y$               | 4    | 4    | 11      | 6       | 3    | $\exp(n)$             | 0.263 |
### Automatic Analysis of Expected Termination Time for Population Protocols

| Flocks-of-bird protocol [11]: $x \geq c$ |  |
|---|---|---|---|---|---|---|
| $c = 2$ | 3 | 6 | 12 | 9 | 2 | $n^3$ | 0.268 |
| $c = 3$ | 4 | 10 | 18 | 15 | 2 | $n^3$ | 0.423 |
| $c = 4$ | 5 | 15 | 22 | 19 | 2 | $n^3$ | 0.597 |
| $c = 5$ | 6 | 21 | 26 | 23 | 2 | $n^3$ | 0.798 |
| $c = 10$ | 11 | 66 | 46 | 43 | 2 | $n^3$ | 3.974 |
| $c = 15$ | 16 | 136 | 66 | 63 | 2 | $n^3$ | 12.121 |
| $c = 20$ | 21 | 231 | 86 | 83 | 2 | $n^3$ | 28.945 |
| $c = 25$ | 26 | 351 | 106 | 103 | 2 | $n^3$ | 58.022 |
| $c = 30$ | 31 | 496 | 126 | 123 | 2 | $n^3$ | 118.855 |
| $c = 35$ | 36 | 666 | 146 | 143 | 2 | $n^3$ | 222.251 |
| $c = 40$ | 41 | 861 | 166 | 163 | 2 | $n^3$ | 366.247 |
| $c = 45$ | 46 | 1081 | 186 | 183 | 2 | $n^3$ | 495.266 |
| $c = 50$ | 51 | 1326 | 206 | 203 | 2 | $n^3$ | 952.841 |
| $c = 55$ | 56 | 1596 | — | — | — | — | TIMEOUT |

| Flocks-of-bird protocol [11]: $x \geq c$ |  |
|---|---|---|---|---|---|---|
| $c = 2$ | 3 | 3 | 12 | 9 | 2 | $n^3$ | 0.256 |
| $c = 3$ | 4 | 5 | 18 | 15 | 2 | $n^3$ | 0.424 |
| $c = 4$ | 5 | 7 | 30 | 27 | 2 | $n^3$ | 0.746 |
| $c = 5$ | 6 | 9 | 54 | 51 | 2 | $n^3$ | 2.541 |
| $c = 7$ | 8 | 13 | 198 | 195 | 2 | $n^3$ | 11.343 |
| $c = 10$ | 11 | 19 | 1542 | 1539 | 2 | $n^3$ | 83.862 |
| $c = 13$ | 14 | 25 | 12294 | 12291 | 2 | $n^3$ | 816.432 |
| $c = 15$ | 16 | 29 | — | — | — | — | TIMEOUT |

| Remainder protocol [8]: $\sum_{1 \leq i \leq m} t \cdot x_i \equiv 0 \pmod{m}$ |  |
|---|---|---|---|---|---|---|
| $m = 2$ | 4 | 7 | 7 | 3 | 3 | $n^3 \cdot \log n$ | 0.198 |
| $m = 3$ | 5 | 12 | 27 | 14 | 3 | $n^3 \cdot \log n$ | 0.811 |
| $m = 4$ | 6 | 18 | 79 | 45 | 3 | $n^3 \cdot \log n$ | 4.062 |
| $m = 5$ | 7 | 25 | 225 | 134 | 3 | $n^3 \cdot \log n$ | 12.479 |
| $m = 7$ | 9 | 42 | 1351 | 846 | 3 | $n^3 \cdot \log n$ | 88.856 |
| $m = 9$ | 11 | 63 | 7035 | 4502 | 3 | $n^3 \cdot \log n$ | 543.931 |
| $m = 10$ | 12 | 75 | — | — | — | — | TIMEOUT |

| Average-and-conquer protocol [9]: $x \geq y$ with parameters $m$ and $d$, assuming $x \neq y$ |  |
|---|---|---|---|---|---|---|
| $m = 3, d = 1$ | 6 | 21 | 41 | 25 | 3 | $n^3 \cdot \log n$ | 1.982 |
| $m = 3, d = 2$ | 8 | 36 | 1948 | 1038 | 5 | $n^3 \cdot \log n$ | 98.711 |
| $m = 5, d = 1$ | 8 | 36 | 1870 | 1119 | 4 | $n^3$ | 80.097 |
| $m = 5, d = 2$ | 10 | 55 | — | — | — | — | TIMEOUT |
| $m = 7, d = 1$ | 10 | 55 | — | — | — | — | TIMEOUT |
| $m = 3, d = 3$ | 10 | 55 | — | — | — | — | TIMEOUT |

| Threshold protocol [4]: $\sum_{1 \leq i \leq k} a_i \cdot x_i < c$ |  |
|---|---|---|---|---|---|---|
| $-x_1 + x_2 < 0$ | 12 | 57 | 21 | 14 | 3 | $n^3$ | 3.012 |
| $-x_1 + x_2 < 1$ | 20 | 155 | 131 | 104 | 3 | $n^3$ | 30.314 |
| $-x_1 + x_2 < 2$ | 28 | 301 | — | — | — | — | TIMEOUT |
| $-2x_1 - x_2 + x_3 + 2x_4 < 0$ | 20 | 155 | 1049 | 834 | 3 | $n^3$ | 166.283 |
| $-2x_1 - x_2 + x_3 + 2x_4 < 1$ | 20 | 155 | 1049 | 834 | 3 | $n^3$ | 155.238 |
| $-2x_1 - x_2 + x_3 + 2x_4 < 2$ | 28 | 301 | — | — | — | — | TIMEOUT |
Logarithmic flock-of-birds protocol$^{13}$ $x \geq c$

| $c$    | 4  | 7  | 18 | 15 | 2  | $n^3$ | 0.571 |
|--------|----|----|----|----|----|-------|-------|
| $c = 3$| 8  | 23 | 25 | 25 | 2  | $n^3$ | 1.926 |
| $c = 15$| 10 | 34 | 130| 127| 2  | $n^3$ | 6.144 |
| $c = 31$| 12 | 47 | 514| 511| 2  | $n^3$ | 13.909|
| $c = 63$| 14 | 62 | 1026| 1023| 2  | $n^3$ | 39.382|
| $c = 127$| 16 | 79 | 4098| 4095| 2  | $n^3$ | 81.000|
| $c = 255$| 20 | 119| 8194| 8191| 2  | $n^3$ | 395.650|
| $c = 1023$| 22 | 142| 1677| 1677| 2  | $n^3$ | 851.861|
| $c = 4095$| 24 | 167| —  | —  | —  | TIMEOUT|

---

$^{10}$ Protocol of Example 3.

$^{11}$ Protocol of Example 2 without the tie-breaking rule $ba \rightarrow bb$ (only correct if $x \neq y$).

$^{12}$ Protocol of Example 2.

$^{13}$ An adapted version of the protocol of [7, Sect. 3] without so-called $k$-way transitions.