On the Power of Anonymous One-Way Communication

Dana Angluin[1], James Aspnes[1]*, David Eisenstat[2], and Eric Ruppert[3]**

[1] Yale University, Department of Computer Science
[2] University of Rochester, Department of Computer Science
[3] York University, Department of Computer Science and Engineering

Abstract. We consider a population of anonymous processes communicating via anonymous message-passing, where the recipient of each message is chosen by an adversary and the sender is not identified to the recipient. Even with unbounded message sizes and process states, such a system can compute only limited predicates on inputs held by the processes. In the finite-state case, we show how the exact strength of the model depends critically on design choices that are irrelevant in the unbounded-state case, such as whether messages are delivered immediately or after a delay, whether a sender can record that it has sent a message, and whether a recipient can queue incoming messages, refusing to accept new messages until it has had a chance to send out messages of its own. These results may have implications for the design of distributed systems where processor power is severely limited, as in sensor networks.

1 Introduction

We introduce and study certain variants of the population protocol model [2, 3] modified to use forms of one-way communication progressively more similar to those of traditional asynchronous message-passing. In the population protocol model, finite-state agents interact in pairs, updating their states according to a joint transition function whose value depends upon the previous states of both agents. Because the new state of both agents may depend on the prior state of the other, we call such an interaction a **bidirectional interaction**. Protocols in this model must work correctly regardless of the order in which these bidirectional interactions occur. Motivating scenarios include models of the propagation of trust in populations of agents [10] and interactions of passively mobile sensors [3]. Similar models of pairwise interaction have been used to study the propagation of rumors in a population of agents [9] and to justify the Chemical Master Equation [13], suggesting that the model of population protocols may be fundamental in several fields.

* Supported in part by NSF grants CNS-0305258 and CNS-0435201.
** Supported in part by the Natural Sciences and Engineering Research Council of Canada.

Because the agents in a population protocol have only a constant number of states, it is impossible for them to adopt distinct identities, making them effectively anonymous. An agent encountering another agent cannot tell in general whether it has interacted with that agent before. Despite these limitations, populations of such agents can compute surprisingly powerful predicates on their initial states under a reasonable global fairness condition. When each agent may interact with every other agent, any predicate over the counts of initial states definable in Presburger arithmetic is computable [3]. When each agent has only a bounded set of neighbors with which it can interact, linear space computable predicates are computable [2].

The assumption of bidirectional interaction, however, may be unrealistic in the context of sensor networks, where radio communication, even between nearby sensors, may not be bidirectional. Moreover, one-way message-passing primitives may be easier to implement in practice. In this paper, we study how the power of the population protocol model changes when the assumption of bidirectional interaction is replaced by certain forms of one-way communication. The pairwise interactions are split into separate send and receive events that each may affect only a single agent.

We consider the effect of two primary attributes of the models: (1) send and receive events may occur simultaneously (**immediate** delivery) or may involve delayed messages subject to various constraints (**delayed** delivery and **queued** delivery), and (2) a sender may be allowed to change its state as a result of sending a message (**transmission**), or not (**observation**). The transmission model is more typical in distributed computing. A web page that increments its counter in response to a visit is an example of such an interaction. Examples of interactions that fit the observation model, where the receiver observes an unknowing sender, include a person reading a post to a discussion forum or a device reading a passive RFID (radio frequency identification) tag. (Of course to preserve the anonymity of our model, the device would not get a unique identity from the tag.) For models with delayed messages, we also consider whether the number of messages in transit is linearly bounded in the population size. Precise definitions are given in Sect. 2.

1.1 Comparing the Models

Comparing the computational power of the resulting models highlights the differences between the forms of one-way communication we consider. As in [3], we assume that every pair of agents eventually come into contact with each other, and seek to characterize the class of predicates on multisets of inputs that are stably computable by protocols in each model. Let $\#(a)$ denote the number of agents assigned input a in the initial configuration. We consider three kinds of predicates: (1) Threshold: $\#(a) \geq t$, (2) Modulo: $\#(a) \equiv j \pmod{k}$, and (3) Comparison: $\#(a) \geq \#(b)$. These three kinds of predicates turn out to be well-suited for characterizing the power of the various versions of our model but they arise in some interesting distributed tasks. The threshold function is applicable to a motivating example of [3]: a network of sensors monitoring individual birds could detect when at least five birds in the flock have an elevated temperature in order to raise an alarm of a possible epidemic. The modulo-k predicate is useful if the system must determine whether processes can be evenly partitioned into groups of size k. A majority voting scheme could use a comparison predicate.

More complicated predicates can be built up from the basic predicates using Boolean operators. We define classes of predicates on finite multisets of symbols from a finite input alphabet Σ as follows. $\mathbf{TH}_k$ is the Boolean closure of all threshold predicates $\#(a) \geq t$ where $a \in \Sigma$ and $t \leq k$. Predicates in $\mathbf{TH}_k$ are determined by the multiplicities of input symbols up to a maximum of k; in particular, predicates in $\mathbf{TH}_1$ are determined by the presence or absence of each input symbol. $\mathbf{TH}_*$ is the union of $\mathbf{TH}_k$ over all positive integers k. **REG** is the Boolean closure of all threshold predicates $\#(a) \geq t$ and modulo predicates $\#(a) \equiv j \pmod{k}$ where $a \in \Sigma$, $t \geq 1$, $j \geq 0$, and $k \geq 2$. Predicates in **REG** are those recognizable by finite-state acceptors when fed the input symbols in any order. **SLIN** is the class of semilinear predicates over multisets of symbols from Σ, that is, the class definable by Presburger predicates [21] over the counts of symbols from Σ. It is the Boolean closure of threshold predicates, modulo predicates and comparisons of linear combinations of input multiplicities. We have the following relationships between these classes:

$$\mathbf{TH}_1 \subset \mathbf{TH}_2 \subset \ldots \subset \mathbf{TH}_* \subset \mathbf{REG} \subset \mathbf{SLIN},$$

where the containments are strict. The predicate $(\#(a) = 1) \wedge (\#(b) \geq 3)$ is in $\mathbf{TH}_3$ but not $\mathbf{TH}_2$. The predicate $(\#(a) \geq 13) \vee (\#(b) \equiv 3 \pmod{5})$ is in **REG**, but not $\mathbf{TH}_*$, which does not contain the modulo predicates. The predicate $((3 * \#(a) + 1) < (5 * \#(b))) \vee (\#(b) = 2)$ is in **SLIN** but not **REG**, which does not contain the comparison predicate. Every predicate in **SLIN** is stably computable by a population protocol in the standard two-way model [3]; whether other predicates are stably computable in this model is open.

We define the class of core-**REG** predicates over an alphabet Σ as follows. A finite multiset of elements of Σ is k-**rich** if it contains each element of Σ with multiplicity at least k. Given a predicate P and a nonnegative integer k, define the k-**core** of P to be $P \wedge Q$ where Q is the property of being k-rich. Then a predicate P is in **core-REG** if some k-core of P is in **REG**.

Our results for the finite-state models appear in Sect. 3. A summary is provided in Fig. 1. For each variant of the model, the corresponding box in the figure describes known facts about the class of predicates that can be stably computed.

The power of the delayed observation model is exactly $\mathbf{TH}_1$; protocols can only detect the presence or absence of each input symbol. The power of the immediate observation model is exactly $\mathbf{TH}_*$. A protocol in this model may determine the multiplicity of each input symbol up to some fixed limit k. Thus, this model is strictly stronger than the delayed observation model. The power of the immediate and delayed transmission models properly includes **REG**, but does not include the comparison predicate. Thus, these two models are strictly

	observation	transmission	transmission with linear message bound
immediate	$=$ **TH**$_*$ (Thm 5)	$\supset$ **REG** (Thm 9) $\not\supseteq$ comparison (Cor 12) $=$ two-way $\cap$ core-**REG** (Thm 13)	*not applicable*
delayed	$=$ **TH**$_1$ (Thm 1)	$\supseteq$ immediate transmission $\not\supseteq$ comparison (Cor 12) $\subseteq$ queued $\cap$ core-**REG** (Thm 11)	$=$ immediate transmission (full paper)
queued	*not applicable*	$\supseteq$ two-way $\supseteq$ **SLIN** (Thm 14 & [3])	$=$ two-way $\supseteq$ **SLIN** (Thm 14 & [3])

Fig. 1. Summary of our results for finite-state models.

more powerful than the immediate observation model. The immediate transmission model is strictly weaker than the standard two-way model and the delayed transmission model is strictly weaker than the queued transmission model.

The queued transmission model, which is essentially an anonymous finite-state version of the usual asynchronous message-passing model, is at least as powerful as the standard two-way model, and equal to it in power with a linear bound on messages in transit. Without such a bound, the queued transmission model admits protocols that spawn an unbounded number of new simulated agents; the exact characterization of the power of the model in this case is an open question.

1.2 Anonymous Communication and Fairness

The question of what computations can be performed in anonymous systems, where processes start with the same state and the same programming, has a long history in theoretical distributed computing. Many early impossibility results such as [1] assume both anonymity and symmetry in the communication model, which limits what can be done without some mechanism for symmetry-breaking. See [12] for a survey of many such impossibility results. More recent work targeted specifically at anonymity has studied what problems are solvable in message-passing systems under various assumptions about the initial knowledge of the processes [6, 7, 22], or in anonymous shared-memory systems where the properties of the supplied shared objects can often (but not always, depending on the details of the model) be used to break symmetry and assign identities [4, 5, 8, 11, 15, 17–20, 23]. This work has typically assumed few limits on the power of the processes in the system other than the symmetry imposed by the model.

Agents in the population protocol model are assumed to be finite-state. Together with a transition rule that depends only on the states of the two interacting agents, the finite-state assumption naturally yields a model in which agents are effectively anonymous. This makes the model much weaker than a typical message-passing model, where processes have identities. On the other hand, in one respect the population protocol model is much stronger than a typical message-passing model: communication between two interacting agents is instantaneous and bidirectional.

Implicit in the structure of a population protocol is that message-passing is rather strongly anonymous: not only does a receiver not learn the identity of the sender, but a sender cannot direct its message to a particular receiver. This is unusual even in anonymous message-passing models, which typically assume that a process can use some sort of local addressing to direct messages to specific neighbors. It also leads to a very weak message-passing model if we adopt the traditional fairness assumption of eventual delivery to all destinations of any message that is sent often enough. We show in Sect. 4 that even with unbounded states and message lengths, this fairness condition provides only enough power to detect the presence or absence of each possible input, giving additional support for the global fairness condition used in the rest of the paper.

2 Model

We give a model that unifies both the standard asynchronous message-passing model, adapted so that processes are anonymous and no longer control the destinations of their messages; and the population protocol model of [3], restricted so that interactions between two agents are one-way. We first describe these two models separately, and then define our combined model and its variations.

2.1 Asynchronous message-passing

In an **asynchronous message-passing model**, processes communicate by sending messages. A process may spontaneously send a message at any time, which is delivered to a recipient at some later time. The recipient may respond to the message by updating its state and possibly sending one or more messages. In the standard model, senders can choose the recipients of their messages, and recipients are aware of the identities of the senders of messages they receive; in our model, we drop these assumptions.

Message-passing systems may be vulnerable to a variety of failures, including failures at processes such as crashes or Byzantine faults, and failures in the message delivery system such as dropped or duplicated messages. We assume fault-free executions. Since message delivery is asynchronous, making any sort of progress requires adopting a fairness condition to exclude executions in which indefinitely-postponed delivery becomes equivalent to no delivery.

A minimal fairness condition might be that if some process sends a particular message m infinitely often, then each other process receives the same message m infinitely often. In Sect. 4, we show that this minimal fairness condition is not enough to solve more than a small class of problems, even in a message-passing model with unbounded states and message sizes. So instead we adopt a stronger global fairness condition derived from that used in [3]. We define this condition formally below.

2.2 Population Protocols

We call this model the **standard two-way model** of population protocols to distinguish it from the one-way models we define in Sect. 2.3. A **population protocol** [3] consists of a finite population V of agents with states drawn from a finite state set Q. The identities of agents $v \in V$ are used in describing the model, but are not accessible to the agents themselves. Agents interact in pairs; each interaction updates the state of both agents according to a joint transition rule $\delta : Q \times Q \to Q \times Q$ that maps pairs of states $(p, q) \mapsto (p', q')$. Interactions are asymmetric: the left-hand agent is called the **initiator** and the right-hand agent the **responder**. We think of the initiator as the **sender** of a message and the responder as the **receiver** of a message, but in the original model information may flow in both directions.

A **configuration** C of a population protocol describes the states of all agents; the state of agent v in C is denoted $C(v)$. An interaction takes a configuration C to a new configuration C' by updating the states of exactly two agents. If there is a transition from C to C', we write $C \to C'$. We write $C \xrightarrow{*} C'$ if there is a sequence of zero or more transitions that transform C to C'. In this case, C' is said to be **reachable** from C.

A **computation** is a sequence of configurations $C_0, C_1, C_2, \ldots$ with $C_i \to C_{i+1}$ for each i. Computations may be finite or infinite. Achieving positive results in this model depends on excluding computations in which subpopulations are isolated from each other or are only permitted to communicate at inopportune times. In [3], a computation was defined to be **fair** if for every configuration C that occurs infinitely often in the computation, if $C \to C'$, then C' also occurs infinitely often in the computation. This condition is intended to capture the effect of a probability 1 property without directly incorporating probabilities; for example, if pairs of agents are selected at random to interact, the resulting computation is fair with probability 1. In Sect. 2.4 we generalize this fairness condition to deal with messages in transit.

To allow agent states to contain information other than the output value, states in Q are mapped to outputs from a finite output alphabet Y by an **output function**. Similarly, inputs from a finite input alphabet Σ are mapped to states in Q by an **input function** $I : \Sigma \to Q$. An input X assigns a symbol from Σ to each agent in the population; the corresponding input configuration is denoted $I(X)$. Because the agents are anonymous and every pair may interact and because we consider predicates, it is immaterial which agent is assigned each symbol [3], and we may consider inputs and configurations as finite multisets. Multisets are denoted by upper-case letters, and individual elements are denoted by lower-case letters. We use the notation $A + B$ for the union of multisets A and B, and $A + a$ for the union of multisets A and $\{a\}$. The notation kA, where k is a non-negative integer and A is a multiset, is used for the multiset in which every element occurs with k times its multiplicity in A.

A configuration C is **output-stable** if, for any C' reachable from C by a sequence of zero or more transitions, the vector of output values in C' is equal to the vector of output values in C. A predicate P on finite multisets X of elements from Σ is **stably computed** by a given protocol if every fair computation of the protocol from an input configuration $I(X)$ eventually reaches an output-stable state in which all agents output the correct value for $P(X)$. As an example, a protocol with inputs $\{0, 1\}$ and identity input and output functions in which $(1, q) \mapsto (1, 1)$ and $(0, q) \mapsto (0, q)$ stably computes the OR of all the initial inputs. Output-stability does not require that the states of individual agents do not change; it is enough that any changes are not visible in the outputs of agents. This fact is exploited by protocols that include "leader bits" or similar tokens that move freely among agents without affecting the output after convergence.

2.3 One-way Communication in Population Protocols

To model one-way communication in a population protocol, we restrict the transition function so that the new state of the sender does not depend on the state of the receiver. There are two natural ways to do this. We may stipulate that an interaction does not change the state of the sender at all. This is an **observation** model, in which the sender is passively observed by the receiver. Formally, if $(p, q) \mapsto (p', q')$, then $p' = p$.

Alternatively, in a **transmission** model, the sender of a message can detect that it has sent the message, but learns nothing about the state of the recipient. This corresponds to requiring that for any two transitions $(p_1, q_1) \mapsto (p'_1, q'_1)$ and $(p_2, q_2) \mapsto (p'_2, q'_2)$, that if $p_1 = p_2$ then $p'_1 = p'_2$. Since each transmission model formally includes the corresponding observation model, it is at least as powerful.

In both cases, the result is that communication is **one-way**: only the receiver obtains any information about its partner's state. We will refer to any protocol with such one-way communication as a **one-way population protocol**. In an **immediate delivery** model, these are the only changes to the basic population protocol model. Immediate delivery models can be thought of as models of interaction.

However, the standard asynchronous message-passing model assumes that (1) processes cannot be compelled to send messages if they do not want to and (2) messages may not be delivered immediately. Including the first feature requires classifying states based on whether or not they are enabled to send messages. For the non-immediate models we assume that send events only occur for states q in some subset Q_S of Q; states in Q_S are called **send-enabled**.

To address (2), we split a joint transition into two separate sending and receiving events. Configurations are extended to include two components: the **population configuration**, giving the states of all the agents, and the multiset of **messages in transit**, which for simplicity we take to be pairs consisting of sender ids and elements of the state space Q. (The sender ids are used only in the model discussed in Sect. 4.) Each transition $(p, q) \mapsto (p', q')$ is split into a **send event** which changes the state of an agent from p to p' and adds p to the multiset of messages in transit, and a **receive event** in which p is removed from the multiset of messages in transit and the state of some agent is updated from q to q'. As with immediate delivery, we can consider both a **delayed transmission** model in which a sender can record that it sent a message and a **delayed observation** model in which a sender cannot.

Both the delayed transmission and delayed observation models require that any agent be prepared to receive any message in any state. This may not give an agent enough time to respond to a message before the next incoming message arrives. With **queued** delivery, an agent can enter into a state in which it refuses to receive messages. Formally, we assume that only states in some subset Q_R of Q can receive messages; states in Q_R are called **receive-enabled**. In the delayed or queued models, the transition rule becomes a partial function whose domain is $Q_S \times Q_R$, where $Q_R = Q$ for delayed transmission or delayed observation and $Q_R \subseteq Q$ for queued transmission.[4]

Separating message transmission and receipt creates the possibility that an agent may receive its own message. This can be thought of as including self-loops in the **interaction graph** controlling which agents can communicate, which we otherwise take to be complete. In general, we assume that this does not occur in the immediate delivery models (which are interaction models) but may occur in the delayed and queued delivery models (once a message is sent it may be delivered to anyone.) In the full paper it will be shown that this has at most a minor effect on the power of the models we consider.

2.4 Fairness Revisited

We generalize the fairness condition from [3], given in Sect. 2.2, to deal with messages in transit. Because we permit partial transition rules, we also extend the definition of a computation to be any sequence of configurations $C_0, C_1, \ldots$ such that for each i, $C_i \to C_{i+1}$ or $C_{i+1} = C_i$. This does not change the reachability relation on computations, but it does permit a simpler definition of fairness that applies to computations that terminate (when no further transitions are enabled) as well as non-terminating ones.

Let $C_0, C_1, \ldots$ be an infinite computation. A population configuration C **occurs infinitely often** in this computation if there are infinitely many j such that C is the population component of C_j. A population configuration C is **infinitely often enabled** in this computation if there exist infinitely many j such that C is the population component of some configuration reachable from C_j. We say that this computation is **fair** if for every population configuration C that is infinitely often enabled in the computation, C occurs infinitely often in the computation.[5]

[4] In an observation model an agent cannot leave a non receive-enabled state; thus we do not consider a queued observation model.

[5] The antecedent of the condition may never be satisfied if the state space is unbounded, as is often implicit in the standard asynchronous message-passing model.

3 The Power of One-Way Population Protocols

We investigate what predicates on the multiset of input symbols are stably computable in the models defined in Sect. 2.3 Note that for each of these models, a direct product construction permits parallel execution of a finite collection of different protocols, and therefore the set of stably computable predicates is closed under Boolean combinations in each model.

3.1 Delayed Observation

The delayed observation model is very weak: an agent is unaware that it has sent a message, and may receive messages that were sent in the distant past. This effectively means that an agent may at any time receive messages containing any state that has ever appeared in the computation. As a result, the most that a protocol can do is detect the presence or absence of particular symbols in the input.

Theorem 1. $\mathbf{TH}_1$ *is the class of predicates stably computable in the delayed observation model.*

Lemma 2. *Let P be a predicate in* $\mathbf{TH}_1$*. Then P is stably computable by a delayed observation protocol.*

Proof. Each state is a subset of the finite input alphabet. Input a is mapped to $\{a\}$. Whenever an agent in state q receives a message q', it updates its state to $q \cup q'$. The output function maps q to the value of P on this set of inputs. By the fairness condition, the value of every state must converge to the set of inputs present in the initial configuration, and the outputs will then be the correct value of P. □

The following cloning technique applies to both the observation models.

Lemma 3. *Suppose a protocol in the delayed or immediate observation model stably computes the protocol P. Suppose $C \overset{*}{\to} D$ and v is an agent such that $C(v) = p$ and $D(v) = q$. Let v' be a new agent, and let C' be C with v' in state p and let D' be D with v' in state q. Then $C' \overset{*}{\to} D'$.*

Proof sketch. We use the computation from C to D to construct a computation from C' to D' by duplicating every message eventually delivered to v in the computation from C to D, and delivering one copy to v and one copy to v'. The agents sending the duplicate messages are unaffected by the change because these are observation models. □

Lemma 4. *Suppose P is stably computed by a delayed observation protocol. Then P is in* $\mathbf{TH}_1$*.*

Proof. We show that for any multiset X of inputs, if $a \in X$ then $P(X + a) = P(X)$, which implies that P is determined solely by the presence or absence of each input symbol and hence is in $\mathbf{TH}_1$.

Consider the finite graph whose nodes are configurations reachable from $I(X)$ that contain no messages in transit, with a directed edge from C to C' if $C \xrightarrow{*} C'$. A **final** strongly connected component of this graph is one from which no other strongly connected component of the graph is reachable. From $I(X)$ we can reach a configuration in a final strongly connected component $\mathcal{F}$ of this graph. Let $\hat{\mathcal{F}}$ denote all the configurations D, including those with undelivered messages, such that $C \xrightarrow{*} D$ for some $C \in \mathcal{F}$. For any configurations D and D' in $\hat{\mathcal{F}}$, $D \xrightarrow{*} D'$ by first delivering all messages in D. This implies that all configurations in $\hat{\mathcal{F}}$ are output-stable.

The set T of states that occur in configurations in $\hat{\mathcal{F}}$ is closed, that is, if $p, q \in T$ and $(p, q) \mapsto (p, q')$, then $q' \in T$. To see this, assume not. Then, take a configuration D in $\hat{\mathcal{F}}$ that contains p and let an agent in state p send a message, putting p into messages in transit. Now mimic a computation from D to a configuration D' in $\hat{\mathcal{F}}$ containing q, leaving the message p undelivered. Then deliver p to an agent in state q, arriving at a configuration in $\hat{\mathcal{F}}$ containing q', a contradiction.

Now consider any a in X. Let C be an output-stable configuration in $\mathcal{F}$ that is reachable from $I(X)$. Let q be the state in configuration C of an agent that began with input a. Then by Lemma 3, a configuration C' equal to C with a new agent in state q is reachable from $I(X + a)$. Because T is closed and the states of C' are all in T, C' is output-stable and therefore $P(X + a) = P(X)$. $\square$

3.2 Immediate Observation

In the immediate observation model, transitions are of the form $(p, q) \mapsto (p, q')$ and there is no multiset of undelivered messages. For any constant k, an immediate observation protocol can count the number of copies of each input symbol up to k, making this model more powerful than the delayed observation model. However, this is also the extent of its power.

Theorem 5. $\mathbf{TH}_*$ *is the class of predicates stably computable in the immediate observation model.*

Lemma 6. *Every predicate in* $\mathbf{TH}_*$ *is stably computable by an immediate observation protocol.*

Proof. By Boolean closure, it suffices to give an immediate observation protocol that stably computes an arbitrary threshold predicate: $\#(a) \geq k$. The states are $0, 1, 2, \ldots, k$. The input map takes a to 1 and every other symbol to 0; the protocol must determine whether there are at least k 1's in the initial configuration. The transitions are $(i, i) \mapsto (i, i + 1)$ for all $i = 1, 2, \ldots, k - 1$ and $(k, i) \mapsto (k, k)$ for all $i = 0, 1, 2, \ldots, k - 1$, where all other transitions leave the argument pair unchanged. The output map takes k to 1 and every other state to 0.

If there are no 1's in the initial configuration, then it never changes. If there are j 1's in the initial configuration for some $0 < j < k$, then any fair computation eventually reaches a configuration in which the only nonzero states are $1, 2, \ldots, j$, and this configuration never changes. In both cases, every output is 0 throughout the computation.

If there are $j \geq k$ 1's in the initial configuration, then any fair computation must reach the configuration in which all states are k, and this configuration never subsequently changes. In this configuration, every output is 1. (The full paper will contain a proof that the number of states used in this protocol is optimal.) $\square$

Consider an immediate observation protocol that stably computes a predicate P. The following property of output-stable configurations of P is very useful. A set $\mathcal{L}$ of finite multisets of elements from some set S is called **linear** if there exist a base element $B \in \mathcal{L}$ and a finite set of periods $P_1, \ldots, P_d$ such that the elements of $\mathcal{L}$ are precisely those of the form $B + m_1 P_1 + \ldots + m_d P_d$, where the m_i are nonnegative integers and the P_i are multisets of elements of S.

Lemma 7. *The set of output-stable accepting (resp., rejecting) configurations is a union of a finite collection of linear sets in which every period consists of a singleton state.*

Proof. A set is **semilinear** if it is a finite union of linear sets. The set $\mathcal{A}$ of output-stable accepting configurations is downward closed, so its complement is upward closed and therefore semilinear by Higman's Lemma [16]. Because the semilinear sets are closed under complement [14], $\mathcal{A}$ is semilinear.

Thus, $\mathcal{A}$ is a finite union of linear sets. Consider one of the linear sets, say $\mathcal{L}$. It has a base element B and a finite collection of periods, $P_1, \ldots, P_d$. Consider the linear set $\mathcal{L}'$ with base B and periods $\{q\}$ for any state q that occurs in some P_i. Clearly, $\mathcal{L} \subseteq \mathcal{L}'$, and we claim $\mathcal{L}' \subseteq \mathcal{A}$, so that replacing each $\mathcal{L}$ by its corresponding $\mathcal{L}'$ gives the decomposition of $\mathcal{A}$ required by the lemma. To see that the claim is true, consider any element C of $\mathcal{L}'$. C consists of B plus multiples of states q in the periods P_i. By taking B plus sufficiently large multiples of the P_i's we get a configuration $C' \in \mathcal{L}$ such that $C \subseteq C'$. Because $\mathcal{A}$ is downward closed, $C \in \mathcal{A}$. The same proof works for the output-stable rejecting configurations. $\square$

Lemma 8. *Let P be a predicate that is stably computed by a protocol in the immediate observation model. Then P is in* $\mathbf{TH}_*$.

Proof. By Lemma 7, the output-stable accepting configurations of the protocol are the union of a finite collection of linear sets $\mathcal{L}_i$ with singleton periods, and similarly for the output-stable rejecting configurations, where the linear sets are $\mathcal{M}_j$. Let k be one more than the maximum cardinality of any of the bases of the $\mathcal{L}_i$'s or $\mathcal{M}_j$'s.

Consider any finite multiset X of inputs for which $\#(a) \geq k$ for some a. Suppose X is accepted; a similar proof applies if X is rejected. If $I(a) = q$

then q occurs with multiplicity at least k in $I(X)$. Consider any output-stable configuration D reachable from C. D is in one of the linear sets $\mathcal{L}_i$. Because the multiplicity of q exceeds the cardinality of the base of $\mathcal{L}_i$, some agent v in state q in $I(X)$ must have state q' in D, where q' is the singleton state of one of the periods of $\mathcal{L}_i$. Thus, $D + q'$ is also in $\mathcal{L}_i$, so $D + q'$ is output-stable and accepting. However, by Lemma 3, $I(X) + q \xrightarrow{*} D + q'$, and $I(X) + q = I(X + a)$, so $X + a$ must also be accepted by the protocol. Thus, for any input symbol a, if $\#(a) \geq k$ in input X, $P(X + a) = P(X)$, which implies that P is in $\mathbf{TH}_k$, and therefore in $\mathbf{TH}_*$. □

3.3 Immediate and Delayed Transmission

The immediate and delayed transmission models can stably compute all threshold and modulo predicates, and therefore all predicates in **REG**. Thus they are more powerful than the immediate observation model.

Theorem 9. *Predicates in* **REG** *are stably computable in the immediate and delayed transmission models.*

Proof. By Boolean closure, it suffices to prove that all the threshold and modulo predicates are stably computable in both models. We assume data values in the set $S = \{0, 1, \ldots, k\}$ and a commutative monoid operation $g(d_1, d_2)$ on this set with identity 0. We describe a protocol to compute the g-sum of all the data values in the input states. The states are (b, d), where $b \in \{0, 1\}$ is a leader bit, and $d \in S$. A transition with sender state (b, d) and receiver state (b', d') updates the sender state to $(0, d)$ and the receiver state to $(1, g(d, d'))$ if $b = b' = 1$, to $(1, d)$ if $b = 1$ and $b' = 0$, and leaves it unchanged otherwise.

The following invariant is preserved by each transition: the g-sum of the data values of those agents and messages in transit with leader bit equal to 1 is the g-sum of all the input data values. By fairness, eventually there will be just one agent (or message in transit) with leader bit equal to 1, and its data value will be the correct g-sum of all the input data values. Again by fairness, that data value will be copied to every agent as the leader bit is passed among them.

For the threshold predicate $\#(a) \geq k$, a is mapped to $(1, 1)$ and all other input symbols are mapped to $(0, 0)$. State (b, d) is mapped to output 1 if and only if $d = k$. The monoid sum $g(d_1, d_2)$ is $\min(k, d_1 + d_2)$. For the modulo predicate $\#(a) \equiv j \pmod{(k + 1)}$ we take the same input function, map (b, d) to output 1 if and only if $d = j$, and take the monoid sum $g(d_1, d_2)$ to be $(d_1 + d_2) \mod (k + 1)$. □

The following theorem shows that **REG** does not exhaust the class of predicates stably computable in the immediate and delayed transmission models. Let \$ be a symbol not in Σ and P a predicate over alphabet Σ. Define $P_\$$ be the predicate over $\Sigma \cup \{\$\}$ that is true if there are at least two agents in the population, there is exactly one \$ in the input, and P is true on the multiset of other input symbols. For example, if P is the comparison predicate, $\#(a) > \#(b)$, then $P_\$$ is the predicate that is true when the input contains exactly one \$ and more a's than b's, which is not in **REG**.

Theorem 10. *Let P be a predicate over Σ that is stably computable in the standard two-way model. Then $P_{\$}$ is stably computable in the immediate and delayed transmission models.*

Proof sketch. We run three protocols in parallel, one to verify that there are at least two agents in the population, one to verify that there is just one \$ in the input, and one that performs a simulation of the two-way protocol computing P on the rest of the input symbols, assuming that the first two conditions are satisfied. The first two conditions are in $\mathbf{TH}_2$ and $\mathbf{TH}_1$, respectively, and are therefore computable, by Theorem 9.

The idea of the simulation is to use the unique input \$ to generate a leader token that passes from one simulated agent to another in the population. The leader token nondeterministically chooses a simulated agent to be the initiator and picks up its state (leaving behind a place marker), chooses another simulated agent to be the responder, updates the responder's state and waits until it returns to the place marker to update the simulated initiator's state, and then repeats the whole sequence. The state of the extra agent (that had the input \$) is updated to reflect the outputs of the simulated agents. □

The following theorem is an important restriction on the power of both transmission models; its proof will appear in the full paper. Recall the definitions of k-rich, k-core, and core-**REG** from Sect. 1.1.

Theorem 11. *Let P be a predicate that is stably computable by an immediate or delayed transmission protocol. Then for some k, the k-core of P is in* **REG**.

Let P be the comparison predicate, $\#(a) > \#(b)$. The 2-core of $P_{\$}$ is empty, and therefore in **REG**, but no k-core of P is in **REG**, yielding the following corollary.

Corollary 12. *The comparison predicate is not stably computable in the immediate or delayed transmission models.*

By generalizing Theorem 10 and combining it with Theorem 11, we get the following characterization of the power of immediate transmission protocols; its proof will appear in the full paper.

Theorem 13. *A predicate P is stably computable in the immediate transmission model if and only if P is stably computable in the standard two-way model and some k-core of P is in* **REG**.

3.4 Queued Transmission

The queued transmission model is the most powerful of the models we consider; it is capable of simulating the standard model of two-way population protocols, and (if no bounds are placed on the size of the multiset of messages in transit) can generate an unbounded number of additional simulated agents. The intuition is that a simulation can use messages in transit to represent agents of the standard population protocol, and collect pairs of simulated agents at real nodes to simulate transitions. To avoid deadlocks, we also include a floating population of "release messages" that trigger nodes to release the simulated agents collected so far.

Theorem 14. *A predicate P is stably computable by a standard two-way population protocol if and only if P is stably computable in the queued transmission model using at most a linear number of messages in transit.*

A detailed proof is given in the full paper. The full paper will also include a proof that the delayed transmission model with a linear bound on messages in transit is equivalent in power to the immediate transmission model, based on Theorems 11 and 14.

4 Local Fairness Is Weak Even with Unbounded States

In this section, we consider an anonymous message-passing model with the following local fairness condition: if some process sends a particular message m infinitely often, then each process receives message m infinitely often. This model turns out to be surprisingly weak. Even if the states of processes and the lengths of messages may grow without bound, protocols in this model cannot distinguish two multisets of inputs if the same set of values appears in each. Since this model subsumes the finite-state models of the preceding sections, it demonstrates why the stronger global fairness condition assumed there is necessary. The definition of $\mathbf{TH}_k$ generalizes straightforwardly to an infinite alphabet Σ.

Theorem 15. *Let Σ denote the (finite or infinite) set of possible input values. A predicate P on finite multisets of elements from Σ is stably computable in the asynchronous message-passing model with the weak fairness condition if and only if P is in $\mathbf{TH}_1$.*

Proof. Consider the delayed observation protocol from the proof of Lemma 2 to determine the set of all inputs that occur in the initial configuration, modified so that each agent sends its state every time it runs. Clearly every message is a subset of the initial set of input values, so there are only finitely many possible messages in each computation. Every message sent by a process with input value x contains the element x, and it sends infinitely many messages, so eventually every process receives a message containing x. Thus, the state of every process eventually consists of the initial set of input values.

For the converse, assume that we have an algorithm to stably compute a predicate P, and let A and B be two multisets of values from Σ such that the same set of values appears in each. Let $n = |A|$ and $n' = |B|$. Let C_0 and C'_0 be initial configurations where processes have inputs from A and B, respectively. We construct two executions α and α' starting from C_0 and C'_0. Let $m_1, m_2, \ldots$ be an arbitrary sequence of messages where every possible message appears infinitely often. We construct the executions α and α' in phases, where phase i will ensure that message m_i gets delivered to everyone if that message has been sent enough times. Let C_i and C'_i be the configurations of α and α' at the end of phase i.

Our goal is to prove the following claim: for all $i \geq 0$ and for all $x \in \Sigma$, the state of each process with input x in C_i is the same as the state of each process with input x in C'_i. Assume that we have constructed the first $i-1$ phases of the two executions so that the claim is satisfied. Suppose we run all processes in lock step from C_{i-1} and C'_{i-1} without delivering any messages. There are two cases.

Case (i): Eventually, after r_i rounds, the run from C_{i-1} will have at least n copies of m_i in transit and, after r'_i rounds, the run from C'_{i-1} will have at least n' copies of m_i in transit. Then, the ith phase of α and α' is constructed by running each process for $\max(r_i, r'_i)$ rounds without delivering any messages, and then delivering one copy of m_i to every process. This ensures the claim will be true for C_i and C'_i.

Case (ii): Otherwise, we allow every process to take one step without delivering any messages. (This clearly satisfies the claim for C_i and C'_i.)

It remains to show that both α and α' satisfy the weak fairness condition, and then it will follow from the claim that $P(A) = P(B)$. First, notice that every process takes infinitely many steps in α and α'. If some process v sends a message m infinitely many times in α or α', it will also be sent infinitely many times by a process with the same input value in the other execution (since a process with a particular input experiences the same sequence of events in both executions). Suppose m is never delivered after phase i to some process w in one of the two executions. Eventually, there will be n copies of m in transit in C_j for some $j > i$ and n' copies of m in transit in $C'_{j'}$ for some $j' > i$. Consider the first occurrence of m in the sequence $m_1, m_2, \ldots$ that comes after m_j and m'_j. During the corresponding phase, m will be delivered to every process, including w, a contradiction. Thus, α and α' satisfy the weak fairness condition. $\square$

5 Conclusion

We defined several models incorporating one-way communication and message-passing into population protocols and compared their ability to compute predicates on multisets of inputs. We have fully characterized the power of the delayed and immediate observation models, the immediate transmission model, and the delayed and queued transmission models with a linear bound on messages in transit. The queued transmission model with a linear bound on messages in transit is equivalent in power to the original model of two-way population protocols. In contrast to traditional message-passing systems, the strongest model is the most asynchronous: in the queued transmission model, messages in transit can effectively act as extra storage. An important feature of the queued transmission model is that receivers can exercise flow control over incoming messages; the delayed transmission model, lacking such flow control, is strictly weaker. The problems of characterizing the power of the delayed and queued transmission models with no bound on messages in transit remains open, as does the related problem from [3] of whether the power of standard two-way model is more than **SLIN**.

References

1. Dana Angluin. Local and global properties in networks of processors. In *Proceedings of the 12th ACM Symposium on Theory of Computing*, pages 82–93, 1980.
2. Dana Angluin, James Aspnes, Melody Chan, Michael J. Fischer, Hong Jiang, and René Peralta. Stably computable properties of network graphs. *IEEE/ACM International Conference on Distributed Computing in Sensor Systems*, June 2005.
3. Dana Angluin, James Aspnes, Zoë Diamadi, Michael J. Fischer, and René Peralta. Computation in networks of passively mobile finite-state sensors. In *Proc. 23rd Annual ACM Symposium on Principles of Distributed Computing*, pages 290–299, 2004.
4. James Aspnes, Gauri Shah, and Jatin Shah. Wait-free consensus with infinite arrivals. In *Proceedings of the 34th ACM Symposium on Theory of Computing*, pages 524–533, 2002.
5. Hagit Attiya, Alla Gorbach, and Shlomo Moran. Computing in totally anonymous asynchronous shared memory systems. *Information and Computation*, 173(2):162–183, March 2002.
6. Paolo Boldi and Sebastiano Vigna. Computing anonymously with arbitrary knowledge. In *Proceedings of the 18th ACM Symposium on Principles of Distributed Computing*, pages 173–179, 1999.
7. Paolo Boldi and Sebastiano Vigna. An effective characterization of computability in anonymous networks. In *Distributed Computing, 15th International Conference*, pages 33–47, 2001.
8. Harry Buhrman, Alessandro Panconesi, Riccardo Silvestri, and Paul Vitanyi. On the importance of having an identity or, is consensus really universal? *Distributed Computing*, 18(3):167–176, 2006.
9. D. J. Daley and D. G. Kendall. Stochastic rumours. *Journal of the Institute of Mathematics and its Applications*, 1:42–55, 1965.
10. Zoë Diamadi and Michael J. Fischer. A simple game for the study of trust in distributed systems. *Wuhan University Journal of Natural Sciences*, 6(1–2):72–82, March 2001. Also appears as Yale Technical Report TR–1207, January 2001.
11. Ömer Eğecioğlu and Ambuj K. Singh. Naming symmetric processes using shared variables. *Distributed Computing*, 8(1):19–38, 1994.
12. Faith Fich and Eric Ruppert. Hundreds of impossibility results for distributed computing. *Distributed Computing*, 16(2-3):121–163, September 2003.
13. Daniel T. Gillespie. A rigorous derivation of the chemical master equation. *Physica A*, 188:404–425, 1992.
14. Seymour Ginsburg. *The Mathematical Theory of Context Free Languages.* McGraw-Hill, New York, 1966.
15. Rachid Guerraoui and Eric Ruppert. What can be implemented anonymously? In *19th International Symposium on Distributed Computing*, pages 244–259, 2005.
16. G. Higman. Ordering by divisibility in abstract algebras. *Proceedings of the London Mathematical Society*, 3(2):326–336, 1952.
17. Prasad Jayanti and Sam Toueg. Wakeup under read/write atomicity. In *Distributed Algorithms, 4th International Workshop*, volume 486 of *LNCS*, pages 277–288, 1990.

18. Shay Kutten, Rafail Ostrovsky, and Boaz Patt-Shamir. The Las-Vegas processor identity problem (How and when to be unique). *Journal of Algorithms*, 37(2):468–494, November 2000.
19. Richard J. Lipton and Arvin Park. The processor identity problem. *Information Processing Letters*, 36(2):91–94, October 1990.
20. Alessandro Panconesi, Marina Papatriantafilou, Philippas Tsigas, and Paul Vitányi. Randomized naming using wait-free shared variables. *Distributed Computing*, 11(3):113–124, August 1998.
21. Mojzesz Presburger. Über die Vollständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchem die Addition als einzige Operation hervortritt. In *Comptes-Rendus du I Congrès de Mathématiciens des Pays Slaves*, pages 92–101, Warszawa, 1929.
22. Naoshi Sakamoto. Comparison of initial conditions for distributed algorithms on anonymous networks. In *Proc. 18th ACM Symposium on Principles of Distributed Computing*, pages 173–179, 1999.
23. Shang-Hua Teng. Space efficient processor identity protocol. *Information Processing Letters*, 34(3):147–154, April 1990.