On the Runtime of Chemical Reaction Networks Beyond Idealized Conditions

Anne Condon
University of British Columbia, Vancouver, Canada

Yuval Emek
Technion – Israel Institute of Technology, Haifa, Israel

Noga Harlev
Technion – Israel Institute of Technology, Haifa, Israel

Abstract

This paper studies the (discrete) *chemical reaction network (CRN)* computational model that emerged in the last two decades as an abstraction for molecular programming. The correctness of CRN protocols is typically established under one of two possible schedulers that determine how the execution advances: (1) a *stochastic scheduler* that obeys the (continuous time) Markov process dictated by the standard model of stochastic chemical kinetics; or (2) an *adversarial scheduler* whose only commitment is to maintain a certain fairness condition. The latter scheduler is justified by the fact that the former one crucially assumes "idealized conditions" that more often than not, do not hold in real wet-lab experiments. However, when it comes to analyzing the *runtime* of CRN protocols, the existing literature focuses strictly on the stochastic scheduler, thus raising the research question that drives this work: Is there a meaningful way to quantify the runtime of CRNs without the idealized conditions assumption?

The main conceptual contribution of the current paper is to answer this question in the affirmative, formulating a new runtime measure for CRN protocols that does not rely on idealized conditions. This runtime measure is based on an adapted (weaker) fairness condition as well as a novel scheme that enables partitioning the execution into short *rounds* and charging the runtime for each round individually (inspired by definitions for the runtime of asynchronous distributed algorithms). Following that, we turn to investigate various fundamental computational tasks and establish (often tight) bounds on the runtime of the corresponding CRN protocols operating under the adversarial scheduler. This includes an almost complete chart of the runtime complexity landscape of predicate decidability tasks.

2012 ACM Subject Classification Theory of computation → Distributed algorithms

Keywords and phrases chemical reaction networks, adversarial runtime, weak fairness, predicate decidability

Digital Object Identifier 10.4230/LIPIcs.DNA.29.3

Related Version *Full Version*: `http://arxiv.org/abs/2307.00647` [13]

Funding *Anne Condon*: This work was supported by an NSERC Discovery grant.
Yuval Emek: This work was supported by VATAT Fund to the Technion Artificial Intelligence Hub (Tech.AI).

1 Introduction

Chemical reaction networks (CRNs) are used to describe the evolution of interacting molecules in a solution [20] and more specifically, the behavior of regulatory networks in the cell [7]. In the last two decades, CRNs have also emerged as a computational model for molecular programming [24, 14]. A protocol in this model is specified by a set of species and a set of reactions, which consume molecules of some species and produce molecules of others.

In a (discrete) CRN computation, inputs are represented as (integral) molecular counts of designated species in the initial system configuration; a sequence of reactions ensues, repeatedly transforming the configuration, until molecular counts of other designated species represent the output. The importance of CRNs as a model of computation is underscored by the wide number of closely related models, including population protocols [2, 4], Petri nets [23], and vector addition systems [21].[1]

The standard model of stochastic chemical kinetics [20], referred to hereafter as the *standard stochastic model*, dictates that the execution of a CRN protocol (operating under fixed environmental conditions) advances as a continuous time Markov process, where the rate of each reaction is determined by the molecular counts of its reactants as well as a reaction specific rate coefficient. This model crucially assumes that the system is "well-mixed", and so any pair of distinct molecules is equally likely to interact, and that the rate coefficients remain fixed.[2] Under the standard stochastic model, CRNs can simulate Turing machines if a small error probability is tolerated [14]. The correctness of some protocols, including Turing machine simulations, depends sensitively on the "idealized conditions" of fixed rate coefficients and a well-mixed system.

However, correctness of many other CRN protocols, such as those which stably compute predicates and functions [2, 4, 11, 18, 8, 10], is premised on quite different assumptions: correct output should be produced on *all* "fair executions" of the protocol, which means that the correctness of these protocols does not depend on idealized conditions. These protocols operate under a notion of fairness, adopted originally in [2], requiring that reachable configurations are not starved; in the current paper, we refer to this fairness notion as *strong fairness*. A celebrated result of Angluin et al. [2, 4] states that with respect to strong fairness, a predicate can be decided by a CRN if and only if it is semilinear.

As the "what can be computed by CRNs?" question reaches a conclusion, the focus naturally shifts to its "how fast?" counterpart. The latter question is important as the analysis of CRN runtime complexity enables the comparison between different CRN protocols and ultimately guides the search for better ones. Even for CRNs designed to operate on all (strongly) fair executions, the existing runtime analyses assume that reactions are *scheduled stochastically*, namely, according to the Markov process of the standard stochastic model, consistent with having the aforementioned idealized conditions. However, such conditions may well not hold in real wet-lab experiments, where additional factors can significantly affect the order at which reactions proceed [25]. For example, temperature can fluctuate, or molecules may be temporarily unavailable, perhaps sticking to the side of a test tube or reversibly binding to another reactant. Consequently, our work is driven by the following research question: *Is there a meaningful way to quantify the runtime of CRNs when idealized conditions do not necessarily hold?*

The Quest for an Adversarial Runtime Measure. We search for a runtime measure suitable for *adversarially scheduled* executions, namely, executions that are not subject to the constraints of the aforementioned idealized conditions. This is tricky since the adversarial scheduler may generate (arbitrarily) long execution intervals during which no progress can be made, even if those are not likely to be scheduled stochastically. Therefore, the "adversarial runtime measure" should neutralize the devious behavior of the scheduler by ensuring that

[1] To simplify the discussions, we subsequently stick to the CRN terminology even when citing literature that was originally written in terms of these related models.

[2] We follow the common assumption that each reaction involves at most two reactants.

the protocol is not unduly penalized from such bad execution intervals. To guide our search, we look for inspiration from another domain of decentralized computation that faced a similar challenge: distributed network algorithms.

While it is straightforward to measure the runtime of (idealized) synchronous distributed protocols, early on, researchers identified the need to define runtime measures also for (adversarially scheduled) asynchronous distributed protocols [6, 16]. The adversarial runtime measures that were formulated in this regard share the following two principles: (P1) partition the execution into *rounds*, so that in each round, the protocol has an opportunity to make progress; and (P2) calibrate the runtime charged to the individual rounds so that if the adversarial scheduler opts to generate the execution according to the idealized benchmark, then the adversarial runtime measure coincides with the idealized one.

Specifically, in the context of asynchronous message passing protocols, Awerbuch [6] translates principle (P1) to the requirement that no message is delayed for more than a single round, whereas in the context of the distributed daemon, Dolev et al. [16] translate this principle to the requirement that each node is activated at least once in every round. For principle (P2), both [6] and [16] take the "idealized benchmark" to be a synchronous execution in which every round costs one time unit.

When it comes to formulating an adversarial runtime measure for CRN protocols, principle (P2) is rather straightforward: we should make sure that on stochastically generated executions (playing the "idealized benchmark" role), the adversarial runtime measure agrees (in expectation) with that of the corresponding continuous time Markov process. Interpreting principle (P1), however, seems more difficult as it is not clear how to partition the execution into rounds so that in each round, the protocol "has an opportunity to make progress".

The first step towards resolving this difficulty is to introduce an alternative notion of fairness, referred to hereafter as *weak fairness*: An execution is weakly fair if a continuously applicable reaction (i.e., one for which the needed reactants are available) is not starved; such a reaction is either eventually scheduled or the system reaches a configuration where the reaction is inapplicable. Using a graph theoretic characterization, we show that any CRN protocol whose correctness is guaranteed on weakly fair executions is correct also on strongly fair executions (see Cor. 3), thus justifying the weak vs. strong terminology choice. It turns out that for predicate decidability, strong fairness is actually not strictly stronger: protocols operating under the weak fairness assumption can decide all (and only) semilinear predicates (see Thm. 8).

It remains to come up with a scheme that partitions an execution of CRN protocols into rounds in which the weakly fair adversarial scheduler can steer the execution in a nefarious direction, but also the protocol has an opportunity to make progress. A naive attempt at ensuring progress would be to end the current round once every applicable reaction is either scheduled or becomes inapplicable; the resulting partition is too coarse though since in general, a CRN does not have to "wait" for *all* its applicable reactions in order to make progress. Another naive attempt is to end the current round once *any* reaction is scheduled; this yields a partition which is too fine, allowing the scheduler to charge the protocol's run-time for (arbitrarily many) "progress-less rounds".

So, which reaction is necessary for the CRN protocol to make progress? We do not have a good answer for this question, but we know who does...

Runtime and Skipping Policies. Our adversarial runtime measure is formulated so that it is the protocol designer who decides which reaction is necessary for the CRN protocol to make progress. This is done by means of a *runtime policy* ϱ, used solely for the runtime analysis, that maps each configuration $\mathbf{c}$ to a *target* reaction $\varrho(\mathbf{c})$. (Our actual definition of

runtime policies is more general, mapping each configuration to a set of target reactions; see Sec. 4.) Symmetrically to the protocol designer's runtime policy, we also introduce a *skipping policy* σ, chosen by the adversarial scheduler, that maps each step $t \geq 0$ to a step $\sigma(t) \geq t$.

These two policies partition a given execution η into successive rounds based on the following inductive scheme: Round 0 starts at step $t(0) = 0$. Assuming that round $i \geq 0$ starts at step $t(i)$, the prefix of round i is determined by the adversarial skipping policy σ so that it lasts until step $\sigma(t(i))$; let $\mathbf{e}^i$ denote the configuration in step $\sigma(t(i))$, referred to as the round's *effective configuration*. Following that, the suffix of round i is determined by the protocol designer's runtime policy ϱ so that it lasts until the earliest step in which the target reaction $\varrho(\mathbf{e}^i)$ of the round's effective configuration $\mathbf{e}^i$ is either scheduled or becomes inapplicable. That is, in each round, the adversarial scheduler determines (by means of the skipping policy) the round's effective configuration, striving to ensure that progress from this configuration is slow, whereas the runtime policy determines when progress has been made from the effective configuration. This scheme is well defined by the choice of weak fairness; we emphasize that this would not be the case with strong fairness.

The partition of execution η into rounds allows us to ascribe a runtime to η by charging each round with a *temporal cost* and then accumulating the temporal costs of all rounds until η terminates.[3] The temporal cost of round i is defined to be the expected (continuous) time until the target reaction $\varrho(\mathbf{e}^i)$ of its effective configuration $\mathbf{e}^i$ is either scheduled or becomes inapplicable in an imaginary execution that starts at $\mathbf{e}^i$ and proceeds according to the stochastic scheduler.[4] In other words, the protocol's runtime is *not* charged for the prefix of round i that lasts until the (adversarially chosen) effective configuration is reached; the temporal cost charged for the round's suffix, emerging from the effective configuration, is the expected time that this suffix would have lasted in a stochastically scheduled execution (i.e., the idealized benchmark).

The asymptotic runtime of the CRN protocol is defined by minimizing over all runtime policies ϱ and then maximizing over all weakly fair executions η and skipping policies σ. Put differently, the protocol designer first commits to ϱ and only then, the (weakly fair) adversarial scheduler determines η and σ.

Intuitively, the challenge in constructing a good runtime policy ϱ (the challenge one faces when attempting to up-bound a protocol's runtime) is composed of two, often competing, objectives (see, e.g., Fig. 1): On the one hand, $\varrho(\mathbf{c})$ should be selected so that every execution η is guaranteed to gain "significant progress" by the time a round whose effective configuration is $\mathbf{c}$ ends, thus minimizing the number of rounds until η terminates. On the other hand, $\varrho(\mathbf{c})$ should be selected so that the temporal cost of such a round is small, thus minimizing the contribution of the individual rounds to η's runtime. In the typical scenarios, a good runtime policy ϱ results in partitioning η into $n^{\Theta(1)}$ rounds, each contributing a temporal cost between $\Theta(1/n)$ and $\Theta(n)$, where n is η's initial molecular count.

To verify that our adversarial runtime measure is indeed compatible with the aforementioned principle (P2), we show that if the (adversarial) scheduler opts to generate the execution η stochastically, then our runtime measure coincides (in expectation) with that of the corresponding continuous time Markov process (see Lem. 5). The adversarial scheduler however can be more malicious than that: simple examples show that in general, the runtime of a CRN protocol on adversarially scheduled executions may be significantly larger than on stochastically scheduled executions (see Fig. 2 and 3).

---

[3] The exact meaning of termination in this regard is made clear in Sec. 2.

[4] Here, it is assumed that the stochastic scheduler operates with no rate coefficients and with a linear volume (a.k.a. "parallel time"), see Sec. 2.

While runtime analyses of CRNs in the presence of common defect modes can be insightful, a strength of our adversarial model is that it is not tied to specific defects in actual CRNs or their biomolecular implementations. In particular, if the adversarial runtime of a CRN matches its stochastic runtime, then we would expect the CRN to perform according to its stochastic runtime even in the presence of defect modes that we may not anticipate. Moreover, in cases where stochastic runtime analysis is complex (involving reasoning about many different executions of a protocol and their likelihoods), it may in fact be easier to determine the adversarial runtime since it only requires stochastic analysis from rounds' effective configurations. For similar reasons, notions of adversarial runtime have proven to be valuable in design of algorithms in both centralized and decentralized domains more broadly, even when they do not capture realistic physical scenarios. Finally, while the analysis task of finding a good runtime policy for a given CRN may seem formidable at first, our experience in analyzing the protocols in this paper is that such a runtime policy is quite easy to deduce, mirroring intuition about the protocol's strengths and weaknesses.

The Runtime of Predicate Decidability. After formulating the new adversarial runtime measure, we turn our attention to CRN protocols whose goal is to decide whether the initial configuration satisfies a given predicate, indicated by the presence of designated Boolean ("yes" and "no") *voter* species in the output configuration. As mentioned earlier, the predicates that can be decided in that way are exactly the semilinear predicates, which raises the following two questions: What is the optimal adversarial runtime of protocols that decide semilinear predicates in general? Are there semilinear predicates that can be decided faster?

A notion that plays an important role in answering these questions is that of CRN *speed faults*, introduced in the impressive work of Chen et al. [10]. This notion captures a (reachable) configuration from which any path to an output configuration includes a (bimolecular) reaction both of whose reactants appear in $O(1)$ molecular counts. The significance of speed faults stems from the fact that any execution that reaches such a "pitfall configuration" requires $\Omega(n)$ time (in expectation) to terminate under the standard stochastic model.[5] The main result of [10] states that a predicate can be decided by a speed fault free CRN protocol (operating under the strongly fair adversarial scheduler) if and only if it belongs to the class of detection predicates (a subclass of semilinear predicates).

The runtime measure introduced in the current paper can be viewed as a quantitative generalization of the fundamentally qualitative notion of speed faults (the quest for such a generalization was, in fact, the main motivation for this work). As discussed in Sec. 4.1, in our adversarial setting, a speed fault translates to an $\Omega(n)$ runtime lower bound, leading to an $\Omega(n)$ runtime lower bound for the task of deciding any non-detection semilinear predicate. On the positive side, we prove that this bound is tight: any semilinear predicate (in particular, the non-detection ones) can be decided by a CRN protocol operating under the weakly fair adversarial scheduler whose runtime is $O(n)$ (see Thm. 8). For detection predicates, we establish a better upper bound (which is also tight): any detection predicate can be decided by a CRN protocol operating under the weakly fair adversarial scheduler whose runtime is $O(\log n)$ (see Thm. 9). Refer to Table 1 for a summary of the adversarial runtime complexity bounds established for predicate decidability tasks; for comparison, Table 2 presents a similar summary of the known stochastic runtime complexity bounds.

[5] The definition of runtime in [10] is based on a slightly different convention which results in scaling the runtime expressions by a $1/n$ factor.

Paper's Outline. The rest of the paper is organized as follows. The CRN model used in this paper is presented in Sec. 2. In Sec. 3, we link the correctness of a CRN protocol to certain topological properties of its configuration digraph. Our new runtime notion for adversarially scheduled executions is introduced in Sec. 4, where we also establish the soundness of this notion and formalize its connection to speed faults. Sec. 5 presents our results for predicate deciding CRNs. These are accompanied by a generic technique for amplifying the molecular count of the voter species in the outcome, introduced in Sec. 6.

2 Chemical Reaction Networks

In this section, we present the *chemical reaction network (CRN)* computational model. For the most part, we adhere to the conventions of the existing CRN literature (e.g., [14, 12, 9]), but we occasionally deviate from them for the sake of simplifying the subsequent discussions. (Refer to Fig. 1a–4a for illustrations of the notions presented in this section.)

A *CRN* is a protocol Π specified by the pair $\Pi = (\mathcal{S}, \mathcal{R})$, where $\mathcal{S}$ is a fixed set of *species* and $\mathcal{R} \subset \mathbb{N}^{\mathcal{S}} \times \mathbb{N}^{\mathcal{S}}$ is a fixed set of *reactions*.[6] For a reaction $\alpha = (\mathbf{r}, \mathbf{p}) \in \mathcal{R}$, the vectors $\mathbf{r} \in \mathbb{N}^{\mathcal{S}}$ and $\mathbf{p} \in \mathbb{N}^{\mathcal{S}}$ specify the stoichiometry of α's *reactants* and *products*, respectively.[7] Specifically, the entry $\mathbf{r}(A)$ (resp., $\mathbf{p}(A)$) indexed by a species $A \in \mathcal{S}$ in the vector $\mathbf{r}$ (resp., $\mathbf{p}$) encodes the number of molecules of A that are consumed (resp., produced) when α is applied. Species A is a *catalyst* for the reaction $\alpha = (\mathbf{r}, \mathbf{p})$ if $\mathbf{r}(A) = \mathbf{p}(A) > 0$.

We adhere to the convention (see, e.g., [11, 17, 15, 10]) that each reaction $(\mathbf{r}, \mathbf{p}) \in \mathcal{R}$ is either *unimolecular* with $\|\mathbf{r}\| = 1$ or *bimolecular* with $\|\mathbf{r}\| = 2$. [8] Note that if all reactions $(\mathbf{r}, \mathbf{p}) \in \mathcal{R}$ are bimolecular and *density preserving*, namely, $\|\mathbf{r}\| = \|\mathbf{p}\|$, then the CRN model is equivalent to the extensively studied *population protocols* model [2, 5, 22] assuming that the population protocol agents have a constant state space.

For a vector (or multiset) $\mathbf{r} \in \mathbb{N}^{\mathcal{S}}$ with $1 \leq \|\mathbf{r}\| \leq 2$, let $\mathcal{R}(\mathbf{r}) = (\{\mathbf{r}\} \times \mathbb{N}^{\mathcal{S}}) \cap \mathcal{R}$ denote the subset of reactions whose reactants correspond to $\mathbf{r}$. In the current paper, it is required that none of these reaction subsets is empty, i.e., $|\mathcal{R}(\mathbf{r})| \geq 1$ for every $\mathbf{r} \in \mathbb{N}^{\mathcal{S}}$ with $1 \leq \|\mathbf{r}\| \leq 2$. Some of the reactions in $\mathcal{R}$ may be *void*, namely, reactions $(\mathbf{r}, \mathbf{p})$ satisfying $\mathbf{r} = \mathbf{p}$; let $\mathrm{NV}(\mathcal{R}) = \{(\mathbf{r}, \mathbf{p}) \in \mathcal{R} \mid \mathbf{r} \neq \mathbf{p}\}$ denote the set of non-void reactions in $\mathcal{R}$. To simplify the exposition, we assume that if $\alpha = (\mathbf{r}, \mathbf{r}) \in \mathcal{R}$ is a void reaction, then $\mathcal{R}(\mathbf{r}) = \{\alpha\}$; this allows us to describe protocol Π by listing only its non-void reactions. For the sake of simplicity, we further assume that $\|\mathbf{r}\| \leq \|\mathbf{p}\|$ for all reactions $(\mathbf{r}, \mathbf{p}) \in \mathcal{R}$.

Configurations. A *configuration* of a CRN $\Pi = (\mathcal{S}, \mathcal{R})$ is a vector $\mathbf{c} \in \mathbb{N}^{\mathcal{S}}$ that encodes the *molecular count* $\mathbf{c}(A)$ of species A in the solution for each $A \in \mathcal{S}$.[9] The molecular count notation is extended to species (sub)sets $\Lambda \subseteq \mathcal{S}$, denoting $\mathbf{c}(\Lambda) = \sum_{A \in \Lambda} \mathbf{c}(A)$. We refer to $\mathbf{c}(\mathcal{S}) = \|\mathbf{c}\|$ as the *molecular count* of the configuration $\mathbf{c}$. Let $\mathbf{c}|_{\Lambda} \in \mathbb{N}^{\Lambda}$ denote the restriction of a configuration $\mathbf{c} \in \mathbb{N}^{\mathcal{S}}$ to a species subset $\Lambda \subseteq \mathcal{S}$.

A reaction $\alpha = (\mathbf{r}, \mathbf{p}) \in \mathcal{R}$ is said to be *applicable* to a configuration $\mathbf{c} \in \mathbb{N}^{\mathcal{S}}$ if $\mathbf{r}(A) \leq \mathbf{c}(A)$ for every $A \in \mathcal{S}$. Let $\mathrm{app}(\mathbf{c}) \subseteq \mathcal{R}$ denote the set of reactions which are applicable to $\mathbf{c}$ and let $\overline{\mathrm{app}}(\mathbf{c}) = \mathcal{R} - \mathrm{app}(\mathbf{c})$, referring to the reactions in $\overline{\mathrm{app}}(\mathbf{c})$ as being *inapplicable* to $\mathbf{c}$. We

[6] Throughout this paper, we denote $\mathbb{N} = \{z \in \mathbb{Z} \mid z \geq 0\}$.
[7] We stick to the convention of identifying vectors in $\mathbb{N}^{\mathcal{S}}$ with multisets over $\mathcal{S}$ expressed as a "molecule summation".
[8] The notation $\|\cdot\|$ denotes the 1-norm ℓ_1.
[9] We consider the discrete version of the CRN model, where the configuration encodes integral molecular counts, in contrast to the continuous model, where a configuration is given by real species densities.

restrict our attention to configurations $\mathbf{c}$ with molecular count $\|\mathbf{c}\| \geq 1$, which ensures that $\text{app}(\mathbf{c})$ is never empty. For a reaction $\alpha \in \text{app}(\mathbf{c})$, let $\alpha(\mathbf{c}) = \mathbf{c} - \mathbf{r} + \mathbf{p}$ be the configuration obtained by applying α to $\mathbf{c}$.[10]

Given two configurations $\mathbf{c}, \mathbf{c}' \in \mathbb{N}^{\mathcal{S}}$, the binary relation $\mathbf{c} \rightharpoonup \mathbf{c}'$ holds if there exists a reaction $\alpha \in \text{app}(\mathbf{c})$ such that $\alpha(\mathbf{c}) = \mathbf{c}'$. We denote the reflexive transitive closure of $\rightharpoonup$ by $\overset{*}{\rightharpoonup}$ and say that $\mathbf{c}'$ is *reachable* from $\mathbf{c}$ if $\mathbf{c} \overset{*}{\rightharpoonup} \mathbf{c}'$. Given a configuration set $Z \subseteq \mathbb{N}^{\mathcal{S}}$, let

$$\text{stab}(Z) \triangleq \left\{\mathbf{c} \in Z \mid \mathbf{c} \overset{*}{\rightharpoonup} \mathbf{c}' \implies \mathbf{c}' \in Z\right\} \quad \text{and} \quad \text{halt}(Z) \triangleq \left\{\mathbf{c} \in Z \mid \mathbf{c} \overset{*}{\rightharpoonup} \mathbf{c}' \implies \mathbf{c}' = \mathbf{c}\right\},$$

observing that the latter set is a (not necessarily strict) subset of the former.

For the sake of simplicity, we restrict this paper's focus to protocols that *respect finite density* [17], namely, $\mathbf{c} \overset{*}{\rightharpoonup} \mathbf{c}'$ implies that $\|\mathbf{c}'\| \leq O(\|\mathbf{c}\|)$. We note that density preserving CRNs inherently respect finite density, however we also allow for reactions that have more products than reactants as long as the CRN protocol is designed so that the molecular count cannot increase arbitrarily. This means, in particular, that although the configuration space $\mathbb{N}^{\mathcal{S}}$ is inherently infinite, the set $\{\mathbf{c}' \in \mathbb{N}^{\mathcal{S}} \mid \mathbf{c} \overset{*}{\rightharpoonup} \mathbf{c}'\}$ is finite (and bounded as a function of $\|\mathbf{c}\|$) for any configuration $\mathbf{c} \in \mathbb{N}^{\mathcal{S}}$.

Executions. An *execution* η of the CRN Π is an infinite sequence $\eta = \langle \mathbf{c}^t, \alpha^t \rangle_{t \geq 0}$ of $\langle \text{configuration}, \text{reaction} \rangle$ pairs such that $\alpha^t \in \text{app}(\mathbf{c}^t)$ and $\mathbf{c}^{t+1} = \alpha^t(\mathbf{c}^t)$ for every $t \geq 0$. It is convenient to think of η as progressing in discrete *steps* so that configuration $\mathbf{c}^t$ and reaction α^t are associated with step $t \geq 0$. We refer to $\mathbf{c}^0$ as the *initial configuration* of η and, unless stated otherwise, denote the molecular count of $\mathbf{c}^0$ by $n = \|\mathbf{c}^0\|$. Given a configuration set $Z \subseteq \mathbb{N}^{\mathcal{S}}$, we say that η *stabilizes* (resp., *halts*) into Z if there exists a step $t \geq 0$ such that $\mathbf{c}^t \in \text{stab}(Z)$ (resp., $\mathbf{c}^t \in \text{halt}(Z)$) and refer to the earliest such step t as the execution's *stabilization step* (resp., *halting step*) with respect to Z.

In this paper, we consider an *adversarial scheduler* that knows the CRN protocol Π and the initial configuration $\mathbf{c}^0$ and determines the execution $\eta = \langle \mathbf{c}^t, \alpha^t \rangle_{t \geq 0}$ in an arbitrary (malicious) way. The execution η is nonetheless subject to the following *fairness* condition: for every $t \geq 0$ and for every $\alpha \in \text{app}(\mathbf{c}^t)$, there exists $t' \geq t$ such that either (I) $\alpha^{t'} = \alpha$; or (II) $\alpha \notin \text{app}(\mathbf{c}^{t'})$. In other words, the scheduler is not allowed to (indefinitely) "starve" a continuously applicable reaction. We emphasize that the mere condition that a reaction $\alpha \in \mathcal{R}$ is applicable infinitely often does not imply that α is scheduled infinitely often.

Note that the fairness condition adopted in the current paper differs from the one used in the existing CRN literature [2, 4, 11, 9]. The latter, referred to hereafter as *strong fairness*, requires that if a configuration $\mathbf{c}$ appears infinitely often in the execution η and a configuration $\mathbf{c}'$ is reachable from $\mathbf{c}$, then $\mathbf{c}'$ also appears infinitely often in η. Strictly speaking, a strongly fair execution η is not necessarily fair according to the current paper's notion of fairness (in particular, η may starve void reactions). However, as we show in Sec. 3, protocol correctness under the current paper's notion of fairness implies protocol correctness under strong fairness (see Cor. 3), where the exact meaning of correctness is defined soon. Consequently, we refer hereafter to the notion of fairness adopted in the current paper as *weak fairness*.

Interface and Correctness. The CRN notions introduced so far are independent of any particular computational task. To correlate between a CRN protocol $\Pi = (\mathcal{S}, \mathcal{R})$ and concrete computational tasks, we associate Π with a (task specific) *interface* $\mathcal{I} = (\mathcal{U}, \mu, \mathcal{C})$ whose

[10] Unless stated otherwise, all vector arithmetic is done component-wise.

semantics is as follows: $\mathcal{U}$ is a fixed set of *interface values* that typically encode the input and/or output associated with the species; $\mu : \mathcal{S} \to \mathcal{U}$ is an *interface mapping* that maps each species $A \in \mathcal{S}$ to an interface value $\mu(A) \in \mathcal{U}$; and $\mathcal{C} \subseteq \mathbb{N}^{\mathcal{U}} \times \mathbb{N}^{\mathcal{U}}$ is a *correctness relation* that determines the correctness of an execution as explained soon.[11]

Hereafter, we refer to the vectors in $\mathbb{N}^{\mathcal{U}}$ as *interface vectors*. The interface of a configuration $\mathbf{c} \in \mathbb{N}^{\mathcal{S}}$ in terms of the input/output that $\mathbf{c}$ encodes is captured by the interface vector

$$\mu(\mathbf{c}) \triangleq \langle \mathbf{c}\left(\{A \in \mathcal{S} \mid \mu(A) = u\}\right)\rangle_{u \in \mathcal{U}} .$$

The abstract interface $\mathcal{I} = (\mathcal{U}, \mu, \mathcal{C})$ allows us to define what it means for a protocol to be correct. To this end, for each configuration $\mathbf{c} \in \mathbb{N}^{\mathcal{S}}$, let $Z_{\mathcal{I}}(\mathbf{c}) = \{\mathbf{c}' \in \mathbb{N}^{\mathcal{S}} \mid (\mu(\mathbf{c}), \mu(\mathbf{c}')) \in \mathcal{C}\}$ be the set of configurations which are mapped by μ to interface vectors that satisfy the correctness relation with $\mu(\mathbf{c})$. A configuration $\mathbf{c}^0 \in \mathbb{N}^{\mathcal{S}}$ is a *valid initial configuration* with respect to $\mathcal{I}$ if $Z_{\mathcal{I}}(\mathbf{c}^0) \neq \emptyset$; an execution is *valid* (with respect to $\mathcal{I}$) if it emerges from a valid initial configuration. A valid execution η is said to be *stably correct* (resp., *haltingly correct*) if η stabilizes (resp., halts) into $Z_{\mathcal{I}}(\mathbf{c}^0)$. The protocol Π is said to be *stably correct* (resp., *haltingly correct*) if every weakly fair valid execution is guaranteed to be stably (resp., haltingly) correct.[12]

The Stochastic Scheduler. While the current paper focuses on the (weakly fair) adversarial scheduler, another type of scheduler that receives a lot of attention in the literature is the *stochastic scheduler*. Here, we present the stochastic scheduler so that it can serve as a "benchmark" for the runtime definition introduced in Sec. 4. To this end, we define the *propensity* of a reaction $\alpha = (\mathbf{r}, \mathbf{p}) \in \mathcal{R}$ in a configuration $\mathbf{c} \in \mathbb{N}^{\mathcal{S}}$, denoted by $\pi_{\mathbf{c}}(\alpha)$, as

$$\pi_{\mathbf{c}}(\alpha) = \begin{cases} \mathbf{c}(A) \cdot \frac{1}{|\mathcal{R}(\mathbf{r})|} , & \mathbf{r} = A \\ \frac{1}{\varphi} \cdot \binom{\mathbf{c}(A)}{2} \cdot \frac{1}{|\mathcal{R}(\mathbf{r})|} , & \mathbf{r} = 2A \\ \frac{1}{\varphi} \cdot \mathbf{c}(A) \cdot \mathbf{c}(B) \cdot \frac{1}{|\mathcal{R}(\mathbf{r})|} , & \mathbf{r} = A + B, A \neq B \end{cases} ,$$

where $\varphi > 0$ is a (global) *volume* parameter.[13] Notice that reaction α is applicable to $\mathbf{c}$ if and only if $\pi_{\mathbf{c}}(\alpha) > 0$. The propensity notation is extended to reaction (sub)sets $Q \subseteq \mathcal{R}$ by defining $\pi_{\mathbf{c}}(Q) = \sum_{\alpha \in Q} \pi_{\mathbf{c}}(\alpha)$. Recalling that $\mathcal{R}(\mathbf{r}) \neq \emptyset$ for each $\mathbf{r} \in \mathbb{N}^{\mathcal{S}}$ with $1 \leq \|\mathbf{r}\| \leq 2$, we observe that

$$\pi_{\mathbf{c}} \triangleq \pi_{\mathbf{c}}(\mathcal{R}) = \|\mathbf{c}\| + \frac{1}{\varphi} \cdot \binom{\|\mathbf{c}\|}{2} .$$

The stochastic scheduler determines the execution $\eta = \langle \mathbf{c}^t, \alpha^t \rangle_{t \geq 0}$ by scheduling a reaction $\alpha \in \mathrm{app}(\mathbf{c}^t)$ in step t, setting $\alpha^t = \alpha$, with probability proportional to α's propensity $\pi_{\mathbf{c}^t}(\alpha)$ in $\mathbf{c}^t$. The assumption that the CRN protocol respects finite density implies that η is (weakly and strongly) fair with probability 1. We define the *time span* of step $t \geq 0$ to be $1/\pi_{\mathbf{c}^t}$,

[11] The abstract interface formulation generalizes various families of computational tasks addressed in the CRN literature, including predicate decision [8, 10, 9] (see also Sec. 5) and function computation [11, 18, 9], as well as the vote amplification task discussed in Sec. 6, without committing to the specifics of one particular family. For example, for the CRDs presented in Sec. 5, we define $\mathcal{U} = (\Sigma \cup \{\bot\}) \times \{0, 1, \bot\}$. The interface mapping μ then maps each species $A \in \mathcal{S}$ to the interface value $\mu(A) = (x, y) \in \mathcal{U}$ defined so that (I) $x = A$ if $A \in \Sigma$; and $x = \bot$ otherwise; and (II) $y = v$ if $A \in \Upsilon_v$; and $y = \bot$ otherwise.

[12] Both notions of correctness have been studied in the CRN literature, see, e.g., [9].

[13] In the standard stochastic model [20], the propensity expression is multiplied by a reaction specific rate coefficient. In the current paper, that merely uses the stochastic scheduler as a benchmark, we make the simplifying assumption that all rate coefficients are set to 1 (c.f. [11, 10]).

i.e., the normalizing factor of the reaction selection probability.[14] Given a step $t^* \geq 0$, the *stochastic runtime* of the execution prefix $\eta^* = \langle \mathbf{c}^t, \alpha^t \rangle_{0 \leq t < t^*}$ is defined to be the accumulated time span $\sum_{t=0}^{t^*-1} 1/\pi_{\mathbf{c}^t}$.

We adopt the convention that the volume is proportional to the initial molecular count $n = \|\mathbf{c}^0\|$ [17]. The assumption that the CRN protocol respects finite density ensures that $\varphi = \Theta(\|\mathbf{c}^t\|)$ for every $t \geq 0$ which means that the volume is sufficiently large to contain all molecules throughout the (stochastic) execution η. This also means that the time span of each step $t \geq 0$ is

$$1/\pi_{\mathbf{c}^t} = \frac{\varphi}{\varphi \cdot \|\mathbf{c}^t\| + \binom{\|\mathbf{c}^t\|}{2}} = \Theta(1/\|\mathbf{c}^t\|) = \Theta(1/n)\,, \tag{1}$$

hence the stochastic runtime of an execution prefix that lasts for t^* steps is $\Theta(t^*/n)$.

3 Correctness Characterization via the Configuration Digraph

It is often convenient to look at CRN protocols through the lens of the following abstract directed graph (a.k.a. digraph): The *configuration digraph* of a protocol $\Pi = (\mathcal{S}, \mathcal{R})$ is a digraph, denoted by D^Π, whose edges are labeled by reactions in $\mathcal{R}$. The nodes of D^Π are identified with the configurations in $\mathbb{N}^\mathcal{S}$; the edge set of D^Π includes an α-labeled edge from $\mathbf{c}$ to $\alpha(\mathbf{c})$ for each configuration $\mathbf{c} \in \mathbb{N}^\mathcal{S}$ and reaction $\alpha \in \mathrm{app}(\mathbf{c})$ (thus the outdegree of $\mathbf{c}$ in D^Π is $|\mathrm{app}(\mathbf{c})|$). Observe that the self-loops of D^Π are exactly the edges labeled by (applicable) void reactions. Moreover, a configuration $\mathbf{c}'$ is reachable, in the graph theoretic sense, from a configuration $\mathbf{c}$ if and only if $\mathbf{c} \xrightarrow{*} \mathbf{c}'$. For a configuration $\mathbf{c} \in \mathbb{N}^\mathcal{S}$, let $D^\Pi_\mathbf{c}$ be the digraph induced by D^Π on the set of configurations reachable from $\mathbf{c}$ and observe that $D^\Pi_\mathbf{c}$ is finite as Π respects finite density. (Refer to Fig. 1b–4b for illustrations of the notions presented in this section.)

The *(strongly connected) components* of the configuration digraph D^Π are the equivalence classes of the "reachable from each other" relation over the configurations in $\mathbb{N}^\mathcal{S}$. We say that a reaction $\alpha \in \mathcal{R}$ *escapes* from a component S of D^Π if every configuration in S admits an outgoing α-labeled edge to a configuration not in S; i.e., $\alpha \in \mathrm{app}(\mathbf{c})$ and $\alpha(\mathbf{c}) \notin S$ for every $\mathbf{c} \in S$ (see, e.g., Fig. 1b). The notion of escaping reactions allows us to express the stable/halting correctness of CRNs in terms of their configuration digraphs.

▶ **Lemma 1.** *A CRN protocol $\Pi = (\mathcal{S}, \mathcal{R})$ is stably (resp., haltingly) correct with respect to an interface $\mathcal{I} = (\mathcal{U}, \mu, \mathcal{C})$ under a weakly fair scheduler if and only if for every valid initial configuration $\mathbf{c}^0 \in \mathbb{N}^\mathcal{S}$, every component S of $D^\Pi_{\mathbf{c}^0}$ satisfies (at least) one of the following two conditions: (1) S admits some (at least one) escaping reaction; or (2) $S \subseteq \mathrm{stab}(Z_\mathcal{I}(\mathbf{c}^0))$ (resp., $S \subseteq \mathrm{halt}(Z_\mathcal{I}(\mathbf{c}^0))$), where $Z_\mathcal{I}(\mathbf{c}^0) = \{\mathbf{c} \in \mathbb{N}^\mathcal{S} \mid (\mu(\mathbf{c}^0), \mu(\mathbf{c})) \in \mathcal{C}\}$.*

To complement Lem. 1, we also express the stable/halting correctness of CRNs in terms of their configuration digraphs under a strongly fair scheduler.

▶ **Lemma 2.** *A CRN protocol $\Pi = (\mathcal{S}, \mathcal{R})$ is stably (resp., haltingly) correct with respect to an interface $\mathcal{I} = (\mathcal{U}, \mu, \mathcal{C})$ under a strongly fair scheduler if and only if for every valid initial configuration $\mathbf{c}^0 \in \mathbb{N}^\mathcal{S}$, every component S of $D^\Pi_{\mathbf{c}^0}$ satisfies (at least) one of the following two conditions: (1) S admits some (at least one) edge outgoing to another component; or (2) $S \subseteq \mathrm{stab}(Z_\mathcal{I}(\mathbf{c}^0))$ (resp., $S \subseteq \mathrm{halt}(Z_\mathcal{I}(\mathbf{c}^0))$), where $Z_\mathcal{I}(\mathbf{c}^0) = \{\mathbf{c} \in \mathbb{N}^\mathcal{S} \mid (\mu(\mathbf{c}^0), \mu(\mathbf{c})) \in \mathcal{C}\}$.*

[14] The time span definition is consistent with the expected time until a reaction occurs under the continuous time Markov process formulation of the standard stochastic model [20] with no rate coefficients.

Combining Lem. 1 and 2, we obtain the following corollary.

▶ **Corollary 3.** *If a CRN protocol* $\Pi = (\mathcal{S}, \mathcal{R})$ *is stably (resp., haltingly) correct with respect to an interface* $\mathcal{I}$ *under a weakly fair scheduler, then* Π *is also stably (resp., haltingly) correct with respect to* $\mathcal{I}$ *under a strongly fair scheduler.*

4 The Runtime of Adversarially Scheduled Executions

So far, the literature on CRN protocols operating under an adversarial scheduler focused mainly on computability, leaving aside, for the most part, complexity considerations. [15] This is arguably unavoidable when working with the strong fairness condition which is inherently oblivious to the chain of reactions that realizes the reachability of one configuration from another. In the current paper, however, we adopt the weak fairness condition which facilitates the definition of a quantitative measure for the runtime of adversarially scheduled executions, to which this section is dedicated. (Refer to Fig. 1c–4c for illustrations of the notions presented in this section.)

Consider a stably (resp., haltingly) correct CRN protocol $\Pi = (\mathcal{S}, \mathcal{R})$. We make extensive use of the following operator: Given a weakly fair execution $\eta = \langle \mathbf{c}^t, \alpha^t \rangle_{t \geq 0}$, a step $t \geq 0$, and a reaction (sub)set $Q \subseteq \mathcal{R}$, let $\tau(\eta, t, Q)$ be the earliest step $s > t$ such that at least one of the following two conditions is satisfied:
(I) $\alpha^{s-1} \in Q$; or
(II) $Q \subseteq \bigcup_{t \leq t' \leq s} \overline{\mathrm{app}}(\mathbf{c}^{t'})$.
(This operator is well defined by the weak fairness of η.)

Intuitively, we think of the operator $\tau(\eta, t, Q)$ as a process that tracks η from step t onward and stops once any Q reaction is scheduled (condition (I)). This by itself is not well defined as the scheduler may avoid scheduling the Q reactions from step t onward. However, the scheduler must prevent the starvation of any continuously applicable reaction in Q, so we also stop the τ-process once the adversary "fulfills this commitment" (condition (II)).

The Policies. Our runtime measure is based on partitioning a given weakly fair execution $\eta = \langle \mathbf{c}^t, \alpha^t \rangle_{t \geq 0}$ into *rounds*. This is done by means of two policies: a *runtime policy* ϱ, determined by the protocol designer, that maps each configuration $\mathbf{c} \in \mathbb{N}^{\mathcal{S}}$ to a non-void reaction (sub)set $\varrho(\mathbf{c}) \subseteq \mathrm{NV}(\mathcal{R})$, referred to as the *target reaction set* of $\mathbf{c}$ under ϱ; and a *skipping policy* σ, determined by the adversarial scheduler (in conjunction with the execution η), that maps each step $t \geq 0$ to a step $\sigma(t) \geq t$.

Round $i = 0, 1, \ldots$ spans the step interval $[t(i), t(i+1))$ and includes a designated *effective step* $t(i) \leq t_{\mathrm{e}}(i) < t(i+1)$. The partition of execution η into rounds is defined inductively by setting

$$t(i) \;=\; \begin{cases} 0\,, & i = 0 \\ \tau\left(\eta, t_{\mathrm{e}}(i-1), \varrho\left(\mathbf{c}^{t_{\mathrm{e}}(i-1)}\right)\right)\,, & i > 0 \end{cases} \qquad \text{and} \qquad t_{\mathrm{e}}(i) \;=\; \sigma(t(i))\,.$$

Put differently, for every round $i \geq 0$ with initial step $t(i)$, the adversarial scheduler first determines the round's effective step $t_{\mathrm{e}}(i) = \sigma(t(i)) \geq t(i)$ by means of the skipping policy σ. Following that, we apply the runtime policy ϱ (chosen by the protocol designer) to the configuration $\mathbf{e}^i = \mathbf{c}^{t_{\mathrm{e}}(i)}$, referred to as the round's *effective configuration*, and obtain the

[15] The one exception in this regard is the work of Chen et al. [10] on speed faults – see Sec. 4.1 and 5.

target reaction set $Q = \varrho(\mathbf{e}^i)$. The latter is then plugged into the operator τ to determine $t(i+1) = \tau(\eta, t_e(i), Q)$. Round i is said to be *target-accomplished* if $\alpha^{t(i+1)-1} \in Q$; otherwise, it is said to be *target-deprived*.

▶ Remark. Our definition of the runtime policy ϱ does not require that the reactions included in the target reaction set $\varrho(\mathbf{c})$ are applicable to the configuration $\mathbf{c} \in \mathbb{N}^{\mathcal{S}}$. Notice though that if all target reactions are inapplicable to $\mathbf{c}$ (which is bound to be the case if $\mathbf{c}$ is halting), then a round whose effective configuration is $\mathbf{c}$ is destined to be target deprived and end immediately after the effective step, regardless of the reaction scheduled in that step. In the full version [13], we investigate several other "natural restrictions" of the runtime policy definition, including fixed policies and singleton target reaction sets, showing that they all lead to significant efficiency loss.

Temporal Cost. We define the *temporal cost* of a configuration $\mathbf{c} \in \mathbb{N}^{\mathcal{S}}$ under a runtime policy ϱ, denoted by $\mathrm{TC}^{\varrho}(\mathbf{c})$, as follows: Let $\eta_r = \langle \mathbf{c}_r^t, \alpha_r^t \rangle_{t \geq 0}$ be a stochastic execution emerging from the initial configuration $\mathbf{c}_r^0 = \mathbf{c}$ and define

$$\mathrm{TC}^{\varrho}(\mathbf{c}) \triangleq \mathbb{E}\left(\sum_{t=0}^{\tau(\eta_r,0,\varrho(\mathbf{c}))-1} 1/\pi_{\mathbf{c}_r^t}\right) = \Theta(1/\|\mathbf{c}\|) \cdot \mathbb{E}\left(\tau(\eta_r, 0, \varrho(\mathbf{c}))\right) ,$$

where the expectation is over the random choice of η_r and the second transition is due to (1). That is, the temporal cost of $\mathbf{c}$ under ϱ is defined to be the expected stochastic runtime of round 0 of η_r with respect to the runtime policy ϱ and the identity skipping policy σ_{id} that maps each step $t \geq 0$ to $\sigma_{\mathrm{id}}(t) = t$ (which means that the effective step of each round is its initial step).

Execution Runtime. Consider a runtime policy ϱ and a skipping policy σ. Let $\eta = \langle \mathbf{c}^t, \alpha^t \rangle_{t \geq 0}$ be a weakly fair valid execution and let $t(i)$, $t_e(i)$, and $\mathbf{e}^i = \mathbf{c}^{t_e(i)}$ be the initial step, effective step, and effective configuration, respectively, of round $i \geq 0$ under ϱ and σ. Fix some step $t^* \geq 0$ and consider the execution prefix $\eta^* = \langle \mathbf{c}^t, \alpha^t \rangle_{0 \leq t < t^*}$. We define the *(adversarial) runtime* of η^* under ϱ and σ, denoted by $\mathrm{RT}^{\varrho,\sigma}(\eta^*)$, by taking $i^* = \min\{i \geq 0 \mid t(i) \geq t^*\}$ and setting

$$\mathrm{RT}^{\varrho,\sigma}(\eta^*) \triangleq \sum_{i=0}^{i^*-1} \mathrm{TC}^{\varrho}\left(\mathbf{e}^i\right) .$$

The *stabilization runtime* (resp., *halting runtime*) of the (entire) execution η under ϱ and σ, denoted by $\mathrm{RT}^{\varrho,\sigma}_{\mathrm{stab}}(\eta)$ (resp., $\mathrm{RT}^{\varrho,\sigma}_{\mathrm{halt}}(\eta)$), is defined to be $\mathrm{RT}^{\varrho,\sigma}\left(\langle \mathbf{c}^t, \alpha^t \rangle_{0 \leq t < t^*}\right)$, where $t^* \geq 0$ is the stabilization (resp., halting) step of η. In other words, we use ϱ and σ to partition η into rounds and mark the effective steps. Following that, we charge each round i that starts before step t^* according to the temporal cost (under ϱ) of its effective configuration $\mathbf{e}^i$.

Looking at it from another angle, using the skipping policy σ, the adversarial scheduler determines the sequence $\mathbf{e}^0, \mathbf{e}^1, \ldots$ of effective configurations according to which the temporal cost $\mathrm{TC}^{\varrho}(\mathbf{e}^i)$ of each round $i \geq 0$ is calculated. By choosing an appropriate runtime policy ϱ, the protocol designer may (1) ensure that progress is made from one effective configuration to the next, thus advancing η towards round $i^* = \min\{i \geq 0 \mid t(i) \geq t^*\}$; and (2) bound the contribution $\mathrm{TC}^{\varrho}(\mathbf{e}^i)$ of each round $0 \leq i < i^*$ to the stabilization runtime $\mathrm{RT}^{\varrho,\sigma}_{\mathrm{stab}}(\eta)$ (resp., halting runtime $\mathrm{RT}^{\varrho,\sigma}_{\mathrm{halt}}(\eta)$). The crux of our runtime definition is that this contribution depends only on the effective configuration $\mathbf{e}^i$, irrespectively of how round i actually develops (see, e.g., Fig. 1c).

▶ Remark. Using this viewpoint, it is interesting to revisit the definitions of [6] and [16] for the runtime of an asynchronous distributed protocol $\mathcal{P}$. Following the discussion in Sec. 1, this runtime is defined as the length of the longest sequence $\mathbf{e}^0, \mathbf{e}^1, \ldots, \mathbf{e}^{i^*-1}$ of "non-terminal"

configurations (of $\mathcal{P}$) such that $\mathbf{e}^i$ is reachable from $\mathbf{e}^{i-1}$ by an execution interval that lasts for at least one round (according to the respective definitions of [6] and [16]). Our adversarial runtime is defined in the same manner, taking $\mathbf{e}^0, \mathbf{e}^1, \ldots, \mathbf{e}^{i^*-1}$ to be the first i^* effective (CRN) configurations, only that we charge each configuration $\mathbf{e}^i$ according to its temporal cost (rather than one "runtime unit" as in [6] and [16]). This difference is consistent with the different "idealized benchmarks": a synchronous schedule in [6] and [16] vs. a stochastic execution in the current paper. The skipping policy σ plays a key role in adversarially generating the sequence $\mathbf{e}^0, \mathbf{e}^1, \ldots, \mathbf{e}^{i^*-1}$ as it "decouples" between the last step of round i, determined by the runtime policy ϱ, and the effective configuration $\mathbf{e}^{i+1}$ of round $i+1$ (see, e.g., Fig. 4c).

The Runtime Function. For $n \geq 1$, let $\mathcal{F}(n)$ denote the set of weakly fair valid executions $\eta = \langle \mathbf{c}^t, \alpha^t \rangle_{t \geq 0}$ of initial molecular count $\|\mathbf{c}^0\| = n$. The *stabilization runtime* (resp., *halting runtime*) of the CRN protocol Π for executions in $\mathcal{F}(n)$, denoted by $\mathrm{RT}^{\Pi}_{\mathrm{stab}}(n)$ (resp., $\mathrm{RT}^{\Pi}_{\mathrm{halt}}(n)$), is defined to be

$$\mathrm{RT}^{\Pi}_{\mathrm{x}}(n) \triangleq \min_{\varrho} \max_{\eta \in \mathcal{F}(n),\, \sigma} \mathrm{RT}^{\varrho,\sigma}_{\mathrm{x}}(\eta)\,,$$

where x serves as a placeholder for stab (resp., halt). This formalizes the responsibility of the protocol designer to specify a runtime policy ϱ, in conjunction with the protocol Π, used for up-bounding Π's stabilization (resp., halting) runtime (see, e.g., Fig. 1c).

The following two lemmas establish the soundness of our adversarial runtime definition: Lem. 4 ensures that the stabilization (resp., halting) runtime function is well defined.[16] In Lem. 5, we show that if the scheduler generates the execution stochastically, then our (adversarial) runtime measure agrees in expectation with the stochastic runtime measure.

▶ **Lemma 4.** *Consider a stably (resp., haltingly) correct protocol $\Pi = (\mathcal{S}, \mathcal{R})$. There exists a runtime policy ϱ such that for every integer $n \geq 1$, execution $\eta \in \mathcal{F}(n)$, and skipping policy σ, the stabilization runtime $\mathrm{RT}^{\varrho,\sigma}_{\mathrm{stab}}(\eta)$ (resp., halting runtime $\mathrm{RT}^{\varrho,\sigma}_{\mathrm{halt}}(\eta)$) is up-bounded as a function of n.*

▶ **Lemma 5.** *Consider a stably (resp., haltingly) correct protocol $\Pi = (\mathcal{S}, \mathcal{R})$. Let $\eta_r = \langle \mathbf{c}^t_r, \alpha^t_r \rangle_{t \geq 0}$ be a stochastic execution emerging from a valid initial configuration $\mathbf{c}^0_r$ and let $t^* \geq 0$ be the stabilization (resp., halting) step of η_r. Then,*

$$\min_{\varrho} \mathbb{E}_{\eta_r} \left(\max_{\sigma} \mathrm{RT}^{\varrho,\sigma}_{\mathrm{x}}(\eta_r) \right) = \mathbb{E}_{\eta_r} \left(\sum_{t=0}^{t^*-1} 1 / \pi_{\mathbf{c}^t_r} \right),$$

where x *serves as a placeholder for* stab *(resp.,* halt*).*

4.1 Speed Faults

Consider a CRN protocol $\Pi = (\mathcal{S}, \mathcal{R})$ which is stably (resp., haltingly) correct with respect to an interface $\mathcal{I} = (\mathcal{U}, \mu, \mathcal{C})$. For a valid initial configuration $\mathbf{c}^0 \in \mathbb{N}^{\mathcal{S}}$, let $Z_{\mathcal{I}}(\mathbf{c}^0) = \{\mathbf{c} \in \mathbb{N}^{\mathcal{S}} \mid (\mu(\mathbf{c}^0), \mu(\mathbf{c})) \in \mathcal{C}\}$ and recall that if a weakly fair execution η of Π emerges from $\mathbf{c}^0$, then η is guaranteed to reach $\mathrm{stab}(Z_{\mathcal{I}}(\mathbf{c}^0))$ (resp., $\mathrm{halt}(Z_{\mathcal{I}}(\mathbf{c}^0))$).

Given a parameter $s > 0$, a configuration $\mathbf{c} \in \mathbb{N}^{\mathcal{S}}$ is said to be a *stabilization s-pitfall* (resp., a *halting s-pitfall*) of the valid initial configuration $\mathbf{c}^0$ if $\mathbf{c}^0 \overset{*}{\rightharpoonup} \mathbf{c}$ and every path from $\mathbf{c}$ to $\mathrm{stab}(Z_{\mathcal{I}}(\mathbf{c}^0))$ (resp., $\mathrm{halt}(Z_{\mathcal{I}}(\mathbf{c}^0))$) in the digraph D^{Π} includes (an edge labeled by) a

[16] Note that in Lem. 4 we use a universal runtime policy that applies to all choices of the initial molecular count n. This is stronger in principle than what the runtime definition actually requires.

reaction whose propensity is at most s/φ (see, e.g., Fig. 2c and 4c). When $s = O(1)$, we often omit the parameter and refer to $\mathbf{c}$ simply as a *stabilization pitfall* (resp., *halting pitfall*). Following the definition of Chen et al. [10], we say that an infinite family $\mathbf{C}^0$ of valid initial configurations has a *stabilization speed fault* (resp., *halting speed fault*) if for every integer $n_0 > 0$, there exists a configuration $\mathbf{c}^0 \in \mathbf{C}^0$ of molecular count $\|\mathbf{c}^0\| = n \geq n_0$ that admits a stabilization (resp., halting) pitfall.

▶ **Lemma 6.** *If an infinite family $\mathbf{C}^0$ of valid initial configurations has a stabilization (resp., halting) speed fault, then for every integer $n_0 > 0$, there exist a configuration $\mathbf{c}^0 \in \mathcal{C}^0$ of molecular count $\|\mathbf{c}^0\| = n \geq n_0$, a weakly fair execution η emerging from $\mathbf{c}^0$, and a skipping policy σ, such that $\mathrm{RT}_{\mathrm{x}}^{\varrho,\sigma}(\eta) \geq \Omega(n)$ for every runtime policy ϱ, where* x *serves as a placeholder for* stab *(resp.,* halt*).*[17]

5 Predicate Decidability

An important class of CRN protocols is that of *chemical reaction deciders (CRDs)* whose goal is to determine whether the initial molecular counts of certain species satisfy a given predicate. In its most general form (see [10, 9]), a CRD is a CRN protocol $\Pi = (\mathcal{S}, \mathcal{R})$ augmented with (1) a set $\Sigma \subset \mathcal{S}$ of *input species*; (2) two disjoint sets $\Upsilon_0, \Upsilon_1 \subset \mathcal{S}$ of *voter species*; (3) a designated *fuel species* $F \in \mathcal{S} - \Sigma$; and (4) a fixed *initial context* $\mathbf{k} \in \mathbb{N}^{\mathcal{S}-(\Sigma\cup\{F\})}$. The CRD is said to be *leaderless* if its initial context is the zero vector, i.e., $\mathbf{k} = \mathbf{0}$.

A configuration $\mathbf{c}^0 \in \mathbb{N}^{\mathcal{S}}$ is valid as an initial configuration of the CRD Π if $\mathbf{c}^0|_{\mathcal{S}-(\Sigma\cup\{F\})} = \mathbf{k}$; to ensure that the initial molecular count $\|\mathbf{c}^0\|$ is always at least 1 (especially when the CRD is leaderless), we also require that $\mathbf{c}^0(F) \geq 1$. In other words, a valid initial configuration $\mathbf{c}^0$ can be decomposed into an *input vector* $\mathbf{c}^0|_\Sigma = \mathbf{x} \in \mathbb{N}^\Sigma$, the initial context $\mathbf{c}^0|_{\mathcal{S}-(\Sigma\cup\{F\})} = \mathbf{k}$, and any number $\mathbf{c}^0(F) \geq 1$ of fuel molecules. We emphasize that in contrast to the initial context, the protocol designer has no control over the *exact* molecular count of the fuel species in the initial configuration.

For $v \in \{0,1\}$, let $\mathcal{D}_v = \left\{\mathbf{c} \in \mathbb{N}^{\mathcal{S}} \mid \mathbf{c}(\Upsilon_v) > 0 \wedge \mathbf{c}(\Upsilon_{1-v}) = 0\right\}$ be the set of configurations that include v-voters and no $(1-v)$-voters. An input vector $\mathbf{x} \in \mathbb{N}^\Sigma$ is said to be *stably accepted* (resp., *haltingly accepted*) by Π if for every valid initial configuration $\mathbf{c}^0 \in \mathbb{N}^{\mathcal{S}}$ with $\mathbf{c}^0|_\Sigma = \mathbf{x}$, every weakly fair execution $\eta = \langle \mathbf{c}^t, \alpha^t \rangle_{t\geq 0}$ stabilizes (resp., halts) into $\mathcal{D}_1$; the input vector $\mathbf{x} \in \mathbb{N}^\Sigma$ is said to be *stably rejected* (resp., *haltingly rejected*) by Π if the same holds with $\mathcal{D}_0$. The CRD Π is stably (resp., haltingly) correct if every input vector $\mathbf{x} \in \mathbb{N}^\Sigma$ is either stably (resp., haltingly) accepted or stably (resp., haltingly) rejected by Π. In this case, we say that Π *stably decides* (resp., *haltingly decides*) the predicate $\psi : \mathbb{N}^\Sigma \to \{0,1\}$ defined so that $\psi(\mathbf{x}) = 1$ if and only if $\mathbf{x}$ is stably (resp., haltingly) accepted by Π.

By definition, the molecular count of the fuel species F in the initial configuration $\mathbf{c}^0$ does not affect the computation's outcome in terms of whether the execution stabilizes (resp., halts) with 0- or 1-voters. Consequently, one can increase the molecular count $\mathbf{c}^0(F)$ of the fuel species in the initial configuration $\mathbf{c}^0$, thus increasing the initial (total) molecular count $n = \|\mathbf{c}^0\|$ for any given input vector $\mathbf{x} \in \mathbb{N}^\Sigma$. Since the runtime of a CRN is expressed in terms of the initial molecular count n, decoupling $\mathbf{x}$ from n allows us to measure the asymptotic runtime of the protocol while keeping $\mathbf{x}$ fixed. In this regard, the CRD Π is said to be *stabilization speed fault free* (resp., *halting speed fault free*) [10] if for every input vector $\mathbf{x} \in \mathbb{N}^\Sigma$, the family of valid initial configurations $\mathbf{c}^0 \in \mathbb{N}^{\mathcal{S}}$ with $\mathbf{c}^0|_\Sigma = \mathbf{x}$ does not admit a stabilization (resp., halting) speed fault (as defined in Sec. 4.1).

[17] As discussed in [10], a speed fault does not imply an $\Omega(n)$ lower bound on the (stochastic) runtime of stochastically scheduled executions since the probability of reaching a pitfall configuration may be small.

5.1 Semilinear Predicates

A predicate $\psi : \mathbb{N}^\Sigma \to \{0,1\}$ is *linear* if there exist a finite set $\mathcal{A} = \mathcal{A}(\psi) \subset \mathbb{N}^\Sigma$ and a vector $\mathbf{b} = \mathbf{b}(\psi) \in \mathbb{N}^\Sigma$ such that $\psi(\mathbf{x}) = 1$ if and only if $\mathbf{x} = \mathbf{b} + \sum_{\mathbf{a}\in\mathcal{A}} k_\mathbf{a}\mathbf{a}$ for some coefficients $k_\mathbf{a} = k_\mathbf{a}(\mathbf{x}) \in \mathbb{N}$, $\mathbf{a} \in \mathcal{A}$. A predicate $\psi : \mathbb{N}^\Sigma \to \{0,1\}$ is *semilinear* if it is the disjunction of finitely many linear predicates. The following theorem is established in the seminal work of Angluin et al. [2, 4].

▶ **Theorem 7** ([2, 4]). *Fix a predicate $\psi : \mathbb{N}^\Sigma \to \{0,1\}$. If ψ is semilinear, then ψ can be haltingly decided under a strongly fair scheduler by a leaderless CRD. If ψ can be stably decided by a CRD under a strongly fair scheduler, then ψ is semilinear.*

In the full version [13], we extend Thm. 7 to weak fairness which allows us to bound the adversarial runtime of the corresponding CRDs and establish the following theorem; notice that the $O(n)$ runtime bound is asymptotically tight – see the speed fault freeness discussion in Sec. 5.2.

▶ **Theorem 8.** *Fix a predicate $\psi : \mathbb{N}^\Sigma \to \{0,1\}$. If ψ is semilinear, then ψ can be haltingly decided under a weakly fair scheduler by a leaderless CRD whose halting runtime is $O(n)$. If ψ can be stably decided by a CRD under a weakly fair scheduler, then ψ is semilinear.*

5.2 Detection Predicates

For a vector $\mathbf{x} \in \mathbb{N}^\Sigma$, let $\mathbf{x}_\downarrow \in \{0,1\}^\Sigma \subset \mathbb{N}^\Sigma$ be the vector defined so that $\mathbf{x}_\downarrow(A) = 1 \iff \mathbf{x}(A) > 0$. A predicate $\psi : \mathbb{N}^\Sigma \to \{0,1\}$ is a *detection* predicate if $\psi(\mathbf{x}) = \psi(\mathbf{x}_\downarrow)$ for all $\mathbf{x} \in \mathbb{N}^\Sigma$ (cf. [1, 10, 19]). Chen et al. [10] prove that a predicate $\psi : \mathbb{N}^\Sigma \to \{0,1\}$ can be stably decided under the strongly fair adversarial scheduler by a stabilization speed fault free CRD if and only if it is a detection predicate. Cor. 3 ensures that the only if direction translates to our weakly fair adversarial scheduler; employing Lem. 6, we conclude that a non-detection predicate cannot be decided by a CRD whose stabilization runtime is better than $\Omega(n)$. For the if direction, the construction in [10] yields leaderless CRDs that haltingly decide ψ whose expected halting runtime under the stochastic scheduler is $O(\log n)$. The following theorem states that the same (asymptotic) runtime upper bound can be obtained under the weakly fair adversarial scheduler; the theorem is proved in the full version [13], where we also explain why the promised upper bound is asymptotically tight.

▶ **Theorem 9.** *For every detection predicate $\psi : \mathbb{N}^\Sigma \to \{0,1\}$, there exists a leaderless CRD that haltingly decides ψ whose halting runtime is $O(\log n)$. Moreover, the CRD is designed so that all molecules in the halting configuration are voters.*

6 Vote Amplification

Recall that CRDs are required to stabilize/halt into configurations $\mathbf{c}$ that include a positive number of v-voter molecules and zero $(1-v)$-voter molecules, where $v \in \{0,1\}$ is determined by the decided predicate according to the input vector. This requirement alone does not rule out the possibility of having a small (yet positive) voter molecular count in $\mathbf{c}$. Indeed, the semilinear predicate CRDs promised in Thm. 8 are designed so that the configuration $\mathbf{c}$ includes a single voter molecule (this is in contrast to the detection predicate CRDs promised in Thm. 9, where all molecules in $\mathbf{c}$ are voters).

In practice though, it may be difficult to obtain a meaningful signal from small molecular counts. Consequently, we aim for *vote amplified* CRDs, namely, CRDs that guarantee to stabilize/halt into configurations in which the voter molecules take all but an ϵ-fraction of

the total molecular count for an arbitrarily small constant $\epsilon > 0$. These are obtained by means of a "generic compiler" that can be applied, in a black-box manner, to any existing CRD, turning it into a vote amplified CRD while preserving the original stabilization/halting correctness. At the heart of this compiler lies a CRN protocol for a standalone computational task, referred to as *vote amplification (VA)*, whose runtime dominates the runtime overhead of the compiler, as stated in the following theorem (proved in the full version [13]).

▶ **Theorem 10.** *Consider a predicate $\psi : \mathbb{N}^\Sigma \to \{0,1\}$ that can be haltingly decided by a (leaderless) CRD in $T_\psi(n)$ time. The existence of a VA protocol that stabilizes (resp., halts) in $T_{\text{amp}}(n)$ time implies the existence of a (leaderless) vote amplified CRD that stably (resp., haltingly) decides ψ in $T_\psi(O(n)) + T_{\text{amp}}(O(n)) + O(\log n)$ time.*

Assuming a stochastic scheduler, Angluin et al. [3] develop a VA protocol that halts in $O(n)$ time. Unfortunately, the protocol of [3] does not meet the topological conditions of Lem. 1, hence the (weakly fair) adversarial scheduler can prevent this protocol from stabilizing (see the full version [13] for more details). Using a completely different technique, we develop a VA protocol whose guarantees are cast in the following theorem.

▶ **Theorem 11.** *There exists a VA protocol (operating under the weakly fair scheduler) that stabilizes in $O(n)$ time and halts in $O(n \log n)$ time.*

Combined with Thm. 10, we obtain a compiler whose stabilization and halting runtime overheads are $O(n)$ and $O(n \log n)$, respectively. Applying this compiler to the CRDs promised in Thm. 8 results in vote amplified CRDs whose stabilization runtime remains $O(n)$, however their halting runtime increases to $O(n \log n)$. The excessive $\log n$ factor would be shaved by a VA protocol that halts in $O(n)$ time whose existence remains an open question.

Task Formalization. A VA protocol is a CRN protocol $\Pi = (\mathcal{S}, \mathcal{R})$ whose species set $\mathcal{S}$ is partitioned into the pairwise disjoint sets $\mathcal{P}_0 \cup \mathcal{P}_1 \cup \mathcal{F}_0 \cup \mathcal{F}_1 = \mathcal{S}$, where for $v \in \{0,1\}$, the species in $\mathcal{P}_v$ are referred to as *permanent v-voters* and the species in $\mathcal{F}_v$ are referred to as *fluid v-voters*. The permanent voters are regarded as part of the task specification and can participate in the reactions of Π only as catalysts (which means that the molecular count of each permanent voter remains invariant throughout the execution).

A configuration $\mathbf{c}^0 \in \mathbb{N}^\mathcal{S}$ is a valid initial configuration for the VA task if there exists a vote $v \in \{0,1\}$ such that $\mathbf{c}^0(\mathcal{P}_v) > 0$ and $\mathbf{c}^0(\mathcal{P}_{1-v}) = 0$, in which case we refer to $\mathbf{c}^0$ as a *v-voting* initial configuration. A configuration $\mathbf{c} \in \mathbb{N}^\mathcal{S}$ is an *amplification* of a v-voting initial configuration $\mathbf{c}^0$ if (1) $\mathbf{c}(A) = \mathbf{c}^0(A)$ for every $A \in \mathcal{P}_0 \cup \mathcal{P}_1$; (2) $\mathbf{c}(\mathcal{F}_v) = \mathbf{c}^0(\{\mathcal{F}_0 \cup \mathcal{F}_1\})$; and (3) $\mathbf{c}(\mathcal{F}_{1-v}) = 0$. In other words, an amplification of a v-voting initial configuration keeps the original permanent voter molecules and shifts all fluid voter molecules to the v-voting side.

The VA protocol Π is stably (resp., haltingly) correct if every weakly fair valid execution $\eta = \langle \mathbf{c}^t, \alpha^t \rangle_{t \geq 0}$ stabilizes (resp., halts) into the (set of) amplifications of $\mathbf{c}^0$. The typical scenario involves a small number of permanent v-voter molecules and the challenge is to ensure that all fluid voter molecules "end up" in $\mathcal{F}_v$. We emphasize that for Π to be correct, the protocol should handle any initial configuration $\mathbf{c}^0|_{\mathcal{F}_0 \cup \mathcal{F}_1}$ of the fluid voters.

The VA Protocol. We now turn to develop the VA protocol $\Pi = (\mathcal{S}, \mathcal{R})$ promised in Thm. 11. For simplicity, assume in this extended abstract that $\mathcal{P}_0$ and $\mathcal{P}_1$ are singleton sets with $\mathcal{P}_0 = \{P_0\}$ and $\mathcal{P}_1 = \{P_1\}$; the general case is handled in the full version [13]. Protocol Π is defined over the fluid voter sets $\mathcal{F}_0 = \{H_0, L_0\}$ and $\mathcal{F}_1 = \{H_1, L_1\}$. Semantically, we think of the H (resp., L) fluid voters as having a high (resp., low) confidence level in their

vote. The reaction set $\mathcal{R}$ of Π includes the following non-void reactions:
β_v^A: $P_v + A \to P_v + H_v$ for every $v \in \{0,1\}$ and $A \in \{H_{1-v}, L_0, L_1\}$;
γ: $H_0 + H_1 \to L_0 + L_1$; and
δ_v: $H_v + L_{1-v} \to 2L_v$ for every $v \in \{0,1\}$.
Informally, these reactions guarantee that the adversary has little leverage because, as we show soon, *all* of the non-void reactions make nontrivial progress in their own different ways.

For the runtime analysis of protocol Π, consider a weakly fair valid execution $\eta = \langle \mathbf{c}^t, \zeta^t \rangle_{t \geq 0}$ of initial molecular count $\|\mathbf{c}^0\| = n$. Assume for simplicity that the initial configuration $\mathbf{c}^0$ is 1-voting which means that $\mathbf{c}^t(P_1) > 0$ and $\mathbf{c}^t(P_0) = 0$ for all $t \geq 0$; the case where $\mathbf{c}^0$ is 0-voting is analyzed symmetrically. Let $m = \mathbf{c}^0(\{H_0, L_0, L_1, H_1\})$ be the initial molecular count of the fluid voters and observe that $\mathbf{c}^t(\{H_0, L_0, L_1, H_1\}) = m$ for every $t \geq 0$.

To capture progress , we assign an integral score $s(\cdot)$ to each fluid voter by setting $s(H_0) = -4$, $s(L_0) = -1$, $s(L_1) = 1$, and $s(H_1) = 2$. Substituting the $s(\cdot)$ scores into each reaction $\alpha \in \mathrm{NV}(\mathcal{R})$ reveals that the sum of scores of α's fluid reactants is strictly smaller than the sum of scores of α's fluid products. Denoting the total score in a configuration $\mathbf{c} \in \mathbb{N}^{\mathcal{S}}$ by $s(\mathbf{c}) = \sum_{A \in \{H_0, L_0, L_1, H_1\}} \mathbf{c}(A) \cdot s(A)$, we deduce that $s(\mathbf{c}^{t+1}) \geq s(\mathbf{c}^t)$ and that $\zeta^t \in \mathrm{NV}(\mathcal{R}) \implies s(\mathbf{c}^{t+1}) > s(\mathbf{c}^t)$ for every $t \geq 0$. Since $-4m \leq s(\mathbf{c}^t) \leq 2m$ for every $t \geq 0$, it follows that η includes, in total, at most $O(m) \leq O(n)$ non-void reactions until it stabilizes.

The last bound ensures that progress is made on each non-void reaction. Accordingly, we choose the runtime policy ϱ so that $\varrho(\mathbf{c}) = \mathrm{NV}(\mathcal{R})$ for all configurations $\mathbf{c} \in \mathbb{N}^{\mathcal{S}}$.[18]

Fix some skipping policy σ and let $\mathbf{e}^i$ be the effective configuration of round $i \geq 0$ under ϱ and σ. Let $i^* = \min\{i \geq 0 \mid \mathbf{e}^i(\{H_0, L_0\}) = 0\}$ be the first round whose effective step appears after η stabilizes. Since the choice of ϱ ensures that each round $0 \leq i < i^*$ is target-accomplished, ending with a non-void reaction, it follows that $i^* \leq O(n)$.

To bound the stabilization runtime of execution η under ϱ and σ, we argue that $\pi_{\mathbf{e}^i}(\mathrm{NV}(\mathcal{R})) \geq \Omega(1)$ for every $0 \leq i < i^*$; by a simple probabilistic argument (elaborated in the full version [13]), this allows us to conclude that $\mathrm{TC}^{\varrho}(\mathbf{e}^i) \leq O(1)$ for every $0 \leq i < i^*$. To this end, notice that if $\mathbf{e}^i(H_1) \geq m/2$, then

$$\pi_{\mathbf{e}^i}(\{\gamma, \delta_1\}) = \tfrac{1}{\varphi} \cdot \mathbf{e}^i(H_1) \cdot \mathbf{e}^i(\{H_0, L_0\}) \geq \Omega(m/n) = \Omega(1)\,.$$

Otherwise ($\mathbf{e}^i(H_1) < m/2$), we know that $\mathbf{e}^i(\{H_0, L_0, L_1\}) > m/2$, hence

$$\pi_{\mathbf{e}^i}\left(\left\{\beta_1^A \mid A \in \{H_0, L_0, L_1\}\right\}\right) = \tfrac{1}{\varphi} \cdot \mathbf{e}^i(\{H_0, L_0, L_1\}) \cdot \mathbf{e}^i(\mathcal{P}_1) \geq \Omega(m/n) = \Omega(1)\,,$$

thus establishing the argument. Therefore, the stabilization runtime of η satisfies

$$\mathrm{RT}^{\varrho,\sigma}_{\mathrm{stab}}(\eta) = \textstyle\sum_{i=0}^{i^*-1} \mathrm{TC}^{\varrho}(\mathbf{e}^i) \leq \sum_{i=0}^{O(n)} O(1) = O(n)\,.$$

The proof of Thm. 11 is completed by showing that protocol Π halts in $O(n \log n)$ time. This part of the proof is deferred to the full version [13].

References

1 Dan Alistarh, Bartłomiej Dudek, Adrian Kosowski, David Soloveichik, and Przemysław Uznański. Robust detection in leak-prone population protocols. In *DNA Computing and Molecular Programming: 23rd International Conference, DNA 23, Austin, TX, USA, September 24–28, 2017, Proceedings 23*, pages 155–171. Springer, 2017.

[18] Although it serves its purpose in the current analysis, for many CRN protocols, a runtime policy whose targets cover all non-void reactions is suboptimal; this is elaborated in the full version [13].

2 Dana Angluin, James Aspnes, Zoë Diamadi, Michael J. Fischer, and René Peralta. Computation in networks of passively mobile finite-state sensors. *Distributed Comput.*, 18(4):235–253, 2006.

3 Dana Angluin, James Aspnes, and David Eisenstat. Fast computation by population protocols with a leader. *Distributed Comput.*, 21(3):183–199, 2008.

4 Dana Angluin, James Aspnes, David Eisenstat, and Eric Ruppert. The computational power of population protocols. *Distrib. Comput.*, 20(4):279–304, 2007.

5 James Aspnes and Eric Ruppert. An introduction to population protocols. In Benoît Garbinato, Hugo Miranda, and Luís E. T. Rodrigues, editors, *Middleware for Network Eccentric and Mobile Applications*, pages 97–120. Springer, 2009.

6 Baruch Awerbuch. Complexity of network synchronization. *J. ACM*, 32(4):804–823, October 1985. doi:10.1145/4221.4227.

7 Hamid Bolouri and James M. Bower. *Computational Modeling of Genetic and Biochemical Networks*. The MIT Press, February 2001. doi:10.7551/mitpress/2018.001.0001.

8 Robert Brijder. Minimal output unstable configurations in chemical reaction networks and deciders. *Nat. Comput.*, 15(2):235–244, 2016.

9 Robert Brijder. Computing with chemical reaction networks: a tutorial. *Nat. Comput.*, 18(1):119–137, 2019.

10 Ho-Lin Chen, Rachel Cummings, David Doty, and David Soloveichik. Speed faults in computation by chemical reaction networks. *Distributed Comput.*, 30(5):373–390, 2017.

11 Ho-Lin Chen, David Doty, and David Soloveichik. Deterministic function computation with chemical reaction networks. *Nat. Comput.*, 13(4):517–534, 2014.

12 Anne Condon. On design and analysis of chemical reaction network algorithms. In Cezar Câmpeanu, editor, *Implementation and Application of Automata - 23rd International Conference, CIAA 2018, Charlottetown, PE, Canada, July 30 - August 2, 2018, Proceedings*, volume 10977 of *Lecture Notes in Computer Science*, pages 1–3. Springer, 2018.

13 Anne Condon, Yuval Emek, and Noga Harlev. On the runtime of crns beyond idealized conditions, 2023. URL: http://arxiv.org/abs/2307.00647.

14 Matthew Cook, David Soloveichik, Erik Winfree, and Jehoshua Bruck. Programmability of chemical reaction networks. In Anne Condon, David Harel, Joost N. Kok, Arto Salomaa, and Erik Winfree, editors, *Algorithmic Bioprocesses*, Natural Computing Series, pages 543–584. Springer, 2009.

15 Rachel Cummings, David Doty, and David Soloveichik. Probability 1 computation with chemical reaction networks. *Nat. Comput.*, 15(2):245–261, 2016.

16 S. Dolev, A. Israeli, and S. Moran. Uniform dynamic self-stabilizing leader election. *IEEE Transactions on Parallel and Distributed Systems*, 8(4):424–440, 1997. doi:10.1109/71.588622.

17 David Doty. Timing in chemical reaction networks. In Chandra Chekuri, editor, *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 772–784. SIAM, 2014.

18 David Doty and Monir Hajiaghayi. Leaderless deterministic chemical reaction networks. *Nat. Comput.*, 14(2):213–223, 2015.

19 Bartłomiej Dudek and Adrian Kosowski. Universal protocols for information dissemination using emergent signals. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 87–99, 2018.

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

21 Richard M. Karp and Raymond E. Miller. Parallel program schemata. *J. Comput. Syst. Sci.*, 3(2):147–195, May 1969. doi:10.1016/S0022-0000(69)80011-5.

22 Othon Michail, Ioannis Chatzigiannakis, and Paul G. Spirakis. *New Models for Population Protocols*. Synthesis Lectures on Distributed Computing Theory. Morgan & Claypool Publishers, 2011.

23 Grzegorz Rozenberg and Joost Engelfriet. *Elementary net systems*, pages 12–121. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998. doi:10.1007/3-540-65306-6_14.

24 David Soloveichik, Matthew Cook, Erik Winfree, and Jehoshua Bruck. Computation with finite stochastic chemical reaction networks. *Nat. Comput.*, 7(4):615–633, 2008.

25 Marko Vasić, Cameron Chalk, Austin Luchsinger, Sarfraz Khurshid, and David Soloveichik. Programming and training rate-independent chemical reaction networks. *Proceedings of the National Academy of Sciences*, 119(24):e2111552119, 2022.

A FIGURES AND TABLES

Table 1 The (adversarial) runtime complexity landscape of predicate decidability CRN protocols operating under the weakly fair adversarial scheduler. The upper bounds (O-notation) hold with a universal quantifier over the predicate family and an existential quantifier over the CRD family; the lower bounds (Ω-notation) hold with a universal quantifier over both the predicate and CRD families. (As usual, $\Theta(f(n))$ should be interpreted as both $O(f(n))$ and $\Omega(f(n))$.)

predicates	leaderless	amplified vote	stabilization runtime	halting runtime
semilinear (non-detection)	yes	yes	$\Theta(n)$	$\Omega(n)$, $O(n \log n)$
	yes	no	$\Theta(n)$	$\Theta(n)$
	no	yes	$\Theta(n)$	$\Omega(n)$, $O(n \log n)$
	no	no	$\Theta(n)$	$\Theta(n)$
detection	yes	yes	$\Theta(\log n)$	$\Theta(\log n)$
	yes	no	$\Theta(\log n)$	$\Theta(\log n)$
	no	yes	$\Theta(\log n)$	$\Theta(\log n)$
	no	no	$\Theta(\log n)$	$\Theta(\log n)$

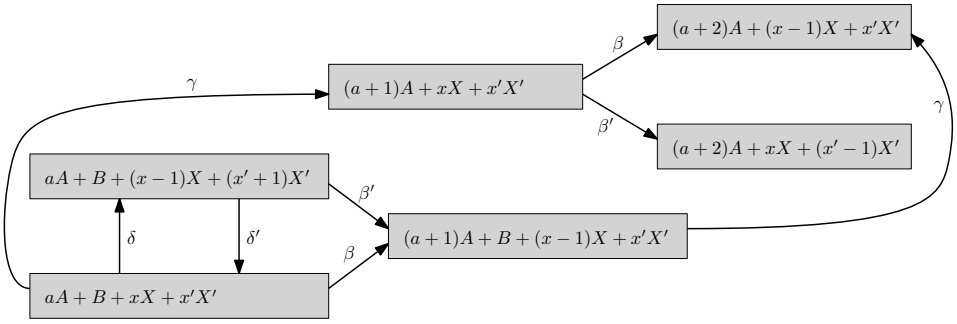
Table 2 The (expected stochastic) runtime complexity landscape of predicate decidability CRN protocols operating under the stochastic scheduler (refer to the full version [13] for details). The upper bounds (O-notation) hold with a universal quantifier over the predicate family and an existential quantifier over the CRD family; the lower bounds (Ω-notation) hold with a universal quantifier over both the predicate and CRD families. (As usual, $\Theta(f(n))$ should be interpreted as both $O(f(n))$ and $\Omega(f(n))$.)

predicates	leaderless	amplified vote	stabilization runtime	halting runtime
semilinear (non-eventually constant)	yes	yes	$\Theta(n)$	$\Theta(n)$
	yes	no	$\Theta(n)$	$\Theta(n)$
	no	yes	$\Omega(\log n)$, $O(n)$	$\Omega(\log n)$, $O(n)$
	no	no	$\Omega(\log n)$, $O(n)$	$\Omega(\log n)$, $O(n)$
eventually constant (non-detection)	yes	yes	$\Omega(\log n)$, $O(n)$	$\Omega(\log n)$, $O(n)$
	yes	no	$\Omega(\log n)$, $O(n)$	$\Omega(\log n)$, $O(n)$
	no	yes	$\Omega(\log n)$, $O(n)$	$\Omega(\log n)$, $O(n)$
	no	no	$\Omega(\log n)$, $O(n)$	$\Omega(\log n)$, $O(n)$
detection	yes	yes	$\Theta(\log n)$	$\Theta(\log n)$
	yes	no	$\Theta(\log n)$	$\Theta(\log n)$
	no	yes	$\Theta(\log n)$	$\Theta(\log n)$
	no	no	$\Theta(\log n)$	$\Theta(\log n)$

$$\mathcal{S} = \{A, B, X, X'\}$$

$$\begin{aligned}
&\beta\colon\ A + X \to 2A\\
&\beta'\colon\ A + X' \to 2A\\
&\gamma\colon\ A + B \to 2A\\
&\delta\colon\ B + X \to B + X'\\
&\delta'\colon\ B + X' \to B + X
\end{aligned}$$

(a) The species and non-void reactions of Π. A configuration $\mathbf{c}^0 \in \mathbb{N}^{\mathcal{S}}$ is valid as an initial configuration of Π if $\mathbf{c}^0(A) = \mathbf{c}^0(B) = 1$. The protocol is designed so that the molecular count of A is non-decreasing, whereas the molecular counts of B and of $\{X, X'\}$ are non-increasing. Moreover, all weakly fair valid executions of Π halt in a configuration that includes only A molecules.



(b) Part of the configuration digraph D^{Π} (excluding void reactions). The configuration set $S = \{aA + B + (x+z)X + (x'-z)X' \mid -x \le z \le x'\}$ forms a component of D^{Π}. Since reactions β and β' are inapplicable in configurations $aA + B + (x+x')X'$ and $aA + B + (x+x')X$, respectively, it follows that these two reactions do not escape from S. Reaction γ on the other hand does escape from S, ensuring that a weakly fair execution cannot remain in S indefinitely. After the B molecule is consumed (by a γ reaction), each configuration constitutes a (singleton) component of D^{Π} and every applicable non-void reaction is escaping.

$$\varrho(\mathbf{c}) = \begin{cases} \{\beta, \beta', \gamma\}, & \mathbf{c}(B) > 0 \\ \{\beta, \beta'\}, & \mathbf{c}(B) = 0 \end{cases}$$

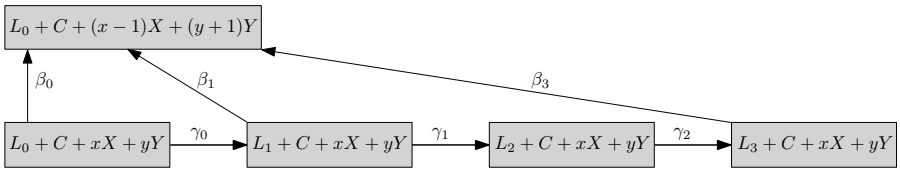
(c) A runtime policy ϱ for Π. Under ϱ, a round with effective configuration $\mathbf{e} = aA + B + xX + x'X'$ is target-accomplished and ends upon scheduling one of the reactions β, β', γ; in particular, it is guaranteed that the next effective configuration $\mathbf{e}'$ satisfies $\mathbf{e}'(A) > a$ (no matter what the adversarial skipping policy is), i.e., progress is made. The adversary may opt to schedule reactions δ and δ' many times before the round ends, however the crux of our runtime definition is that this does not affect the round's temporal cost. Specifically, since $\pi_{\mathbf{c}}(\{\beta, \beta', \gamma\}) = \frac{a(x+x'+1)}{\varphi}$ for every $\mathbf{c} \in S$ (recall the definition of component S from Fig. 1b), it follows, by a simple probabilistic argument, that $\mathrm{TC}^{\varrho}(\mathbf{e}) = \frac{\varphi}{a(x+x'+1)}$. A similar (simpler in fact) argument leads to the conclusion that if $\mathbf{e} = aA + xX + x'X'$, then $\mathrm{TC}^{\varrho}(\mathbf{e}) = \frac{\varphi}{a(x+x')}$. This allows us to show that $\mathrm{RT}^{\Pi}_{\mathrm{halt}}(n) = O(\log n)$.

Figure 1 A CRN protocol $\Pi = (\mathcal{S}, \mathcal{R})$ demonstrating how a carefully chosen runtime policy guarantees significant progress in each round while up-bounding the round's temporal cost.

$\mathcal{S} = \{L_0, L_1, \ldots, L_{k+1}, C, X, Y\}$

β_i: $L_i + X \to L_0 + Y$ for $0 \le i \le k+1$, $i \neq k$
γ_i: $L_i + C \to L_{i+1} + C$ for $0 \le i \le k$

(a) The species and non-void reactions of Π, where k is an arbitrarily large constant. A configuration $\mathbf{c}^0 \in \mathbb{N}^{\mathcal{S}}$ is valid as an initial configuration of Π if $\mathbf{c}^0(L_0) = 1$, $\mathbf{c}^0(C) = 1$, and $\mathbf{c}^0(\{L_1, \ldots, L_{k+1}\}) = 0$. The protocol is designed so that $\mathbf{c}(\{L_0, L_1, \ldots, L_{k+1}\}) = 1$ for any configuration $\mathbf{c}$ reachable from a valid initial configuration (i.e., $L_0, L_1, \ldots, L_{k+1}$ are "leader species"). Species C is a catalyst for any reaction it participates in and $\mathbf{c}(C) = 1$ for any configuration $\mathbf{c}$ reachable from a valid initial configuration. The execution progresses by shifting all X molecules into Y molecules. We are interested in the stabilization of Π's executions into the (set of) configurations $\mathbf{c} \in \mathbb{N}^{\mathcal{S}}$ satisfying $\mathbf{c}(X) < \mathbf{c}(Y)$, although the executions actually halt once $\mathbf{c}(X) = 0$ (and $\mathbf{c}(L_{k+1}) = 1$).



(b) Part of the configuration digraph D^{Π} (excluding void reactions) for $k = 2$. Notice that each configuration constitutes a (singleton) component of D^{Π} and every applicable non-void reaction is escaping.

$$\varrho(\mathbf{c}) = \mathrm{app}(\mathbf{c}) \cap \mathrm{NV}(\mathcal{R})$$

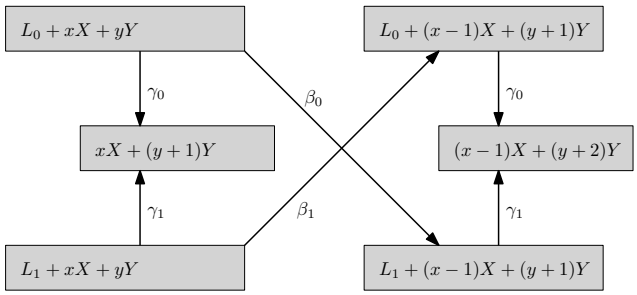
(c) A runtime policy ϱ for Π. Under ϱ, every round ends once any non-void reaction is applied to the round's effective configuration. In the full version [13], we show that $\mathrm{RT}^{\varrho,\sigma}_{\mathrm{stab}}(\eta) \le O(n^2)$ for any skipping policy σ and weakly fair valid execution η of initial molecular count n. It turns out that this bound is tight: The (weakly fair) adversarial scheduler can generate the execution $\eta = \langle \mathbf{c}^t, \alpha^t \rangle$ by scheduling $\alpha^t = \gamma_i$ if $\mathbf{c}^t(L_i) = 1$ for some $0 \le i \le k$; and $\alpha^t = \beta_{k+1}$ if $\mathbf{c}^t(L_{k+1}) = 1$. Using the identity skipping policy σ_{id} and assuming that $\mathbf{c}^0(X) = x_0$ and $\mathbf{c}^0(Y) = 0$, it is easy to show (see the full version [13]) that η visits the configuration $\mathbf{c}_y = L_k + C + (x_0 - y)X + yY$ for every $0 \le y \le x_0/2$ before it stabilizes and that each such visit constitutes the effective configuration of the corresponding round, regardless of the runtime policy ϱ'. Since each such configuration $\mathbf{c}_y$ is a 2-pitfall (recall the definition from Sec. 4.1), we deduce that $\mathrm{TC}^{\varrho'}(\mathbf{c}_y) = \Omega(n)$, which sums up to $\mathrm{RT}^{\varrho',\sigma_{\mathrm{id}}}_{\mathrm{stab}}(\eta) = \Omega(n^2)$. The interesting aspect of protocol Π is that with high probability, a stochastic execution stabilizes without visiting the pitfall configurations $\mathbf{c}_y$ even once, which allows us to conclude that the expected stochastic stabilization runtime of Π is $O(n)$ – see the full version [13] for details.

Figure 2 A CRN protocol $\Pi = (\mathcal{S}, \mathcal{R})$ demonstrating that the adversarial stabilization runtime may be significantly larger than the expected stochastic runtime due to (asymptotically many) pitfall configurations.

$$\mathcal{S} = \{L_0, L_1, X, Y\}$$

$$\begin{aligned}
\beta_0&\colon\ L_0 + Y \to 2Y\\
\beta_1&\colon\ L_1 + Y \to 2Y\\
\gamma_0&\colon\ L_0 + X \to L_1 + Y\\
\gamma_1&\colon\ L_1 + X \to L_0 + Y
\end{aligned}$$

(a) The species and non-void reactions of Π. A configuration $\mathbf{c}^0 \in \mathbb{N}^{\mathcal{S}}$ is valid as an initial configuration of Π if $\mathbf{c}^0(\{L_0, L_1\}) = 1$ and $\mathbf{c}^0(Y) \geq \mathbf{c}^0(X)$. The protocol is designed so that the molecular counts of $\{L_0, L_1\}$ and of X are non-increasing, whereas the molecular count of Y is non-decreasing. Moreover, all weakly fair valid executions of Π halt in a configuration that includes zero L_0 and L_1 molecules.



(b) Part of the configuration digraph D^{Π} (excluding void reactions). Notice that each configuration constitutes a (singleton) component of D^{Π} and every applicable non-void reaction is escaping. Furthermore, if $\mathbf{c} \rightharpoonup \mathbf{c}'$, then $\mathrm{app}(\mathbf{c}) \cap \mathrm{app}(\mathbf{c}') \cap \mathrm{NV}(\mathcal{R}) = \emptyset$.

$$\varrho(\mathbf{c}) = \begin{cases} \beta_0, & \mathbf{c}(L_0) = 1 \\ \beta_1, & \mathbf{c}(L_1) = 1 \end{cases}$$

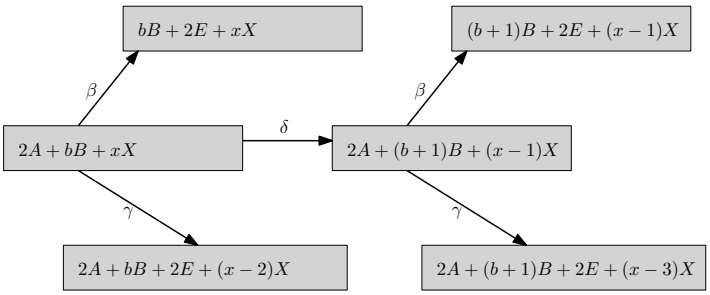
(c) A runtime policy ϱ for Π. Under ϱ, a round whose (non-halting) effective configuration is $\mathbf{e}$ ends once the execution reaches any configuration $\mathbf{c} \neq \mathbf{e}$. This may result from scheduling the target reaction $\varrho(\mathbf{e})$, in which case the round is target-accomplished; otherwise, the round ends because $\varrho(\mathbf{e})$ is inapplicable in $\mathbf{c}$, in which case the round is target-deprived. Since $\pi_{L_i+xX+yY}(\beta_i) = y/\varphi$ for $i \in \{0,1\}$, it follows, by a simple probabilistic argument, that $\mathrm{TC}^{\varrho}(L_i + xX + yY) = \varphi/y$. This allows us to show that $\mathrm{RT}^{\Pi}_{\mathrm{halt}}(n) = O(n)$. In the full version [13], it is shown that the $O(n)$ halting runtime bound is (asymptotically) tight under the (weakly fair) adversarial scheduler; in contrast, the expected stochastic runtime of Π under the stochastic scheduler is only $O(1)$. This gap holds despite the fact that the executions that realize the $\Omega(n)$ runtime lower bound do not reach a pitfall configuration.

Figure 3 A CRN protocol $\Pi = (\mathcal{S}, \mathcal{R})$ demonstrating that the adversarial runtime may be significantly larger than the expected stochastic runtime even though the protocol does not admit a speed fault.

$$\mathcal{S} = \{A, B, E, X\}$$

$$\begin{aligned}
&\beta\colon\ A + A \to 2E \\
&\gamma\colon\ X + X \to 2E \\
&\delta\colon\ B + X \to 2B \\
&\chi_Z\colon\ E + Z \to 2E \qquad \text{for every } Z \in \{A, B, X\}
\end{aligned}$$

(a) The species and non-void reactions of Π. A configuration $\mathbf{c}^0 \in \mathbb{N}^{\mathcal{S}}$ is valid as an initial configuration of Π if $\mathbf{c}^0(A) = 2$, $\mathbf{c}^0(B) = 1$, and $\mathbf{c}^0(E) = 0$. The protocol is designed so that once the first (two) E molecules are produced, species E spreads by means of the χ reactions until it takes over the entire test tube. The first E molecules can be produced either through reaction β, that remains applicable as long as E was not produced, or through reaction γ, that may become inapplicable if reaction δ is scheduled many times. In any case, all weakly fair valid executions of Π halt in a configuration that includes only E molecules.



(b) Part of the configuration digraph D^{Π} (excluding void reactions). Notice that each configuration constitutes a (singleton) component of D^{Π} and every applicable non-void reaction is escaping.

$$\varrho(\mathbf{c}) = \begin{cases} \{\beta, \gamma\}, & \mathbf{c}(E) = 0 \\ \{\chi_A, \chi_B, \chi_X\}, & \mathbf{c}(E) > 0 \end{cases}$$

(c) A runtime policy ϱ for Π. Under ϱ, a round whose (non-halting) effective configuration $\mathbf{e}$ satisfies $\mathbf{e}(E) = 0$ ends once (two) E molecules are produced, either through reaction β or through reaction γ (either way, the round is target accomplished); in particular, there can be at most one such round, that is, the first round of the execution. To maximize the temporal cost charged for this round, the adversarial scheduler devises (the execution and) the skipping policy σ so that $\mathbf{e}(X) < 2$ which means that reaction γ is inapplicable in $\mathbf{e}$ (this requires that σ generates a "large skip"). Such a configuration $\mathbf{e}$ is a halting 2-pitfall as reaction β must be scheduled in order to advance the execution and the propensity of β is $2/\varphi$. We conclude, by a simple probabilistic argument, that $\mathrm{TC}^{\varrho}(\mathbf{e}) = \varphi/2$. A round whose (non-halting) effective configuration $\mathbf{e}$ satisfies $\mathbf{e}(E) > 0$ ends once a χ configuration is scheduled, thus ensuring that the execution's next effective configuration $\mathbf{e}'$ satisfies $\mathbf{e}'(E) > \mathbf{e}(E)$. Using standard arguments, one can prove that the total contribution of (all) these rounds to the halting runtime of an execution η with initial molecular count n is up-bounded by $O(\log n)$. Therefore, together with the contribution of the first "slow" round, we get $\mathrm{RT}^{\varrho,\sigma}_{\mathrm{halt}}(\eta) = \Theta(n)$. Note that the same runtime policy ϱ leads to a much better halting runtime of $\mathrm{RT}^{\varrho,\sigma_{\mathrm{id}}}_{\mathrm{halt}}(\eta) = \Theta(\log n)$ if the adversarial scheduler opts to use the identity skipping policy σ_{id} instead of the aforementioned skipping policy σ. As a direct consequence of Lem. 6, we conclude that ϱ is asymptotically optimal for Π, hence $\mathrm{RT}^{\Pi}_{\mathrm{halt}}(n) = \Theta(n)$.

Figure 4 A CRN protocol $\Pi = (\mathcal{S}, \mathcal{R})$ demonstrating that a non-trivial skipping policy results in a significantly larger runtime, compared to the identity skipping policy.