Natural Computing
https://doi.org/10.1007/s11047-021-09877-9

Population-induced phase transitions and the verification of chemical reaction networks

James I. Lathrop[1] · Jack H. Lutz[1] · Robyn R. Lutz[1] · Hugh D. Potter[1] · Matthew R. Riley[1]

Accepted: 16 October 2021

Abstract
We show that very simple molecular systems, modeled as chemical reaction networks, can have behaviors that exhibit dramatic phase transitions at certain population thresholds. Moreover, the magnitudes of these thresholds can thwart attempts to use simulation, model checking, or approximation by differential equations to formally verify the behaviors of such systems at realistic populations. We show how formal theorem provers can successfully verify some such systems at populations where other verification methods fail.

Keywords Chemical reaction networks · Molecular programming · Phase transitions · Population protocols · Verification

1 Introduction

Chemical reaction networks, mathematical abstractions similar to Petri nets, are used as a programming language to specify the dynamic behaviors of engineered molecular systems. Existing software can compile chemical reaction networks into DNA strand displacement systems that simulate them with growing generality and precision (Soloveichik et al. 2009; Chen et al. 2013; Badelt et al. 2017; Thubagere et al. 2017). Programming is a challenging discipline in any case, but this is especially true of molecular programming, because chemical reaction networks—in addition to being Turing universal (Soloveichik et al. 2008; Cook et al. 2009; Fages et al. 2017) and hence subject to all the uncomputable aspects of sequential, imperative programs–are, like the systems that they specify, distributed, asynchronous, and probabilistic. Since many envisioned applications of molecular programming will be safety critical (Wooley John and Lin Herbert 2005; Zhang and Seelig 2011; Douglas et al. 2012; Liu et al. 2013; Li et al. 2018; Apoorva et al. 2020, 20), programmers thus seek to create chemical reaction networks that can be *verified* to correctly carry out their design intent.

One principle that is sometimes used in chemical reaction network design is the *small population heuristic* (Lakin et al. 2012; Cardelli et al. 2016; Ellis et al. 2019). The idea here is to verify various stages of a design by model checking or software simulation to ferret out bugs in the design prior to laboratory experimentation or deployment. Since the number of states of a molecular system is typically much larger than its population (the number of molecules present), and since molecular systems typically have very large populations, this model checking or simulation can usually only be carried out on populations that are far smaller than those of the intended molecular systems. It is nevertheless reasonable to hope that, if a system is going to consist of a very large number of "devices" of various sorts, then any unforeseen errors in these devices' interactions will manifest themselves even with very small populations of each device. It is this reasonable hope that is the underlying premise of the small population heuristic. (Note that the small population heuristic can be regarded as

✉ Jack H. Lutz
lutz@iastate.edu

James I. Lathrop
jil@iastate.edu

Robyn R. Lutz
rlutz@iastate.edu

Hugh D. Potter
hdpotter@iastate.edu

Matthew R. Riley
mrriley@iastate.edu

[1] Iowa State University, Ames, IA, USA

Published online: 17 November 2021

a molecular version of the small scope hypothesis (Jackson 2019).

The question that we address here is whether real molecular systems can thwart the small population heuristic. That is, can a real molecular system behave very differently at large populations than at small populations? If so, *how sensitive* can its behavior be to its population, and *how simple a mechanism* can achieve such sensitivity?

In order to ensure that we are only investigating population effects, we focus our attention on chemical reaction networks that are *population protocols* in the sense that their populations remain constant throughout their operations. If we have such a chemical reaction network, and if we vary its initial population *and nothing else*, then we are assured that any resulting variations of behavior are due solely to the differing populations.

In this paper we show that very simple chemical reaction networks can be very sensitive to their own populations. In fact, they can exhibit *population-induced phase transitions*, behaving one way below a threshold population and behaving very differently above that threshold. After reviewing chemical reaction networks in Sect. 2, we present in Sect. 3 a chemical reaction network $\mathbf{N}_1$, and we prove that $\mathbf{N}_1$ exhibits a population-induced phase transition in the following sense. There are two parameters, m and n, in the construction. For this discussion, we may take $m = 34$ and $n = 67$, but the construction is general. There are $n + 2$ reactions among $n + 3$ species (molecule types) in $\mathbf{N}_1$. A species Z_0 is given an initial population of p, and all other species counts are initially 0. Each reaction of $\mathbf{N}_1$ has two reactants and two products, so the total population of $\mathbf{N}_1$ is p at all times. There are in $\mathbf{N}_1$ two distinguished species, B and R. These "blue" and "red" species are abstract stand-ins for two different behaviors of $\mathbf{N}_1$. Our construction exploits the inherent nonlinearity of chemical kinetics to ensure that, if $p < 2^m$, then $\mathbf{N}_1$ terminates with essentially all its population blue, while if $p \geq 2^m$, then $\mathbf{N}_1$ terminates with essentially all its population red. Thus $\mathbf{N}_1$ exhibits a sharp phase transition at the population threshold $p = 2^m$.

Our construction is very simple. The chemical reaction network $\mathbf{N}_1$ changes its behavior at the threshold $p = 2^m$ by merely computing successive bits of p, starting at the least significant bit. This mechanism is so simple that it could be hidden, by accident or by malice, in a larger chemical reaction network. Moreover, for suitable values of m (e.g., $m = 34$, so that the threshold $p = 2^m$ is roughly 1.7×10^{10}),

1. any attempt to model-check or simulate $\mathbf{N}_1$ will perforce use a population much less than the threshold and conclude that $\mathbf{N}_1$ will always turn blue; while
2. any realistic wet-lab molecular implementation of $\mathbf{N}_1$ will have a population greater than the threshold and thus turn red.

If the behaviors represented by blue and red here are a desired, "good" behavior of $\mathbf{N}_1$ (or of a network containing $\mathbf{N}_1$) and an undesired, "bad" behavior of this network, respectively, then the possibility of such a phase transition is a serious challenge to verifying the correct behavior of the chemical reaction network. Simply put, this is a context in which the small population heuristic can lead us astray.

There is a dual *large population heuristic* that is used even more often than the small population heuristic. A theorem of Kurtz (Kurtz 1972; Anderson and Kurtz 2011, 2015) draws a connection between the behavior of a *stochastic* chemical reaction network (the type of chemical reaction network used in our work here and in much of molecular programming) at large populations and the behavior of a deterministic chemical reaction network, which is governed by a system of ordinary differential equations. Kurtz's theorem involves several preconditions and caveats, and it does not always transparently equate stochastic and deterministic behavior. When it does apply, however, we can use a mathematical software package to numerically solve the deterministic system and thereby understand the behavior of the stochastic chemical reaction network at sufficiently large populations.

In Sect. 4 we add a single reaction to the chemical reaction network $\mathbf{N}_1$, creating a chemical reaction network $\mathbf{N}_2$ that we prove (in Theorem 4.6) to exhibit two *coupled population-induced phase transitions* in the following sense. If $p < 2^m$ or $p \geq 2^n$, then $\mathbf{N}_2$ terminates with essentially all its population blue, while if $2^m \leq p < 2^n$, then $\mathbf{N}_2$ terminates with essentially all its population red. Thus $\mathbf{N}_2$ exhibits sharp phase transitions at the two population thresholds, $p = 2^m$ and $p = 2^n$. These phase transitions are *coupled* in that exceeding the second threshold returns the behavior of $\mathbf{N}_2$ to its behavior below the first threshold. For suitable values of m and n (e.g. $m = 34$ and $n = 67$ as above, so that the thresholds $p = 2^m$ and $p = 2^n$ are roughly 1.7×10^{10} and 1.5×10^{20}), this implies (see Fig. 1) that

1. any attempt to model-check or simulate $\mathbf{N}_2$ will perforce use a population much less than the smaller threshold and conclude that $\mathbf{N}_2$ will always turn blue, and
2. any realistic wet-lab molecular implementation of $\mathbf{N}_2$ will have a population between the two thresholds and thus turn red.

As we discuss later, when we analyze $\mathbf{N}_2$ with a numerical approach based on differential equations, we also do not observe a red outcome. The chemical reaction network $\mathbf{N}_2$

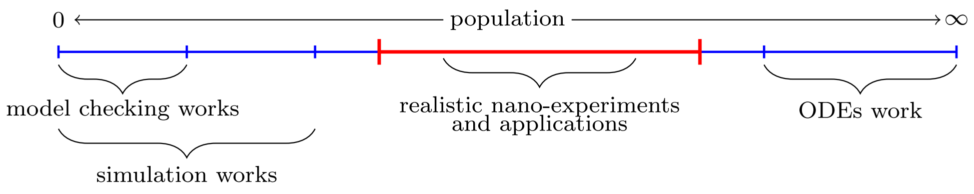


Fig. 1 Scales at which different verification methods (simulation, model checking, and ODEs) work. The gap in the middle shows the scale at which none of these methods will catch the "produce red" behavior of the system design. This gap is problematic because it is the scale of realistic programmed molecular systems. We show in Sect. 6.4 how such systems can be verified using automated theorem proving

thus exemplifies a class of contexts in which the small population heuristic and the large population heuristic can both lead us astray.

The chemical reaction network $\mathbf{N}_2$ is a very simple model of coupled phase transitions with a simple correctness proof. However, as we explain in Sect. 5, $\mathbf{N}_2$ typically takes a *very* long time to terminate with initial populations p above the 2^n threshold. If one tries to simulate $\mathbf{N}_2$ with such initializations, even for small values of m and n, the simulations fail to terminate in a feasible amount of time. This raises the question whether coupled transitions *require* such a long time to termination.

In Sect. 5 we present a chemical reaction network $\mathbf{N}_3$ that, like $\mathbf{N}_2$, has coupled phase transitions at initial populations $p = 2^m$ and $p = 2^n$. However, $\mathbf{N}_3$ reaches termination more quickly than $\mathbf{N}_2$ for initial populations $p \geq 2^n$. The correctness proof for $\mathbf{N}_3$ is not as elegant as the one for $\mathbf{N}_2$, but we have verified it in Isabelle. Hence coupled phase transitions with faster termination can occur.

We emphasize that the phase transitions in the chemical reaction networks $\mathbf{N}_1$, $\mathbf{N}_2$, and $\mathbf{N}_3$ occur at thresholds in their *absolute populations*. In contrast, phase transitions in chemical reaction networks for approximate majority (Angluin et al. 2008; Cardelli and Csikász-Nagy 2012; Condon et al. 2017) occur at threshold *ratios between subpopulations*, and phase transitions in bacterial quorum sensing (Miller and Bassler 2001) occur at threshold *population densities*.

Section 6 discusses the consequences of our results for the verification of programmed molecular systems in some detail. Here we summarize these consequences briefly. Phase transitions are ubiquitous in natural and engineered systems (Dana 2010; Randall 2017; Cannon et al. 2018). Our results are thus cautionary, but they should not be daunting. Fifteen years after Turing proved the undecidability of the halting problem, Rice (1951, 1953) proved his famous generalization stating that *every* nontrivial input/output property of programs is undecidable. Rice's theorem saves valuable time, but it has never prevented computer scientists from developing specific programs in disciplined ways that enable them to be verified. Similarly, Sects. 3 and 4 give mathematical *proofs* that the chemical reaction networks $\mathbf{N}_1$ and $\mathbf{N}_2$ have the properties described above, and Sect. 6 describes how we have implemented such proofs in the Isabelle proof assistant (Nipkow et al. 2002; Paulson et al. 2019). As molecular programming develops, simulators, model checkers, theorem provers, and other tools will evolve with it, as will disciplined scientific judgment about how and when to use such tools.

A preliminary version of a portion of this work was presented at the 26th International Conference on DNA Computing and Molecular Programming (September, 2020) and published (Lathrop et al. 2020) in the proceedings of that conference. Material that is new to the present paper includes proofs of the lemmas and corollaries in Sects. 3 and 4; the completely new Sect. 5 presenting a chemical reaction network with coupled phase transitions and faster termination than the chemical reaction network in Sect. 4; and, in Sect. 6, verification issues for this faster chemical reaction network and an expanded discussion of verification using differential equations.

2 Chemical reaction networks

Chemical reaction networks (CRNs) are abstract models of molecular processes in well-mixed solutions. They are roughly equivalent to three models used in distributed computing, namely, Petri nets, population protocols, and vector addition systems (Cook et al. 2009). This paper uses stochastic chemical reaction networks.

For our purposes, a (*stochastic*) *chemical reaction network* $\mathbf{N}$ consists of finitely many *reactions*, each of which has the form

$$A + B \rightarrow C + D, \tag{2.1}$$

where A, B, C, and D (not necessarily distinct) are *species*, i.e., abstract types of molecules. Intuitively, if this reaction occurs in a solution at some time, then one A and one B disappear from the solution and are replaced by one C and one D, these things happening instantaneously. A *state* of the chemical reaction network $\mathbf{N}$ with species $A_1, \ldots, A_s$ at a particular moment of time is the vector $(a_1, \ldots, a_s)$, where each a_i is the nonnegative integer count of the molecules of species A_i in solution at that moment. Note that we are using the so called "lower-case convention" for denoting species counts.

In the full stochastic chemical reaction network model, each reaction also has a positive real *rate constant*, and the random behavior of $\mathbf{N}$ obeys a continuous-time Markov chain derived from these rate constants. However, our results here are so robust that they hold for *any* assignment of rate constants, so we need not concern ourselves with rate constants or continuous-time Markov chains. In fact, for this paper, we can consider the reaction (2.1) to be the if-statement

$$\begin{aligned}&\text{if } a > 0 \text{ and } b > 0 \text{ then } a, b, c, d := \\ &a - 1, b - 1, c + 1, d + 1,\end{aligned} \tag{2.2}$$

where ":=" is parallel assignment. The reaction (2.1) is *enabled* in a state q of $\mathbf{N}$ if $a > 0$ and $b > 0$ in q; otherwise, this reaction is *disabled* in q. A state q of $\mathbf{N}$ is *terminal* if no reaction is enabled in q.

A *trajectory* of a chemical reaction network $\mathbf{N}$ is a sequence $\tau = (q_i \mid 0 \leq i < \ell)$ of states of $\mathbf{N}$, where $\ell \in \mathbb{Z}^+ \cup \{\infty\}$ is the *length* of τ and, for each $i \in \mathbb{N}$ with $i + 1 < \ell$, there is a reaction of $\mathbf{N}$ that is enabled in q_i and whose effect, as defined by (2.2), is to change the state of $\mathbf{N}$ from q_i to q_{i+1}. A trajectory $\tau = (q_i \mid 0 \leq i < \ell)$ is *terminal* if $\ell < \infty$ and $q_{\ell-1}$ is a terminal state of $\mathbf{N}$.

Assume for this paragraph that the context specifies an initial state q_0 of $\mathbf{N}$, as it does in this paper. A state q of $\mathbf{N}$ is *accessible* if there is a finite trajectory $\tau = (q_i \mid 0 \leq i < \ell)$ of $\mathbf{N}$ with $q_{\ell-1} = q$. A *full trajectory* of $\mathbf{N}$ is a trajectory $\tau = (q_i \mid 0 \leq i < \ell)$ that is either terminal or infinite.

The fact that each reaction (2.1) has two *reactants* (A and B) and two *products* (C and D) means that $\mathbf{N}$ is a *population protocol* (Angluin et al. 2007). This condition implies that the total population of all species never changes in the course of a trajectory. If such a chemical reaction network has s species and initial population p, its state space is thus the $(s-1)$-dimensional integer simplex

$$\Delta^{s-1}(p) = \left\{ (a_1, \ldots, a_s) \in \mathbb{N}^s \;\middle|\; \sum_{i=1}^{s} a_i = p \right\}. \tag{2.3}$$

Note that $|\Delta^{s-1}(p)| = \binom{p+s-1}{s-1}$. Of course, fewer than this many states may be reachable from a particular initial state of $\mathbf{N}$.

A full trajectory $\tau = (q_i \mid 0 \leq i < \ell)$ of a CRN $\mathbf{N}$ is (*strongly*) *fair* (Kwiatkowska 1989; Baier and Katoen 2008) if it has the property that, for every state q and reaction ρ that is enabled in q,

$$(\exists^\infty i) q_i = q \Rightarrow (\exists^\infty j)[q_j = q \text{ and } \rho \text{ occurs at } j \text{ in } \tau], \tag{2.4}$$

where $(\exists^\infty i)$ means "there exist infinitely many i such that." Note that every terminal trajectory of $\mathbf{N}$ is vacuously fair, because it does not satisfy the hypothesis of (2.4).

The stochastic kinetics of chemical reaction networks implies that, regardless of the rate constants of the reactions, for every population protocol $\mathbf{N}$ and every initial population p of $\mathbf{N}$, there is a real number $\varepsilon > 0$ such that, for every state q of $\mathbf{N}$ and reaction ρ that is enabled in q, the probability that ρ occurs in q depends only on q and is at least ε. This in turn implies that, with probability 1, $\mathbf{N}$ follows a fair trajectory. Hence, if $\mathbf{N}$ has a given behavior on all fair trajectories, then $\mathbf{N}$ has that behavior with probability 1.

We use the following two facts in Sect. 4. The first is an obvious consequence of the definition of fairness.

Observation 2.1 If $\tau = (q_i \mid 0 \leq i < \ell)$ is a fair trajectory of a population protocol $\mathbf{N}$, then, for every reaction ρ of $\mathbf{N}$,

$$(\exists^\infty i)[\rho \text{ is enabled in } q_i] \Rightarrow (\exists^\infty j)[\rho \text{ occurs at } j \text{ in } \tau]. \tag{2.5}$$

A famous theorem of Harel (1986; Kozen 2006) implies that the general problem of deciding whether a chemical reaction network terminates on all fair trajectories is undecidable. Nevertheless, the following lemma gives a useful sufficient condition for termination on all fair trajectories. This lemma undoubtedly follows from a very old result on fairness, but we do not know a proper reference at the time of this writing. A proof appears in the Appendix.

Lemma 2.2 (*fair termination lemma*) *If a population protocol with a specified initial state has a terminal trajectory from every accessible state, then all its fair trajectories are terminal.*

3 Single phase transition

This section presents the chemical reaction network $\mathbf{N}_1$ and proves that it exhibits a population-induced phase transition as described in the introduction.

Fix $m, n, p \in \mathbb{Z}^+$ with $n > m + 1$. Let $\mathbf{N}_1$ be a chemical reaction network consisting of the $n + 1$ ζ-*reactions*

$$\zeta_i \equiv Z_i + Z_i \to \begin{cases} Z_{i+1} + B & (0 \leq i < m) \\ Z_{i+1} + R & (m \leq i < n) \\ Z_i + R & (i = n) \end{cases}$$

and the χ-*reaction*

$$\chi \equiv B + R \to R + R.$$

All results here hold regardless of the rate constants of these $n+2$ reactions.

We initialize $\mathbf{N}_1$ with $z_0 = p$ and all other counts 0.

Intuitively, B is *blue*, R is *red*, and the species Z_i are all *colorless*.

Lemma 3.1 $\mathbf{N}_1$ *terminates on all possible trajectories.*

Proof Every reaction in $\mathbf{N}_1$ reduces the rank

$$3\sum_{i=0}^{n} z_i + 2b + r.$$

□

Notation For $1 \le k \le n+1$, let

$$S_k = \sum_{i=0}^{k-1} 2^i z_i,$$

noting that this quantity depends on the state of $\mathbf{N}_1$.

Lemma 3.2 *Let $0 \le j \le n$ and $1 \le k \le n+1$.*

1. If $j \ne k-1$, then the reaction ζ_j preserves the value of S_k.
2. If $j = k-1$, then the reaction ζ_j reduces the value of S_k.

Proof Let $0 \le j \le n$ and $1 \le k \le n+1$.

1. This holds trivially if $k \le j \le n$, so assume that $0 \le j < k-1$. Let $z_0, \ldots, z_{k-1}$ be the counts of $Z_0, \ldots, Z_{k-1}$ just before an occurrence of ζ_j, let $z'_0, \ldots, z'_{k-1}$ be the counts just after, and let $S'_k = \sum_{i=0}^{k-1} 2^i z'_i$. Since ζ_j occurs, we must have $z_j \ge 2$. If we let $I = \{0, \ldots, k-1\} \setminus \{j, j+1\}$, then each
$$z'_i = \begin{cases} z_i & \text{if } i \in I \\ z_i - 2 & \text{if } i = j \\ z_i + 1 & \text{if } i = j+1, \end{cases}$$
so we have
$$S'_k = \sum_{i\in I} 2^i z_i + 2^j(z_j - 2) + 2^{j+1}(z_{j+1}+1) = S_k.$$
2. Assume that $j = k-1$, and use the notations $z_0, \ldots, z_{k-1}$, $z'_0, \ldots, z'_{k-1}$, and S'_k as in part 3 of this proof. If $1 \le k \le n$, then each
$$z'_i = \begin{cases} z_i & \text{if } 0 \le i < k-1 \\ z_i - 2 & \text{if } i = k-1. \end{cases}$$
If $k = n+1$, then each
$$z'_i = \begin{cases} z_i & \text{if } 0 \le i < k-1 \\ z_i - 1 & \text{if } i = k-1. \end{cases}$$
Either way, $S'_k < S_k$.

□

Corollary 3.3 *For every $1 \le k \le n+1$, the inequality $S_k \le p$ is an invariant of $\mathbf{N}_1$.*

Proof Let $1 \le k \le n+1$. We have $S_k = p$ in the initial state that we have specified. The χ-reaction trivially preserves the value of S_k and Lemma 3.2 tells us that the reactions $\zeta_0, \ldots, \zeta_n$ all preserve the condition $S_k \le p$. □

Corollary 3.4 *If $1 \le k \le n$ and $z_k > 0$ in some accessible state of $\mathbf{N}_1$, then $p \ge 2^k$.*

Proof Assume the hypothesis. Then Corollary 3.3 tells us that, in the given accessible state,

$$p \ge S_{k+1} \ge 2^k z_k \ge 2^k.$$

□

In the following, for $d \in \mathbb{Z}^+$, we use both the mod-d *congruence* (equivalence relation)

$$a \equiv b \bmod d,$$

which asserts of integers $a, b \in \mathbf{Z}$ that $b - a$ is divisible by d, and the **mod**-d *operation*

$$b \ \mathbf{mod}\ d$$

whose value, for $b \in \mathbb{Z}$, is the unique $r \in \mathbb{Z}$ such that $0 \le r < d$ and $r \equiv b \bmod d$.

Corollary 3.5 *The congruence*

$$S_n \equiv p \bmod 2^n \tag{3.1}$$

is an invariant of $\mathbf{N}_1$.

Proof The initialization of $\mathbf{N}_1$ ensures that (3.1) holds in the initial state. It is clear that the reactions ζ_n and χ preserve the value of S_n, and Lemma 3.2 tells us that the reactions $\zeta_1, \ldots, \zeta_{n-2}$ preserve the value of S_n. Hence it suffices to show that the reaction ζ_{n-1} preserves the truth of (3.1).

Assume that (3.1) holds just prior to the occurrence of ζ_{n-1}. Let $z_0, \ldots, z_{n-1}$ be the counts of $Z_0, \ldots, Z_{n-1}$ just before this occurrence, let $z'_0, \ldots, z'_{n-1}$ be the counts just after this occurrence, and let $S'_n = \sum_{i=0}^{n-1} 2^i z'_i$. Since ζ_{n-1} occurs, we must have $z_{n-1} \ge 2$. For each $0 \le i < n$ we have

$$z'_n = \begin{cases} z_i & \text{if } 0 \le i < n-1 \\ z_i - 2 & \text{if } i = n-1, \end{cases}$$

so

$$\begin{aligned} S'_n &= \sum_{i=0}^{n-2} 2^i z_i + 2^{n-1}(z_{n-1} - 2) \\ &= S_n - 2^n \\ &\equiv S_n \bmod 2^n. \end{aligned}$$

Since $S_n \equiv p \bmod 2^n$, it follows that $S'_n \equiv p \bmod 2^n$, i.e., that (3.1) holds just after this occurrence of ζ_{n-1}. □

Corollary 3.6 *For every* $1 \leq k \leq n$, *the condition*

$$\Theta_k \equiv z_k = \cdots = z_n = 0 \Rightarrow S_k = p$$

is an invariant of $\mathbf{N}_1$.

Proof Let $1 \leq k \leq n$. The condition Θ_k holds trivially in the initial state that we have specified. The reactions ζ_n and χ trivially preserve the value of S_k, so let $0 \leq j < n$. It suffices to show that ζ_j preserves the condition Θ_k. For this, assume that Θ_k holds just prior to an occurrence of ζ_j. Let $z_0, \ldots, z_n$ be the counts of $Z_0, \ldots, Z_n$ just prior to this occurrence of ζ_j, and let $z'_0, \ldots, z'_n$ be the counts just after this occurrence. Since ζ_j occurs, we must have $z_j \geq 2$ and $z'_{j+1} \geq 1$. To see that Θ_k holds just after this occurrence of ζ_j, assume that $z'_k = \cdots = z'_n = 0$. Then $z_k = \cdots = z_n = 0$ and $j+1 < k$, so Lemma 3.2 tells us that ζ_j preserves the value of S_k. Hence Θ_k holds just after this occurrence of ζ_j. □

Corollary 3.7 *Let* $(q_0, \ldots, q_t)$ *be a trajectory of* $\mathbf{N}_1$, *where* q_t *is a terminal state, and let* $1 \leq k \leq n$. *If* $p \geq 2^k$, *then there exists* $1 \leq s \leq t$ *such that* $z_k > 0$ *in* q_s.

Proof Let $(q_0, \ldots, q_t)$ be a trajectory of $\mathbf{N}_1$, and let $1 \leq k \leq n$. Assume that $p \geq 2^k$ and that there does not exist $s \in \{1, \ldots, t\}$ such that $z_k > 0$. It suffices to show that q_t is not terminal.

Since there does not exist $s \in \{1, \ldots, t\}$ such that $z_k > 0$ in q_s, it must be the case that $z_k = \cdots = z_n = 0$ in q_t. It follows by Corollary 3.6 that $S_k = p$ holds in q_t. Since $\sum_{i=0}^{k-1} 2^i = 2^k - 1$ and we have $p \geq 2^k$ by assumption, this implies that there exists $0 \leq i < k$ such that $z_i > 1$ in q_t. Hence the reaction ζ_i is enabled in q_t, so q_t is not terminal. □

Notation For each $r \in \{0, \ldots, 2^n - 1\}$, let $\lambda(r)$ be the number of 1s in the n-bit binary representation of r (leading 0s allowed), and let

$$\varepsilon = \begin{cases} \lambda(p) & \text{if } p < 2^n \\ 1 + \lambda(p \textbf{ mod } 2^n) & \text{if } p \geq 2^n. \end{cases}$$

Note that ε is an integer depending on n and p, and that ε is negligible in the sense that $\varepsilon = o(p)$ as $p \to \infty$.

The Boolean value of a condition φ is $[\![\varphi]\!] = \textbf{if } \varphi \textbf{ then } 1 \textbf{ else } 0$.

Theorem 3.8 $\mathbf{N}_1$ *terminates on all trajectories in the state* $(z_0, \ldots, z_n, b, r)$ *specified as follows.*

(i) $z_{n-1} \cdots z_0$ *is the n-bit binary expansion of* $p \textbf{ mod } 2^n$.

(ii) $z_n = [\![p \geq 2^n]\!]$.

(iii) $b = (p - \varepsilon) \cdot [\![p < 2^m]\!]$.

(iv) $r = (p - \varepsilon) \cdot [\![p \geq 2^m]\!]$.

Proof Lemma 3.1 tells us that $\mathbf{N}_1$ terminates on all trajectories. Let $q = (z_0, \ldots, z_n, b, r)$ be a terminal state of $\mathbf{N}_1$, and note the following.

(a) For all $0 \leq i \leq n$, ζ_i is not enabled in q, so $z_i \in \{0, 1\}$.

(b) χ is not enabled in q, so $b = 0$ or $r = 0$.

(c) By (a), $S_n \leq \sum_{i=0}^{n-1} 2^i = 2^n - 1$, so Corollary 3.5 tells us that $S_n = p \textbf{ mod } 2^n$, i.e., that *(i)* holds.

(d) If $p < 2^n$, then Corollary 3.4 tells us that $z_n = 0$. If $p \geq 2^n$, then Corollary 3.7 tells us that $z_n \geq 1$ somewhere along every trajectory leading to q. Since z_n can never become 0 after becoming positive, this implies that $z_n = 1$ in q. Hence *(ii)* holds.

(e) By (c) and (d) we have $\sum_{i=0}^{n} z_i = \varepsilon$.

(f) Since $b + r + \sum_{i=0}^{n} z_i$ is an invariant of $\mathbf{N}_1$, (b) and (e) tell us that one of b and r is $p - \varepsilon$ and the other is 0.

(g) If $p < 2^m$, then Corollary 3.4 tells us that $z_m = \cdots = z_n = 0$ holds throughout every trajectory leading to q. This implies that none of the reactions $\zeta_m, \ldots, \zeta_n$ occurs along any trajectory leading to q, whence $r = 0$.

(h) If $p \geq 2^m$, then Corollary 3.7 tells us that $z_m > 0$ holds somewhere along every trajectory leading to q. This implies that the reaction ζ_{m-1} occurs, whence r becomes positive, somewhere along every trajectory leading to q. Since r can never become 0 after becoming positive, this implies that $r > 0$.

(i) By (f), (g), and (h), *(iii)* and *(iv)* hold. □

Since ε is negligible with respect to p, Theorem 3.8 says that $\mathbf{N}_1$ terminates in an overwhelmingly blue state if $p < 2^m$ and in an overwhelmingly red state if $p \geq 2^m$. This is a very sharp phase transition at the population threshold 2^m.

4 Coupled phase transitions

Let m, n, p, and $\mathbf{N_1}$ be as in Sect. 3, and let $\mathbf{N_2}$ be a CRN consisting of the $n+2$ reactions of $\mathbf{N_1}$ and the ω-*reaction*

$$\omega \equiv R + Z_n \to B + Z_n.$$

This section proves that $\mathbf{N_2}$ exhibits two coupled population-induced phase transitions as described in the introduction. That is, exceeding the second threshold returns the behavior of $\mathbf{N_2}$ to its behavior below the first threshold.

We use the same initialization for $\mathbf{N_2}$ as for $\mathbf{N_1}$. Again, all our results hold regardless of the rate constants of the $n+3$ reactions of $\mathbf{N_2}$.

Routine inspection verifies the following.

Observation 4.1 Lemma 3.2 and Corollaries 3.3-3.7 hold for $\mathbf{N_2}$ as well as for $\mathbf{N_1}$.

If $p < 2^n$, then Corollary 3.4 tells us that z_n never becomes positive in $\mathbf{N_2}$, so the ω-reaction never occurs in $\mathbf{N_2}$. Thus, for $p < 2^n$, $\mathbf{N_2}$ behaves exactly like $\mathbf{N_1}$.

On the other hand, if $p \geq 2^n$, then the behavior of $\mathbf{N_2}$ is very different from that of $\mathbf{N_1}$. For example, in contrast with Lemma 3.1, we have the following.

Lemma 4.2 *If $p \geq 2^n$, then not all trajectories of $\mathbf{N}_2$ terminate.*

Proof Assume that $p \geq 2^n$. Let

$$\zeta = \zeta_0^{2^{n-1}} \zeta_1^{2^{n-2}} \ldots \zeta_{n-1}^{2^0}$$

denote a sequence consisting of 2^{n-1} consecutive occurrences of the reaction ζ_0, followed by 2^{n-2} occurrences of ζ_1, etc. Since $p \geq 2^n$ each of these $2^n - 1$ reactions is enabled when its turn comes. After the sequence ζ has occurred, we have

$$z_0 = p - 2^n,$$
$$z_1 = \ldots = z_{n-1} = 0,$$
$$z_n = 1,$$
$$b = \sum_{i=0}^{m-1} 2^{n-(i+1)} = 2^n(1 - 2^{-m}),$$
$$r = \sum_{i=m}^{n-1} 2^{n-(i+1)} = 2^{n-m} - 1.$$

Recalling that $n > m + 1$, we have $r \geq 3$ here, so the reaction ω is enabled after ζ has occurred. In fact ζ can (in principle) be followed by the infinite sequence

$$\omega, \chi, \omega, \chi, \ldots$$

of reactions, so $\mathbf{N}_2$ has a nonterminating trajectory. $\square$

It is easy to see that the infinite trajectory of $\mathbf{N}_2$ exhibited in the proof of Lemma 4.2 is not fair. In fact, we prove below that all fair paths of $\mathbf{N}_2$ terminate. First, however, we note that $\mathbf{N}_2$, like $\mathbf{N}_1$, has a unique terminal state.

Let ε be as defined before Theorem 3.8.

Lemma 4.3 *If $p \geq 2^n$ and $\mathbf{N}_2$ terminates, then it does so in the state $(z_0, \ldots, z_n, b, r)$ specified as follows.*

(i) $z_{n-1} \cdots z_0$ *is the n-bit binary expansion of* $p \,\mathbf{mod}\, 2^n$.
(ii) $z_n = 1$.
(iii) $b = p - \varepsilon$.
(iv) $r = 0$.

Proof Let q be an accessible state of $\mathbf{N}_2$ that is terminal. The proofs of *(i)* and *(ii)* are the same as in Theorem 3.8, together with the fact that the ω-reaction does not alter the value of z_n. Since $z_n = 1$ and the ω-reaction is disabled in state q, *(iv)* holds in q. Finally, since

$$p = \sum_{i=0}^{n} z_i + b + r = \varepsilon + b + r,$$

(iii) follows from *(i)*, *(ii)*, and *(iv)*. $\square$

Lemma 4.4 *On any fair trajectory of $\mathbf{N}_2$, after finitely many steps, all ζ-reactions are permanently disabled.*

Proof For each $0 \leq j \leq n$, let Φ_j be the assertion that, on any fair trajectory of $\mathbf{N}_2$, after finitely many steps the reaction ζ_j is permanently disabled. It suffices to prove that Φ_j holds for all $0 \leq j \leq n$. We do this by induction on j.

Let $0 \leq j \leq n$ and assume that Φ_i holds for all $0 \leq i < j$. It suffices to show that Φ_j holds. For this, let $\tau = (q_i | 0 \leq i < \infty)$ be a trajectory in which ζ_j is enabled infinitely often. It suffices to show that τ is not fair. We have two cases.

Case 1: Some ζ_i for $0 \leq i < j$ is enabled infinitely often. Then τ is not fair by the induction hypothesis.

Case 2: There exists $k^* \in \mathbb{N}$ such that, for all $k \geq k^*$ the reactions ζ_i, for $0 \leq i < j$, are all disabled in q_k. Then z_j does not increase at any step of τ from k^* onward. Since every occurrence of ζ_j decreases z_j, this implies that ζ_j only occurs finitely many times after k^*, hence only finitely many times in τ. Since ζ_j is enabled infinitely often along τ it follows by Observation 2.1 that τ is not fair. $\square$

Lemma 4.5 *With any initialization, all fair trajectories of the chemical reaction network $\mathbf{N}\chi\omega$, consisting of just the reactions χ and ω, are terminal.*

Proof If $\mathbf{N}\chi\omega$ is initialized with $z_n = 0$, then there is only one full trajectory, which is terminal, so it suffices to prove the lemma for initializations with $z_n > 0$. Let q_0 be any such initial state, and let $\hat{p}$ be the value of $b + r + z_n$ in q_0. Since z_n and $\hat{p}$ are invariants of $\mathbf{N}\chi\omega$, a state of $\mathbf{N}\chi\omega$ is completely determined by the value of r. We thus refer to "the state r" of $\mathbf{N}\chi\omega$, for $0 \leq r \leq \hat{p} - z_n$. Note that in this terminology the unique terminal state is 0.

For each state r of $\mathbf{N}\chi\omega$ with initial state q_0 the trajectory $\tau_r = (r, r-1, \ldots, 1, 0)$ given by r consecutive occurrences of ω is a terminal trajectory from r, so the fair termination lemma (Lemma 2.2) tells us that all fair trajectories of $\mathbf{N}\chi\omega$ are terminal. $\square$

Recall the notation defined just before Theorem 3.8. The following result is our main theorem.

Theorem 4.6 *Let $(z_0, \ldots, z_n, b, r)$ be the state of $\mathbf{N}_2$ specified as follows.*

(i) *$z_{n-1} \cdots z_0$ is the n-bit binary expansion of $p \,\mathbf{mod}\, 2^n$.*
(ii) $z_n = [\![p \geq 2^n]\!]$.
(iii) $b = (p - \varepsilon) * [\![p < 2^m \text{ or } p \geq 2^n]\!]$.
(iv) $r = (p - \varepsilon) * [\![2^m \leq p < 2^n]\!]$.

If $p < 2^n$, then $\mathbf{N_2}$ terminates in this state on all trajectories. If $p \geq 2^n$, then $\mathbf{N_2}$ terminates in this state on all fair trajectories.

Proof If $p < 2^n$, then Corollary 3.3 tells us that z_n never becomes positive in $\mathbf{N_2}$, so ω is never enabled. Hence, in this case $\mathbf{N_2}$ behaves exactly like $\mathbf{N_1}$. Theorem 3.8 tells us that $\mathbf{N_2}$ terminates on all trajectories to the state satisfying *(i)* and *(ii)* above and, since $[\![p < 2^m]\!] = [\![p < 2^m \text{ or } p \geq 2^n]\!]$ and $[\![p \geq 2^m]\!] = [\![2^m \leq p < 2^n]\!]$, also satisfying *(iii)* and *(iv)* above.

If $p \geq 2^n$, then Lemmas 4.4 and 4.5 together tell us that $\mathbf{N_2}$ terminates on all fair trajectories. Since $[\![p \geq 2^n]\!] = 1, [\![p < 2^m \text{ or } p \geq 2^n]\!] = 1$, and $[\![2^m \leq p < 2^n]\!] = 0$, Lemma 4.3 tells us that termination must occur in the state satisfying (i)-(iv) above. □

Since ε is again negligible with respect to p, Theorem 4.6 says that $\mathbf{N_2}$ terminates in an overwhelmingly blue state if $p < 2^m$ or $p \geq 2^n$ but in an overwhelmingly red state if $2^m \leq p < 2^n$. Hence $\mathbf{N_2}$ exhibits very sharp phase transitions at the population thresholds 2^m and 2^n. As noted in the Introduction and elaborated in Sect. 6 below, this has significant implications for the verification of chemical reaction networks.

5 Coupled phase transitions with faster termination

As we have demonstrated with both human and machine-verified proofs, the chemical reaction network $\mathbf{N_2}$ terminates on *all* trajectories and in the intended state (color) in the lower (blue) and middle (red) ranges of initial population. Moreover, in these two ranges, $\mathbf{N_2}$ is efficient in the sense that every occurrence of a reaction makes progress toward the terminal state. In fact, Lemma 3.1 assures us that $\mathbf{N_2}$ halts after at most $3p$ reactions, where p is the initial population. We have also shown that, in the upper (blue) range of initial populations, $\mathbf{N_2}$ terminates on *all fair* trajectories, always in the same, blue state. However, an easy modification of the proof of Lemma 4.2 shows that there is no upper bound whatsoever on the lengths of these fair trajectories. The last epoch of any such trajectory consists entirely of χ-reactions and ω-reactions. We have not specified rate constants of the reactions in $\mathbf{N_2}$, or defined a probability model of the behavior of $\mathbf{N_2}$ in any other way, but in any reasonable such model, the χ-reactions and ω-reactions would impose a random walk on the relative populations r and b of red and blue molecules, respectively, with the sum $b+r$ fixed. If the initial population of $\mathbf{N_2}$ is significantly above the threshold of the upper blue region, then r will be far greater than b at the beginning of this random walk, and the χ-reaction will intuitively be trying to exterminate the blue molecules. In fact, the χ-reaction will succeed in doing this many times (with the ω-reaction subsequently producing a new blue molecule) before the random walk eventually enables the ω-reaction to finally and permanently exterminate the red molecules. Indeed, simulations of $\mathbf{N_2}$ with simple probability assumptions (e.g., all rate constants 1) and even very modest values of m and n fail to terminate in practical amounts of time with initial populations in the upper blue region.

We have chosen the chemical reaction network $\mathbf{N_2}$ as our primary exemplar of coupled phase transitions because its behavior admits a reasonably elegant correctness proof that is very much in the spirit of distributed computing theory. However, a discerning reader might well be concerned that coupled phase transitions can only occur in simple chemical reaction networks that do not terminate in feasible amounts of time.

In this section we dispel this concern by presenting a chemical reaction network $\mathbf{N_3}$ that has coupled phase transitions like those of $\mathbf{N_2}$, but terminates more quickly under reasonable assumptions.

The chemical reaction network $\mathbf{N_3}$ uses the same ladder mechanism as $\mathbf{N_2}$ as an entry point for the phase transition. As with $\mathbf{N_1}$, we fix $m, n, p \in \mathbb{Z}^+$, with $n > m + 1$, and define n ζ-reactions

$$\zeta_i \equiv Z_i + Z_i \to \begin{cases} Z_{i+1} + B & (0 \leq i < m) \\ Z_{i+1} + R & (m \leq i < n) \end{cases}.$$

To perform the transition to red at $p = 2^m$, however, $\mathbf{N_3}$ uses the species $Z_m, \ldots, Z_{n-1}$ (which can only be created when $p \geq 2^m$) as catalysts to convert blue to red via the following additional $n - m$ reactions

$$Z_i + B \to Z_i + R, \quad m \leq i < n.$$

Since $\mathbf{N_3}$ creates only a finite amount of blue molecules in the lower stages of the ladder, these catalytic reactions are sufficient to convert all blue into red.

To perform the transition back to blue at $p = 2^n$, $\mathbf{N_3}$ uses the Q species, which is created at the end of the ladder, reachable only when $p \geq 2^n$. We include the following $n + 3$ additional reactions:

$$\begin{aligned} Z_{n-1} + Z_{n-1} &\to Q + B \\ Q + R &\to Q + B \\ Q + Q &\to Q + R \\ Q + Z_i &\to Q + Q, \quad i \leq n. \end{aligned}$$

The Q species catalytically converts Z_i into Q and R into B. It also catalytically converts itself into blue molecules. Therefore if $\mathbf{N_3}$ produces a single Q species, the population must eventually convert into blue molecules, leaving behind a single Q molecule.

Note that we still require a fairness condition to prove that $\mathbf{N_3}$ terminates if $p \geq 2^n$, because we can identify a class of infinite-length state-space trajectories wherein B is catalytically exchanged for R and vice-versa. For example, if both a Z_m molecule and a Q molecule are present, the reactions

$$Z_m + B \rightarrow Z_m + R$$
$$Q + R \rightarrow Q + B$$

can alternate indefinitely.

As described in Sect. 6, we have performed the same verification techniques for $\mathbf{N_3}$ as for $\mathbf{N_2}$ with very similar results. Model checking and stochastic simulation cannot address population counts outside the lower blue region due to computational limitations, and therefore always predict a blue result. ODE simulation also predicts a blue result. As with $\mathbf{N_2}$, we cannot conclusively say whether this is due to numerical issues, and without a more extensive analysis, we cannot guarantee that the preconditions of Kurtz's theorem apply. It is clear, however, that ODE simulation fails for $\mathbf{N_3}$. We have again used Isabelle/HOL to verify $\mathbf{N_3}$, producing a machine-verified mathematical proof that applies at any population scale, and correctly predicts the two population-induced phase transitions.

Erik Winfree (2020) recently designed another coupled phase transition chemical reaction network that terminates more quickly than $\mathbf{N_2}$. This reaction network, unlike $\mathbf{N_3}$, retains the "random walk" final epoch of $\mathbf{N_2}$ but uses a rapidly produced catalyst to shorten this epoch.

6 Implications for verification

The coupled phase transitions in the chemical reaction network $\mathbf{N_2}$ make it difficult to verify its behavior. In this section we describe the use and limitations of verifying the chemical reaction network using simulation, model checking and differential equations. None of these methods detected that the system turned red when the population reached 2^m. We then describe how the use of an interactive theorem prover enabled us to verify the chemical reaction network's behavior at both phase transitions, i.e., that it turned from blue to red at 2^m and from red to blue at 2^n. The fact that theorem proving could verify behavior that was otherwise not verified for the chemical reaction network suggests that interactive theorem proving may have a useful role to play in future verification of a class of chemical reaction networks. Recall that the chemical reaction networks $\mathbf{N_1}$ and $\mathbf{N_2}$ have fixed populations throughout any given execution, and that their initial states have z_0 as the entire population.

6.1 Simulation

The MATLAB SimBiology package is widely used to explore the behavior of a number of devices (molecules) executing concurrently (MATLAB 2019). Using SimBiology, simulations of the $\mathbf{N_2}$ chemical reaction network were performed on an Intel processor computer with a processor clock of 5.0 GHz and 64GB of RAM. Several simulations were performed with increasing populations z_0. With a population of 10^7, the simulation performed as expected. However, with a population of 10^8, the simulation failed and terminated with no output or error message. Thus, the stochastic simulation was unable to detect that the behavior of the $\mathbf{N_2}$ chemical reaction network could experience a phase transition.

Since it is currently not possible to simulate $\mathbf{N_1}$, $\mathbf{N_2}$ and $\mathbf{N_3}$ to completion with $n = 65$, it is not possible to visualize the phase transitions of these CRNs using simulation. Hence, we investigate the running time of smaller versions of these two CRNs, which confirms our intuition and theorems. For simulating smaller versions of of these CRNs, we reduce the set of reactions so that $m = 5$ and $n = 10$.

Figure 2 shows the behavior of the smaller version of $\mathbf{N_2}$ at populations near the phase transitions. Population 127 is just below the first phase transition and the top graph shows that the population quickly turns to blue and remains blue. Population 128 is the lowest population above the first phase transition, and it turns red in approximately 3.3 seconds. The largest population where the CRN can remain in the red state indefinitely is 1023. Finally, the smallest population above the second phase transition is 1024. This is shown in the bottom graph of the figure and does not exhibit the eventual blue behavior, because the expected time of the random walk for $n = 10$ is still very large.

This random walk can be seen in Fig. 3 where part of the simulation is magnified to show the individual counts of the red species. It is clear from the graph that even converting 10 red species to blue is unlikely in feasible time.

Finally, Fig. 4 shows the red and blue population counts for the smaller version of $\mathbf{N_3}$. Here we see that a population of size 1024 turns nearly all red before turning essentially all blue in approximately 5 seconds.

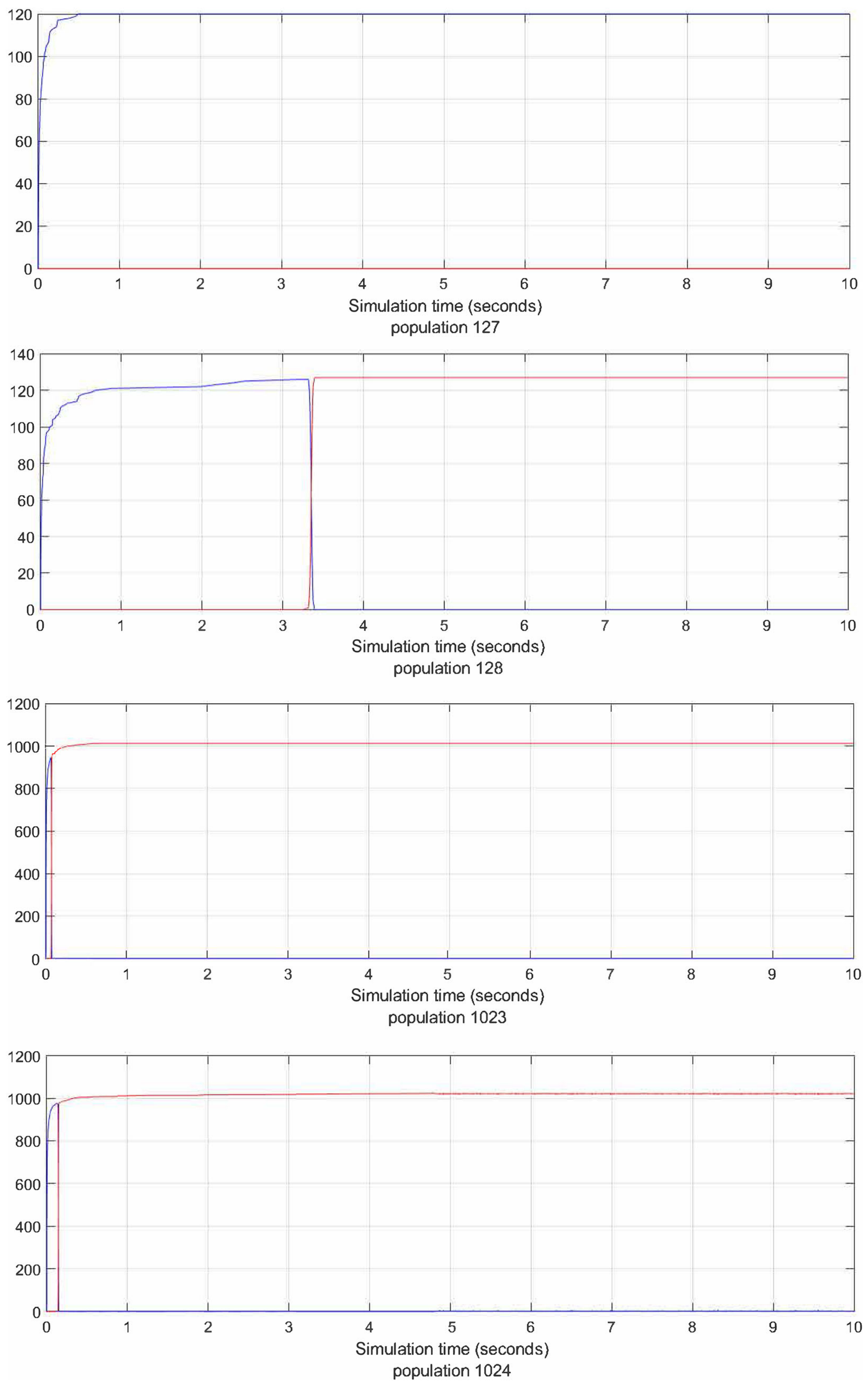


Fig. 2 Simulation of $\mathbf{N_2}$ with $m = 5$ and $n = 10$ at different population sizes

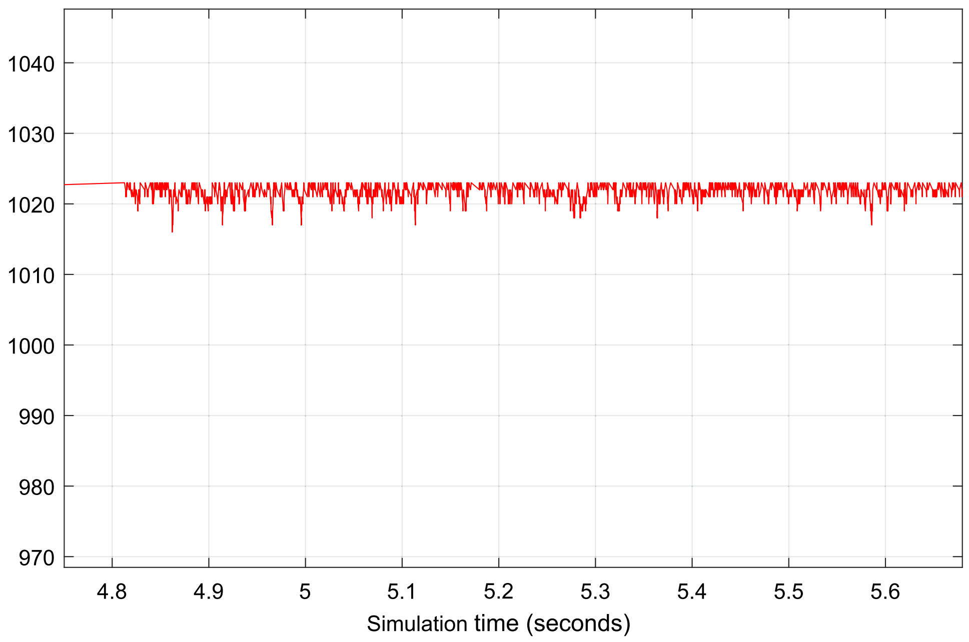


Fig. 3 Example of random walk with initial population 1024 in $\mathbf{N_2}$ with $m = 5$ and $n = 10$, failing to turn blue in feasible time

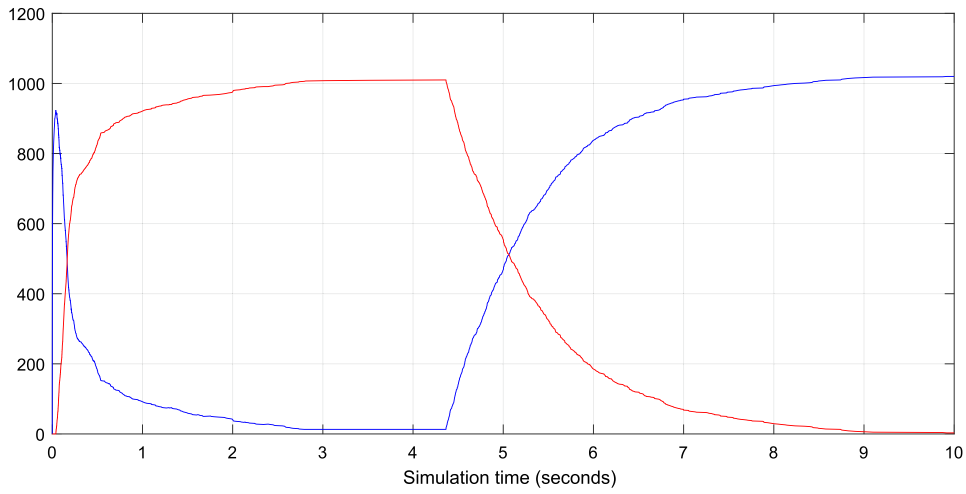


Fig. 4 Simulation of $\mathbf{N_3}$ with $m = 5$ and $n = 10$ and population 1024 with fast change to blue

6.2 Model checking

The chemical reaction networks $\mathbf{N_2}$ and $\mathbf{N_3}$ simulated in SimBiology and described above were also verified using the PRISM 4.6 probabilistic model checker (Kwiatkowska et al. 2011). Kwiatkowska and Thachuk, among others, have described the use of PRISM for the probabilistic verification of chemical reaction networks for biological systems (Kwiatkowska and Thachuk 2014).

To verify the chemical reaction network behavior we first converted the $\mathbf{N_2}$ and $\mathbf{N_3}$ models to SBML using the export function in SimBiology, and then converted the SBML models to PRISM using the sbml2prism conversion tool supplied with the PRISM software. PRISM was used to verify six key properties of the $\mathbf{N_2}$ and $\mathbf{N_3}$ chemical reaction networks at multiple populations. For example, one of the properties stated that "$P >= 1[\mathrm{F\ G}\ r = 0]$", i.e., that with probability 1, the eventual state of the R species has 0 molecules, and never changes from that. With a population of 100, PRISM generated the CTMC state model in 1.65 seconds using the same processor and memory as for the SimBiology simulations, and the verification of the six properties required less than 2 seconds of CPU time.

For a population of 100 molecules, 97 correctly turn blue and 0 correctly turn red, since the latter only happens when the initial population is larger than 2^{34}. PRISM also verified that in the final state the species count was $z_0 = 0$, $z_1 = 0$, $z_2 = 1$ and $z_5 = 1$, based on the binary expansion of one hundred.

However, we were unable to model check $\mathbf{N_2}$ or $\mathbf{N_3}$ with a population of 400 due to the rapid increase in states and limited memory. Thus, model checking confirmed the expected behavior of the $\mathbf{N_2}$ and $\mathbf{N_3}$ chemical reaction networks for a population of 100 but could not detect the behavioral change to red when the population increased.

Advanced methods to prune a model so that meaningful model checking can occur include symmetry reduction (Heath et al. 2006), statistical model checking (Cardelli et al. 2018), and automated partial exploration of the model (Pavese et al. 2016). Recent work by Cauchi, et al. using formal synthesis allowed verification of systems with 10 continuous variables (Cauchi et al. 2019). However, even these methods would not be likely to help with the exceedingly large number of states when the number of molecules is scaled to a realistic value for experiments.

6.3 Differential equations

We have seen how model checking and simulation fail to detect the "red" behavior in our chemical reaction networks $\mathbf{N_2}$ and $\mathbf{N_3}$ due to the processing time and memory required for a large population. The red behavior also is not detected when these CRNs are approximated by deterministic semantics. In this model, a chemical reaction network is represented by a system of polynomial autonomous differential equations. Our purpose here is to investigate the usefulness of the large population heuristic in this context; we do not make any claims that our results respect the preconditions and caveats of Kurtz's theorem (Kurtz 1972), which provides a mathematical link between deterministic and high-population stochastic systems.

In general, the system of differential equations induced by a chemical reaction network is difficult or impossible to solve exactly, and numerical methods are often used to approximate solutions. Here, we utilized MATLAB and the SimBiology package (MATLAB 2019) to numerically integrate the system of differential equations for $\mathbf{N_2}$ and $\mathbf{N_3}$. We found that both CRNs reached and remained in a predominantly blue state for the duration of the simulation, again missing the red behavior.

We identify three potential causes for these failures. One potential cause is numerical failure; it may be that MATLAB's numerical integration was not robust enough to capture the relevant deterministic behavior, or that we did not let the simulation run long enough to converge. (We note that, although we expect $\mathbf{N_3}$ to have a reasonable convergence time, we expect $\mathbf{N_2}$ to take an extreme amount of time to converge, at least in the stochastic case.) Another potential cause is that, as suggested by Kurtz's theorem, the deterministic system might correctly approximate high-population stochastic behavior, which falls above the second phase-transition threshold (and well above the range of a realistic wet-lab implementation of these CRNs.) Finally, it may be that the stochastic and deterministic behaviors of one or both of these CRNs are not actually closely related, and the deterministic results do not imply anything conclusive about the underlying stochastic system. Regardless of the cause, however, we see that differential equation methods are not sufficient to capture the red behavior of $\mathbf{N}_2$ and $\mathbf{N}_3$.

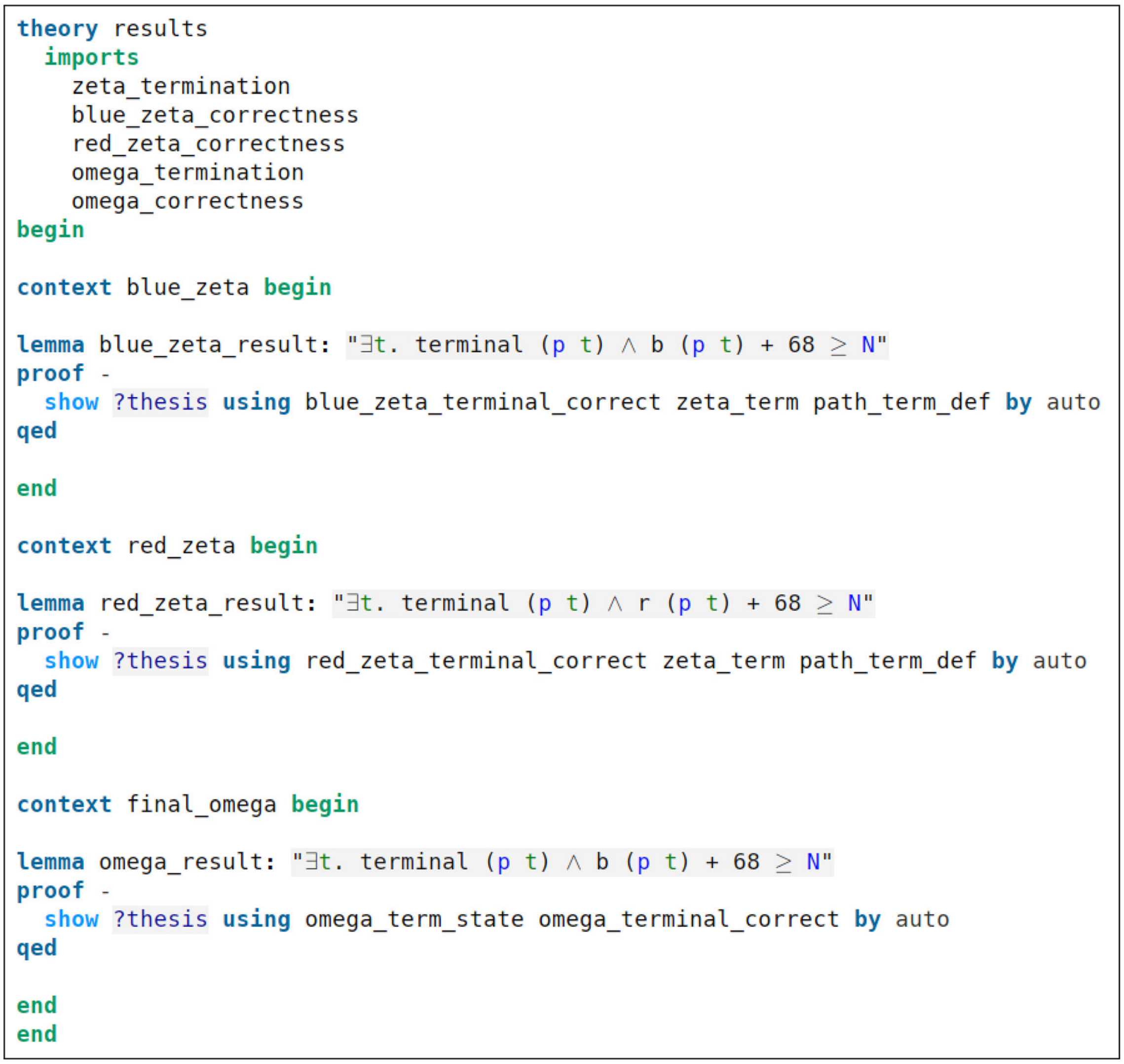
6.4 Theorem proving

The simulation, model checking, and differential equations approaches to chemical reaction network verification outlined above all make some simplifying assumptions: reduced state space or generalization to the continuum. In the case of our chemical reaction network, these assumptions lead to an incorrect verification result.

Interactive theorem proving, however, offers an exact approach that is guaranteed to apply at every scale. In the interactive theorem proving paradigm, users create a machine-checkable mathematical proof of verification properties in collaboration with a software system. Model checking also constructs a mathematical proof of correctness, but it relies more on a complete or semi-complete search of the state space in question. By contrast, the goal of interactive theorem proving is to construct a more traditional mathematical proof that is also machine-checkable. The result then applies to any population scale; a mathematical proof parameterized by population p is valid at every possible value of p.

In a typical interactive theorem proving session, a user starts with a base of trusted facts generated from axioms and assumptions, and uses well-understood rules like modus ponens and double negation removal to construct new trusted facts and lemmas. As with a conventional mathematical proof, the user's goal is to add new trusted facts in a strategic way until reaching the goal of the proof.

We have verified our chemical reaction networks with Isabelle/HOL (Nipkow and Klein 2014; Nipkow et al. 2002), a popular interactive theorem prover with several useful proof automation features. Instead of working at the level of rules like modus ponens, users can instruct Isabelle to execute more general proof methods that can apply sequences of basic rules without user direct input. For example, Isabelle can often prove the equivalence of predicate logic formulas with only one user-generated method invocation. Once invoked, such a method attempts to automatically construct a series of low-level logical rules whose application proves the equivalence. An Isabelle proof, then, consists of a directed acyclic graph of facts, connected by applications of these methods. The user's task is to choose a chain of intermediate goal facts in a way



```
theory results
  imports
    zeta_termination
    blue_zeta_correctness
    red_zeta_correctness
    omega_termination
    omega_correctness
begin

context blue_zeta begin

lemma blue_zeta_result: "∃t. terminal (p t) ∧ b (p t) + 68 ≥ N"
proof -
  show ?thesis using blue_zeta_terminal_correct zeta_term path_term_def by auto
qed

end

context red_zeta begin

lemma red_zeta_result: "∃t. terminal (p t) ∧ r (p t) + 68 ≥ N"
proof -
  show ?thesis using red_zeta_terminal_correct zeta_term path_term_def by auto
qed

end

context final_omega begin

lemma omega_result: "∃t. terminal (p t) ∧ b (p t) + 68 ≥ N"
proof -
  show ?thesis using omega_term_state omega_terminal_correct by auto
qed

end
end
```

Fig. 5 The end of the Isabelle proof for $\mathbf{N}_2$, which summarizes its results in three lemmas. The `context` statements bring our assumptions about the population size into context. The `using` statements bring in trusted facts from the rest of our proof and supply them as arguments to Isabelle's `auto` proof method. The identifier `p` refers to an arbitrary trajectory that is part of each context. Isabelle displays all statements with a white or light gray background to indicate that it has checked them completely, and they are valid

that allows Isabelle to connect them easily on the way to the overall goal.

Isabelle also provides the powerful Sledgehammer automation tool, which makes calls to external proof systems to automate aspects of proof creation. Sledgehammer takes a goal fact as input and attempts to generate a method invocation that proves it, operating at one level of abstraction above the proof methods invocations discussed above. Since it is often unclear which method to invoke (or which arguments to supply to it), this functionality can increase proof construction speed substantially.

We have used Isabelle to verify that our chemical reaction networks have the desired behavior for all possible initializations. That is, if we initialize them with $p < 2^{34}$ or $p \geq 2^{67}$, the chemical reaction networks terminate with majority blue, but if we initialize them with $2^{34} \leq p < 2^{67}$, they terminate with majority red. As expected, theorem proving is able to verify behavior correctly in all regions, including the middle region that is inaccessible to model checking, simulation, and ODE methods. Figure 5 shows an image taken from the end of our Isabelle proof for $\mathbf{N}_2$; it contains the three goal facts that we successfully verified, which summarize the behavior of the chemical reaction network.

Our Isabelle proof for $\mathbf{N}_2$ is loosely based on the proofs presented in Sects. 3 and 4. Whereas those proofs define two chemical reaction networks $\mathbf{N}_1$ and $\mathbf{N}_2$, we associate assumptions about the population p with various parts of our proof. Within the scope of the assumption that $p < 2^{34}$, for example, we are able to prove that $\mathbf{N}_2$ terminates with

Fig. 6 This Isabelle code from our proof for $\mathbf{N}_2$ defines a terminal state as a state with no outgoing reactions; K is a relation that encodes which state transitions our reaction set allows. We also show a sample lemma that helps prove termination: if we identify a countdown expression f and a constant C such that all states with $f < C$ are terminal, then our system is guaranteed to terminate

```
theory termin
  imports piptcrn
begin

definition terminal :: state ⇒ bool where terminal s1 = (¬(∃ s2. K s1 s2))
definition nonterm :: state ⇒ bool where nonterm s = (¬(terminal s))

definition path-term :: (nat ⇒ state) ⇒ bool where
  (path-term p) = (∃ t. (terminal (p t)))

definition state-term :: (state ⇒ bool) where
  (state-term s) = (∀ (p :: (nat ⇒ state)).
     ((∃ t. ((p t) = s))
      ⟶ (path-term p)))

lemma dec-imp-term:
    fixes f :: state ⇒ nat
    fixes p :: nat ⇒ state
    fixes c :: nat
    assumes evterm: ((f s) ≤ c) ⟶ (terminal s)
    assumes dec: ∀ i. ((¬(terminal (p i))) ⟶ (
      (f (p (i + 1)) < (f (p i)))))
    shows path-term p
proof –
  {
    fix n::nat
    have ((∃ t. ((f (p t)) ≤ n)) ⟶ (path-term p))
    proof (induction n)
      case 0
      then show ?case
        using dec gr-implies-not0 path-term-def by blast
    next
      case (Suc n)
      then show ?case
        by (metis dec le-SucE less-Suc-eq-le path-term-def)
    qed
  }
  then show ?thesis by blast
qed

end
```

majority blue. Our Isabelle proof for $\mathbf{N}_3$ functions in a similar way.

We refer to the three population scales as the lower blue region, the middle red region, and the upper blue region. Under the assumptions associated with each region, our proofs must show both termination and correctness; i.e., we must show that our chemical reaction network reaches a final state where no reactions are possible, and that any possible final state has the specified red or blue population.

As in Lemma 3.1, we show termination in the lower two regions via a "countdown" expression that is guaranteed to decrease with every reaction. See Fig. 6 for our Isabelle definitions of termination in our machine-checked $\mathbf{N}_2$ proof and a general lemma we proved that allows us to use the countdown technique. In the upper blue region of both $\mathbf{N}_2$ and $\mathbf{N}_3$, it is impossible to prove termination without assuming that executions are fair. Our Isabelle proof includes Equation 2.4 as an unproven assumption; we are not interested in unfair trajectories, but since they exist we cannot prove that all trajectories are fair. For convenience, we also include Observation 2.1 as an assumption. These two fairness assumptions allow us to prove that our chemical reaction networks terminate in the upper blue region as well.

Our correctness proofs rely heavily on the sum $S_{68} = \sum_{i=0}^{67} 2^i z_i$, using the notation of Sect. 3, which is an invariant in the lower two regions. In the upper blue region, it is an invariant until at least one Z_{67} (in $\mathbf{N}_2$) or Q (in $\mathbf{N}_3$) is produced. This invariant allows us to reason about the composition of terminal states. In the lower blue region, for example, we know that no red can ever be produced; the chemical reaction network can only produce its first red molecule alongside Z species that would make the invariant too large. We can therefore prove that any terminal state in that region must be majority blue.

7 Conclusion

Taken together, the near-ubiquity of phase transitions in nature (Dana 2010; Cannon et al. 2018), the sheer size of molecular populations, and the simplicity of the chemical reaction networks that we have shown to exhibit population-induced phase transitions, indicate that molecular programming will present us with many exceptions to the otherwise useful notion that most bugs can be demonstrated with small counterexamples. As we have seen, this presents a significant challenge to the verification of chemical reaction networks. Here we suggest some directions of current and future research that might help meet this challenge.

A great deal of creative work has produced a steady scaling up of model checking to larger and larger state spaces (Clarke et al. 2009; Chrszon et al. 2018; Abdulla et al. 2018; Bortolussi et al. 2019; Lomuscio and Pirovano 2019; Ceska et al. 2019). Perhaps the most hopeful approach for dealing with population-induced phase changes, or with more general population-sensitive behaviors, is the model checking of parametrized systems (Abdulla et al. 2018).

Our results clearly demonstrate the advantage of including theorem proving (by humans and by software) in the verification toolbox for chemical reaction networks and other molecular programming languages. This in turn suggests that software proof assistants such as Isabelle (Nipkow et al. 2002; Nipkow and Klein 2014) be augmented with features to deal more directly with chemical reaction networks and with population-sensitive phenomena. It would also be useful to know how much of such work could be carried out with automated (as opposed to interactive) theorem provers such as Vampire (Kovács and Voronkov 2013).

Some future programmed molecular applications will be safety-critical, such as in health diagnostics and therapeutics. It is likely that evidence that such systems behave as intended will be required for certification by regulators prior to deployment. Toward providing such evidence, Nemouchi et al. have recently shown how a descriptive language for safety cases can be incorporated into Isabelle in order to formalize argument-based safety assurance cases (Nemouchi et al. 2019).

We conclude with a more focused, theoretical question. Our chemical reaction network $\mathbf{N}_1$ exhibits its phase transition on all trajectories, while $\mathbf{N}_2$ exhibits its coupled phase transitions only on all fair trajectories. Is there a chemical reaction network that achieves $\mathbf{N}_2$'s coupled phase transitions on *all* trajectories?

Appendix: Proof of Fair Termination Lemma

Lemma 2.2 *(fair termination lemma) If a population protocol with a specified initial state has a terminal trajectory from every accessible state, then all its fair trajectories are terminal.*

Proof Let $\mathbf{N}$ be a population protocol with initial state q_0, and assume that $\mathbf{N}$ has a terminal trajectory from every accessible state. Let $\tau = (q_i \mid 0 \le i < \infty)$ be an infinite trajectory of $\mathbf{N}$. It suffices to show that τ is not fair. For each state q of $\mathbf{N}$, let

$$I_q = \{i \in \mathbb{N} \mid q_i = q\}.$$

Since $\mathbf{N}$ is a population protocol, it has finitely many accessible states, so there is a state q^* of $\mathbf{N}$ such that the set I_{q^*} is infinite. This state q^* is accessible, so our assumption tells us that there is a finite trajectory $\tau^* = (q_i^* \mid 0 \le i < \ell)$ of $\mathbf{N}$ such that $q_0^* = q^*$ and $q_{\ell-1}^*$ is terminal. Now $I_{q_0^*} = I_{q^*}$ is infinite and $I_{q_{\ell-1}^*} = \emptyset$ (because $q_{\ell-1}^*$ is terminal, so it does not appear in the infinite trajectory τ), so there exists $0 \le k < \ell - 1$ such that $I_{q_k^*}$ is infinite and $I_{q_{k+1}^*}$ is finite. Let $q^{**} = q_k^*$, and let ρ be the reaction that takes q_k^* to q_{k+1}^*. Then ρ is enabled in q^{**} and there exist infinitely many i such that $q_i = q^{**}$ (because $I_{q^{**}}$ is infinite), but there are only finitely many j for which $q_j = q^{**}$ and ρ occurs at j in τ (because $I_{q_{k+1}^*}$ is finite). Hence τ is not fair. □

Acknowledgements We thank Neil Lutz for technical assistance. The second and third authors thank Erik Winfree for his hospitality while they did part of this work during a 2020 sabbatical visit at the California Institute of Technology. We thank the anonymous reviewers of both the conference and journal versions of this paper for multiple useful suggestions. We thank Erik Winfree for correcting an "off-by-one" error in our conference paper and for telling us about his fast chemical reaction network mentioned here at the end of Section 5. This research was supported in part by National Science Foundation Grants 1545028, 1900716, and 1909688.

References

Abdulla PA, Sistla AP, Talupur M (2018) Model checking parameterized systems. In: Clarke EM, Henzinger TA, Veith H, Bloem R (eds) Handbook of Model Checking. Springer, pp 685–725

Anderson DF, Kurtz TG (2011) Continuous time Markov chain models for chemical reaction networks. In: Heinz K, Gianluca S, Mario di B, Douglas D, (eds) Design and Analysis of Biomolecular Circuits. Springer, pp 3–42

Anderson DF, Kurtz TG (eds) (2015) Stochastic analysis of biochemical systems, volume 1.2. Springer International Publishing

Angluin D, Aspnes J, Eisenstat D (2008) A simple population protocol for fast robust approximate majority. Distrib Comput 21(2):87–102

Angluin D, Aspnes J, Eisenstat D, Ruppert E (2007) The computational power of population protocols. Distrib Comput 20(4):279–304

Badelt S, Shin SW, Johnson RF, Dong Q, Thachuk C, Winfree E (2017) A general-purpose CRN-to-DSD compiler with formal verification, optimization, and simulation capabilities. In: Proceedings of the 23rd International Conference on DNA Computing and Molecular Programming, Lecture Notes in Computer Science, pp 232–248

Baier C, Katoen J-P (2008) Principles of model checking (Representation and mind series). The MIT Press, USA

Bortolussi L, Cardelli L, Kwiatkowska M, Laurenti L (2019) Central limit model checking. ACM Trans Comput Log 20(4):19:1-19:35

Cannon S, Miracle S, Randall D (2018) Phase transitions in random dyadic tilings and rectangular dissections. SIAM J Discret Math 32(3):1966–1992

Cardelli L, Csikász-Nagy A (2012) The cell cycle switch computes approximate majority. Scientific Reports, 2

Cardelli L, Kwiatkowska M, Whitby M (2016) Chemical reaction network designs for asynchronous logic circuits. In: International Conference on DNA-Based Computers. Springer, pp 67–81

Cardelli L, Kwiatkowska M, Whitby M (2018) Chemical reaction network designs for asynchronous logic circuits. Nat Comput 17(1):109–130

Cauchi N, Laurenti L, Lahijanian M, Abate A, Kwiatkowska M, Cardelli L (2019) Efficiency through uncertainty: scalable formal synthesis for stochastic hybrid systems. In: Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control, HSCC 2019, Montreal, QC, Canada, April 16-18, 2019., pp 240–251

Ceska M, Jansen N, Junges S, Katoen JP (2019) Shepherding hordes of Markov chains. In: Proceedings of the International Conference on Tools and Algorithms for the Construction and Analysis of Systems, pages 172–190. Springer

Chen Y-J, Dalchau N, Srinivas N, Phillips A, Cardelli L, Soloveichik D, Seelig G (2013) Programmable chemical controllers made from DNA. Nat Nanotechnol 8(10):755–762

Chrszon P, Dubslaff C, Klüppelholz S, Baier C (2018) ProFeat: feature-oriented engineering for family-based probabilistic model checking. Formal Asp Comput 30(1):45–75

Clarke EM, Allen Emerson E, Sifakis J (2009) Model checking: algorithmic verification and debugging. Commun ACM 52(11):74–84

Condon A, Hajiaghayi M, Kirkpatrick DG, Manuch J (2017) Simplifying analyses of chemical reaction networks for approximate majority. In: Proceedings of the 23rd International Conference on DNA Computing and Molecular Programming, volume 10467 of Lecture Notes in Computer Science, pages 188–209. Springer

Cook M, Soloveichik D, Winfree E, Bruck J (2009) Programmability of chemical reaction networks. In: Condon A, Harel D, Kok JN, Salomaa A, Winfree E (eds) Algorithmic Bioprocesses. Springer, Natural Computing Series, pp 543–584

Dietz H. Synthetic DNA machines to fight viruses and other troubles. Matter to Life Lecture Series, Technical University of Munich

Douglas SM, Bachelet I, Church GM (2012) A logic-gated nanorobot for targeted transport of molecular payloads. Science 335(6070):831–834

Ellis SJ, Klinge TH, Lathrop JI, Lutz JH, Lutz RR, Miner AS, Potter HD (2019) Runtime fault detection in programmed molecular systems. ACM Trans Softw Eng Methodol 28(2):6:1-6:20

Fages F, Le Guludec G, Bournez O, Pouly A (2017) Strong Turing completeness of continuous chemical reaction networks and compilation of mixed analog-digital programs. In: Proceedings of the 15th International Conference on Computational Methods in Systems Biology, pages 108–127. Springer International Publishing

Harel D (1986) Effective transformations on infinite trees, with applications to high undecidability, dominoes, and fairness. J ACM 33(1):224–248

Heath J, Kwiatkowska M, Norman G, Parker D, Tymchyshyn O (2006) Probabilistic model checking of complex biological pathways. In: Computational Methods in Systems Biology, pages 32–47, Berlin, Heidelberg. Springer Berlin Heidelberg

Jackson D (2019) Alloy: a language and tool for exploring software designs. Commun ACM 62(9):66–76

Kovács L, Voronkov A (2013) First-order theorem proving and vampire. In Computer Aided Verification - 25th International Conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Proceedings, volume 8044 of Lecture Notes in Computer Science, pages 1–35. Springer

Kozen D (2006) Theory of computation. Texts in Computer Science. Springer

Kurtz TG (1972) The relationship between stochastic and deterministic models for chemical reactions. J Chem Phys 57(7):2976–2978

Kwiatkowska M, Norman G, Parker D (2011) PRISM 4.0: Verification of probabilistic real-time systems. In: Proceedings of the 23rd International Conference on Computer Aided Verification, volume 6806 of Lecture Notes in Computer Science, pages 585–591. Springer

Kwiatkowska M, Thachuk C (2014) Probabilistic model checking for biology. Softw Syst Safety 36:165–189

Kwiatkowska MZ (1989) Survey of fairness notions. Inf Softw Technol 31(7):371–386

Lakin MR, Parker D, Cardelli L, Kwiatkowska M, Phillips A (2012) Design and analysis of DNA strand displacement devices using probabilistic model checking. J R Soc Interface 9(72):1470–1485

Lathrop JI, Lutz JH, Lutz RR, Potter HD, Riley MR (2020) Population-induced phase transitions and the verification of chemical reaction networks. In: 26th International Conference on DNA Computing and Molecular Programming, LIPIcs, pages 5:1–5:17. Schloss Dagstuhl

Li S, Jiang Q, Liu S, Zhang Y, Tian Y, Song C, Wang J, Zou Y, Anderson GJ, Han J-Y, Chang Y, Liu Y, Zhang C, Chen L, Zhou G, Nie G, Yan H, Ding B, Zhao Y (2018) A DNA nanorobot functions as a cancer therapeutic in response to a molecular trigger in vivo. Nat Biotechnol 36:258

Liu X, Liu Y, Yan H (2013) Functionalized DNA nanostructures for nanomedicine. Isr J Chem 53(8):555–566

Lomuscio A, Pirovano E (2019) A counter abstraction technique for the verification of probabilistic swarm systems. In: Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS'19, pages 161–169

MATLAB (2019) version 9.7.0 (R2019b, Update 4). The MathWorks Inc., Natick, Massachusetts

Miller B, Bassler L (2001) Quorum sensing in bacteria. Annu Rev Microbiol 55(1):165–199 **(PMID: 11544353)**

Nemouchi Y, Foster S, Gleirscher M, Kelly T (2019) Isabelle/SACM: Computer-assisted assurance cases with integrated formal methods. In: Proceedings of the 15th International Conference on Integrated Formal MethodsIFM 2019, volume 11918 of Lecture Notes in Computer Science, pages 379–398. Springer

Nipkow T, Klein G (2014) Concrete semantics-With Isabelle/HOL. Springer, Berlin

Nipkow T, Paulson LC, Wenzel M (2002) Isabelle/HOL, volume 2283 of Lecture Notes in Computer Science. Springer-Verlag Berlin Heidelberg, 1 edition

Paulson LC, Nipkow T, Wenzel M (2019) From LCF to Isabelle/HOL. Formal Asp Comput 31(6):675–698

Pavese E, Braberman V, Uchitel S (2016) Less is more: Estimating probabilistic rewards over partial system explorations. ACM Trans Softw Eng Methodol 25(2):16:1-16:47

Dana R (2010) Phase transitions in sampling algorithms and the underlying random structures. In: Kaplan H (ed) Proceedings Scandinavian Symposium and Workshops on Algorithm Theory SWAT, vol 6139. Lecture Notes in Computer Science, page 309. Springer

Randall D (2017) Phase Transitions and Emergent Phenomena in Random Structures and Algorithms (Keynote Talk). In: 31st International Symposium on Distributed Computing (DISC 2017), volume 91 of Leibniz International Proceedings in Informatics (LIPIcs), pages 3:1–3:2. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik,

Rice HG (1951) Classes of recursively enumerable sets and their decision problems. Ph.D thesis, Syracuse University

Rice HG (1953) Classes of recursively enumerable sets and their decision problems. Trans Am Math Soc 74:358–366

Apoorva S, Akshaya A, Junling G, Samir M (2020) Layered self-assemblies for controlled drug delivery: A translational overview. Biomaterials 242:119929

Soloveichik D, Cook M, Winfree E, Bruck J (2008) Computation with finite stochastic chemical reaction networks. Nat Comput 7(4):615–633

Soloveichik D, Seelig G, Winfree E (2009) DNA as a universal substrate for chemical kinetics. In: Proceedings of the 14th International Meeting on DNA Computing, volume 5347 of Lecture Notes in Computer Science, pages 57–69. Springer

Thubagere AJ, Thachuk C, Berleant J, Johnson RF, Ardelean DA, Cherry KM, Qian L (2017) Compiler-aided systematic construction of large-scale DNA strand displacement circuits using unpurified components. Nature Communications, 8

Erik Winfree (2020) personal communication

Wooley John C, Lin Herbert S (2005) Catalyzing inquiry at the interface of computing and biology. National Academies Press, USA

Zhang DY, Seelig G (2011) Dynamic DNA nanotechnology using strand-displacement reactions. Nat Chem 3(2):103–113