Noname manuscript No.
(will be inserted by the editor)

Stable Leader Election in Population Protocols Requires Linear Time

David Doty · David Soloveichik

Abstract A population protocol *stably elects a leader* if, for all n, starting from an initial configuration with n agents each in an identical state, with probability 1 it reaches a configuration $\mathbf{y}$ that is *correct* (exactly one agent is in a special leader state ℓ) and *stable* (every configuration reachable from $\mathbf{y}$ also has a single agent in state ℓ). We show that any population protocol that stably elects a leader requires $\Omega(n)$ expected "parallel time"—$\Omega(n^2)$ expected total pairwise interactions—to reach such a stable configuration. Our result also informs the understanding of the time complexity of chemical self-organization by showing an essential difficulty in generating exact quantities of molecular species quickly.

Acknowledgements. The authors thank Anne Condon and Monir Hajiaghayi for several insightful discussions. We also thank the attendees of the 2014 Workshop on Programming Chemical Reaction Networks at the Banff International Research Station, where the first incursions were made into the solution of the problem of PP stable leader election. We are also grateful to anonymous reviewers whose comments have significantly improved the presentation.

A preliminary version of this article appeared as [18]; the current version has been revised for clarity, and includes several omitted proofs.

David Doty
Department of Computer Science
University of California, Davis
Davis, CA, USA
E-mail: doty@ucdavis.edu
Author was supported by NSF grants CCF-1619343, CCF-1219274, and CCF-1162589 and the Molecular Programming Project under NSF grant 1317694.

David Soloveichik
Department of Electrical and Computer Engineering
University of Texas, Austin
Austin, TX, USA
E-mail: david.soloveichik@utexas.edu
Author was supported by an NIGMS Systems Biology Center grant P50 GM081879 and NSF grant CCF-1618895.

1 Introduction

Background. Population protocols (PPs) were introduced by Angluin, Aspnes, Diamadi, Fischer, and Peralta [4] as a model of distributed computing in which the agents have very little computational power and no control over their schedule of interaction with other agents. They also can be thought of as a special case of Petri nets and vector addition systems [21, 22], which were introduced in the 1960s as a model of concurrent processing. In addition to being an appropriate model for electronic computing scenarios such as mobile sensor networks, they are a useful abstraction of "fast-mixing" physical systems such as animal populations [24], chemical reaction networks, and gene regulatory networks [12].

A PP is defined by a finite set Λ of *states* that each agent may have,[1] together with a *transition function* $\delta : \Lambda\times\Lambda \to \Lambda\times\Lambda$.[2] Given states $r_1, r_2, p_1, p_2 \in \Lambda$, if $\delta(r_1, r_2) = (p_1, p_2)$ (denoted $r_1, r_2 \to p_1, p_2$)

[1] Some recent work on PPs [1–3, 10, 11, 20] allows the number of states to grow with the number of agents. This paper uses the original model [4] with state set that is constant with respect to the population size. (See section "Related work".)

[2] Some work allows "non-deterministic" transitions, in which the transition function maps to subsets of $\Lambda\times\Lambda$. Our results are independent of whether the transition function is deterministic or nondeterministic in this manner.

and a pair of agents in respective states r_1 and r_2 interact, then their states become p_1 and p_2.[3] A *configuration* of a PP is a vector $\mathbf{c} \in \mathbb{N}^\Lambda$ describing, for each state $s \in \Lambda$, the *count* $\mathbf{c}(s)$ of how many agents are in state s. Executing a transition $r_1, r_2 \to p_1, p_2$ alters the configuration by decrementing the counts of r_1 and r_2 by 1 each and incrementing p_1 and p_2 by 1 each. Possibly some of r_1, r_2, p_1, p_2 are equal to each other, so the count of a state could change by 0, 1, or 2.

Associated with a PP is a set of *valid initial configurations* that we expect the PP to be able to handle.[4] Agents interact in a pairwise manner and change state based on the transition function. The next pair of agents to interact is chosen uniformly at random among the n agents. If no transition rule applies, the interaction is a "null transition" $r_1, r_2 \to r_1, r_2$, in which the agents interact but don't change state. We count the expected number of *interactions* until some event occurs, and then define the "parallel time" until this event as the expected number of interactions divided by the number of agents n. This measure of time is based on the natural parallel model where each agent participates in a constant number of interactions in one unit of time, hence $\Theta(n)$ total interactions are expected per unit time [6]. In this paper all references to "time" refer to parallel time.

In order to define error-free computation in PPs, we rely on to the model of *stable* computation [8]. The model defines computation to be complete when the PP gets to a configuration that is correct[5] and "stable" in the sense that no subsequent sequence of transitions can take the PP to an incorrect configuration. The model of stable computation disallows error even in an "adversarial" schedule of transitions: we require that from every configuration reachable by *any* sequence of transitions from the initial configuration, it is possible to reach to a correct stable configuration. Since the configuration space is finite, stable computation is equivalent to requiring, under the randomized model, that a correct stable configuration is reached with probability 1. Thus, although it may appear at first glance that correctness and expected time are defined with respect to different models of transition sequences—adversarial vs random—due to this equivalence they are both seen to be definable in the randomized model. This notion of stable computation is also equivalent to requiring that every *fair* sequence of transitions reaches a correct stable configuration, where "fair" means that every configuration infinitely often reachable is infinitely often reached [8]. For the arguments of this paper, the most convenient definition of stable computation is the first one, combinatorial in terms of reachability.

A PP works "with a leader" if there is a special "leader" state ℓ, and every valid initial configuration $\mathbf{i}$ satisfies $\mathbf{i}(\ell) = 1$. This is in contrast to a uniform initial configuration ($\mathbf{i}(x) = n$ for some state x and $\mathbf{i}(y) = 0$ for all states $y \neq x$) or an initial configuration only encoding the input ($\mathbf{i}(x_i) = n_i$ for $i \in \{1, \ldots, k\}$ to represent any input $(n_1, n_2, \ldots, n_k) \in \mathbb{N}^k$). It is known that the predicates $\phi : \mathbb{N}^k \to \{0, 1\}$ stably computable by PPs are exactly the semilinear predicates, whether an initial leader is allowed or not [8]. Although the initial leader does not alter the class of computable predicates, it may allow faster computation. For example, the fastest known PPs to stably compute semilinear predicates without a leader take as long as $\Theta(n)$ to converge.[6] In contrast, with a leader, it is known that any semilinear predicate can be stably computed with expected convergence time $O(\log^5 n)$ [6]. Thus, in certain circumstances, the presence of a initial leader seems to give PPs more computational power (e.g., to converge quickly). Angluin, Aspnes, and Eisenstat [6] asked whether polylogarithmic time stable computation of semilinear predicates is possible without a leader; absent a positive answer, the presence of a leader appears to add power to the model.

Statement of main result. Motivated in part by the apparent speedup possible with an initial leader, we ask how quickly a leader may be elected from a configuration lacking one. We pose the problem as follows: design a PP $\mathcal{P}$ with two special states x (the initial state) and ℓ (the leader state, which may or may not be identical to x) such that, for every $n \in \mathbb{N}$,

[3] In the most generic model, there is no restriction on which agents are permitted to interact. If one prefers to think of the agents as existing on nodes of a graph, then it is the complete graph K_n for a population of n agents.

[4] The set of valid initial configurations for a "self-stabilizing" PP is $\mathbb{N}^\Lambda$, where leader election is provably impossible [9]. We don't require the PP to work if started in any possible configuration, but rather allow potentially "helpful" initial configurations as long as they don't already have small count states (see "α-dense" below).

[5] What "correct" means depends on the task. For computing a predicate, for example, Λ is partitioned into "yes" and "no" voters, and a "correct" configuration is one in which every state present has the correct vote.

[6] See "Open questions" for the distinction between time to *converge* and time to *stabilize*. In this paper, the time lower bound we prove is on stabilization.

from the initial configuration $\mathbf{i}_n$ defined as $\mathbf{i}_n(x) = n$ and $\mathbf{i}_n(y) = 0$ for all other states y, has the following property. For every configuration $\mathbf{c}$ reachable from $\mathbf{i}_n$, there is a configuration $\mathbf{y}$ reachable from $\mathbf{c}$ that *has a stable leader*, meaning that in all configurations $\mathbf{y}'$ reachable from $\mathbf{y}$ (including $\mathbf{y}$ itself), $\mathbf{y}'(\ell) = 1$.

There is a simple $O(n)$ expected time PP for stable leader election, with (assuming $x \equiv \ell$) the single transition $\ell, \ell \to \ell, f$. Our main theorem shows that *every* PP that stably elects a leader requires time $\Omega(n)$ to reach a configuration with a stable leader; thus the previous PP is asymptotically optimal. Section 3.3 discusses why some straightforward approaches to proving a time lower bound for leader election fail.

Composing leader election with other tasks. We have motivated the importance of leader election in population protocols, in part, by reference to tasks that seem to require a leader to be present initially. However, a fast leader election protocol would only alleviate the need for an initial leader if the election could be composed with the subsequent task. Simply combining all transitions for the leader election with the transitions for the task does not necessarily work: prior to the conclusion of the leader election, the presence of multiple leaders may result in unintended transitions.

A number of ad-hoc schemes successfully compose leader election with computation that requires a leader. For example, in ref. [4], the stable computation of the "remainder protocol" depends on leader election to stabilize on a single leader which collects the remainder information. As leaders drop out they transfer their information to the surviving leaders. However, to facilitate composition of leader election with a subsequent task, it would be easiest to have a leader election protocol with a stronger termination criterion than the "stabilizing" criterion we study, namely the "terminating" criterion in which the leader "knows" when it has been elected, and only then would it trigger the subsequent task to begin. One way to formalize "knowing when it has been elected" is to require that the system *never* has more than one leader.[7] However, it is simple to prove that leader election with this terminating convention is impossible: Dividing the initial population in half and preventing the two halves from interacting, each half (being a valid initial configuration itself) must elect a separate leader, violating the requirement that its count never exceeds 1.

Thus the question of how to systematically compose leader election with arbitrary downstream computation is itself open. Nevertheless, we show that even if the protocol follows the more liberal stabilizing criterion, in which the leader need not "know" when it has been elected, stabilization to a single leader *still* requires linear time.

Multiple leader states, multiple leaders, and other initial configurations. A more general notion of leader election is to identify a subset $\Psi \subset \Lambda$ of states that are all considered leader states, and to require the PP to eventually reach a configuration $\mathbf{y}$ in which $\sum_{\ell \in \Psi} \mathbf{y}(\ell) = 1$, and this sum is 1 in every configuration reachable from $\mathbf{y}$. This corresponds more appropriately to how leader states actually coordinate computation in PPs: a leader agent must remember some state information in between transitions (hence it changes state while remaining the unique leader). Our techniques actually show this stronger result as well (as explained in Section 3.2). Further, our result implies that a PP cannot elect any fixed quantity of leaders (e.g. exactly 256) or variable quantity of leaders under a fixed bound (e.g. at most 256) in sublinear expected time.

In the simplest formulation of the task of leader election, we always start with n agents in state x (as described above). Can we capture more generally leader election from a configuration "without a pre-existing leader"? Intuitively, we want to exclude initial configurations with states present in small but non-zero count. We can exclude such initial configurations, but allow otherwise deliberately prepared starting conditions, using the notion of α-dense configurations: any state present in the initial configuration has count $\geq \alpha n$. Our general negative result (Theorem 3.8) implies that even starting with best-case initial configurations, as long as, for some constant $\alpha > 0$, they are all α-dense, sublinear time leader election is impossible. An open question relates to weakening the notion of α-dense (see below).

Chemical reaction networks. The main result and proof are stated in the language of PPs; however, the result holds for more general systems that have PPs as a special case. The discrete, stochastic chemical

[7] If it were possible to detect when a leader election protocol such as $\ell, \ell \to \ell, f$ has stabilized, in the sense that each agent carries a bit $\{g, s\}$ in which all agents start with g and only transition to s after a single ℓ exists, then one could consider the product state (ℓ, s) to be the "true" leader state, and (ℓ, g) is considered only a "candidate" leader state that may have count > 1 prior to the stabilization to a single ℓ.

reaction network (CRN) model has been extensively used in the natural sciences to model chemical kinetics in a well-mixed solution [19]. The CRN model is also used prescriptively for specifying the behavior of synthetic chemical systems [14,23]. A CRN is a finite set of *species* (corresponding to PP states) such as X, Y, Z, and *reactions* (corresponding to PP transitions) such as $X + Y \xrightarrow{k_1} Z$ or $Y \xrightarrow{k_2} 2X + Z$. CRNs can be thought of as a generalization of PPs in which spontaneous transitions are possible (unimolecular reactions), the transition may cause the number of agents to change (if the reaction has a different number of products than reactants), and each transition has an associated constant k that affects its probability of being selected.

As an essential form of self-organization, biological cells seem able to precisely control the count of certain molecules (centriole number [15] is a well studied example). How chemical systems transform relatively uncontrolled initial conditions to precisely controlled amounts of desired species is still not well understood. Our negative result applied to CRNs[8] implies that generating with probability 1 an exact positive count of a certain species, whether 1 or 256, is *necessarily* slower ($\Omega(n)$ time) than, for example, destroying all molecules of the species (through the reaction $X \to \varnothing$), which takes $O(\log n)$ time.

Open questions. Although we measure computation time with respect to stabilization—the ultimate goal of stable computation—some work uses a different goalpost for completion. Consider a PP stably electing a leader, and one particular transition sequence that describes its history. We can say the transition sequence *converged* at the point when the count of the leader is the same in every subsequently reached configuration (if the PP is correct, this count should be 1). In contrast, recall that the point of stabilization is when the count of the leader is the same in every subsequently *reachable* configuration (whether actually reached in the transition sequence or not). Measuring time to stabilization in the randomized model, as we do here, measures the expected time until the probability of changing the output becomes 0. To help illustrate the difference between these two subtly different concepts, Section 3.3 shows some examples of PPs that converge before stabilizing.

Our proof shows only that stabilization must take expected $\Omega(n)$ time. However, convergence could occur much earlier in a transition sequence than stabilization. We leave as an open question whether there is a PP that stably elects a leader and converges in expected $o(n)$ time. We reiterate that there are PPs that work with a leader to stably compute semilinear predicates with convergence time $O(\log^5 n)$ [6]. Thus if stable leader election can converge in expected sublinear time, by coupling the two PPs it might be possible to achieve stable computation of arbitrary semilinear predicates with sublinear convergence time.

It should be noted that the optimal stabilization time for stably computing semilinear predicates, even with an initial leader, is still an open question. The stably computing PPs converging in $O(\log^5 n)$ time [6] provably require expected time $\Omega(n)$ to stabilize, and it is unknown whether faster stabilization is possible.

Going beyond stable computation, the open question of Angluin, Aspnes, and Eisenstat [6] asks whether their efficient high-probability simulation of a space-bounded Turing machine by a PP could remove the assumption of an initial leader. That simulation has some small probability $\epsilon > 0$ of failure, so if one could elect a leader with a small probability $\epsilon' > 0$ of error and subsequently use it to drive the simulation, by the union bound the total probability of error would be at most $\epsilon + \epsilon'$ (i.e., still close to 0). However, it remains an open question whether the necessary PP exists.

Our general negative result applies to α-dense initial configurations. However, is sublinear time stable leader election possible from other kinds of initial configurations that satisfy our intuition of not having preexisting leaders? It is known, for example, that for each $0 < \varepsilon < 1$, an initial configuration with $\Theta(n)$ agents in one state and $\Theta(n^\varepsilon)$ in another state can elect a leader in expected time $O(\log^2 n)$ with high probability [6], although this protocol has a positive probability of failure. In Section 3.3 we give an example PP that stably elects a leader in $O(n^{1/2} \log n)$ time starting from an initial configuration with $\Theta(n)$ agents in one state and $\Theta(n^{1/4})$ in another state. In general we want to better characterize the initial configurations for which sublinear time leader election is possible.

[8] Our result holds for any CRN that obeys Theorem 4.3, the precise constraints of which are specified in [17] (those constraints automatically apply to all PPs). Importantly, to generalize to CRNs, we never assume that the count of agents is fixed, but rather use n to indicate the initial count.

Related work. Alistarh and Gelashvili [2] showed that relaxing the requirement of $O(1)$ states to $O(\log^3 n)$ states allows for a leader to be stably elected in expected time $O(\log^3 n)$. (Indeed, our proof technique fails if the number of states is not constant with respect to n.) Alistarh, Aspnes, Eisenstat, Gelashvili, and Rivest [1] have further refined our understanding of PPs with non-constant states by showing: (1) a time complexity lower bound: even with up to $O(\log \log n)$ states, any stable leader election protocol still requires "near linear" ($\Omega(n/\text{polylog } n)$) expected time to stabilize, and (2) a time complexity upper bound: reducing the $O(\log^3 n)$ state requirement of [2] to $O(\log^2 n)$, at the cost of requiring $O(\log^9 n)$ expected time to stabilize. They are furthermore able to apply these techniques to show similar lower and upper bounds for the majority problem [7], where the initial majority among a population of states x and y should eventually occupy the whole population.

Whether a constant bound on the number of states is appropriate depends upon the situation being modeled by the PP. In some settings—e.g., sensor networks—it is reasonable that employing larger "swarms" may be helped by slightly increasing the memory per sensor (say, logarithmically with n). However, when modeling biological regulatory networks, for example, each state corresponds to an existing chemical species, and $O(1)$ states is natural.

2 Preliminaries

If Λ is a finite set (in this paper, of *states*), we write $\mathbb{N}^\Lambda$ to denote the set of functions $\mathbf{c} : \Lambda \to \mathbb{N}$. Equivalently, we view an element $\mathbf{c} \in \mathbb{N}^\Lambda$ as a vector of $|\Lambda|$ nonnegative integers, with each coordinate "labeled" by an element of Λ. Given $s \in \Lambda$ and $\mathbf{c} \in \mathbb{N}^\Lambda$, we refer to $\mathbf{c}(s)$ as the *count of s in* $\mathbf{c}$. Let $\|\mathbf{c}\| = \|\mathbf{c}\|_1 = \sum_{s\in\Lambda} \mathbf{c}(s)$ denote the total number of agents. We write $\mathbf{c} \leq \mathbf{c}'$ to denote that $\mathbf{c}(s) \leq \mathbf{c}'(s)$ for all $s \in \Lambda$. Since we view vectors $\mathbf{c} \in \mathbb{N}^\Lambda$ equivalently as multisets of elements from Λ, if $\mathbf{c} \leq \mathbf{c}'$ we say $\mathbf{c}$ is a *subset* of $\mathbf{c}'$. It is sometimes convenient to use multiset notation to denote vectors, e.g., $\{x, x, y\}$ and $\{2x, y\}$ both denote the vector $\mathbf{c}$ defined by $\mathbf{c}(x) = 2$, $\mathbf{c}(y) = 1$, and $\mathbf{c}(z) = 0$ for all $z \notin \{x, y\}$. Given $\mathbf{c}, \mathbf{c}' \in \mathbb{N}^\Lambda$, we define the vector component-wise operations of addition $\mathbf{c}+\mathbf{c}'$, subtraction $\mathbf{c}-\mathbf{c}'$, and scalar multiplication $m\mathbf{c}$ for $m \in \mathbb{N}$. For a set $\Delta \subset \Lambda$, we view a vector $\mathbf{c} \in \mathbb{N}^\Delta$ equivalently as a vector $\mathbf{c} \in \mathbb{N}^\Lambda$ by assuming $\mathbf{c}(s) = 0$ for all $s \in \Lambda \setminus \Delta$.

The following lemma is used frequently in reasoning about population protocols.

Lemma 2.1 (Dickson's Lemma [16]) *Any infinite sequence $\mathbf{x}_0, \mathbf{x}_1, \ldots \in \mathbb{N}^k$ has an infinite nondecreasing subsequence $\mathbf{x}_{i_0} \leq \mathbf{x}_{i_1} \leq \ldots$, where $i_0 < i_1 < \ldots \in \mathbb{N}$.*

2.1 Population Protocols

A *population protocol (PP)* is a pair $\mathcal{P} = (\Lambda, \delta)$,[9] where Λ is a finite set of *states*, and $\delta : \Lambda\times\Lambda \to \Lambda\times\Lambda$ is the (symmetric) *transition function*. A *configuration* of a PP is a vector $\mathbf{c} \in \mathbb{N}^\Lambda$, with the interpretation that $\mathbf{c}(s)$ agents are in state s. By convention, the value $n \in \mathbb{Z}^+$ represents the total number of agents $\|\mathbf{c}\|$. A *transition* is a 4-tuple $\alpha = (r_1, r_2, p_1, p_2) \in \Lambda^4$, written $\alpha : r_1, r_2 \to p_1, p_2$, such that $\delta(r_1, r_2) = (p_1, p_2)$. This paper typically defines a PP by a list of transitions, with δ implicit. If an agent in state r_1 interacts with an agent in state r_2, then they change states to p_1 and p_2. For every pair of states r_1, r_2 without an explicitly listed transition $r_1, r_2 \to p_1, p_2$, there is an implicit *null* transition $r_1, r_2 \to r_1, r_2$ in which the agents interact but do not change state.

More formally, given a configuration $\mathbf{c}$ and transition $\alpha : r_1, r_2 \to p_1, p_2$, we say that α is *applicable* to $\mathbf{c}$ if $\mathbf{c} \geq \{r_1, r_2\}$, i.e., $\mathbf{c}$ contains 2 agents, one in state r_1 and one in state r_2. If α is applicable to $\mathbf{c}$, then write $\alpha(\mathbf{c})$ to denote the configuration $\mathbf{c} - \{r_1, r_2\} + \{p_1, p_2\}$ (i.e., the configuration that results from applying α to $\mathbf{c}$); otherwise $\alpha(\mathbf{c})$ is undefined. A finite or infinite sequence of transitions (α_i) is a *transition sequence* (or *path*). Applying a finite or infinite transition sequence (α_i) starting at configuration $\mathbf{c}_0$ induces a finite or infinite sequence of configurations $(\mathbf{c}_0, \mathbf{c}_1, \ldots)$ such that, for all $\mathbf{c}_i$ $(i \geq 1)$,

[9] We give a slightly different formalism than that of [8] for population protocols. The main difference is that since we are not deciding a predicate, there is no notion of inputs being mapped to states or states being mapped to outputs. Another difference is that we assume the transition function is symmetric (so there is no notion of a "sender" and "receiver" agent as in [8]; the unordered pair of states completely determines the next pair of states). However, the results of this paper hold even if we allow the transition function to be non-symmetric or even to be non-deterministic (allowing transitions such as $a, b \to c, d$ and $a, b \to x, y$ to coexist).

$\mathbf{c}_i = \alpha_{i-1}(\mathbf{c}_{i-1})$.[10] If a finite transition sequence q, when applied to the starting configuration $\mathbf{c}$, ends with $\mathbf{c}'$, we write $\mathbf{c} \Longrightarrow_q \mathbf{c}'$. We write $\mathbf{c} \Longrightarrow \mathbf{c}'$ if such a transition sequence exists (i.e., it is possible for the system to reach from $\mathbf{c}$ to $\mathbf{c}'$) and we say that $\mathbf{c}'$ is *reachable* from $\mathbf{c}$. If it is understood from context what is the initial configuration $\mathbf{i}$, then say $\mathbf{c}$ is simply *reachable* if $\mathbf{i} \Longrightarrow \mathbf{c}$. Note that this notation omits mention of $\mathcal{P}$; we always deal with a single PP at a time, so it is clear from context which PP is defining the transitions. If a transition $\alpha : r_1, r_2 \to p_1, p_2$ has the property that for $i \in \{1, 2\}$, $r_i \notin \{p_1, p_2\}$, or if $(r_1 = r_2$ and $(r_i \neq p_1$ or $r_i \neq p_2))$, then we say that α *consumes* r_i. In other words, applying α reduces the count of r_i. We similarly say that α *produces* p_i if it increases the count of p_i.

2.2 Time Complexity

In any configuration the next interaction is chosen by selecting a pair of agents uniformly at random and applying transition function δ. To measure time we count the expected total number of interactions (including null transitions such as $a, b \to a, b$ in which the agents interact but do not change state), and divide by the number of agents n. (In the population protocols literature, this is often called "parallel time"; i.e. n interactions among a population of n agents corresponds to one unit of time). Let $\mathbf{c} \in \mathbb{N}^\Lambda$ and $C \subseteq \mathbb{N}^\Lambda$. Denote the probability that the PP reaches from $\mathbf{c}$ to some configuration $\mathbf{c}' \in C$ by $\mathsf{Pr}[\mathbf{c} \Longrightarrow C]$. If $\mathsf{Pr}[\mathbf{c} \Longrightarrow C] = 1$,[11] define the *expected time to reach from* $\mathbf{c}$ *to* C, denoted $\mathsf{T}[\mathbf{c} \Longrightarrow C]$, to be the expected number of interactions to reach from $\mathbf{c}$ to some $\mathbf{c}' \in C$, divided by the number of agents n.

3 Main Results

3.1 Impossibility of Sublinear Time Stable Leader Election

We consider the following *stable leader election* problem. Suppose that each PP $\mathcal{P} = (\Lambda, \delta)$ we consider has a specially designated state $\ell \in \Lambda$, which we call the *leader state*. Informally, the goal of stable leader election is to be guaranteed to reach a configuration with count 1 of ℓ (a leader has been "elected"), from which no transition sequence can change the count of ℓ (the leader is "stable"). We also assume there is a special initial state x (it could be that $x \equiv \ell$ but it is not required), such that the only valid initial configurations $\mathbf{i}$ are of the form $\mathbf{i}(x) > 0$ and $\mathbf{i}(y) = 0$ for all states $y \in \Lambda \setminus \{x\}$. We write $\mathbf{i}_n$ to denote such an initial configuration with $\mathbf{i}_n(x) = n$.

Definition 3.1 *A configuration* $\mathbf{y}$ *is* stable *if, for all* $\mathbf{y}'$ *such that* $\mathbf{y} \Longrightarrow \mathbf{y}'$, $\mathbf{y}'(\ell) = \mathbf{y}(\ell)$ *(in other words, after reaching* $\mathbf{y}$*, the count of* ℓ *cannot change);* $\mathbf{y}$ *is said to have a* stable leader *if it is stable and* $\mathbf{y}(\ell) = 1$.

The following definition captures our notion of stable leader election. It requires the PP to be "guaranteed" eventually to reach a configuration with a stable leader.

Definition 3.2 *We say a PP* stably elects a leader *if, for all* $n \in \mathbb{Z}^+$*, for all* $\mathbf{c}$ *such that* $\mathbf{i}_n \Longrightarrow \mathbf{c}$*, there exists* $\mathbf{y}$ *with a stable leader such that* $\mathbf{c} \Longrightarrow \mathbf{y}$.

In other words, letting Y denote the set of configurations with a stable leader, every reachable configuration can reach to Y. It is well-known [8] that Definition 3.2 is equivalent to requiring $\mathsf{Pr}[\mathbf{i}_n \Longrightarrow Y] = 1$ for all $n \in \mathbb{Z}^+$.

We note that our PP model captures anonymous nodes defined on a complete communication graph. Thus, agents in the same state are truly indistinguishable. Consequently, our formalism does not discern whether the agent that becomes the single leader stays the leader, or whether the leader state moves among agents (reminiscent of token passing).

Definition 3.3 *Let* $t : \mathbb{Z}^+ \to \mathbb{R}^+$*, and let* Y *be the set of all configurations with a stable leader. We say a PP stably elects a leader* in time $t(n)$ *if, for all* $n \in \mathbb{Z}^+$, $\mathsf{T}[\mathbf{i}_n \Longrightarrow Y] \leq t(n)$.

Our main theorem says that stable leader election requires at least linear time to stabilize:

Theorem 3.4 *If a PP stably elects a leader in time* $t(n)$*, then* $t(n) = \Omega(n)$.

Thus a PP that elects a leader in sublinear time cannot do so stably, i.e., it must have a positive probability of failure.

The high-level strategy to prove Theorem 3.4 is as follows. With high probability the PP initially

[10] When the initial configuration to which a transition sequence is applied is clear from context, we may overload terminology and refer to $(\mathbf{c}_0, \mathbf{c}_1, \ldots)$ as a transition sequence or path.

[11] Since PP's have a finite reachable configuration space, this is equivalent to requiring that for all $\mathbf{x}$ reachable from $\mathbf{c}$, there is a $\mathbf{c}' \in C$ reachable from $\mathbf{x}$.

goes from configuration $\mathbf{i}_n$ to configuration $\mathbf{x}_n$, such that in the sequence $(\mathbf{x}_n)$ for increasing population size n, the count of each state grows without bound as $n \to \infty$ (indeed the count of each state grows linearly with n); this follows from Theorem 4.3. We then show that any such configuration must have an "$O(1)$-bottleneck transition" before reaching a configuration with a stable leader (informally this means that every transition sequence from $\mathbf{x}_n$ to a configuration $\mathbf{y}$ with a stable leader must have a transition in which both input states have count $O(1)$, depending on the PP but not on n). Since it takes expected time $\Omega(n)$ to execute a transition when both states have constant count, from any such configuration it requires linear time to stably elect a leader. Since one of these configurations is reached from the initial configuration with high probability, those configurations' contribution to the overall expected time dominates, showing that the expected time to stably elect a leader is linear.

3.2 More General Impossibility Result in Terms of Inapplicable Transitions and Dense Configurations

Rather than proving Theorem 3.4 using the notion of leader stability directly, we prove a more general result concerning the notion of a set of inapplicable transitions. We generalize in two ways. (1) A configuration $\mathbf{y}$ is stable by Definition 3.1 if no transition altering the count of ℓ is applicable in any configuration reachable from $\mathbf{y}$; Definition 3.5 generalizes this to an arbitrary subset Q of transitions. (2) The valid initial configurations of Section 3.1 are those with $\mathbf{i}_n(x) = n$ and $\mathbf{i}_n(y) = 0$ for all $y \in \Lambda \setminus \{x\}$; Theorem 3.8 generalizes this to any set I of configurations that are all "α-dense": any state can be initially present as long as it is present in "large count" (at least a constant fraction of the population; see Definition 3.6). We also require the allowed initial configurations to satisfy a weak sort of "closure under doubling" property: namely, that there is an infinite subset $I' \subseteq I$ such that $2I' \subseteq I$. In other words, there are infinitely many $\mathbf{i} \in I$ such that $2\mathbf{i} \in I$. This is true if I is closed under addition, for example if I is the set of uniform initial configurations, or if I is the set of all α-dense configurations for some $\alpha > 0$ (since doubling a configuration preserves its density).

Definition 3.5 *Let Q be a set of transitions. A configuration $\mathbf{y} \in \mathbb{N}^\Lambda$ is said to be Q-*stable *if no transition in Q is applicable in any configuration reachable from $\mathbf{y}$.*

Let $I \subseteq \mathbb{N}^\Lambda$ and Q be a set of transitions. Let Y be the set of Q-stable configurations reachable from some configuration in I. We say that a PP $\mathcal{P} = (\Lambda, \delta)$ *Q-stabilizes from I* if, for any $\mathbf{i} \in I$, $\mathsf{Pr}[\mathbf{i} \Longrightarrow Y] = 1$.[12] If I and Q are understood from context, we say that $\mathcal{P}$ *stabilizes*. For a time bound $t(n)$, we say that $\mathcal{P}$ stabilizes *in expected time $t(n)$* if, for all $\mathbf{i} \in I$ such that $\|\mathbf{i}\| = n$, $\mathsf{T}[\mathbf{i} \Longrightarrow Y] \leq t(n)$.

Definition 3.6 *Let $0 < \alpha \leq 1$. We say that a configuration $\mathbf{c}$ is* α-dense *if for all $s \in \Lambda$, $\mathbf{c}(s) > 0$ implies that $\mathbf{c}(s) \geq \alpha\|\mathbf{c}\|$, i.e., all states present in $\mathbf{c}$ occupy at least an α fraction of the total count of agents.*

In order to reason about the behavior of PPs for larger and larger population sizes, we consider infinite sequences of configurations that "slice" across different population counts n. In other words, these sequences consist of configurations satisfying certain criteria that are reachable from ever-larger initial configurations. (Note that such configurations are not reachable from each other since they have different numbers of agents.) When C spans infinitely many population sizes, the following definition expresses a basic distinction in how state counts scale with increasing population size:

Definition 3.7 *For an (infinite) set/sequence of configurations C, let $\mathsf{bdd}(C)$ be the set of states*

$$\{\ s \in \Lambda \mid (\exists b \in \mathbb{N})(\forall \mathbf{c} \in C)\ \mathbf{c}(s) < b\ \}.$$

Let $\mathsf{unbdd}(C) = \Lambda \setminus \mathsf{bdd}(C)$.

Remark 1 Note that if $C = (\mathbf{c}_m)$ is a nondecreasing sequence, then for all $k \in \mathbb{N}$, there is $\mathbf{c}_m$ such that for all $s \in \mathsf{unbdd}(\mathbf{c}_m)$, $\mathbf{c}_m(s) \geq k$. (Note that if C is not nondecreasing, the conclusion can fail; e.g., $\mathbf{c}_m(s_1) = m, \mathbf{c}_m(s_2) = 0$ for m even and $\mathbf{c}_m(s_1) = 0, \mathbf{c}_m(s_2) = m$ for m odd.)

The following is our most general theorem, which the rest of the paper is devoted to proving.

Theorem 3.8 *Let $\mathcal{P} = (\Lambda, \delta)$ be a PP, Q be any subset of transitions of $\mathcal{P}$, $\alpha > 0$, and $I \subseteq \mathbb{N}^\Lambda$ be a set of α-dense initial configurations such that there is an infinite subset $I' \subseteq I$ such that $2I' \subseteq I$. Let Y be the set of all Q-stable configurations reachable*

[12] Recall that the condition $\mathsf{Pr}[\mathbf{i} \Longrightarrow Y] = 1$ is equivalent to $[(\forall \mathbf{c} \in \mathbb{N}^\Lambda)\ \mathbf{i} \Longrightarrow \mathbf{c}$ implies $(\exists \mathbf{y} \in Y)\ \mathbf{c} \Longrightarrow \mathbf{y}]$.

from I. Suppose $\mathcal{P}$ Q-stabilizes from I in expected time $o(n)$. Then there are infinitely many $\mathbf{v} \in Y$ such that $\forall s \in \mathsf{bdd}(Y)$, $\mathbf{v}(s) = 0$.

In other words, if some states have "small" count in all reachable stable configurations, then there is a reachable stable configuration in which those states have count 0. A PP $\mathcal{P}$ that stably elects a leader is a PP in which Q is the set of transitions that alter the count of ℓ, $I = \{ \mathbf{i}_n \mid n \in \mathbb{N} \}$ (note all $\mathbf{i}_n$ are 1-dense and all of I is closed under doubling), Y is the set of configurations reachable from I with a stable leader, and $\mathcal{P}$ Q-stabilizes from I. Hence by Theorem 3.8, if $\mathcal{P}$ stabilizes in expected time $o(n)$, there is a $\mathbf{v}$ that is both stable and reachable, where $\mathbf{v}(\ell) = 0$, a contradiction. Thus Theorem 3.4 follows from Theorem 3.8.

We can also use Theorem 3.8 to prove that stable leader election requires linear time under the more relaxed requirement that there is a set $\Psi \subset \Lambda$ of "leader states," and the goal of the PP is to reach a configuration $\mathbf{y}$ in which $\sum_{\ell \in \Psi} \mathbf{y}(\ell) = 1$ and stays 1 in any configuration reachable from $\mathbf{y}$. Choosing Q as the set of transitions that alter that sum, Theorem 3.8 implies this form of stable leader election also requires $\Omega(n)$ expected time.

The rest of the paper is organized as follows. We conclude Section 3 with two example PPs showing that sublinear time leader election *is possible* if we relax the α-dense requirement in Theorem 3.8—showing that this requirement is indeed necessary. These examples are particularly useful in discarding certain simple proof techniques that naturally come to mind. Then Section 4 develops the technical tools, which we use in Section 5 to complete the proof of Theorem 3.8. Throughout the rest of this paper, fix $\mathcal{P} = (\Lambda, \delta)$, α, I, and Q as in the statement of Theorem 3.8.

3.3 Why Simple Proofs Fail

It is tempting to believe that the main theorem follows by a simple argument based on reasoning about the last transition to change the count of the leader. Indeed, if we start with more than one leader, and no transition rule can produce a new leader, then we can easily prove the impossibility of sublinear time leader election as follows. To quickly reduce from two leaders to one, the other agent's state must be numerous in the population, so the same transition could occur again. This would leave us with no leaders and no possibility to make a new leader. However, if transitions *can* produce new leaders, then the argument cannot reason only about the last transition involving the leader. We illustrate this using two examples, which the authors have found helpful in ruling out plausible-sounding but ultimately insufficient ideas for proving a negative result.

We describe two PPs that stably elect a leader in sublinear time starting from initial configurations that are not α-dense (for $\alpha > 0$ independent of n). (Since the initial configurations are not α-dense these PPs do not contradict the statement of our main theorem.) In both examples, with high probability exactly one transition involving the leader occurs. In the first example the transition produces precisely one leader in a configuration that previously had none, whereas in the second example, it consumes precisely one leader in a configuration that previously had two. (Clearly, these are the only two possible forms of the final transition involving the leader.) The examples imply that any proof of the main result cannot be based solely on reasoning about the final transition, but must additionally establish that configurations such as the initial configurations of these PPs cannot be reached with high probability in sublinear time.

Consider the following PP, with initial configuration $\mathbf{i}$ given by $\mathbf{i}(r) = n^{1/4}$, $\mathbf{i}(x) = n - n^{1/4}$, and transitions:

$$r, r \to \ell, k \tag{3.1}$$
$$r, k \to k, k \tag{3.2}$$
$$x, k \to k, k \tag{3.3}$$
$$\ell, \ell \to \ell, k \tag{3.4}$$

Transition (3.1) is the only one possible initially, and it takes expected time $\Theta(n^{1/2})$ to occur for the first time, producing a single leader. Transition (3.4) ensures that if transition (3.1) occurs more than once, the PP will eventually stabilize to a single leader. However, with high probability transitions (3.2) and (3.3) consume all r and x *before* (3.1) executes a second time. After exactly one instance of transition (3.1) occurs, let a *speed fault* denote the event that transition (3.1) occurs again (this is the same speed fault concept studied in ref. [13]). For convenience, for state $s \in \Lambda$, let s also denote the count of that state in the configuration considered.

Conditioned on the next interaction being non-null, i.e., it is either a speed fault (transition (3.1)) or moves closer to converting all x and r to k (transition (3.2) or (3.3)), the probability of a speed fault in

any particular configuration is $\frac{r(r-1)}{r(r-1)+2k(n-k-1)} < \frac{n^{1/2}}{k(n-k-1)}$, since $r(r-1)$ is the number of ways of choosing two agents in state r (leading to a speed fault), and $2k(n-k-1)$ is the number of ways of choosing an agent in state k and another agent in state either r or x, when $\ell = 1$ (increasing the count of k), and therefore $r + x = n - k - 1$. By the union bound, the probability that a speed fault occurs in between $k = 1$ and $k = n - 1$ (at which point transition (3.1) is disabled and the PP stabilizes) is at most

$$n^{1/2} \sum_{k=1}^{n-2} \frac{1}{k(n-k-1)} = n^{1/2} O\left(\frac{\log n}{n}\right) = O\left(\frac{\log n}{n^{1/2}}\right).$$

Whether or not a speed fault occurs, to produce ℓ, transition (3.1) must occur for the first time, taking expected time $O(n^{1/2})$. Let T be the random variable denoting the time to stabilization *after* transition (3.1) has occurred for the first time. If a speed fault occurs, then transition (3.4) must execute enough times to reduce ℓ to 1, which requires expected time $O(n)$ [6]. Thus $\mathsf{E}[\mathrm{T}|\text{speed fault}] = O(n)$. Now to analyze $\mathsf{E}[\mathrm{T}|\text{no speed fault}]$, note that if only transitions (3.2) and (3.3) existed, then after producing a single k, the expected time for transitions (3.2) and (3.3) to convert all x and r into k would be $O(\log n)$ (this is known as an "epidemic" [4]). If we consider that transition (3.1) is also competing with transitions (3.2) and (3.3), and then we condition on transition (3.1) not occurring before all r are converted to k, then this conditioning can only reduce this expected time. Thus $\mathsf{E}[T|\text{no speed fault}] = O(\log n)$.

Thus, the total expected time to stabilize to a single leader is at most

$$\begin{aligned} &O(n^{1/2}) + \mathsf{Pr}[\text{speed fault}] \cdot \mathsf{E}[\mathrm{T}|\text{speed fault}] \\ &+\mathsf{Pr}[\text{no speed fault}] \cdot \mathsf{E}[\mathrm{T}|\text{no speed fault}] \\ &\leq O(n^{1/2}) + O\left(\frac{\log n}{n^{1/2}}\right) \cdot O(n) + 1 \cdot O(\log n) \\ &= O(n^{1/2} \log n), \end{aligned}$$

i.e., sublinear time.

The above PP uses a non-dense initial configuration since $\mathbf{i}(r) = o(n)$. Thus, although it does not directly contradict the existence of a linear time lower bound from dense configurations, it points out that any proof based on reasoning about the last transition to alter the count of ℓ must disallow the possibility that a leader is elected from some *intermediate* configuration in the manner described above. With high probability all states obtain count $\Omega(n)$ in a constant amount of time;[13] however, it is possible to subsequently reduce some states to sublinear count after super-constant time. *A priori*, it is conceivable that after, say, $O(\log n)$ time, the PP reaches a non-dense configuration, with $\ell = 0$ and $r \approx n^{1/4}$ similar to $\mathbf{i}$ above, which would then elect a leader in sublinear time by producing a single ℓ with high probability.

This example shows the difference between convergence and stabilization. Assuming no speed fault occurs, the PP converges when the first transition (3.1) occurs, but it does not stabilize until transition (3.2) reduces the count of r below 2, disabling transition (3.1) from occurring again.

We now consider another example PP. Even if the final change of ℓ takes it from 2 to 1, it is *a priori* conceivable that the final transition $r, \ell \to p_1, p_2$ to consume ℓ has count $o(n)$ of r, so that, although a second execution of the transition is *possible*, the second execution requires sufficiently long expected time that the system, in the meantime, likely consumes all remaining copies of r, along with a mechanism to ensure that a leader is elected even if the second leader is also consumed by an r. The following PP achieves this, with initial configuration $\mathbf{i}$ given by $\mathbf{i}(\ell) = 2$, $\mathbf{i}(r) = n^{1/2}$, $\mathbf{i}(x) = n - \mathbf{i}(r) - \mathbf{i}(\ell)$, with transitions

$$r, \ell \to r, \ell' \tag{3.5}$$
$$\ell', x \to \ell', k \tag{3.6}$$
$$k, x \to k, k \tag{3.7}$$
$$k, r \to k, k \tag{3.8}$$
$$\ell', \ell' \to \ell, k \tag{3.9}$$

An analysis similar to the previous PP shows that the expected time to stabilize to $\ell = 1$ is $O(n^{1/2} \log n)$. Informally, transition (3.5) consumes one copy of ℓ after expected time $O(n^{1/2})$. Transition (3.6) subsequently produces k in expected time $O(1)$, and transitions (3.7) and (3.8) remove all r and x in expected time $O(\log n)$. With high probability this happens before transition (3.5) can execute a second time, but if not, then $\ell' = 2$, so transition (3.9) guarantees that a single leader is stably elected (as above, if this is needed, it requires expected time $\Omega(n)$, but it is needed with such low probability that the overall expected time remains sublinear).

[13] This is the main theorem of [17], of which Theorem 4.3 is a corollary.

In summary, the two PPs above demonstrate that the proof cannot be based solely on reasoning about the final transition to alter ℓ, no matter whether that transition increases ℓ from 0 to 1 or decreases it from 2 to 1.

4 Technical Tools

4.1 Bottleneck Transitions Require Linear Time

This section proves a straightforward observation used in the proof of our main theorem. It states that, if to get from a configuration $\mathbf{x} \in \mathbb{N}^\Lambda$ to some configuration in a set $Y \subseteq \mathbb{N}^\Lambda$, it is necessary to execute a transition $r_1, r_2 \to p_1, p_2$ in which the counts of r_1 and r_2 are both at most some number b, then the expected time to reach from $\mathbf{x}$ to some configuration in Y is $\Omega(n/b^2)$.

Let $b \in \mathbb{N}$. We say that transition $\alpha : r_1, r_2 \to p_1, p_2$ is a *b-bottleneck* for configuration $\mathbf{c}$ if $\mathbf{c}(r_1) \leq b$ and $\mathbf{c}(r_2) \leq b$. We say a transition sequence q such that $\mathbf{x} \Longrightarrow_q \mathbf{y}$ has a b-bottleneck transition if some transition in q is a b-bottleneck for the configuration where it is applied.

Observation 4.1 *Let $b \in \mathbb{N}$, $\mathbf{x} \in \mathbb{N}^\Lambda$, and $Y \subseteq \mathbb{N}^\Lambda$ such that $\mathsf{Pr}[\mathbf{x} \Longrightarrow Y] = 1$. If every transition sequence taking $\mathbf{x}$ to a configuration $\mathbf{y} \in Y$ has a b-bottleneck transition, then $\mathsf{T}[\mathbf{x} \Longrightarrow Y] \geq \frac{n-1}{2(b \cdot |\Lambda|)^2}$.*

Proof The probability that a particular transition $r_1, r_2 \to p_1, p_2$ occurs in any configuration $\mathbf{c}$ where $\mathbf{c}(r_1) \leq b$ and $\mathbf{c}(r_2) \leq b$ is at most $\frac{2b^2}{n(n-1)}$.[14] There are no more than $|\Lambda|^2$ different transitions.[15] By the union bound, the probability that in any configuration $\mathbf{c}$, any b-bottleneck transition occurs is no more than $|\Lambda|^2 \frac{2b^2}{n(n-1)}$. Thus we can bound the number of interactions until the first b-bottleneck transition occurs by a geometric random variable with success probability at most $\frac{2(b \cdot |\Lambda|)^2}{n(n-1)}$, whence the expected number of interactions until the first b-bottleneck is at least $\frac{n(n-1)}{2(b \cdot |\Lambda|)^2}$. Since the parallel time is defined as the number of interactions divided by n, this corresponds to $\frac{n-1}{2(b \cdot |\Lambda|)^2}$ expected time. By assumption, the set of configurations Y is reached only after a b-bottleneck transition occurs. Therefore, the statement of the lemma follows. □

Corollary 4.2 *Let $\gamma > 0$, $b \in \mathbb{N}$, $\mathbf{c} \in \mathbb{N}^\Lambda$, and $X, Y \subseteq \mathbb{N}^\Lambda$ such that $\mathsf{Pr}[\mathbf{c} \Longrightarrow X] \geq \gamma$, $\mathsf{Pr}[\mathbf{c} \Longrightarrow Y] = 1$, and every transition sequence from every $\mathbf{x} \in X$ to some $\mathbf{y} \in Y$ has a b-bottleneck transition. Then $\mathsf{T}[\mathbf{c} \Longrightarrow Y] \geq \gamma \frac{n-1}{2(b \cdot |\Lambda|)^2}$.*

4.2 Sublinear Time from Dense Configurations Implies Bottleneck Free Path from Configurations with Every State "Populous"

The following theorem, along with Corollary 4.2, fully captures the probability theory necessary to prove our main theorem.[16] Given it and Corollary 4.2, Theorem 3.8 is provable (through Lemma 4.4) using only combinatorial arguments about reachability between configurations.

For ease of notation, we assume throughout this paper that all states in Λ are *producible*, meaning they have positive count in some reachable configuration. Otherwise the following theorem applies only to states that are actually producible. Recall that for $\alpha > 0$, a configuration $\mathbf{c}$ is *α-dense* if for all $s \in \Lambda$, $\mathbf{c}(s) > 0$ implies that $\mathbf{c}(s) \geq \alpha \|\mathbf{c}\|$. Say that $\mathbf{c} \in \mathbb{N}^\Lambda$ is *full* if $(\forall s \in \Lambda)\ \mathbf{c}(s) > 0$, i.e., every state is present. The following theorem states that with high probability, a PP will reach from an α-dense configuration to a configuration in which all states are present (full) in "large" count (β-dense, for some $0 < \beta < \alpha$).

Theorem 4.3 (adapted from [17]) *Let $\mathcal{P} = (\Lambda, \delta)$ be a PP and $\alpha > 0$. Then there are constants $\epsilon, \beta > 0$ such that, letting $X = \{\ \mathbf{x} \in \mathbb{N}^\Lambda \mid \mathbf{x}$ is full and β-dense $\}$, for all sufficiently large α-dense configurations $\mathbf{i}$, $\mathsf{Pr}[\mathbf{i} \Longrightarrow X] \geq 1 - 2^{-\epsilon \|\mathbf{i}\|}$.*

The following lemma reduces the problem of proving Theorem 3.8 to a combinatorial statement involving only reachability among configurations (and

[14] If $r_1 \neq r_2$ and $\mathbf{c}(r_1) = \mathbf{c}(r_2) = b$, then the probability to pick the first agent in one of the states r_1 or r_2 is $\frac{2b}{n}$, and the probability to pick the second agent in the other state is $\frac{b}{n-1}$, so the total probability of both is $\frac{2b^2}{n(n-1)}$. The case for $r_1 = r_2$ gives $\frac{b}{n}$ for the first times $\frac{b-1}{n-1}$ for the second, resulting in lower total probability $\frac{b^2-b}{n(n-1)}$.

[15] With a nondeterministic transition function, the total number of transitions would replace the quantity $|\Lambda|^2$ in the conclusion, but it would remain a constant independent of the size of the initial configuration.

[16] Theorem 4.3 was proven in a more general model for Chemical Reaction Networks (CRNs) that obey a certain technical condition [17]. As observed in that paper, the class of CRNs obeying that condition includes all PPs, so the theorem holds unconditionally for PPs. The theorem proved in [17] is more general than Theorem 4.3, but we have stated a corollary of it here. A similar statement is implicit in the proof sketch of Lemma 5 of a technical report on a variant model called "urn automata" that has PPs as a special case [5].

the lack of bottleneck transitions between them). In Section 5 we will prove Theorem 3.8 by showing that the existence of the configurations $\mathbf{x}_m$ and $\mathbf{y}_m$ and the transition sequence p_m in the following lemma implies that we can reach a Q-stable configuration $\mathbf{v} \in \mathbb{N}^\Gamma$, where $\Gamma = \mathsf{unbdd}(Y)$ and Y is the set of Q-stable configurations reachable from I.

Lemma 4.4 *Let $\alpha > 0$. Let $\mathcal{P} = (\Lambda, \delta)$ be a PP such that, for some set of transitions Q and infinite set of α-dense initial configurations I, $\mathcal{P}$ Q-stabilizes from I in expected time $o(n)$. Then for all $m \in \mathbb{N}$, there is an n_0 such that for all $\mathbf{i} \in I$ with $\|\mathbf{i}\| \geq n_0$, there is a configuration $\mathbf{x}_m$ reachable from $\mathbf{i}$ and transition sequence p_m such that: (1) $\mathbf{x}_m(s) \geq m$ for all $s \in \Lambda$, (2) $\mathbf{x}_m \Longrightarrow_{p_m} \mathbf{y}_m$, where $\mathbf{y}_m$ is Q-stable, and (3) p_m has no m-bottleneck transition.*

Proof Intuitively, the lemma follows from the fact that states $\mathbf{x}_m$ are reached with high probability by Theorem 4.3, and if no paths such as p_m existed, then all paths from $\mathbf{x}_m$ to a stable configuration would have a bottleneck and require linear time. Since $\mathbf{x}_m$ is reached with high probability, this would imply the entire expected time is linear.

For any configuration $\mathbf{x}_m$ reachable from some configuration in I, there is a transition sequence p_m satisfying condition (2) by the fact that $\mathcal{P}$ Q-stabilizes from I. It remains to show we can find $\mathbf{x}_m$ and p_m satisfying conditions (1) and (3).

By Theorem 4.3 there exist ϵ, β (which depend only on $\mathcal{P}$ and α) such that, starting in any sufficiently large initial configuration $\mathbf{i}$, with probability at least $1 - 2^{-\epsilon n}$, $\mathcal{P}$ reaches a configuration $\mathbf{x}$ where all states $s \in \Lambda$ have count at least βn, where $n = \|\mathbf{i}\|$. For all $\mathbf{i}$, let $X_\mathbf{i} = \{\mathbf{x} \mid \mathbf{i} \Longrightarrow \mathbf{x}$ and $(\forall s \in \Lambda)\, \mathbf{x}(s) \geq \beta\|\mathbf{i}\|\}$. Given any $m \in \mathbb{N}$, let n_0 be a lower bound on n such that: Theorem 4.3 applies for all $n \geq n_0$, $1 - 2^{-\epsilon n_0} \geq \frac{1}{2}$, and further $n_0 \geq m/\beta$. Then, for all $\mathbf{i} \in I$ such that $\|\mathbf{i}\| = n \geq n_0$, $\mathsf{Pr}[\mathbf{i} \Longrightarrow X_\mathbf{i}] \geq \frac{1}{2}$. Choose any $n \geq n_0$ for which there is $\mathbf{i} \in I$ with $\|\mathbf{i}\| = n$. Then any $\mathbf{x}_m \in X_\mathbf{i}$ satisfies condition (1): $\mathbf{x}_m(s) \geq m$ for all $s \in \Lambda$. We now show that by choosing $\mathbf{x}_m$ from $X_\mathbf{i}$ for a large enough n, we can find a corresponding p_m satisfying condition (3) as well.

Suppose for the sake of contradiction that for some m we cannot satisfy condition (3) when choosing $\mathbf{x}_m$ as above, no matter how large we make n. This means that, letting Y be set of Q-stable configurations, for infinitely many $\mathbf{i} \in I$, (and therefore infinitely many population sizes $n = \|\mathbf{i}\|$), all transition sequences from $X_\mathbf{i}$ to Y have an m-bottleneck. Then Corollary 4.2, letting $\mathbf{c} = \mathbf{i}$, $\gamma = \frac{1}{2}$, $X = X_\mathbf{i}$, and $b = m$, tells us that $\mathsf{T}[\mathbf{i} \Longrightarrow Y] \geq \frac{1}{2} \frac{n-1}{2(m \cdot |\Lambda|)^2} = \Omega(n)$, a contradiction since $\mathcal{P}$ is supposed to Q-stabilize from I in expected time $o(n)$, i.e., $\mathsf{T}[\mathbf{i} \Longrightarrow Y] = o(n)$. □

4.3 Transition Ordering Lemma

The following lemma was first proven (in the more general model of Chemical Reaction Networks) in [13]. We provide a proof for the sake of self-containment. Intuitively, the lemma states that a "fast" transition sequence (meaning one without a bottleneck transition) that decreases certain states from large counts to small counts must contain transitions of a certain restricted form. In particular the form is as follows: if Δ is the set of states whose counts decrease from large to small, then we can write the states in Δ in some order $d_1, d_2, \ldots, d_k$, such that for each $1 \leq i \leq k$, there is a transition α_i that consumes d_i, and every other state involved in α_i is either not in Δ, or comes later in the ordering. These transitions will later be used to do controlled "surgery" on fast transition sequences, because they give a way to alter the count of d_i, by inserting or removing the transitions α_i, knowing that this will not affect the counts of $d_1, \ldots, d_{i-1}$.

Lemma 4.5 (Adapted from [13]) *Let $b_1, b_2 \in \mathbb{N}$ such that $b_2 > |\Lambda| \cdot b_1$. Let $\mathbf{x}, \mathbf{y} \in \mathbb{N}^\Lambda$ such that $\mathbf{x} \Longrightarrow \mathbf{y}$ via transition sequence q that does not contain a b_2-bottleneck. Define*

$$\Delta = \{\ d \in \Lambda \mid \mathbf{x}(d) \geq b_2 \text{ and } \mathbf{y}(d) \leq b_1 \ \}.$$

Then there is an order on Δ, so that we may write $\Delta = \{d_1, d_2, \ldots, d_k\}$, such that, for all $i \in \{1, \ldots, k\}$, there is a transition α_i of the form $d_i, s_i \to o_i, o_i'$, such that $s_i, o_i, o_i' \notin \{d_1, \ldots, d_i\}$, and α_i occurs at least $(b_2 - |\Lambda| \cdot b_1)/|\Lambda|^2$ times in q.

Proof We define the ordering based on increasing sets $\varnothing = \Delta_0 \subset \Delta_1 \subset \Delta_2 \subset \ldots \Delta_{k-1} \subset \Delta_k = \Delta$, where for each $1 \leq i \leq k$, $\Delta_i = \Delta_{i-1} \cup \{d_i\}$.

We define the ordering inductively "in reverse," by first defining d_k, then d_{k-1}, etc. For all $1 \leq i \leq k$, define $\Phi_i : \mathbb{N}^\Lambda \to \mathbb{N}$ for all configurations $\mathbf{c}$ by $\Phi_i(\mathbf{c}) = \sum_{d \in \Delta_i} \mathbf{c}(d)$. Φ_k is well-defined since $\Delta_k = \Delta$, and Φ_i is well-defined once we have defined $d_{i+1}, \ldots, d_k$, because $\Delta_i = \Delta \setminus \{d_{i+1}, \ldots, d_k\}$.

Because $\mathbf{y}(d) \leq b_1$ for all $d \in \Delta$, it follows that $\Phi_i(\mathbf{y}) \leq i \cdot b_1 \leq |\Lambda| \cdot b_1$. Recall that $\mathbf{x}(d) \geq b_2$ for all $d \in \Delta$. Let r be the suffix of q after the last

configuration $\mathbf{c}'$ along q such that $\Phi_i(\mathbf{c}') \geq b_2$. Then in all configurations $\mathbf{c}$ in r, $\mathbf{c}(d) < b_2$ for all $d \in \Delta_i$. Because $\Phi_i(\mathbf{c}') \geq b_2$ but $\Phi_i(\mathbf{y}) \leq |\Lambda| \cdot b_1$, r contains a subsequence u of transitions, each of which strictly decreases Φ_i, and the total decrease in Φ_i over all of u is at least $(b_2 - |\Lambda| \cdot b_1)$ between configurations $\mathbf{c}'$ and $\mathbf{y}$.

Let $\alpha : r_1, r_2 \to p_1, p_2$ be a transition in u. Since α strictly decreases Φ_i, $r_1 \in \Delta_i$ or $r_2 \in \Delta_i$; assume without loss of generality that $r_1 \in \Delta_i$. Further, since u does not contain a b_2-bottleneck, and all configurations $\mathbf{c}$ along u have $\mathbf{c}(d) < b_2$ for all $d \in \Delta_i$, for α not to be a b_2-bottleneck, we must have $r_2 \notin \Delta_i$. Since exactly one state in Δ_i decreases its count, $p_1 \notin \Delta_i$ and $p_2 \notin \Delta_i$, or else α would not decrease Φ_i. Let $d_i = r_1, s_i = r_2, o_i = p_1$, and $o_i' = p_2$.

Then α decreases Φ_i by exactly 1. Since there are at least $b_2 - |\Lambda| \cdot b_1$ instances of such transitions in u, and there are at most $|\Lambda|^2$ total types of transitions, by the pigeonhole principle at least one transition type must repeat in u at least $(b_2 - |\Lambda| \cdot b_1)/|\Lambda|^2$ times. □

It is instructive to observe how Lemma 4.5 can fail if the transition sequence q contains a b_2-bottleneck. Consider the linear-time leader election PP given by the transition $\ell, \ell \to \ell, f$ with initial configuration $\mathbf{x} = \{n\ell\}$ and final configuration $\mathbf{y} = \{1\ell, (n-1)f\}$. In this case, $\Delta = \{\ell\}$, but once the count of ℓ drops below b_2, subsequent transitions are b_2-bottlenecks. Thus, the hypothesis of the lemma is not obeyed, and indeed, there is no transition $\ell, s \to o, o'$ such that $\ell \notin \{s, o, o'\}$.

5 Proof of Theorem 3.8

Let $I' \subseteq I$ be an infinite subset of I such that $2I' \subseteq I$. Recall Lemma 4.4. We use it (letting I in Lemma 4.4 be I') to construct infinite sequences $(\mathbf{x}_m)$ and $(\mathbf{y}_m)$ of configurations and (p_m) of paths as follows. Let $m \in \mathbb{N}$. Lemma 4.4 tells us that there is an n_0 such that for all $\mathbf{i} \in I'$ with $\|\mathbf{i}\| \geq n_0$, there is a configuration $\mathbf{x}_m$ reachable from $\mathbf{i}$ and transition sequence p_m such that: (1) $\mathbf{x}_m(s) \geq m$ for all $s \in \Lambda$, (2) $\mathbf{x}_m \Longrightarrow_{p_m} \mathbf{y}_m$, where $\mathbf{y}_m$ is Q-stable, and (3) p_m has no m-bottleneck transition.

By Dickson's Lemma (Lemma 2.1) there is an infinite subsequence of values of m for which both $(\mathbf{x}_m)$ and $(\mathbf{y}_m)$ are nondecreasing. Without loss of generality, we take $(\mathbf{x}_m)$, $(\mathbf{y}_m)$, and (p_m) to be these subsequences. Let $\Delta = \mathsf{bdd}((\mathbf{y}_m))$ and $\Gamma = \mathsf{unbdd}((\mathbf{y}_m)) = \Lambda \setminus \Delta$. Note that since each $\mathbf{y}_m \in Y$ (the set of stable configurations reachable from some $\mathbf{i} \in I$), we have that $\mathsf{bdd}(Y) \subseteq \mathsf{bdd}((\mathbf{y}_m))$. Thus, we prove the theorem by showing that for infinitely many $\mathbf{y} \in Y$, for all $s \in \Delta$, $\mathbf{y}(s) = 0$.

Note that stability is closed downward: subsets of a Q-stable configuration are Q-stable. For any fixed $\mathbf{v}^\Gamma \in \mathbb{N}^\Gamma$, $\mathbf{v}^\Gamma \leq \mathbf{y}_m$ for sufficiently large m, by the definition of Γ (the states that grow unboundedly in $\mathbf{y}_m$ as $m \to \infty$). All $\mathbf{y}_m$ are Q-stable. Thus *any* configuration $\mathbf{v}^\Gamma \in \mathbb{N}^\Gamma$ is automatically Q-stable. This is why Claims 1, 2, and 3 of this proof center around reaching configurations that have count 0 of every state in Δ.

Overview of Claims 1–3. Recall the path $\mathbf{x}_m \Longrightarrow_{p_m} \mathbf{y}_m$ from Lemma 4.4. Intuitively, Claim 1 below says that because this path is m-bottleneck free, Lemma 4.5 applies, and its transitions can appended to the path to consume all states in Δ from $\mathbf{y}_m$, resulting in a configuration $\mathbf{z}_m^\Gamma$ that contains only states in Γ. If this is possible directly as stated, this would correspond to the formal claim that $\mathbf{x}_m \Longrightarrow_{p_m} \mathbf{y}_m \Longrightarrow_{p_m'} \mathbf{z}_m$, where $\mathbf{z}_m \in \mathbb{N}^\Gamma$ contains no states in Δ. However, we do not know how to prove this directly (although it may be true). Instead, we show in Claim 1 that $\mathbf{y}_m$ can reach to such a $\mathbf{z}_m^\Gamma \in \mathbb{N}^\Gamma$ *if some extra agents in special states are supplied*, i.e., that there exists $\mathbf{e} \in \mathbb{N}^\Lambda$ (the extra agents) such that $\mathbf{y}_m + \mathbf{e} \Longrightarrow \mathbf{z}_m^\Gamma$. By additivity $\mathbf{x}_m + \mathbf{e} \Longrightarrow_{p_m} \mathbf{y}_m + \mathbf{e}$, so $\mathbf{x}_m + \mathbf{e} \Longrightarrow \mathbf{z}_m^\Gamma$.

So where will these extra agents come from? Although we talk about them as if they are somehow physically added, in actuality, we'll start with a larger initial configuration and "guide" some of the agents to the desired states that make up $\mathbf{e}$. Claims 2 and 3 explain how this happens.

Claim 2, also relying on Lemma 4.5, is a way to produce the states corresponding to $\mathbf{e}$ needed for Claim 1. Claim 2 states, intuitively, that any such $\mathbf{e} \in \mathbb{N}^\Lambda$ can be produced from $\mathbf{x}_m$, as long as this is done "in the context" of extra states (corresponding to $\mathbf{p} \in \mathbb{N}^\Lambda$). However, unlike the extra states $\mathbf{e} \in \mathbb{N}^\Lambda$ as used in Claim 1, the states in $\mathbf{p}$ are "recovered".

To understand why Claim 2 is useful, it is helpful to look how the proof of Claim 3 works. The initial configuration $\mathbf{i}$ in Claim 3 can be split into two halves $\mathbf{i}'$ where $\mathbf{i} = 2\mathbf{i}'$. We have $\mathbf{i}' \Longrightarrow \mathbf{x}_m$, so $\mathbf{i} \Longrightarrow 2\mathbf{x}_m$. The proof of Claim 3 works by employing Claim 2 to produce $\mathbf{e}$ from one copy of $\mathbf{x}_m$ (while at the same time turning $\mathbf{x}_m$ into $\mathbf{w}^\Gamma \in \mathbb{N}^\Gamma$), using the other copy of $\mathbf{x}_m$ as the "context" $\mathbf{p}$ that allows Claim 2 to work. Once $\mathbf{e}$ is produced and the second copy of $\mathbf{x}_m$ is recovered, Claim 1 is then used to

show that $\mathbf{x}_m + \mathbf{e} \Longrightarrow \mathbf{z}^\Gamma$. In other words, having already eliminated Γ states from the first copy of $\mathbf{x}_m$, turning it into $\mathbf{w}^\Gamma$, we use $\mathbf{e}$ to eliminate Γ states from the other copy of $\mathbf{x}_m$, turning it into $\mathbf{z}^\Gamma$. Thus $\mathbf{i} = 2\mathbf{i}' \Longrightarrow 2\mathbf{x}_m \Longrightarrow \mathbf{x}_m + \mathbf{w}^\Gamma + \mathbf{e} \Longrightarrow \mathbf{z}^\Gamma + \mathbf{w}^\Gamma = \mathbf{v}^\Gamma \in \mathbb{N}^\Gamma$. We argued above that $\mathbf{v}^\Gamma$ is Q-stable, proving Theorem 3.8.

Claim 1 *There is* $\mathbf{e} \in \mathbb{N}^\Lambda$ *such that for all large enough* m*, there is* $\mathbf{z}_m^\Gamma \in \mathbb{N}^\Gamma$*, such that* $\mathbf{x}_m + \mathbf{e} \Longrightarrow \mathbf{z}_m^\Gamma$.

Example. We first illustrate Claim 1 through an example. Define a PP by the transitions

$$b, a \to f, c \tag{5.1}$$
$$b, c \to f, a \tag{5.2}$$
$$a, c \to f, f \tag{5.3}$$
$$f, c \to f, b \tag{5.4}$$
$$f, b \to f, f \tag{5.5}$$

For convenience, for state $s \in \Lambda$, let s also denote the count of that state in the configuration considered. Let configuration $\mathbf{x}_m$ be where $f = 100$, $a = 100$, $b = 100$, $c = 100$. Suppose a transition sequence p_m without an m-bottleneck ($m = 100$) takes the PP from $\mathbf{x}_m$ to $\mathbf{y}_m$, in which $a = 3$, $b = 2$, $c = 1$, and $f = 394$. Then in the language of Lemma 4.5, $\Delta = \{a, b, c\}$; these states go from "large" count in $\mathbf{x}_m$ to "small" count in $\mathbf{y}_m$.

Our strategy is to add interactions to p_m in order to reach a configuration $\mathbf{z}_m^\Gamma$ with $a = b = c = 0$. There are two issues we must deal with. First, to get rid of a we may try to add 3 instances of (5.1) at the end of p_m. However, there is only enough b for 2 instances. To eliminate such dependency, in Claim 1, whenever we add a transition $b, a \to f, c$, we add an extra agent in state b to $\mathbf{e}$. (In general if we consume r_2 by adding transition $r_1, r_2 \to p_1, p_2$, we add an extra agent in state r_1 to $\mathbf{e}$.) Second, we need to prevent circularity in consuming and producing states. Imagine trying to add more executions of (5.1) to get a to 0 and more of (5.2) to get c to 0; this will fail because these transitions conserve the quantity $a + c$. To drive each of these states to 0, we must find some ordering on them so that each can be driven to 0 using a transition that does not affect the count of any state previously driven to 0.

Lemma 4.5 gives us a way to eliminate such dependency systematically. In the example above, we can find the ordering $d_1 \equiv a$, $d_2 \equiv c$, and $d_3 \equiv b$, with respective transitions (5.1) to drive a to 0 (3 executions), (5.4) to drive c to 0 (4 executions: 1 to consume the 1 copy of c in $\mathbf{y}_m$, and 3 more to consume the extra 3 copies that were produced by the 3 extra executions of (5.1)), and (5.5) to drive b to 0 (6 executions: 2 to consume 2 copies of b in $\mathbf{y}_m$, and 4 more to consume the extra 4 copies that were produced by the 4 extra executions of (5.4)).

Proof (of Claim 1) Intuitively, the proof works as follows. Recall that $\mathbf{x}_m \Longrightarrow_{p_m} \mathbf{y}_m$ and p_m does not contain an m-bottleneck. The goal is to get from configuration $\mathbf{y}_m$ (which may be positive on some elements of Δ) to $\mathbf{z}_m^\Gamma$ (which is 0 on all elements of Δ). (Recall that Δ and Γ partition the set of states Λ.) We will show that we can append to the end of p_m transitions $\alpha_i : d_i, s_i \to o_i, o_i'$, for $i \in \{1, \ldots, k\}$—in that order—such that for all i, $d_i \in \Delta$ and $o_i, o_i' \notin \{d_1, \ldots, d_i\}$. We use Lemma 4.5 to find the necessary transitions. We add enough α_i transitions to consume all copies of d_i. (We'll use c_i to indicate the number of transitions added.) However, this will also consume copies of s_i, so we add more agents in state s_i to $\mathbf{e}$ to account for this. (We'll use $\mathbf{e}_i$ to represent the additional copies of s_i added, with eventually $\mathbf{e} = \sum_{i=1}^k \mathbf{e}_i$.) Once we have added enough α_i transitions to make the count of d_i equal to 0, by the fact that for all j, $o_j, o_j' \notin \{d_1, \ldots, d_j\}$, subsequently added transitions α_j for $j > i$ will not produce d_i (so its count will stay 0), nor will it require consuming d_i (so the transitions will be applicable). However, prior to reaching the point where we add α_i transitions, if $d_i = o_j$ or o_j' for $j < i$, then the excess copies of d_i generated by the extra α_j transitions mean that we may need to add more than $\mathbf{y}_m(d_i)$ copies of α_i to consume all the copies of d_i. The resulting configuration will be $\mathbf{z}_m^\Gamma \in \mathbb{N}^\Gamma$.

More formally, we choose large enough m such that the counts of species in Δ are no longer changing with m in $(\mathbf{y}_m)$; thus, the same $\mathbf{e}$ and the same appended transitions will suffice for all larger m. Recall that $\mathbf{x}_m \Longrightarrow_{p_m} \mathbf{y}_m$ and p_m does not contain an m-bottleneck. We'll apply Lemma 4.5 on this path with $b_2 = m$ and let b_1 be the largest count of any species in Δ anywhere in the sequence $(\mathbf{y}_m)$. (If necessary, increase m further to ensure $b_2 > |\Lambda| \cdot b_1$.) Note that with these parameters, $\Delta = \mathsf{bdd}((\mathbf{y}_m))$ exactly matches the set of states "Δ" defined in the statement of Lemma 4.5. Then this lemma tells us that there is an ordering on Δ, so that we can write $\Delta = \{d_1, \ldots, d_k\}$, such that for each $1 \le i \le k$, there is a transition $\alpha_i : d_i, s_i \to o_i, o_i'$ such that $d_i \in \Delta$ and $s_i, o_i, o_i' \notin \{d_1, \ldots, d_i\}$. (The fact that

$s_i \notin \{d_1, \ldots, d_i\}$ is not used in this Claim, but it will be essential for Claim 2).

For all $i \in \{1, \ldots, k\}$, let

- $c_i = \mathbf{y}_m(d_i) + \sum_{j=1}^{i-1} \{c_j o_j, c_j o'_j\}$ (note that if $o_j = o'_j$, then $\{c_j o_j, c_j o'_j\} = \{2c_j o_j\}$)
- $\mathbf{e}_i = \{c_i s_i\}$

Given a transition α and $j \in \mathbb{N}$, let $j \cdot \alpha$ denote the transition sequence consisting of j copies of α. For all $i \in \{0, \ldots, k\}$, define $\mathbf{z}_{m,i}$ as follows, where $\mathbf{z}_{m,0} = \mathbf{y}_m$:

$$\begin{aligned}
&\mathbf{z}_{m,0} + \mathbf{e}_1 \Longrightarrow_{c_1 \cdot \alpha_1} \mathbf{z}_{m,1} \\
&\mathbf{z}_{m,1} + \mathbf{e}_2 \Longrightarrow_{c_2 \cdot \alpha_2} \mathbf{z}_{m,2} \\
&\qquad \ldots \\
&\mathbf{z}_{m,k-1} + \mathbf{e}_k \Longrightarrow_{c_k \cdot \alpha_k} \mathbf{z}_{m,k}.
\end{aligned}$$

For all $i \in \{0, \ldots, k\}$, define path $p_{m,i}$ inductively to be $p_{m,i-1}$ followed by $c_i \cdot \alpha_i$, where the base case is $p_{m,0} = p_m$, so that $\mathbf{x}_m + \sum_{j=1}^{i} \mathbf{e}_j \Longrightarrow_{p_{m,i}} \mathbf{z}_{m,i}$. We prove by induction on i that:

1. $p_{m,i}$ is a valid transition sequence (i.e., it never has a transition in a configuration in which the input states are not present),
2. for all $j \in \{1, \ldots, i\}$, $\mathbf{z}_{m,i}(d_j) = 0$, and
3. for all $j \in \{i+1, \ldots, k\}$, $\mathbf{z}_{m,i}(d_j) = \mathbf{y}_m(d_j) + \left(\sum_{\ell=1}^{i} \{c_\ell o_\ell, c_\ell o'_\ell\}\right)(d_j)$ (in particular, $\mathbf{z}_{m,i}(d_{i+1}) = c_{i+1}$, which is the amount that we remove in path $c_{i+1} \cdot \alpha_{i+1}$).

The base case $i = 0$ for 1 follows from the fact that p_m is a valid transition sequence to apply to $\mathbf{x}_m$. The base case is vacuous for 2. For 3, observe that $\mathbf{z}_{m,0} = \mathbf{y}_m$ and the sum is empty when $i = 0$.

Assume the inductive case for $i - 1$. We have $\mathbf{z}_{m,i} = \mathbf{z}_{m,i-1} + \mathbf{e}_i + c_i\{o_i, o'_i\} - c_i\{d_i, s_i\}$ where $c_i\{o_i, o'_i\} - c_i\{d_i, s_i\}$ is the effect of applying transition α_i for c_i times. Since $\mathbf{e}_i = \{c_i s_i\}$, we have $\mathbf{z}_{m,i} = \mathbf{z}_{m,i-1} + c_i\{o_i, o'_i\} - c_i\{d_i\}$. By the induction hypothesis 3 for $i - 1$, $\mathbf{z}_{m,i-1}(d_i) = c_i$. Since $d_i \notin \{o_i, o'_i\}$, this implies that $c_i \alpha_i$ is a valid path (which establishes the inductive case for 1). Further, for all $j \in \{1, \ldots, i-1\}$, $d_j \notin \{o_i, o'_i, d_i\}$, which implies that the amount of d_j is not changed by α_i, and by inductive hypothesis 2, for all $j \in \{1, \ldots, i-1\}$, $\mathbf{z}_{m,i}(d_j) = 0$. Since also $\mathbf{z}_{m,i}(d_i) = 0$, we establish the inductive case for 2.

Inductive case 3 is proven as follows. Let $j \in \{i+1, \ldots, k\}$. Induction hypothesis 3 gives that $\mathbf{z}_{m,i-1}(d_j) = \mathbf{y}_m(d_j) + \left(\sum_{\ell=1}^{i-1} \{c_\ell o_\ell, c_\ell o'_\ell\}\right)(d_j)$. Let $\mathbf{b} = \{c_i o_i, c_i o'_i\}$ be the new term in the sum for the inductive case i; we must show that $\mathbf{z}_{m,i} = \mathbf{z}_{m,i-1} + \mathbf{b}$. Then $\mathbf{b}(d_j) = 0$ if d_j is not an output state of α_i, $\mathbf{b}(d_j) = c_i$ if d_j is exactly one output state, and $\mathbf{b}(d_j) = 2c_i$ if d_j is both output states. Thus after applying $c_i \cdot \alpha_i$ to $\mathbf{z}_{m,i-1} + \mathbf{e}_i$ to result in configuration $\mathbf{z}_{m,i}$, we have increased the count of d_j by exactly $\mathbf{b}(d_j)$, resulting in $\mathbf{z}_{m,i}(d_j) = \mathbf{y}_m(d_j) + \left(\sum_{\ell=1}^{i} \{c_\ell o_\ell, c_\ell o'_\ell\}\right)(d_j)$, proving the inductive case for 3.

To complete the proof we let $\mathbf{z}^{\Gamma}_m = \mathbf{z}_{m,k}$, for the final value k. Inductive case 2 on this final value k, shows that for all $j \in \{1, \ldots, k\}, \mathbf{z}_{m,k}(d_j) = \mathbf{z}^{\Gamma}_m(d_j) = 0$, thus proving that $\mathbf{z}^{\Gamma}_m \in \mathbb{N}^{\Gamma}$. Finally, note that $\mathbf{e} = \sum_{i=1}^{k} \mathbf{e}_i$ in the statement of the claim. □

Intuitively, Claim 2 below works toward generating the vector of states $\mathbf{e}$ that we needed for Claim 1. The "cost" for Claim 2 is that the path must be taken "in the context" of additional agents in states captured by $\mathbf{p}$. Importantly, the net effect of the path preserves $\mathbf{p}$, which will give us a way to "interleave" Claims 1 and 2 as shown in Claim 3.

Claim 2 *For all $\mathbf{e} \in \mathbb{N}^{\Lambda}$, there is $\mathbf{p} \in \mathbb{N}^{\Lambda}$, such that for all large enough m, there is $\mathbf{w}^{\Gamma}_m \in \mathbb{N}^{\Gamma}$, such that $\mathbf{p} + \mathbf{x}_m \Longrightarrow \mathbf{p} + \mathbf{w}^{\Gamma}_m + \mathbf{e}$.*

Example. Recall the example above illustrating Claim 1. Claim 2 is more difficult than Claim 1 for two reasons. First, we need to be able to obtain any counts of states a, b, c, f in $\mathbf{e}$, and not only ensure that $a = b = c = 0$. Second, we no longer have the freedom to consume extra states (i.e., $\mathbf{e}$ in Claim 1). Note that $\mathbf{p}$ cannot fulfill the same role as $\mathbf{e}$ did in Claim 1 because $\mathbf{p}$ must be recovered at the end.

For concreteness, suppose $\mathbf{e}$ consists of $a = 7$, $b = 2$, $c = 0$, $f = 3$. To start with, note that handling state f in $\mathbf{e}$ is easy: recall f is in $\Gamma = \Lambda \setminus \Delta$ and is present in "large" count in $\mathbf{y}_m$. We can simply "siphon" the required number of agents in state f into $\mathbf{e}$ leaving the rest as $\mathbf{w}^{\Gamma}_m$. For the rest of $\mathbf{e}$, recall that $\mathbf{y}_m$ has $a = 3$, $b = 2$, $c = 1$. How can we generate an additional 4 copies of a? Note that all transitions preserve or decrease the sum $a + b + c$. Thus we cannot solely add interactions to p_m to get to our desired $\mathbf{e}$. The key is that we can increase a by removing existing interactions from p_m that consumed it. Indeed, Lemma 4.5 helps us by giving a lower bound on the number of instances of transitions (5.1),(5.4),(5.5) that must have occurred in p_m. (Note that in Claim 1, we didn't need to use the fact that these transitions occurred in p_m. Now, we need to ensure that there are enough instances for us to remove.)

In our case, to increase a by 4, we can remove 4 instances of interaction (5.1) from p_m, resulting in $a = 7, b = 6, c = -3$.[17] To get $c = 0$ as desired, we can remove 3 instances of transition (5.4), resulting in $a = 7, b = 3, c = 0$. Finally, we add 1 instance of transition (5.5) to get $a = 7, b = 2, c = 0$ as desired.

Note that unlike in Claim 1, we have more potential for circularity now because we cannot add the other input to a transition as $\mathbf{e}$. For example, we can't use transition (5.3) to affect c because it affects a (which we have previously driven to the desired count). Luckily, the ordering given by Lemma 4.5 avoids any circularity because the other input and both of the outputs come later in the ordering.

Importantly, as we remove or add interactions to p_m, we could potentially drive the count of some state negative—but only temporarily because the *final* counts $(\mathbf{w}^\Gamma + \mathbf{e})$ are nonnegative. Performing these interactions in the context of more agents $(\mathbf{p})$ ensures that the path is valid.

Proof (of Claim 2) Define $\mathbf{e}^\Delta \in \mathbb{N}^\Delta$ by $\mathbf{e}^\Delta(d) = \mathbf{e}(d)$ for all $d \in \Delta$ and $\mathbf{e}^\Delta(s) = 0$ for all $s \in \Gamma$, and define $\mathbf{e}^\Gamma \in \mathbb{N}^\Gamma$ similarly, so that $\mathbf{e} = \mathbf{e}^\Delta + \mathbf{e}^\Gamma$. (Recall that Δ and Γ partition the set of states Λ.) We proceed by proving the following claim, which focuses only on the $\mathbf{e}^\Delta$ part of $\mathbf{e}$, and which additionally ensures that $\mathbf{w}_m^\Gamma$ grows on all states in Γ as $m \to \infty$:

(*) *For all $\mathbf{e}^\Delta \in \mathbb{N}^\Delta$, there is $\mathbf{p} \in \mathbb{N}^\Lambda$, such that for all large enough m, there is $\mathbf{w}_m^\Gamma \in \mathbb{N}^\Gamma$, such that $\mathbf{p} + \mathbf{x}_m \Longrightarrow \mathbf{p} + \mathbf{w}_m^\Gamma + \mathbf{e}^\Delta$ and $\mathsf{unbdd}((\mathbf{w}_m^\Gamma)) = \Gamma$.*

Supposing this claim is true, it is easy to complete the proof of Claim 2 by handling positive $\mathbf{e}^\Gamma$ as follows. Since $\mathsf{unbdd}((\mathbf{w}_m^\Gamma)) = \Gamma$, given any $\mathbf{e}^\Gamma$, $\mathbf{w}_m^\Gamma - \mathbf{e}^\Gamma$ is non-negative for large enough m. Then $\mathbf{p} + \mathbf{x}_m \Longrightarrow \mathbf{p} + \mathbf{w}_m^\Gamma + \mathbf{e}^\Delta = \mathbf{p} + (\mathbf{w}_m^\Gamma - \mathbf{e}^\Gamma) + \mathbf{e}$. In other words we apply claim (*) to produce $\mathbf{e}^\Delta$, and then "siphon" the remaining states $\mathbf{e}^\Gamma$ from $\mathbf{w}_m^\Gamma$ to produce $\mathbf{e} = \mathbf{e}^\Delta + \mathbf{e}^\Gamma$, which maintains the required conclusions of Claim 2 with $(\mathbf{w}_m^\Gamma - \mathbf{e}^\Gamma)$ replacing $\mathbf{w}_m^\Gamma$.

We now show how to prove the above claim (*). Recall that $\mathbf{x}_m \Longrightarrow_{p_m} \mathbf{y}_m$ and p_m does not contain an m-bottleneck. Intuitively, we will try to modify p_m so that in the end we get exactly $\mathbf{e}^\Delta$ of Δ. As in the proof of Claim 1, we will use the fact that p_m does not contain an m-bottleneck and Lemma 4.5 to find transitions affecting Δ in a non-circular manner. However, unlike in Claim 1, we cannot simply consume additional states (i.e., $\mathbf{e} \in \mathbb{N}^\Lambda$ in Claim 1) to ensure that the count of the "other input state s_i" does not become negative. Rather, to increase the amounts of s_i we will remove certain transition instances originally in p_m. It turns out that even with removing transitions, our modification to p_m may still *temporarily* take certain states negative if we start from $\mathbf{x}_m$. However, executing the path in the context of $\mathbf{p}$ provides "buffer room" to ensure that no counts ever go below zero.

More formally, as in the proof of Claim 1 apply Lemma 4.5 with $b_2 = m$ and let b_1 be the largest count of any state in Δ anywhere in the sequence $(\mathbf{y}_m)$. The lower bound on $b_2 = m$ is determined below ("bound on the amount of fixing"). Lemma 4.5 tells us that there is an ordering on Δ, so that we can write $\Delta = \{d_1, \ldots, d_k\}$, such that for each $1 \le i \le k$, there is a transition $\alpha_i : d_i, s_i \to o_i, o_i'$ such that $d_i \in \Delta$ and $s_i, o_i, o_i' \notin \{d_1, \ldots, d_i\}$, and α_i occurs at least $(b_2 - |\Lambda| \cdot b_1)/|\Lambda|^2$ times in p_m. Note that the final condition was not necessary to prove Claim 1 since its proof only added transitions to p_m. However, since the current proof removes transitions as well, we require this condition to ensure that there are sufficiently many existing instances to be removed.

We iteratively fix the counts of states in Δ one by one, in the ordering given, i.e. we first adjust p_m to fix d_1, then we fix d_2 (while showing that the fixing of d_2 cannot affect the count of d_1 in any configuration, so it remains fixed), etc. We start with $\mathbf{e}_0^\Delta(s) = \mathbf{y}_m(s)$ for $s \in \Delta$ and $\mathbf{e}_0^\Delta(s) = 0$ for $s \in \Gamma$, and $\mathbf{w}_{m,0}^\Gamma(s) = \mathbf{y}_m(s)$ for $s \in \Gamma$ and $\mathbf{w}_{m,0}^\Gamma(s) = 0$ for $s \in \Delta$. Having fixed $d_1, \ldots, d_{i-1}$, and obtaining new $\mathbf{e}_{i-1}^\Delta$, $\mathbf{w}_{m,i-1}^\Gamma$ (which could now be negative) such that $\mathbf{e}_{i-1}^\Delta$ agrees with the desired $\mathbf{e}^\Delta$ over $d_1, \ldots, d_{i-1}$, we process d_i as follows. If $\delta_i = \mathbf{e}^\Delta(d_i) - \mathbf{e}_{i-1}^\Delta(d_i) < 0$: add δ_i instances of transition α_i at the end of the transition sequence. If $\delta_i > 0$: remove δ_i instances of α_i where they occur in the transition sequence; property (3) ensures that q contains enough instances of α_i (see below). Let $\mathbf{e}_i^\Delta$ be the counts of the states in Δ at the end of this path. By property (2) and (3), adding or removing instances of α_i affects only the counts of states in Γ and $d_{i+1}, \ldots, d_k$. Since we fix these counts in the prescribed order, when we are done, the counts of each d_i is equal to its count in $\mathbf{e}^\Delta$ (ie $\mathbf{e}^\Delta = \mathbf{e}_k^\Delta$), while counts of elements of Γ have been altered (letting $\mathbf{w}_m^\Gamma = \mathbf{w}_{m,k}^\Gamma$). We now claim that for large enough m, $\mathbf{w}_{m,k}^\Gamma$ is nonnegative, and that $\mathbf{p}$ can be indepen-

[17] Note the need for $\mathbf{p}$ to ensure that the total count never goes negative. In writing "$a = 7, b = 6, c = -3$", we are examining the effect only on $\mathbf{y}_m$ of modifying p_m, but in applying the lemma, the starting configuration is $\mathbf{x}_m + \mathbf{p}$, not merely $\mathbf{x}_m$, so the actual count of each state s will be $\mathbf{p}(s)$ larger than just stated.

dent of m. Finally, we derive a bound on the number of transition instances that we may need to remove, which determines another bound on m (ie b_2) to ensure that there are enough instances by property (4) above.

Note that the amount of fixing we need to do only depends on the desired $\mathbf{e}^{\Delta}$ as well as on the counts of Δ states in $\mathbf{y}_m$. Because $\mathbf{y}_m$ are nondecreasing, and $\Delta = \mathsf{bdd}((\mathbf{y}_m))$, for large enough m, the counts of Δ states in $\mathbf{y}_m$ stop changing, and the amount of fixing depends only on the desired $\mathbf{e}^{\Delta}$. This implies that the $\mathbf{p}$ we need to add to ensure that no counts go negative can be independent of m. Further, for large enough m, the difference between $\mathbf{w}_m^{\Gamma}$ and $\mathbf{y}_m$ is independent of m, and thus $\mathsf{unbdd}(\mathbf{w}_m^{\Gamma}) = \mathsf{unbdd}(\mathbf{y}_m) = \Gamma$, as needed for claim (*). This also implies that for large enough m, $\mathbf{w}_{m,k}^{\Gamma}$ is nonnegative.

Bound on the amount of fixing: We now derive a bound the number of transition instances that must be added/removed, in order to justify that this bound depends only on $\mathbf{e}$, but is independent of m. Define the quantity $c_b = \max_{m \in \mathbb{N}, d \in \Delta} |\mathbf{y}_m(d) - \mathbf{e}^{\Delta}(d)|$ as the maximum amount that any state in Δ deviates from its desired count. We add or remove at most $|\delta_1| \leq c_b$ instances of α_1, which affects the count of states in $\Gamma \cup \{d_2, \ldots, d_k\}$ by at most $2c_b$ (it could be 2 per transition if the transition is $\alpha_1 : d_1, s \to s', s'$ for some state $s' \in \Lambda$). Thus, $|\delta_2| \leq c_b + 2|\delta_1|$ (the original c_b error plus the additional error from altering the number of α_1 transitions). In general, $|\delta_i| \leq c_b + 2(|\delta_1| + \cdots + |\delta_{i-1}|) \leq 3^{i-1}c_b$. Thus if we let $m = b_2 \geq k \cdot b_1 + 3^{k-1}c_b|\Lambda|^2$, we will have enough transition instances by property (4) to remove $((b_2 - |\Lambda| \cdot b_1)/|\Lambda|^2 = 3^{i-1}c_b)$. □

Claim 3 *For infinitely many* $\mathbf{i} \in I$*, there is* $\mathbf{v}^{\Gamma} \in \mathbb{N}^{\Gamma}$ *such that* $\mathbf{i} \Longrightarrow \mathbf{v}^{\Gamma}$.

Proof Intuitively, Claim 3 follows by expressing $\mathbf{i} = 2\mathbf{i}'$ where $\mathbf{i}' \in I'$ and $\mathbf{i}' \Longrightarrow \mathbf{x}_m$, so $2\mathbf{i}' \Longrightarrow 2\mathbf{x}_m$. We then apply Claim 2 to one copy of $\mathbf{x}_m$ (with the other $\mathbf{x}_m$ playing the role of $\mathbf{p}$) to get to a configuration with the correct $\mathbf{e}$ for Claim 1, and then apply Claim 1 to remove all states in Δ.

Choose m large enough to satisfy the conditions stated below as they are needed. By Claim 1, there is $\mathbf{e} \in \mathbb{N}^{\Lambda}$ and $\mathbf{z}^{\Gamma} \in \mathbb{N}^{\Gamma}$ such that $\mathbf{x}_m + \mathbf{e} \Longrightarrow_{p_m} \mathbf{z}^{\Gamma}$. Apply Claim 2 on $\mathbf{e}$ (making sure m is large enough to satisfy the claim on $\mathbf{e}$). Thus, there is $\mathbf{p} \in \mathbb{N}^{\Lambda}$ and $\mathbf{w}^{\Gamma} \in \mathbb{N}^{\Gamma}$ such that $\mathbf{p} + \mathbf{x}_m \Longrightarrow_{p_m} \mathbf{p} + \mathbf{w}^{\Gamma} + \mathbf{e}$. If m is large enough that $\mathbf{x}_m \geq \mathbf{p}$, then $\mathbf{x}_m + \mathbf{x}_m \Longrightarrow_{p_m} \mathbf{x}_m + \mathbf{w}^{\Gamma} + \mathbf{e}$. Then, by Claim 1, $\mathbf{x}_m + \mathbf{w}^{\Gamma} + \mathbf{e} \Longrightarrow_{p_m} \mathbf{w}^{\Gamma} + \mathbf{z}^{\Gamma}$. To complete the claim, we let $\mathbf{v}^{\Gamma} = \mathbf{w}^{\Gamma} + \mathbf{z}^{\Gamma}$. □

Finally, Theorem 3.8 is proven because $\mathbf{v}^{\Gamma}$ is Q-stable and it contains zero count of states in Δ. To see that $\mathbf{v}^{\Gamma}$ is Q-stable recall that $\mathbf{v}^{\Gamma} \leq \mathbf{y}_{m'}$ for sufficiently large m' since $\Gamma = \mathsf{unbdd}((\mathbf{y}_m))$ and $\mathbf{v}^{\Gamma}$ contains only states in Γ. Since stability is closed downward, and $\mathbf{y}_{m'}$ is Q-stable, we have that $\mathbf{v}^{\Gamma}$ is Q-stable as well.

References

1. Dan Alistarh, James Aspnes, David Eisenstat, Rati Gelashvili, and Ronald L. Rivest. Time-space trade-offs in molecular computation. Technical Report 1602.08032, arXiv, 2016.
2. Dan Alistarh and Rati Gelashvili. Polylogarithmic-time leader election in population protocols. In *ICALP 2015: Proceedings of the 42nd International Colloquium on Automata, Languages, and Programming, Kyoto, Japan*, 2015.
3. Dan Alistarh, Rati Gelashvili, and Milan Vojnović. Fast and exact majority in population protocols. In *PODC 2015: Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing*, pages 47–56. ACM, 2015.
4. Dana Angluin, James Aspnes, Zoë Diamadi, Michael Fischer, and René Peralta. Computation in networks of passively mobile finite-state sensors. *Distributed Computing*, 18:235–253, 2006. Preliminary version appeared in PODC 2004.
5. Dana Angluin, James Aspnes, Zoë Diamadi, Michael J. Fischer, and René Peralta. Urn automata. Technical Report YALEU/DCS/TR-1280, Yale University, November 2003.
6. Dana Angluin, James Aspnes, and David Eisenstat. Fast computation by population protocols with a leader. *Distributed Computing*, 21(3):183–199, September 2008. Preliminary version appeared in DISC 2006.
7. Dana Angluin, James Aspnes, and David Eisenstat. A simple population protocol for fast robust approximate majority. *Distributed Computing*, 21(2):87–102, July 2008.
8. Dana Angluin, James Aspnes, David Eisenstat, and Eric Ruppert. The computational power of population protocols. *Distributed Computing*, 20(4):279–304, 2007.
9. Dana Angluin, James Aspnes, Michael J Fischer, and Hong Jiang. Self-stabilizing population protocols. In *Principles of Distributed Systems*, pages 103–117. Springer, 2006.
10. James Aspnes, Joffroy Beauquier, Janna Burman, and Devan Sohier. Time and space optimal counting in population protocols. 2016.
11. Joffroy Beauquier, Janna Burman, Simon Clavière, and Devan Sohier. Space-optimal counting in population protocols. In *DISC 2015: Proceedings of the 29th International Symposium on Distributed Computing*, pages 631–646, 2015.
12. James M Bower and Hamid Bolouri. *Computational modeling of genetic and biochemical networks*. MIT press, 2004.

13. Ho-Lin Chen, Rachel Cummings, David Doty, and David Soloveichik. Speed faults in computation by chemical reaction networks. In *DISC 2014: Proceedings of the 28th International Symposium on Distributed Computing, Austin, TX, USA*, pages 16–30, 2014.
14. Yuan-Jyue Chen, Neil Dalchau, Niranjan Srinivas, Andrew Phillips, Luca Cardelli, David Soloveichik, and Georg Seelig. Programmable chemical controllers made from DNA. *Nature Nanotechnology*, 8(10):755–762, 2013.
15. Inês Cunha-Ferreira, Inês Bento, and Mónica Bettencourt-Dias. From zero to many: control of centriole number in development and disease. *Traffic*, 10(5):482–498, 2009.
16. Leonard E. Dickson. Finiteness of the odd perfect and primitive abundant numbers with n distinct prime factors. *American Journal of Mathematics*, 35(4):413–422, October 1913.
17. David Doty. Timing in chemical reaction networks. In *SODA 2014: Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 772–784, January 2014.
18. David Doty and David Soloveichik. Stable leader election in population protocols requires linear time. In *DISC 2015: Proceedings of the 29th International Symposium on Distributed Computing*, Lecture Notes in Computer Science, pages 602–616. Springer Berlin Heidelberg, 2015.
19. Daniel T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *Journal of Physical Chemistry*, 81(25):2340–2361, 1977.
20. Tomoko Izumi, Keigo Kinpara, Taisuke Izumi, and Koichi Wada. Space-efficient self-stabilizing counting population protocols on mobile sensor networks. *Theoretical Computer Science*, 552:99–108, 2014.
21. Richard M Karp and Raymond E Miller. Parallel program schemata. *Journal of Computer and System Sciences*, 3(2):147–195, 1969.
22. Carl A Petri. Communication with automata. Technical report, DTIC Document, 1966.
23. David Soloveichik, Georg Seelig, and Erik Winfree. DNA as a universal substrate for chemical kinetics. *Proceedings of the National Academy of Sciences*, 107(12):5393, 2010. Preliminary version appeared in DNA 2008.
24. Vito Volterra. Variazioni e fluttuazioni del numero dindividui in specie animali conviventi. *Mem. Acad. Lincei Roma*, 2:31–113, 1926.