The Computational Power of Discrete Chemical Reaction Networks with Bounded Executions

David Doty
Computer Science, University of California – Davis, CA, USA

Ben Heckmann
CIT, Technical University of Munich, Germany
Computer Science, University of California – Davis, CA, USA

Abstract

Chemical reaction networks (CRNs) model systems where molecules interact according to a finite set of *reactions* such as $A + B \to C$, representing that if a molecule of A and B collide, they disappear and a molecule of C is produced. CRNs can compute Boolean-valued predicates $\phi : \mathbb{N}^d \to \{0, 1\}$ and integer-valued functions $f : \mathbb{N}^d \to \mathbb{N}$; for instance $X_1 + X_2 \to Y$ computes the function $\min(x_1, x_2)$, since starting with x_i copies of X_i, eventually $\min(x_1, x_2)$ copies of Y are produced.

We study the computational power of *execution bounded* CRNs, in which only a finite number of reactions can occur from the initial configuration (e.g., ruling out reversible reactions such as $A \rightleftharpoons B$). The power and composability of such CRNs depend crucially on some other modeling choices that do not affect the computational power of CRNs with unbounded executions, namely whether an initial leader is present, and whether (for predicates) all species are required to "vote" for the Boolean output. If the CRN starts with an initial leader, and can allow only the leader to vote, then all semilinear predicates and functions can be stably computed in $O(n \log n)$ parallel time by execution bounded CRNs.

However, if no initial leader is allowed, all species vote, and the CRN is "non-collapsing" (does not shrink from initially large to final $O(1)$ size configurations), then execution bounded CRNs are severely limited, able to compute only *eventually constant* predicates. A key tool is a characterization of execution bounded CRNs as precisely those with a nonnegative *linear potential function* that is strictly decreased by every reaction [6].

2012 ACM Subject Classification Theory of computation → Models of computation

Keywords and phrases chemical reaction networks, population protocols, stable computation

Digital Object Identifier 10.4230/LIPIcs.DISC.2024.20

Related Version *Full Version*: https://arxiv.org/abs/2405.08649

Funding *David Doty*: NSF awards 2211793, 1900931, 1844976, and DoE EXPRESS award SC0024467.
Ben Heckmann: NSF award 1844976.

1 Introduction

Chemical reaction networks (CRNs) are a fundamental tool for understanding and designing molecular systems. By abstracting chemical reactions into a set of finite, rule-based transformations, CRNs allow us to model the behavior of complex chemical systems. For instance, the CRN with a single reaction $2X \to Y$, produces one Y every time two X molecules randomly react together, effectively calculating the function $f(x) = \lfloor x/2 \rfloor$ if the initial count of X is interpreted as the input and the eventual count of Y as the output. A commonly studied special case of CRNs is the *population protocol* model of distributed computing [3], in which each reaction has exactly two reactants and two products, e.g., $A + B \to C + D$. This model assumes idealized conditions where reactions can proceed indefinitely, constrained only by the availability of reactants in the well-mixed solution.

Precisely the *semilinear* predicates $\phi : \mathbb{N}^d \to \{0, 1\}$ [1] and functions $f : \mathbb{N}^d \to \mathbb{N}$ [5] can be computed *stably*, roughly meaning that the output is correct no matter the order in which reactions happen. In population protocols or other CRNs with a finite reachable configuration space, this means that the output is correct with probability 1 under a stochastic scheduler that picks the next molecules to react at random. However, existing constructions to compute semilinear predicates and functions use CRNs with *unbounded executions*, meaning that it is possible to execute infinitely many reactions from the initial configuration. CRNs with *bounded executions* have several advantages. With an absolute guarantee on how many reactions will happen before the CRN terminates, wet-lab implementations need only supply a bounded amount of fuel to power the reactions. Such CRNs are simpler to reason about: each reaction brings it "closer" to the answer. They also lead to a simpler definition of stable computation than is typically employed: an execution bounded CRN stably computes a predicate/function if it gets the correct answer after sufficiently many reactions.

To study this topic, we study networks that must eventually reach a configuration where no further reactions can occur, regardless of the sequence of reactions executed. This restriction is nontrivial because the techniques of [5, 7] rely on reversible reactions (leading to unbounded executions) catalyzed by species we expect to be depleted once a computational step has terminated. This trick seems to add computational power to our system by undoing certain reactions as long as a specific species is present. Consider the following CRN computing $f(x_1, x_2, x_3) = \min(x_1 - x_2, x_3)$. The input values x_i are given by the counts of X_i, and the output by the count of Z molecules in the stable state:

$$X_1 \to Y \tag{1}$$
$$X_2 + Y \to \varnothing \tag{2}$$
$$Y + X_3 \to Z \tag{3}$$
$$Z + X_2 \to X_2 + X_3 + Y \tag{4}$$

Reactions (1) and (2) compute $x_1 - x_2$, storing the result in the count of Y. Next, reaction (3) can be applied exactly $\min(y, x_3)$ times. But since the order of reactions is a stochastic process, we might consume copies of Y in (3), before all of x_2 is subtracted from it. Therefore, we add reaction (4), using X_2 as a catalyst to undo reaction (3) as long as copies of X_2 are present, indicating that the first step of computation has not terminated. However, this means the above CRN does not have bounded exectutions, since reactions (3) and (4) can be alternated in an infinite execution. A similar technique is used in [5], where semilinear sets are understood as a finite union of linear sets, shown to be computable in parallel by CRNs. A reversible, catalyzed reaction finally converts the output of one of the CRNs to the global output. Among other questions, we explore how the constructions of [5] and [7] can be modified to provide equal computational power while guaranteeing bounded execution.

The paper is organized as follows. Section 3 defines execution boundedness (Definition 3.1). We introduce alternative characterizations of the class for use in later proofs, such as the lack of self-covering execution paths. Section 4 and 5 contain the main positive results of the paper and provide the concrete constructions used to decide semilinear sets and functions using execution bounded CRNs whose initial configurations contain a single leader. Section 6 discusses the limitations of execution bounded CRNs, introducing the concept of a "linear potential function" as a core characterization of these systems. We demonstrate that entirely execution bounded CRNs that are leaderless and non-collapsing (such as all population protocols), can only stably decide trivial semilinear predicates: the *eventually constant* predicates (Definition 6.6).

2 Preliminaries

We use established notation from [5, 7] and stable computation definitions from [3] for (discrete) chemical reaction networks.

2.1 Notation

Let $\mathbb{N}$ denote the nonnegative integers. For any finite set Λ, we write $\mathbb{N}^\Lambda$ to mean the set of functions $f : \Lambda \to \mathbb{N}$. Equivalently, $\mathbb{N}^\Lambda$ can be interpreted as the set of vectors indexed by the elements of Λ, and so $\mathbf{c} \in \mathbb{N}^\Lambda$ specifies nonnegative integer counts for all elements of Λ. $\mathbf{c}(i)$ denotes the i-th coordinate of $\mathbf{c}$, and if $\mathbf{c}$ is indexed by elements of Λ, then $\mathbf{c}(Y)$ denotes the count of species $Y \in \Lambda$. We sometimes use multiset notation for such vectors, e.g., $\{A, 3C\}$ for the vector $(1, 0, 3)$, assuming there are three species A, B, C. If $\Sigma \subseteq \Lambda$, then $\mathbf{i} \upharpoonright \Sigma$ denotes restriction of $\mathbf{i}$ to Σ.

For two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^k$, we write $\mathbf{x} \geqq \mathbf{y}$ to denote that $\mathbf{x}(i) \geq \mathbf{y}(i)$ for all $1 \leq i \leq k$, $\mathbf{x} \geq \mathbf{y}$ to denote that $\mathbf{x} \geqq \mathbf{y}$ but $\mathbf{x} \neq \mathbf{y}$, and $\mathbf{x} > \mathbf{y}$ to denote that $\mathbf{x}(i) > \mathbf{y}(i)$ for all $1 \leq i \leq k$. In the case that $\mathbf{y} = \mathbf{0}$, we say that $\mathbf{x}$ is *nonnegative*, *semipositive*, and *positive*, respectively. Similarly define $\leqq, \leq, <$.

For a matrix or vector $\mathbf{x}$, define $\|\mathbf{x}\| = \|\mathbf{x}\|_1 = \sum_i |\mathbf{x}(i)|$, i ranges over all the entries of $\mathbf{x}$.

2.2 Chemical Reaction Networks

A *chemical reaction network* (CRN) is a pair $\mathcal{C} = (\Lambda, R)$, where Λ is a finite set of chemical *species*, and R is a finite set of reactions over Λ, where each *reaction* is a pair $(\mathbf{r}, \mathbf{p}) \in \mathbb{N}^\Lambda \times \mathbb{N}^\Lambda$ indicating the *reactants* $\mathbf{r}$ and *products* $\mathbf{p}$. A *population protocol* [1] is a CRN in which all reactions $(\mathbf{r}, \mathbf{p})$ obey $\|\mathbf{r}\| = \|\mathbf{p}\| = 2$. (Note that CRNs, including population protocols, do not assume any underlying "communication graph" and model a well-mixed system in which each equal-sized of molecules is as likely to collide and react as any other.) We write reactions such as $A + 2B \to A + 3C$ to represent the reaction $(\{A, 2B\}, \{A, 3C\})$. A *configuration* $\mathbf{c} \in \mathbb{N}^\Lambda$ of a CRN assigns integer counts to every species $S \in \Lambda$. When convenient, we use the notation $\{n_1 S_1, n_2 S_2, \ldots, n_k S_k\}$ to describe a configuration $\mathbf{c}$ with $n_i \in \mathbb{N}$ copies of species S_i, i.e., $\mathbf{c}(S_i) = n_i$, and any species that is not listed is assumed to have a zero count. If some configuration $\mathbf{c}$ is understood from context, for a species S, we write $\#S$ to denote $\mathbf{c}(S)$. A reaction $(\mathbf{r}, \mathbf{p})$ is said to be *applicable* in configuration $\mathbf{c}$ if $\mathbf{r} \leqq \mathbf{c}$. If the reaction $(\mathbf{r}, \mathbf{p})$ is applicable, applying it results in configuration $\mathbf{c}' = \mathbf{c} - \mathbf{r} + \mathbf{p}$, and we write $\mathbf{c} \to \mathbf{c}'$.

An *execution* $\mathcal{E}$ is a finite or infinite sequence of one or more configurations $\mathcal{E} = (\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2, \ldots)$ such that, for all $i \in \{1, \ldots, |\mathcal{E}| - 1\}, \mathbf{c}_{i-1} \to \mathbf{c}_i$ and $\mathbf{c}_{i-1} \neq \mathbf{c}_i$. $\mathbf{x} \Rightarrow_P \mathbf{y}$ denotes that P is finite, starts at $\mathbf{x}$, and ends at $\mathbf{y}$. In this case we say $\mathbf{y}$ is *reachable* from $\mathbf{x}$. Let $\mathsf{reach}(\mathbf{x}) = \{\mathbf{y} \mid \mathbf{x} \Rightarrow \mathbf{y}\}$. Note that the reachability relation is *additive*: if $\mathbf{x} \Rightarrow \mathbf{y}$, then for all $\mathbf{c} \in \mathbb{N}^\Lambda$, $\mathbf{x} + \mathbf{c} \Rightarrow \mathbf{y} + \mathbf{c}$.

For a CRN $\mathcal{C} = (\Lambda, R)$ where $|\Lambda| = n$ and $|R| = m$, define the $n \times m$ *stoichiometric matrix* $\mathbf{M}$ of $\mathcal{C}$ as follows. The species are ordered $S_1, \ldots, S_n$, and the reactions are ordered $(\mathbf{r}_1, \mathbf{p}_1), \ldots, (\mathbf{r}_m, \mathbf{p}_m)$, and $\mathbf{M}_{ij} = \mathbf{p}_j(S_i) - \mathbf{r}_j(S_i)$. In other words, $\mathbf{M}_{ij}$ is the net amount of S_i produced when executing the j'th reaction. For instance, if the CRN has two reactions $S_1 \to S_2 + 2S_3$ and $3S_2 + S_3 \to S_1 + S_2 + S_3$, then $\mathbf{M} = \begin{pmatrix} -1 & 1 \\ 1 & -2 \\ 2 & 0 \end{pmatrix}$.

▶ Remark 2.1. Let $\mathbf{u} \in \mathbb{N}^R$. Then the vector $\mathbf{Mu} \in \mathbb{Z}^\Lambda$ represents the change in species counts that results from applying reactions by amounts described in $\mathbf{u}$. In the above example, if $\mathbf{u} = (2, 1)$, then $\mathbf{Mu} = (-1, 0, 4)$, meaning that executing the first reaction twice ($\mathbf{u}(1) = 2$) and the second reaction once ($\mathbf{u}(2) = 1$) causes S_1 to decrease by 1, S_2 to stay the same, and S_3 to increase by 4.

2.3 Stable computation with CRNs

To capture the result of computations done by a CRN, we generalize the definitions to include information about how to interpret the final configuration after letting the CRN run until the result cannot change anymore (characterized below as *stable computation*). Computation primarily involves two classes of functions: 1. evaluating predicates $\phi : \mathbb{N}^k \to \{0, 1\}$ to determine properties of the input, and 2. executing general functions that map an input configuration to an output, denoted as $f : \mathbb{N}^k \to \mathbb{N}$.

The definitions below reference *input species* $\Sigma \subseteq \Lambda$ and an *initial context* $\mathbf{s} \in \mathbb{N}^{\Lambda \setminus \Sigma}$. If $\mathbf{s} = \mathbf{0}$ we say that CRN is *leaderless*. The initial context may be any constant multiset of species, though in practice it tends to be a single "leader" molecule. Furthermore, other initial contexts such as $\{2A, 3B\}$ could be produced from a single leader L via a reaction $L \to 2A + 3B$, so we may assume without loss of generality that the initial context, if it is nonzero, is simply a single leader. In both cases, we say $\mathbf{i} \in \mathbb{N}^\Lambda$ is a *valid initial configuration* if $\mathbf{i} = \mathbf{s} + \mathbf{x}$, where $\mathbf{x}(S) = 0$ for all $S \in \Lambda \setminus \Sigma$; i.e., $\mathbf{i}$ is the initial context plus only input species.

A *chemical reaction decider* (CRD) is a tuple $\mathcal{D} = (\Lambda, R, \Sigma, \Upsilon_1, \Upsilon_0, \mathbf{s})$, where (Λ, R) is a CRN, $\Sigma \subseteq \Lambda$ is the set of *input species*, $\Upsilon_1 \subseteq \Lambda$ is the set of *yes voters*, and $\Upsilon_0 \subseteq \Lambda$ is the set of *no voters*, such that $\Upsilon_1 \cap \Upsilon_0 = \emptyset$, and $\mathbf{s} \in \mathbb{N}^{\Lambda \setminus \Sigma}$ is the *initial context*. If $\Upsilon_1 \cup \Upsilon_0 = \Lambda$, we say the CRD is *all-voting*. We define a global output partial function $\Phi : \mathbb{N}^\Lambda \dashrightarrow \{0, 1\}$ as follows. $\Phi(\mathbf{c})$ is undefined if either $\mathbf{c} = \mathbf{0}$, or if there exist $S_0 \in \Upsilon_0$ and $S_1 \in \Upsilon_1$ such that $\mathbf{c}(S_0) > 0$ and $\mathbf{c}(S_1) > 0$. In other words, we require a unanimous vote as our output. We say $\mathbf{c}$ is *stable* if, for all $\mathbf{c}'$ such that $\mathbf{c} \Rightarrow \mathbf{c}'$, $\Phi(\mathbf{c}) = \Phi(\mathbf{c}')$. We say a CRD $\mathcal{D}$ *stably decides* the predicate $\psi : \mathbb{N}^\Sigma \to \{0, 1\}$ if, for any valid initial configuration $\mathbf{i} \in \mathbb{N}^\Lambda$, letting $\mathbf{i}_0 = \mathbf{i} \upharpoonright \Sigma$, for all configurations $\mathbf{c} \in \mathbb{N}^\Lambda, \mathbf{i} \Rightarrow \mathbf{c}$ implies $\mathbf{c} \Rightarrow \mathbf{c}'$ such that $\mathbf{c}'$ is stable and $\Phi(\mathbf{c}') = \psi(\mathbf{i}_0)$. We associate to a predicate ψ the set $A = \psi^{-1}(1)$ of inputs on which ψ outputs 1, so we can equivalently say the CRD *stably decides* the set A.

A chemical reaction computer (CRC) is a tuple $\mathcal{C} = (\Lambda, R, \Sigma, Y, \mathbf{s})$, where (Λ, R) is a CRN, $\Sigma \subset \Lambda$ is the set of *input species*, $Y \in \Lambda \setminus \Sigma$ is the *output species*, and $\mathbf{s} \in \mathbb{N}^{\Lambda \setminus \Sigma}$ is the *initial context*. A configuration $\mathbf{o} \in \mathbb{N}^\Lambda$ is *stable* if, for every $\mathbf{c}$ such that $\mathbf{o} \Rightarrow \mathbf{c}, \mathbf{o}(Y) = \mathbf{c}(Y)$, i.e. the output can never change again. We say that $\mathcal{C}$ *stably computes* a function $f : \mathbb{N}^k \to \mathbb{N}$ if for any valid initial configuration $\mathbf{i} \in \mathbb{N}^\Sigma$ and any $\mathbf{c} \in \mathbb{N}^\Lambda, \mathbf{i} \Rightarrow \mathbf{c}$ implies $\mathbf{c} \Rightarrow \mathbf{o}$ such that $\mathbf{o}$ is stable and $f(\mathbf{i} \upharpoonright \Sigma) = \mathbf{o}(Y)$.

2.4 Time model

The following model of stochastic chemical kinetics is widely used in quantitative biology and other fields dealing with chemical reactions between species present in small counts [8]. It ascribes probabilities to execution sequences, and also defines the time of reactions, allowing us to study the computational complexity of the CRN computation in Sections 4 and 5. If the volume is defined to be the total number of molecules, then the time model is essentially equivalent to the notion of *parallel time* studied in population protocols [2]. In this paper, the rate constants of all reactions are 1, and we define the kinetic model with this assumption. A reaction is *unimolecular* if it has one reactant and *bimolecular* if it has two reactants. We use no higher-order reactions in this paper.

The kinetics of a CRN is described by a continuous-time Markov process as follows. Given a fixed volume $v > 0$, the *propensity* of a unimolecular reaction $\alpha : X \to \ldots$ in configuration $\mathbf{c}$ is $\rho(\mathbf{c}, \alpha) = \mathbf{c}(X)$. The propensity of a bimolecular reaction $\alpha : X + Y \to \ldots$, where $X \neq Y$, is $\rho(\mathbf{c}, \alpha) = \frac{\mathbf{c}(X)\mathbf{c}(Y)}{v}$. The propensity of a bimolecular reaction $\alpha : X + X \to \ldots$ is $\rho(\mathbf{c}, \alpha) = \frac{1}{2}\frac{\mathbf{c}(X)(\mathbf{c}(X)-1)}{v}$. The propensity function determines the evolution of the system as follows. The time until the next reaction occurs is an exponential random variable with rate $\rho(\mathbf{c}) = \sum_{\alpha \in R} \rho(\mathbf{c}, \alpha)$ (note that $\rho(\mathbf{c}) = 0$ if no reactions are applicable to $\mathbf{c}$). The probability that next reaction will be a particular α_{next} is $\frac{\rho(\mathbf{c}, \alpha_{\text{next}})}{\rho(\mathbf{c})}$.

The kinetic model is based on the physical assumption of well-mixedness that is valid in a dilute solution. Thus, we assume the *finite density constraint*, which stipulates that a volume required to execute a CRN must be proportional to the maximum molecular count obtained during execution [12]. In other words, the total concentration (molecular count per volume) is bounded. This realistically constrains the speed of the computation achievable by CRNs.

For a CRD or CRC stably computing a predicate/function, the *stabilization time* is the function $t : \mathbb{N} \to \mathbb{N}$ defined for all $n \in \mathbb{N}$ as $t(n) =$ the worst-case expected time to reach from any valid initial configuration of size n to a stable configuration.

2.5 Semilinear sets, predicates, functions

▶ **Definition 2.2.** *A set $L \subseteq \mathbb{N}^d$ is* linear *if there are vectors $\mathbf{b}, \mathbf{p}_1, \ldots, \mathbf{p}_k$ such that $L = \{\mathbf{b} + n_1\mathbf{p}_1 + \cdots + n_k\mathbf{p}_k \mid n_1, \ldots, n_k \in \mathbb{N}\}$. A set is* semilinear *if it is a finite union of linear sets. A predicate $\phi : \mathbb{N}^d \to \{0, 1\}$ is* semilinear *if the set $\phi^{-1}(1)$ is semilinear. A function $f : \mathbb{N}^d \to \mathbb{N}$ is* semilinear *if its* graph *$\{(\mathbf{x}, y) \in \mathbb{N}^{d+1} \mid f(\mathbf{x}) = y\}$ is semilinear.*

The following is a known characterization of the computational power of CRNs [3, 4].

▶ **Theorem 2.3** ([3, 4])**.** *A predicate/function is stably computable by a CRD/CRC if and only if it is semilinear.*

▶ **Definition 2.4.** *$T \subseteq \mathbb{N}^d$ is a* threshold *set is if there are constants $c, w_1, \ldots, w_d \in \mathbb{Z}$ such that $T = \{\mathbf{x} \in \mathbb{N}^d \mid w_1\mathbf{x}(1) + \cdots + w_d\mathbf{x}(d) \leq c\}$. $M \subseteq \mathbb{N}^d$ is a* mod *set if there are constants $c, m, w_1, \ldots, w_d \in \mathbb{N}$ such that $M = \{\mathbf{x} \in \mathbb{N}^d \mid w_1\mathbf{x}(1) + \cdots + w_d\mathbf{x}(d) \equiv c \mod m\}$.*

The following well-known characterization of semilinear sets is useful.

▶ **Theorem 2.5** ([9])**.** *A set is* semilinear *if and only if it is a Boolean combination (union, intersection, complement) of threshold and mod sets.*

3 Execution bounded chemical reaction networks

In this section, we define execution bounded CRNs and state an alternate characterization of the definition.

▶ **Definition 3.1.** *A CRN $\mathcal{C}$ is* execution bounded from configuration $\mathbf{x}$ *if all executions $\mathcal{E} = (\mathbf{x}, \ldots)$ starting at $\mathbf{x}$ are finite. A CRD or CRC $\mathcal{C}$ is* execution bounded *if it is execution bounded from every valid initial configuration. $\mathcal{C}$ is* entirely execution bounded *if it is execution bounded from every configuration.*

This is a distinct concept from the notion of "bounded" CRNs studied by Rackoff [11] (studied under the equivalent formalism of vector addition systems). That paper defines a CRN to be *bounded* from a configuration $\mathbf{x}$ if $|\mathsf{reach}(\mathbf{x})|$ is finite (and shows that the decision problem of determining whether this is true is EXPSPACE-complete.) We use the term *execution bounded* to avoid confusion with this concept.

We first observe an equivalent characterization of execution bounded that will be useful in the negative results of Section 6.

▶ **Definition 3.2.** *A execution* $\mathcal{E} = (\mathbf{x}_1, \mathbf{x}_2, \dots)$ *is* self-covering *if for some* $i < j$, $\mathbf{x}_i \leqq \mathbf{x}_j$. *It is* strictly self-covering *if* $\mathbf{x}_i \leq \mathbf{x}_j$. *We also refer to these as (strict) self-covering* paths.[1]

▶ **Lemma 3.3.** *A CRN is execution bounded from* $\mathbf{x}$ *if and only if there is no self-covering path from* $\mathbf{x}$.

4 Execution bounded CRDs stably decide all semilinear sets

In this section, we will show that execution bounded CRDs have the same computational power as unrestricted CRDs. The following is the main result of this section.

▶ **Theorem 4.1.** *Exactly the semilinear sets are stably decidable by execution bounded CRDs. Furthermore, each can be stably decided with expected stabilization time* $\Theta(n \log n)$.

Since semilinear sets are Boolean combinations of mod and threshold predicates, we prove this theorem by showing that execution bounded CRDs can decide mod and threshold sets individually as well as any Boolean combination in the following lemmas. To ensure execution boundedness in the last step, we require the following property.

▶ **Definition 4.2.** *Let* $\mathcal{D}$ *be a CRD with voting species* Υ. *We say* $\mathcal{D}$ *is* single-voting *if for any valid initial configuration* $\mathbf{i} \in \mathbb{N}^\Sigma$ *and any* $\mathbf{c} \in \mathbb{N}^\Lambda$ *s.t.* $\mathbf{i} \Rightarrow \mathbf{c}$, $\sum_{V \in \Upsilon} \mathbf{c}(V) = 1$, *i.e., exactly one voter is present in every reachable configuration.*

Lemmas 4.3 and 4.4 are proven in the full version of this paper.

▶ **Lemma 4.3.** *Every mod set* $M = \left\{ (x_1, \dots, x_d) \mid \sum_{i=1}^d w_i x_i \equiv c \bmod m \right\}$ *is stably decidable by an execution bounded, single-voting CRD with expected stabilization time* $\Theta(n \log n)$.

We design a CRD $\mathcal{D}$ with exactly one leader present at all times, cycling through m "states" while consuming the input and accepting on state c. Let $\Sigma = \{X_1, \dots, X_d\}$ be the set of input species and start with only one L_0 leader, i.e. set the initial context $\mathbf{s}(L_0) = 1$ and $\mathbf{s}(S) = 0$ for all other species. For each $i \in \{1, \dots, d\}, j \in \{0, \dots, m-1\}$ add the following reaction: $X_i + L_j \to L_{j+w_i \bmod m}$. Let only L_c vote *yes* and all other species *no*, i.e. $\Upsilon = \{L_c\}$. For any valid initial configuration, $\mathcal{D}$ reaches a stable configuration which votes *yes* if and only if the input is in the mod set, and *no* otherwise.

▶ **Lemma 4.4.** *Every threshold set* $T = \left\{ (x_1, \dots, x_d) \mid \sum_{i=1}^d w_i x_i \geq t \right\}$ *is stably decidable by an execution bounded, single-voting CRD with expected stabilization time* $\Theta(n \log n)$.

We design a CRD $\mathcal{D}$ which multiplies the input molecules according to their weight and consumes positive and negative units alternatingly using a single leader. Once no more reaction is applicable, the leader's state will indicate whether or not there are positive units left and the threshold is met. Let $\Sigma = \{X_1, \dots, X_d\}$ be the set of input species and $\Upsilon = \{L_Y\}$

[1] Rackoff [11] uses the term "self-covering" to mean what we call *strictly self-covering* here, and points out that Karp and Miller [10] showed that $|\mathsf{reach}(\mathbf{x})|$ is infinite if and only if there is a strictly self-covering path from $\mathbf{x}$. The distinction between these concepts is illustrated by the CRN $A \rightleftharpoons B$. From any configuration $\mathbf{x}$, $\mathsf{reach}(\mathbf{x})$ is finite ($|\mathsf{reach}(\mathbf{x})| = \mathbf{x}(A) + \mathbf{x}(B) + 1$), and there is no strict self-covering path. However, from (say) $\{A\}$, there is a (nonstrict) self-covering path $\{A\} \Rightarrow \{B\} \Rightarrow \{A\}$, and by repeating, this CRN has an infinite cycling execution within its finite configuration space $\mathsf{reach}(\{A\}) = \{\{A\}, \{B\}\}$.

the *yes* voter. We first add reactions to multiply the input species by their respective weights. For all $i \in \{1, \ldots, d\}$, add the reaction:

$$X_i \to \begin{cases} w_i P & \text{if } w_i > 0 \\ -w_i N & \text{if } w_i < 0 \\ \emptyset & \text{otherwise} \end{cases} \tag{5}$$

P and N represent "positive" and "negative" units respectively. Now add reactions to consume P and N alternatingly using a leader until we run out of one species:

$$L_Y + N \to L_N \tag{6}$$
$$L_N + P \to L_Y \tag{7}$$

Finally, initialize the CRD with one L_Y and the threshold number t copies of P (or $-tN$ if t is negative), i.e. $\mathbf{s}(L_Y) = 1$, $\mathbf{s}(P) = t$ if $t > 0$, or $\mathbf{s}(N) = -t$ if $t < 0$, and $\mathbf{s}(S) = 0$ for all other species. For any valid initial configuration, $\mathcal{D}$ reaches a stable configuration which votes *yes* if and only if the weighted sum of inputs is above the threshold, and *no* otherwise.

▶ **Lemma 4.5.** *If sets $X_1, X_2 \subseteq \mathbb{N}^d$ are stably decided by some execution bounded, single-voting CRD, then so are $X_1 \cup X_2$, $X_1 \cap X_2$, and $\overline{X_1}$ with expected stabilization time $O(n \log n)$.*

Proof. To stably decide $\overline{X_1}$, swap the yes and no voters.

For $\cup$ and $\cap$, consider a construction where we decide both sets separately and record both of their votes in a new voter species. For this, we allow the set of all voters to be a strict subset of all species. We first add reactions to duplicate our input with reactions of the form

$$X_i \to X_{i,1} + X_{i,2} \tag{8}$$

by two separate CRDs. Subsequently, we add reactions to record the separate votes in one of four new voter species: $V_{NN}, V_{NY}, V_{YN}, V_{YY}$. The first and second CRN determine the first and second subscript respectively. For $b \in \{Y, N\}$ and if S_b, T_b are voters of $\mathcal{C}_1$ and $\mathcal{C}_2$ respectively, add the reactions:

$$S_b + V_{\bar{b}?} \to S_b + V_{b?} \tag{9}$$
$$T_b + V_{?\bar{b}} \to T_b + V_{?b} \tag{10}$$

Above, the ? subscript is shorthand for "any bit"; e.g. if N_1 is the *no* voter of the first CRD, we would add two reactions $N_1 + L_{YN} \to N_1 + L_{NN}$ and $N_1 + L_{YY} \to N_1 + L_{NY}$. We let the *yes* voters be: $\Upsilon = \{V_{NY}, V_{YN}, V_{YY}\}$ to stably decide $X_1 \cup X_2$ or $\Upsilon = \{V_{YY}\}$ to stably decide $X_1 \cap X_2$.

Reaction (8) will complete in $O(\log n)$ time and is clearly execution bounded since the input X_i is finite and not produced in any reaction. Consequently, two separate CRNs run in $\Theta(n \log n)$ time as shown in Lemma 4.3 and Lemma 4.4. After stabilization of the parallel CRNs, we expect reaction (9) and (10) to happen exactly once. Each molecule involved is a leader and has count 1 in volume n. This leads to a rate of $\lambda = \frac{1 \cdot 1}{n}$, so the expected time for one reaction to happen is $O(n)$. It is important to note that reactions (9) and (10) do not result in unbounded executions due to the unanimous vote in parallel CRDs. In both mod sets and threshold sets, the leader changes its vote a maximum of $|\mathbf{i}|$ times, with only ever one leader present at any time. Again, we start with only one V_{bb} voter present initially and no reaction changes the count of voters, making our construction single-voting. ◀

Since semilinear predicates are exactly Boolean combinations of threshold and mod predicates, Lemmas 4.3–4.5 imply Theorem 4.1.

We can also prove the same result for all-voting CRDs. Note, however, that such CRDs cannot be "composed" using the constructions of Lemma 4.5 and Theorem 5.4, which crucially relied on the assumption that the CRDs being used as "subroutines" are single-voting.

▶ **Theorem 4.6.** *Every semilinear set is stably decidable by an execution bounded, all-voting CRD, with expected stabilization time* $O(n \log n)$.

5 Execution bounded CRCs stably compute all semilinear functions

In this section we shift focus from computing Boolean-valued predicates $\phi : \mathbb{N}^d \to \{0,1\}$ to integer-valued functions $f : \mathbb{N}^d \to \mathbb{N}$, showing that execution bounded CRCs can stably compute the same class of functions (semilinear) as unrestricted CRCs.

Similar to [5, 7], we compute semilinear functions by decomposing them into "affine pieces", which we will show can be computed by execution bounded CRNs and combined by using semilinear predicates to decide which linear function to apply for a given input.[2]

We say a partial function $f : \mathbb{N}^k \dashrightarrow \mathbb{N}$ is *affine* if there exist vectors $\mathbf{a} \in \mathbb{Q}^k$, $\mathbf{c} \in \mathbb{N}^k$ with $\mathbf{x} - \mathbf{c} \geq \mathbf{0}$ and nonnegative integer $b \in \mathbb{N}$ such that $f(\mathbf{x}) = \mathbf{a}^\top(\mathbf{x} - \mathbf{c}) + b$. For a partial function f we write $\operatorname{dom} f$ for the *domain* of f, the set of inputs for which f is defined. This definition of affine function may appear contrived, but the main utility of the definition is that it satisfies Lemma 5.3. For convenience, we can ensure to only work with integer valued molecule counts by multiplying by $\frac{1}{d}$ after the dot product, where d may be taken to be the least common multiple of the denominators of the rational coefficients in the original definition such that $n_i = d \cdot \mathbf{a}(i)$: $f(\mathbf{x}) = b + \sum_{i=1}^k \mathbf{a}(i)(\mathbf{x}(i) - \mathbf{c}(i)) \iff f(\mathbf{x}) = b + \frac{1}{d}\sum_{i=1}^k n_i(\mathbf{x}(i) - \mathbf{c}(i))$.

We say that a partial function $\hat{f} : \mathbb{N}^k \to \mathbb{N}^2$ is a *diff-representation* of f if $\operatorname{dom} f = \operatorname{dom} \hat{f}$ and, for all $\mathbf{x} \in \operatorname{dom} f$, if $(y_P, y_C) = \hat{f}(\mathbf{x})$, then $f(\mathbf{x}) = y_P - y_C$, and $y_P = O(f(\mathbf{x}))$. In other words, $\hat{f}$ represents f as the difference of its two outputs y_P and y_C, with the larger output y_P possibly being larger than the original function's output, but at most a multiplicative constant larger [7].

▶ **Lemma 5.1.** *Let* $f : \mathbb{N}^k \to \mathbb{N}$ *be an affine partial function. Then there is a diff-representation* $\hat{f} : \mathbb{N}^k \longrightarrow \mathbb{N}^2$ *of* f *and an execution bounded CRC that monotonically stably computes* $\hat{f}$ *in expected stabilization time* $O(n)$.

Proof. Define a CRC C with input species $\Sigma = \{X_1, \ldots, X_k\}$ and output species $\Gamma = \{Y^P, Y^C\}$. We need to ensure that after stabilizing, $y = \#Y^P - \#Y^C$

To account for the b offset, start with b copies of Y^P.

For the c_i offset, we must reduce the number of X_i by c_i. Since the result will be used in the next reaction, we want to produce a new species X_i' and require X_i' to not be consumed during the computation. We achieve this by adding reactions that let X_i consume itself c_i times (keeping track with a subscript) and converting X_i to X_i' once c_i has been reached. For the sake of notation below, assume input species X_i is actually named $X_{i,1}$. For each $i \in \{1, \ldots, k\}$ and $m, p \in \{1, \ldots, c_i\}$, if $m + p \leq c_i$, add the reaction

$$X_{i,m} + X_{i,p} \to X_{i,m+p} \tag{11}$$

[2] While this proof generalizes to multivariate output functions as in [5, 7], to simplify notation we focus on single output functions. Multi-valued functions $f : \mathbb{N}^d \to \mathbb{N}^l$ can be equivalently thought of as l separate single output functions $f_i : \mathbb{N}^d \to \mathbb{N}$, which can be computed in parallel by independent CRCs.

If $m + p > c_i$, add the reaction

$$X_{i,m} + X_{i,p} \to X_{i,c_i} + (m + p - c_i)\, X_i' \tag{12}$$

Runtime: In volume n, the rate of reactions (11) and (12) would be $\lambda \approx \frac{(x_i)^2}{n}$ (x_i molecules have the chance to react with any of the $x_i - 1$ others), so the expected time for the next reaction is $\frac{n}{(x_i)^2}$. The expected time for the whole process is $\sum_{i=1}^{x_i} \frac{n}{i^2} = n \sum_{i=1}^{x_i} \frac{1}{i^2} = O(n)$. Further, the reactions are execution bounded since both strictly decrease the number of their reactants and exactly $x_i - 1$ reactions will happen.

To account for the n_i/d coefficient, we multiply by n_i, then divide by d using similar reactions as for the subtraction. To multiply by n_i, add the following reaction for each $i \in \{1, \ldots, k\}$:

$$X_i' \to \begin{cases} n_i D_1^P, & \text{if } n_i > 0 \\ (-n_i)\, D_1^C, & \text{if } n_i < 0 \end{cases} \tag{13}$$

For each $m, p \in \{1, \ldots, d-1\}$, if $m + p \leq d - 1$, add the reactions

$$D_m^P + D_p^P \to D_{m+p}^P \tag{14}$$
$$D_m^C + D_p^C \to D_{m+p}^C \tag{15}$$

If $m + p \geq d$, add the reactions

$$D_m^P + D_p^P \to D_{m+p-d}^P + Y^P \tag{16}$$
$$D_m^C + D_p^C \to D_{m+p-d}^C + Y^C \tag{17}$$

Reactions (13) complete in expected time $O(\log n)$, while (16) and (17) complete in $O(n)$ by a similar analysis as for the first two reactions. As for execution boundedness, (13) is only applicable once for every X_i'; all other reactions start with a number of reactants which are a constant factor of X_i' and decrease the count of their reactants by one in each reaction. ◀

We require the following result due to Chen, Doty, Soloveichik [5], guaranteeing that any semilinear function can be built from affine partial functions.

▶ **Lemma 5.2** ([5]). *Let $f : \mathbb{N}^d \to \mathbb{N}$ be a semilinear function. Then there is a finite set $\{f_1 : \mathbb{N}^d \to \mathbb{N}, \ldots, f_m : \mathbb{N}^d \to \mathbb{N}\}$ of affine partial functions, where each $\operatorname{dom} f_i$ is a linear set, such that, for each $\mathbf{x} \in \mathbb{N}^d$, if $f_i(\mathbf{x})$ is defined, then $f(\mathbf{x}) = f_i(\mathbf{x})$, and $\bigcup_{i=1}^m \operatorname{dom} f_i = \mathbb{N}^d$.*

We strengthen Lemma 5.2 to show we may assume each $\operatorname{dom} f_i$ is disjoint from the others. This is needed not only to prove Theorem 5.4, but to correct the proof of Lemma 4.4 in [5], which implicitly assumed the domains are disjoint.

▶ **Lemma 5.3.** *Let $f : \mathbb{N}^d \to \mathbb{N}$ be a semilinear function. Then there is a finite set $\{f_1 : \mathbb{N}^d \to \mathbb{N}, \ldots, f_m : \mathbb{N}^d \to \mathbb{N}\}$ of affine partial functions, where each $\operatorname{dom} f_i$ is a linear set, and $\operatorname{dom} f_i \cap \operatorname{dom} f_j = \emptyset$ for all $i \neq j$, such that, for each $\mathbf{x} \in \mathbb{N}^d$, if $f_i(\mathbf{x})$ is defined, then $f(\mathbf{x}) = f_i(\mathbf{x})$, and $\bigcup_{i=1}^m \operatorname{dom} f_i = \mathbb{N}^d$.*

The next theorem shows that semilinear functions can be computed by execution bounded CRCs in expected time $O(n \log n)$.

▶ **Theorem 5.4.** *Let $f : \mathbb{N}^d \to \mathbb{N}$ be a semilinear function. Then there is an execution bounded CRC that stably computes f with expected stabilization time $O(n \log n)$.*

Proof. We employ the same construction of [5] with minor alterations. A CRC with input species $\Sigma = \{X_1, \ldots, X_d\}$ and output species $\Gamma = \{Y\}$. By Lemma 5.3, we decompose our semilinear function into partial affine functions (with linear, disjoint domains), which can be computed in parallel by Lemma 5.1. Further, we decide which function to use by computing the predicate $\phi_i =$ "x $\in$ dom f_i" (Theorem 4.1). We interpret each $\widehat{Y}_i^P$ and $\widehat{Y}_i^C$ as an "inactive" version of "active" output species Y_i^P and Y_i^C. Let L_i^Y, L_i^N be the *yes* and *no* voters respectively voting whether $\mathbf{x}$ lies in the domain of i-th partial function. Now, we convert the function result of the applicable partial affine function to the global output by adding the following reactions for each $i \in \{1, \ldots, m\}$.

$$L_i^Y + \widehat{Y}_i^P \to L_i^Y + Y_i^P + Y \tag{18}$$
$$L_i^N + Y_i^P \to L_i^N + M_i \tag{19}$$
$$M_i + Y \to \widehat{Y}_i^P \tag{20}$$

Reaction (18) produces an output copy of species Y and (19) and (20) reverse the first reaction using only bimolecular reactions. Both are catalyzed by the vote of the i-th predicate result. Also add reactions

$$L_i^Y + \widehat{Y}_i^C \to L_i^Y + Y_i^C \tag{21}$$
$$L_i^N + Y_i^C \to L_i^N + \widehat{Y}_i^C \tag{22}$$

and

$$Y_i^P + Y_i^C \to K \tag{23}$$
$$K + Y \to \varnothing \tag{24}$$

Reactions (21) and (22) activate and deactivate the "negative" output values and reactions (23) and (24) allow two active partial outputs to cancel out and consume the excess Y in the process. When the input is in the domain of function i, exactly one copy of L_i^Y will be present, otherwise one copy of L_i^N. Since we know that the predicate computation is execution bounded and produces at most one voter, the catalytic reaction will also happen at most as often as the leader changes its vote. Therefore, it is also execution bounded.

The underlying CRNs computing the predicates and functions have expected stabilization time $O(n \log n)$. Once they have stabilized, the slowest reactions described above are those where a leader (L_i^Y or L_i^N) must convert all outputs, which also takes expected time $O(n \log n)$ by a coupon collector argument. ◀

6 Limitations of execution bounded CRNs

The main positive results of the paper (Theorems 4.1 and 5.4) rely on the assumption that valid initial configurations have a single leader (in particular, they are execution bounded only from configurations with a single leader, but not from arbitrary configurations). Theorem 4.6 shows that we may assume the CRD deciding a semilinear set is all-voting. However, for the "constructive" results Lemma 4.5 and Theorem 5.4, which compose the output of a CRD $\mathcal{D}$ with downstream computation, using $\mathcal{D}$ as a "subroutine" to stably compute a more complex set/function, the constructions crucially use the assumption that $\mathcal{D}$ is single-voting (i.e., only the leader of $\mathcal{D}$ votes) to argue the resulting composed CRN is execution bounded. In this section we show these assumptions are necessary, proving that execution bounded CRNs without those constraints are severely limited in their computational abilities.

We use a result of Czerner, Guttenberg, Helfrich, and Esparza [6], showing that entirely execution bounded CRNs (from every configuration) can be characterized by a simpler property of having a "linear potential function" that essentially measures how close the CRN is to reaching a terminal configuration. We use this characterization to prove that entirely execution bounded CRNs can stably decide only limited semilinear predicates (eventually constant, Definition 6.6), assuming all species vote, and that molecular counts cannot decrease to $O(1)$ in stable configurations (see Definition 6.4).

6.1 Linear potential functions

We define a *linear potential function* of a CRN to be a nonnegative linear function of configurations that each reaction strictly decreases.

▶ **Definition 6.1.** *A* linear potential function $\Phi : \mathbb{R}^{\Lambda}_{\geq 0} \to \mathbb{R}_{\geq 0}$ *for a CRN is a nonnegative linear function, such that for each reaction* $(\mathbf{r}, \mathbf{p})$, $\Phi(\mathbf{p}) - \Phi(\mathbf{r}) < 0$.

Note that for a configuration $\mathbf{x}$, since $\Phi(\mathbf{x}) = \sum_{S \in \Lambda} v_S \mathbf{x}(S) \geq 0$, it must be nondecreasing in each species, i.e., all coefficients v_S must be nonnegative (though some are permitted to be 0). Intuitively, we can think of Φ as assigning a nonnegative "mass" to each species (the mass of S is v_S), such that each reaction removes a positive amount of mass from the system. Note also that since Φ is linear, the above is equivalent to requiring that $\Phi(\mathbf{p} - \mathbf{r}) < 0$, if we extend Φ to a linear function $\Phi : \mathbb{R}^{\Lambda} \to \mathbb{R}$ on vectors with negative elements.

A CRN may or may not have a linear potential function. Although it is not straightforward to "syntactically check" a CRN to see if has a linear potential function, it is efficiently decidable: a CRN has a linear potential function if and only if the following system of linear inequalities has a solution (which can be solved in polynomial time using linear programming techniques; the variables to solve for are the v_S for each $S \in \Lambda$), where the i'th reaction has reactants $\mathbf{r}_i$ and products $\mathbf{p}_i$, and species $S \in \Lambda$ has mass $v_S \geq 0$: $(\forall i) \sum_{S \in \Lambda} [\mathbf{p}_i(S) - \mathbf{r}_i(S)] v_S < 0$. For example, for the reactions $A + A \to B + C$ and $B + B \to A$, for each reaction to strictly decrease the potential function $\Phi(\mathbf{x}) = v_A \mathbf{x}(A) + v_B \mathbf{x}(B) + v_C \mathbf{x}(C)$, Φ must satisfy $2v_A > v_B + v_C$ and $2v_B > v_A$. In this case, $v_A = 1, v_B = 1, v_C = 0$ works.

▶ Remark 6.2. A system of linear inequalities with rational coefficients has a real solution if and only if it has a rational solution. For any homogeneous system (where all inequalities are comparing to 0), any positive scalar multiple of a solution is also a solution. By clearing denominators, a system has a rational solution if and only if it has an integer solution. Thus, one can equivalently define a linear potential function to be a function $\Phi(\mathbf{x}) = \sum_{S \in \Lambda} v_S \mathbf{x}(S)$ such that each $v_S \in \mathbb{N}$, i.e., we may assume $\Phi : \mathbb{N}^{\Lambda} \to \mathbb{N}$. In particular, since Φ is decreased by each reaction, it is decreased by at least 1.

The following theorem due to Czerner, Guttenberg, Helfrich, and Esparza, is crucial to proving limitations on execution bounded CRNs such as Theorem 6.5 and Theorem 6.7.

▶ **Theorem 6.3** ([6]). *A CRN has a linear potential function if and only if it is entirely execution bounded.*

6.2 Impossibility of stably deciding majority and parity

In this section, we prove Theorem 6.5, which is a special case of our main negative result, Theorem 6.7. We give a self-contained proof of Theorem 6.5 because it is simpler and serves as an intuitive warmup to some of the key ideas used in proving Theorem 6.7, without the complexities of dealing with arbitrary semilinear sets.

Theorem 6.5 shows a limitation on the computational power of entirely execution bounded, all-voting CRNs, but it requires an additional constraint on the CRN for the result to hold (and we later give counterexamples showing that this extra hypothesis is provably necessary), described in the following definition.

▶ **Definition 6.4.** *Let $\mathcal{D}$ be a CRD. The* output size *of $\mathcal{D}$ is the function $s : \mathbb{N} \to \mathbb{N}$ defined $s(n) = \min_{\mathbf{x},\mathbf{y}}\{\|\mathbf{y}\| \mid \mathbf{x} \Rightarrow \mathbf{y}, \|\mathbf{x}\| = n, \mathbf{x}$ is a valid initial configuration, $\mathbf{y}$ is stable$\}$, the size of the smallest stable configuration reachable from any valid initial configuration of size n. A CRD is* non-collapsing *if $\lim_{n\to\infty} s(n) = \infty$.*

Put another way, $\mathcal{D}$ is *collapsing* if there is a constant c such that, from infinitely many initial configurations $\mathbf{x}$, $\mathcal{D}$ can reach a stable configuration of size at most c. All population protocols are non-collapsing, since every reaction preserves the configuration size.

▶ **Theorem 6.5.** *No non-collapsing, all-voting, entirely execution bounded CRD can stably decide the majority predicate $[X_1 \geq X_2?]$ or the parity predicate $[X \equiv 1 \mod 2?]$.*

Proof. Let $\mathcal{D} = (\Lambda, R, \Sigma, \Upsilon_\mathrm{Y}, \Upsilon_\mathrm{N}, \mathbf{s})$ be a CRD obeying the stated conditions, and suppose for the sake of contradiction that $\mathcal{D}$ stably decides the majority predicate (so $\Sigma = \{X_1, X_2\}$).

We consider the sequence of stable configurations $\mathbf{a}_1, \mathbf{b}_1, \mathbf{a}_2, \mathbf{b}_2, \ldots$ defined as follows. Let $\mathbf{a}_1$ be a stable configuration reachable from initial configuration $\mathbf{s} + \{X_1, X_2\}$; since the correct answer is yes, all species present in $\mathbf{a}_1$ vote yes. Now add a single copy of X_2. By additivity, the configuration $\mathbf{a}_1 + \{X_2\}$ is reachable from $\mathbf{s} + \{X_1, 2X_2\}$, for which the correct answer in this case is no. Thus, since $\mathcal{D}$ stably decides majority, from $\mathbf{a}_1 + \{X_2\}$, a stable "no" configuration is reachable; call this $\mathbf{b}_1$. Now add a single X_1. Since the correct answer is yes, from $\mathbf{b}_1 + \{X_1\}$ a stable "yes" configuration is reachable, call it $\mathbf{a}_2$.

Continuing in this way, we have a sequence of stable configurations $\mathbf{a}_1, \mathbf{b}_1, \mathbf{a}_2, \mathbf{b}_2, \ldots$ where all species in $\mathbf{a}_i$ vote yes and all species in $\mathbf{b}_i$ vote no. Since $\mathcal{D}$ is non-collapsing, the size of the configurations $\mathbf{a}_i$ and $\mathbf{b}_i$ increases without bound as $i \to \infty$. (Possibly $\|\mathbf{a}_{i+1}\| < \|\mathbf{a}_i\|$, i.e., the size is not necessarily monotonically increasing, but for all sufficiently large $j > i$, we have $\|\mathbf{a}_j\| > \|\mathbf{a}_i\|$.) Since all species vote, for some constant $\delta > 0$, to get from $\mathbf{a}_i + \{X_2\}$ to $\mathbf{b}_i$, at least $\delta\|\mathbf{a}_i\|$ reactions must occur. This is because all species in $\mathbf{a}_i$ must be removed since they vote yes, and each reaction removes at most $O(1)$ molecules. (Concretely, let $\delta = 1/\max_{(\mathbf{r},\mathbf{p})\in R} \|\mathbf{r}\| - \|\mathbf{p}\|$, i.e., 1 over the most net molecules consumed in any reaction.) Similarly, to get from $\mathbf{b}_i + \{X_1\}$ to $\mathbf{a}_{i+1}$, at least $\delta\|\mathbf{b}_i\|$ reactions must occur.

Since $\mathcal{D}$ is entirely execution bounded, by Theorem 6.3, $\mathcal{D}$ has a linear potential function $\Phi(\mathbf{x}) = \mathbf{v}\cdot\mathbf{x}$, where $\mathbf{v} \geq \mathbf{0}$. Adding a single X_2 to $\mathbf{a}_i$ increases Φ by the constant $\mathbf{v}(X_2)$. Since $\|\mathbf{a}_i\|$ grows without bound, the number of reactions to get from $\mathbf{a}_i + \{X_2\}$ to $\mathbf{b}_i$ increases without bound as $i \to \infty$, and since each reaction strictly decreases Φ by at least 1, the total change in Φ that results from adding X_2 and then going from $\mathbf{a}_i + \{X_2\}$ to $\mathbf{b}_i$ is unbounded in i, so unboundedly negative for sufficiently large i (negative once i is large enough that $\delta\|\mathbf{a}_i\| \geq \mathbf{v}(X_2) + 2$). Similarly, adding a single X_1 to $\mathbf{b}_i$ and going from $\mathbf{b}_i + \{X_1\}$ to $\mathbf{a}_{i+1}$, the resulting total change in Φ is unbounded and (for large enough i) negative.

Φ starts this process at the constant $\Phi(\mathbf{s} + \{X_1, X_2\})$. Before $\|\mathbf{a}_i\|$ and $\|\mathbf{b}_i\|$ are large enough that $\delta\|\mathbf{a}_i\| \geq \mathbf{v}(X_2) + 2$ and $\delta\|\mathbf{b}_i\| \geq \mathbf{v}(X_1) + 2$ (i.e., large enough that the net change in Φ is negative resulting from adding a single input and going to the next stable configuration), Φ could increase, if $\Phi(\{X_1\})$ (resp. $\Phi(\{X_2\})$) is larger than the net decrease in Φ due to following reactions to get from $\mathbf{a}_i + \{X_2\}$ to $\mathbf{b}_i$ (resp. from $\mathbf{b}_i + \{X_1\}$ to $\mathbf{a}_i$).

However, since $\mathcal{D}$ is non-collapsing, this can only happen for a constant number of i (so Φ never reaches more than a constant above its initial value $\Phi(\mathbf{s} + \{X_1, X_2\})$), after which Φ strictly decreases after each round of this process. At some point in this process, $\mathcal{D}$ will not be able to reach all the way to the next $\mathbf{a}_i$ or $\mathbf{b}_i$ without Φ becoming negative, a contradiction.

The argument for parity is similar, but instead of alternating adding X_1 then X_2, in each round we always add one more X to flip the correct answer. ◀

Theorem 6.5 is false without the non-collapsing hypothesis. The following collapsing, leaderless (but all-voting and entirely execution bounded) CRD stably decides majority: Species X_1, x_1 vote yes, while X_2, x_2 vote no:

$$X_1 + X_2 \to x_1 + x_2$$
$$X_1 + x_2 \to X_1$$
$$X_2 + x_1 \to X_2$$
$$x_1 + x_2 \to x_1$$

It has bounded executions from every configuration: $\min(\#X_1, \#X_2)$ of the first reaction can occur, and the other reactions decrease molecular count, so are limited by the total configuration size. However, it is collapsing since, for any n, there exists an input of size n that reaches a stable configuration of size 1. Theorem 6.5 is similarly false without the all-voting hypothesis; for each of the reactions with one product above, add another non-voting product W. This converts the CRD to be non-collapsing but not all-voting. Of course, the execution bounded hypothesis is also necessary: the original population protocols paper [1] showed that all-voting, non-collapsing, leaderless population protocols can stably decide all semilinear predicates.

The following collapsing, all-voting, leaderless (but entirely execution bounded) CRD stably decides parity. Let the input species be named X_1. Species X_1 votes yes, X_0 votes no:

$$X_1 + X_1 \to X_0$$
$$X_1 + X_0 \to X_1$$
$$X_0 + X_0 \to X_0$$

6.3 Impossibility of stably deciding not eventually constant predicates

We now present our main negative result, Theorem 6.7, which generalizes Theorem 6.5 to show that such CRNs can stably decide only very limited (eventually constant) predicates.

▶ **Definition 6.6.** *Let $\phi : \mathbb{N}^d \to \{0,1\}$ be a predicate. We say ϕ is* eventually constant *if there is $n_0 \in \mathbb{N}$ such that ϕ is constant on $\mathbb{N}^d_{\geq n_0} = \{\mathbf{x} \in \mathbb{N}^d \mid (\forall i \in \{1, \ldots, d\})\ \mathbf{x}(i) \geq n_0\}$, i.e., either $\phi^{-1}(0) \cap \mathbb{N}^d_{\geq n_0} = \emptyset$ or $\phi^{-1}(1) \cap \mathbb{N}^d_{\geq n_0} = \emptyset$.*

In other words, although ϕ may have an infinite number of each output, "sufficiently far from the boundary of the positive orthant" (where all coordinates exceed n_0), only one output appears. A complete proof appears in the full version of this paper.

▶ **Theorem 6.7.** *If a non-collapsing, all-voting, entirely execution bounded CRD stably decides a predicate ϕ, then ϕ is eventually constant.*

Proof sketch. This proof is similar to that of Theorem 6.5. In that proof, we repeatedly add a "constant amount of additional input $\{X_2\}$ or $\{X_1\}$, which flips the output". For more general semilinear, but not eventually constant, predicates, we dig into the structure of the

semilinear set to find a sequence of constant-size vectors representing additional inputs that flip the correct output. Any predicate that is not eventually constant has infinitely many yes inputs and infinitely many no inputs, but in general they could be increasingly far apart: e.g., $\phi(\mathbf{x}) = 1$ if and only if $2^n \leq \|\mathbf{x}\| < 2^{n+1}$ for even n. For the potential function argument to work, each subsequent input needs to be at most a constant larger than the previous.

But if ϕ is *semilinear* (and not eventually constant) then we can show that there is a sequence of increasing inputs $\mathbf{x}_0 \leq \mathbf{x}_1 \leq \mathbf{x}_2 \leq \ldots$, each a *constant* distance from the next ($\|\mathbf{x}_{j+1} - \mathbf{x}_j\| = O(1)$), flipping the output ($\phi(\mathbf{x}_j) \neq \phi(\mathbf{x}_{j+1})$). Roughly, this is true for one of two reasons. Using Theorem 2.5, ϕ is a Boolean combination of threshold and mod sets. Either the mod sets are not combined to be trivially $\emptyset$ or $\mathbb{N}^d$, in which case we can find some vector $\mathbf{v}$ that, followed infinitely far from some starting point $\mathbf{x}_0$ (so $\mathbf{x}_i = \mathbf{x}_0 + i\mathbf{v}$) periodically hits both yes inputs ($\phi(\mathbf{x}_j) = 1$) and no inputs ($\phi(\mathbf{x}_j) = 0$). Otherwise, the mod sets can be removed and simplify the Boolean combination to only threshold sets, in which case the infinite sequence $\mathbf{x}_0, \mathbf{x}_1, \ldots$ can be obtained by moving along a threshold hyperplane that separates yes from no inputs. ◀

The statement of Theorem 6.5 does not mention the concept of a leader, but it would typically apply to leaderless CRDs. A CRD may be execution bounded from configurations with a single leader, but not execution bounded when multiple leaders are present (preventing the use of Theorem 6.3, which requires the CRD to be execution bounded from *all* configurations). For example, in Lemma 4.5, reaction (9) occurs finitely many times if the leader/voter S_Y or S_N has count 1. However, if S_Y and S_N can be present simultaneously (e.g., if we start with two leaders), then the reactions $S_Y + V_{NN} \to S_Y + V_{YN}$ and $S_N + V_{YN} \to S_N + V_{NN}$ can flip between V_{NN} and V_{YN} infinitely often in an unbounded execution.

If the CRN is leaderless, however, we have the following, which says that if it is execution bounded from *valid initial* configurations, then it is execution bounded from *all* configurations.

▶ **Lemma 6.8.** *If a leaderless CRD or CRC is execution bounded, then it is entirely execution bounded.*

Proof sketch. Since $\mathcal{C}$ is leaderless, the sum of two valid initial configurations is also valid. Thus if we can produce some species from a valid initial configuration, we can produce arbitrarily large counts of all species by adding up sufficiently many initial configurations. This means that for any configuration $\mathbf{x}$, from any sufficiently large valid initial configuration $\mathbf{i}$, some $\mathbf{y} \geqq \mathbf{x}$ is reachable from $\mathbf{i}$. But if $\mathcal{C}$ is execution bounded from $\mathbf{i}$, since $\mathbf{i} \Rightarrow \mathbf{y}$, it must also be execution bounded from $\mathbf{y}$, thus also from $\mathbf{x}$ since by additivity any reactions applicable to $\mathbf{x}$ are also applicable to $\mathbf{y}$. ◀

Lemma 6.8 lets us replace "entirely execution bounded" in Theorem 6.7 with "leaderless and execution bounded":

▶ **Corollary 6.9.** *If a non-collapsing, all-voting, leaderless, execution bounded CRD stably decides a predicate ϕ, then ϕ is eventually constant.*

In particular, since the original model of population protocols [1] defined them as leaderless and all-voting – and since population protocols are non-collapsing – we have the following.

▶ **Corollary 6.10.** *If an execution bounded population protocol stably decides a predicate ϕ, then ϕ is eventually constant.*

7 Conclusion

A key question remains open: *Can execution bounded CRNs compute semilinear functions and predicates within polylogarithmic time?* Angluin, Aspnes and Eisenstat [2] introduced a fast population protocol that simulates a register machine with high probability, and can be made probability 1 with semilinear predicates. However, this construction seems inherently unbounded in executions.

References

1 Dana Angluin, James Aspnes, Zoë Diamadi, Michael J Fischer, and René Peralta. Computation in networks of passively mobile finite-state sensors. In *PODC 2004: Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing*, pages 290–299, 2004. doi:10.1145/1011767.1011810.

2 Dana Angluin, James Aspnes, and David Eisenstat. Fast computation by population protocols with a leader. *Distributed Computing*, 21(3):183–199, September 2008. doi:10.1007/S00446-008-0067-Z.

3 Dana Angluin, James Aspnes, David Eisenstat, and Eric Ruppert. The computational power of population protocols. *Distributed Computing*, 20(4):279–304, 2007. doi:10.1007/S00446-007-0040-2.

4 Ho-Lin Chen, David Doty, Wyatt Reeves, and David Soloveichik. Rate-independent computation in continuous chemical reaction networks. *Journal of the ACM*, 70(3), May 2023. doi:10.1145/3590776.

5 Ho-Lin Chen, David Doty, and David Soloveichik. Deterministic function computation with chemical reaction networks. *Natural Computing*, 13(4):517–534, 2014. Preliminary version appeared in DNA 2012. doi:10.1007/s11047-013-9393-6.

6 Philipp Czerner, Roland Guttenberg, Martin Helfrich, and Javier Esparza. Fast and succinct population protocols for Presburger arithmetic. *Journal of Computer and System Sciences*, 140:103481, 2024. doi:10.1016/J.JCSS.2023.103481.

7 David Doty and Monir Hajiaghayi. Leaderless deterministic chemical reaction networks. *Natural Computing*, 14(2):213–223, 2015. Preliminary version appeared in DNA 2013. doi:10.1007/S11047-014-9435-8.

8 Daniel T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *Journal of Physical Chemistry*, 81(25):2340–2361, 1977.

9 S. Ginsburg and E. H. Spanier. Semigroups, Presburger formulas, and languages. *Pacific Journal of Mathematics*, 16(2):285–296, 1966.

10 Richard M Karp and Raymond E Miller. Parallel program schemata. *Journal of Computer and system Sciences*, 3(2):147–195, 1969. doi:10.1016/S0022-0000(69)80011-5.

11 Charles Rackoff. The covering and boundedness problems for vector addition systems. *Theoretical Computer Science*, 6(2):223–231, 1978. doi:10.1016/0304-3975(78)90036-1.

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