JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 16, Number 3, 2009

Pp. 501–522
DOI: 10.1089/cmb.2008.0063

Robust Stochastic Chemical Reaction Networks and Bounded Tau-Leaping

DAVID SOLOVEICHIK

ABSTRACT

The behavior of some stochastic chemical reaction networks is largely unaffected by slight inaccuracies in reaction rates. We formalize the robustness of state probabilities to reaction rate deviations, and describe a formal connection between robustness and efficiency of simulation. Without robustness guarantees, stochastic simulation seems to require computational time proportional to the total number of reaction events. Even if the concentration (molecular count per volume) stays bounded, the number of reaction events can be linear in the duration of simulated time and total molecular count. We show that the behavior of robust systems can be predicted such that the computational work scales linearly with the duration of simulated time and concentration, and only polylogarithmically in the total molecular count. Thus our asymptotic analysis captures the dramatic speedup when molecular counts are large, and shows that for bounded concentrations the computation time is essentially invariant with molecular count. Finally, by noticing that even robust stochastic chemical reaction networks are capable of embedding complex computational problems, we argue that the linear dependence on simulated time and concentration is likely optimal.

Key words: tau-leaping, computational complexity, stochastic processes, robustness.

1. INTRODUCTION

THE STOCHASTIC CHEMICAL REACTION NETWORK (SCRN) model of chemical kinetics is used in chemistry, physics, and computational biology. It describes interactions involving integer number of molecules as Markov jump processes (Érdi and Tóth, 1989; Gillespie, 1992; McQuarrie, 1967; van Kampen, 1997), and is used in domains where the traditional model of deterministic continuous mass action kinetics is invalid due to small molecular counts. Small molecular counts are prevalent in biology: for example, over 80% of the genes in the *Escherichia coli* chromosome are expressed at fewer than a hundred copies per cell, with some key control factors present in quantities under a dozen (Guptasarma, 1995; Levin, 1999). Indeed, experimental observations and computer simulations have confirmed that stochastic effects can be physiologically significant (Elowitz et al., 2002; McAdams and Arkin, 1997; Suel et al., 2006). Consequently, the stochastic model is widely employed for modeling cellular processes (Arkin et al., 1998)

Department of CNS, California Institute of Technology, Pasadena, California.

and is included in numerous software packages (Adalsteinsson et al., 2004; Kierzek, 2002; Vasudeva and Bhalla, 2004).[1] The stochastic model becomes equivalent to the classical law of mass action when the molecular counts of all participating species are large (Ethier and Kurtz, 1986; Kurtz, 1972).

Gillespie's stochastic simulation algorithm (SSA) can be used to model the behavior of SCRNs (Gillespie, 1977). However, simulation of systems of interest often requires an unfeasible amount of computational time. Some work has focused on optimizing simulation of large SCRNs—many different species and reaction channels. For example, certain tricks can improve the speed of deciding which reaction occurs next if there are many possible choices (Gibson and Bruck, 2000). However, for the purposes of this paper we suppose that the number of species and reactions is relatively small, and that it is fundamentally the number of reaction occurrences in a given interval of time that presents the difficulty. Because SSA simulates every single reaction event, simulation is slow when the number of reaction events is large.

On the face of it, simulation should be possible without explicitly modeling every reaction occurrence. In the mass action limit, fast simulation is achieved using numerical ODE solvers. The complexity of the simulation does not scale at all with the actual number of reaction occurrences but with overall simulation time and the concentration of the species. If the volume gets larger without a significant increase in concentration, mass action ODE solvers achieve a profound difference in computation time compared to SSA.[2] Moreover maximum concentration is essentially always bounded, because the model is only valid for solutions dilute enough to be well mixed, and ultimately because of the finite density of matter. However, mass action simulation can only be applied if molecular counts of *all* the species are large. Even one species that maintains a low molecular count and interacts with other species prevents the use of mass action ODE solvers.

Another reason why it seems that it should be possible to simulate stochastic chemical systems quickly, is that for many systems the behavior of interest does not depend crucially upon details of events. For example biochemical networks tend to be robust to variations in concentrations and kinetic parameters (Alon, 2007; Morohashi et al., 2002). If these systems are robust to many kinds of perturbations, including sloppiness in simulation, can we take advantage of this to speed up simulation? For example, can we approach the speed of ODEs but allow molecular counts of some species to be small? Indeed, tau-leaping algorithms (Cao et al., 2006, Gillespie, 2001, 2007, Rathinam et al., 2003), are based on the idea that if we allow reaction propensities to remain constant for some amount of time τ, but therefore deviate slightly from their correct values, we don't have to explicitly simulate every reaction that occurs in this period of time (and can thus "leap" by amount of time τ).

In this paper we formally define robustness of the probability that the system is in a certain state at a certain time to perturbations in reaction propensities. We also provide a method for proving that certain simple systems are robust. We then describe a new approximate stochastic simulation algorithm called bounded tau-leaping (BTL), which naturally follows from our definition of robustness, and provably provides correct answers for robust systems. In contrast to Gillespie's and others' versions of tau-leaping, in each step of our algorithm the leap time, rather than being a function of the current state, is a random variable. This algorithm naturally avoids some pitfalls of tau-leaping: the concentrations cannot become negative, and the algorithm scales to SSA when necessary, in a way that there is always at least one reaction per leap. However, in the cases when there are "opposing reactions" (canceling or partially cancelling each other) other forms of tau-leaping may be significantly faster (Rathinam and El Samad, 2007).

BTL seems more amenable to theoretical analysis than Gillespie's versions (Gillespie, 2001, 2003, Cao et al., 2006), and may thus act as a stand-in for approximate simulation algorithms in analytic investigations. In this paper we use the language and tools of computational complexity theory to formally study how the number of leaps that BTL takes varies with the maximum molecular count m, time span of the simulation t, and volume V. In line with the basic computational complexity paradigm, our analysis is asymptotic and worst-case. "Asymptotic" means that we do not evaluate the exact number of leaps but rather look at the

[1] Some stochastic simulation implementations on the web: Systems Biology Workbench: *http://sbw.sourceforge.net*; BioSpice: *http://biospice.lbl.gov*; Stochastirator: *http://opnsrcbio.molsci.org*; STOCKS: *http://www.sysbio.pl/stocks*; BioNetS: *http://x.amath.unc.edu:16080/BioNetS*; SimBiology package for MATLAB: *http://www.mathworks.com/products/simbiology/index.html*.

[2] As an illustrative example, a prokaryotic cell and a eukaryotic cell may have similar concentrations of proteins but vastly different volumes.

functional form of the dependence of their number on m, t, and V. This is easier to derive and allows for making fundamental distinctions (e.g., an exponential function is fundamentally larger than a polynomial function) without getting lost in the details. "Worst-case" means that we will not study the behavior of our algorithm on any particular chemical system but rather upper-bound the number of leaps our algorithm takes independent of the chemical system. This will allow us to know that no matter what the system we are trying to simulate, it will not be worse than our bound.

In this computational complexity paradigm, we show that indeed robustness helps. We prove an upper bound on the number of steps our algorithm takes that is logarithmic in m, and linear in t and total concentration $C = m/V$. This can be contrasted with the exact SSA algorithm which, in the worst case, takes a number of steps that is linear in m, t, and C. Since a logarithmic dependence is much smaller than a linear one, BTL is provably "closer" to the speed of ODE solvers for mass action systems which have no dependence on m.[3]

Finally we ask whether it is possible to improve upon BTL for robust systems, or did we exhaust the speed gains that can be obtained due to robustness? In the last section of the paper, we connect this question to a conjecture in computer science that is believed to be true. With this conjecture we prove that there are robust systems whose behavior cannot be predicted in fewer computational steps than the number of leaps that BTL makes, ignoring multiplicative constant factors and powers of $\log m$. We believe other versions of tau-leaping have similar worst-case complexities as our algorithm, but proving equivalent results for them remains open.

2. MODEL AND DEFINITIONS

A *Stochastic Chemical Reaction Network* (SCRN) $\mathcal{S}$ specifies a set of N *species* S_i ($i \in \{1, \ldots, N\}$) and M *reactions* R_j ($j \in \{1, \ldots, M\}$). The *state* of $\mathcal{S}$ is a vector $\vec{x} \in \mathbb{N}^N$ indicating the integral molecular counts of the species.[4] A reaction R_j specifies a reactants' stoichiometry vector $\vec{r}_j \in \mathbb{N}^N$, a products' stoichiometry vector $\vec{p}_j \in \mathbb{N}^N$, and a real-valued rate constant $k_j > 0$. We describe reaction stoichiometry using a standard chemical "arrow" notation; for example, if there are three species, the reaction R_j: $S_1 + S_2 \to S_1 + 2S_3$ has reactants vector $\vec{r}_j = (-1, -1, 0)$ and products vector $\vec{p}_j = (1, 0, 2)$. A reaction R_j is *possible* in state $\vec{x}$ if there are enough reactant molecules: $(\forall i)\ x_i - r_{ij} \geq 0$. Then if reaction R_j occurs (or "fires") in state $\vec{x}$, the state changes to $\vec{x} + \vec{\nu}_j$, where $\vec{\nu}_j \in \mathbb{Z}^N$ is the state change vector for reaction R_j defined as $\vec{\nu}_j = \vec{p}_j - \vec{r}_j$. We follow Gillespie and others and allow unary ($S_i \to \ldots$) and bimolecular ($2S_i \to \ldots$ or $S_i + S_{i'} \to \ldots$, $i \neq i'$) reactions only. Sometimes the model is extended to higher-order reactions (van Kampen, 1997), but the merit of this is a matter of some controversy.

Let us fix an SCRN $\mathcal{S}$. Given a starting state $\vec{x}_0$ and a fixed volume V, we can define a continuous-time Markov process we call an *SSA process*[5] $\mathcal{C}$ of $\mathcal{S}$ according to the following stochastic kinetics. Given a current state $\vec{x}$, the propensity function a_j of reaction R_j is defined so that $a_j(\vec{x})dt$ is the probability that one R_j reaction will occur in the next infinitesimal time interval $[t, t + dt)$. If R_j is a unimolecular reaction $S_i \to \ldots$ then the propensity is proportional to the number of molecules of S_i currently present since each is equally likely to react in the next time instant; specifically, $a_j(\vec{x}) = k_j x_i$ for reaction rate constant k_j. If R_j is a bimolecular reaction $S_i + S_{i'} \to \ldots$, where $i \neq i'$, then the reaction propensity is proportional to $x_i x_{i'}$, which is the number of ways of choosing a molecule of S_i and a molecule of $S_{i'}$, since each pair is equally likely to react in the next time instant. Further, the probability that a particular pair reacts in the next time instant is inversely proportional to the volume, resulting in the propensity function $a_j(\vec{x}) = k_j \frac{x_i x_{i'}}{V}$. If R_j is a bimolecular reaction $2S_i \to \ldots$ then the number of ways of choosing two molecules of S_i to react is $\frac{x_i(x_i-1)}{2}$, and the propensity function is $a_j(\vec{x}) = k_j \frac{x_i(x_i-1)}{2V}$.

---

[3] Indeed, the total molecular count m can be extremely large compared to its logarithm—e.g., Avogadro's number $= 6 \times 10^{23}$, while its $\log_2$ is only 79.

[4] $\mathbb{N} = \{0, 1, 2, \ldots\}$ and $\mathbb{Z} = \{\ldots, -1, 0, 1, \ldots\}$.

[5] It is exactly the stochastic process simulated by Gillespie's Stochastic Simulation Algorithm (SSA) (Gillespie, 1977).

Since the propensity function a_j of reaction R_j is defined so that $a_j(\vec{x})dt$ is the probability that one R_j reaction will occur in the next infinitesimal time interval $[t, t + dt)$, state transitions in the SSA process are equivalently described as follows: If the system is in state $\vec{x}$, no further reactions are possible if $\sum a_j(\vec{x}) = 0$. Otherwise, the time until the next reaction occurs is an exponential random variable with rate $\sum_j \alpha_j(\vec{x})$. The probability that next reaction will be a particular R_{j*} is $\alpha_{j*}(\vec{x})/\sum_j \alpha_j(\vec{x})$.

We are interested in predicting the behavior of SSA processes. While there are potentially many different questions that we could be trying to answer, for simplicity we define the *prediction problem* as follows. Given an SSA process $\mathcal{C}$, a time t, a state $\vec{x}$, and $\delta \geq 0$, predict[6] whether $\mathcal{C}$ is in $\vec{x}$ at time t, such that the probability that the prediction is incorrect is at most δ. In other words we are interested in algorithmically generating values of a Bernoulli random variable $I(\vec{x}, t)$ such that the probability that $I(\vec{x}, t) = 1$ when $\mathcal{C}$ is not in $\vec{x}$ at time t plus the probability that $I(\vec{x}, t) = 0$ when $\mathcal{C}$ is in $\vec{x}$ at time t is at most δ. We assume δ is some small positive constant. We can easily extend the prediction problem to a set of states Γ rather than a single target state $\vec{x}$ by asking to predict whether the process is in any of the states in Γ at time t. Since Γ is meant to capture some qualitative feature of the SSA process that is of interest to us, it is called an *outcome*.

By decreasing the volume V (which speeds up all bimolecular reactions), increasing t, or allowing for more molecules (up to some bound m) we are increasing the number of reaction occurrences that we may need to consider. Thus for a fixed SCRN, one can try to upper bound the computational complexity of the prediction problem as a function of V, t, and m. Given a molecular count bound m, we define the *bounded-count prediction problem* as before, but allowing an arbitrary answer if the molecular count exceeds m within time t. Suppose $\mathcal{P}$ is a bounded-count prediction problem with molecular count bound m, error bound δ, about time t and an SSA process in which the volume is V. We then say $\mathcal{P}$ is a (m, t, C, δ)-*prediction problem* where $C = m/V$ is a bound on the maximum concentration.[7] Fixing some small δ, we study how the computational complexity of solving (m, t, C, δ)-prediction problems may scale with increasing m, t, and C. If the (m, t, C, δ)-prediction problem is regarding an outcome Γ consisting of multiple states, we require the problem of deciding whether a particular state is in Γ to be easily solvable. Specifically we require it to be solvable in time at most polylogarithmic in m, which is true for any natural problem.

It has been observed that permitting propensities to deviate slightly from their correct values, allows for much faster simulation, especially if the molecular counts of some species are large. This idea forms the basis of approximate stochastic simulation algorithms such as tau-leaping (Gillespie, 2001). As opposed to the exact SSA process described above, consider letting the propensity function vary stochastically. Specifically, we define new propensity functions $a'_j(\vec{x}, t) = \xi_j(t)a_j(\vec{x})$ where $\{\xi_j(t)\}$ are random variables indexed by reaction and time. The value of $\xi_j(t)$ describes the deviation from the correct propensity of reaction R_j at time t, and should be close to 1. For any SSA process $\mathcal{P}$ we can define a new stochastic process called a *perturbation* of $\mathcal{P}$ through the choice of the distributions of $\{\xi_j(t)\}$. Note that the new process may not be Markov, and may not possess Poisson transition probabilities. If there is a $0 < \rho < 1$ such that $\forall j, t$, $(1 - \rho) \leq \xi_j(t) \leq (1 + \rho)$, then we call the new process a ρ-*perturbation*. There may be systems exhibiting behavior such that any slight inexactness in the calculation of propensities quickly gets amplified and results in qualitatively different behavior. However, for some processes, if ρ is a small constant, the ρ-perturbation may be a good approximation of the SSA process. That a ρ-perturbation is bounded multiplicatively (i.e., that $\xi_j(t)$ acts multiplicatively) corresponds to our intuitive notion that proportionally larger deviations are required to have an effect if the affected propensity is large.

We now define our notion of robustness. Intuitively, we want the prediction problem to not be affected even if reaction propensities vary slightly. Formally, we say an SSA process $\mathcal{C}$ *is* (ρ, δ)-*robust* with respect to state $\vec{x}$ at time t if for any ρ-deviating process $\tilde{\mathcal{C}}$ based on $\mathcal{C}$, the probability of being in $\vec{x}$ at time t

[6]We phrase the prediction problem in terms appropriate for a simulation algorithm. An alternative formulation would be the problem of estimating the probability that the SSA process is in $\vec{x}$ at time t. To be able to solve this problem using a simulation algorithm we can at most require that with probability at least δ_1 the estimate is within δ_2 of the true probability for some constants $\delta_1, \delta_2 > 0$. This can be attained by running the simulation algorithm a constant number of times.

[7]Maximum concentration C is a more natural measure of complexity compared to V because similar to m and t, computational complexity increases as C increases.

is within plus or minus δ of the corresponding probability for $\mathcal{C}$. This definition can be extended to an outcome Γ similar to the definition on the prediction problem. Finally, we say an SSA process $\mathcal{C}$ is (ρ, δ)-*robust with respect to a prediction problem* (or bounded-count prediction problem) $\mathcal{P}$ if $\mathcal{C}$ is (ρ, δ)-robust with respect to the same state (or outcome) as specified in $\mathcal{P}$, at the same time t as specified in $\mathcal{P}$.

For simplicity, we often use asymptotic notation. The notation $O(1)$ is used to denote an unspecified positive constant. This constant is potentially different every time the expression $O(1)$ appears.

3. ROBUSTNESS EXAMPLES

In this section, we elucidate our notion of robustness by considering some examples. In general, the question of whether a given SSA process is (ρ, δ)-robust for a particular outcome seems a difficult one. The problem is especially hard because we have to consider every possible ρ-perturbation—thus, we may not even be able to give an approximate characterization of robustness by simulation with SSA. However, we can characterize the robustness of certain (simple) systems.

For an SSA process or ρ-perturbation $\mathcal{C}$, and outcome Γ, let $F^{\Gamma}(\mathcal{C}, t)$ be the probability of being in Γ at time t. Consider the SCRN shown in Figure 1a. We start with 300 molecules of S_1 and S_3 each, and are interested in the outcome Γ of having at least 150 molecules of S_4. The dashed line with circles shows F for the correct SSA process $\mathcal{C}$. (All plots of F are estimated from 10^3 SSA runs.) The two dashed lines without circles show F for two "extremal" ρ-perturbations: $\tilde{\mathcal{C}}^{+\rho}$ with constant $\xi_j(t) = 1 + \rho$, and $\tilde{\mathcal{C}}^{-\rho}$ with constant $\xi_j(t) = 1 - \rho$. What can we say about other ρ-perturbations, particularly where the $\xi_j(t)$ have much more complicated distributions? It turns out that for this SCRN and Γ, we can prove that any ρ-perturbation falls within the bounds set by the two extremal ρ-perturbations $\tilde{\mathcal{C}}^{-\rho}$ and $\tilde{\mathcal{C}}^{+\rho}$. Thus, F for any ρ-perturbation falls within the dashed lines. Formally, $\mathcal{C}$ is monotonic with respect to Γ using the definition of monotonicity in Appendix A.3. This is easily proven by Lemma A.5 because every species is a reactant in at most one reaction. Then by Lemma A.4, $F^{\Gamma}(\tilde{\mathcal{C}}^{-\rho}, t) \leq F^{\Gamma}(\tilde{\mathcal{C}}, t) \leq F^{\Gamma}(\tilde{\mathcal{C}}^{+\rho}, t)$ for any ρ-perturbation $\tilde{\mathcal{C}}$.

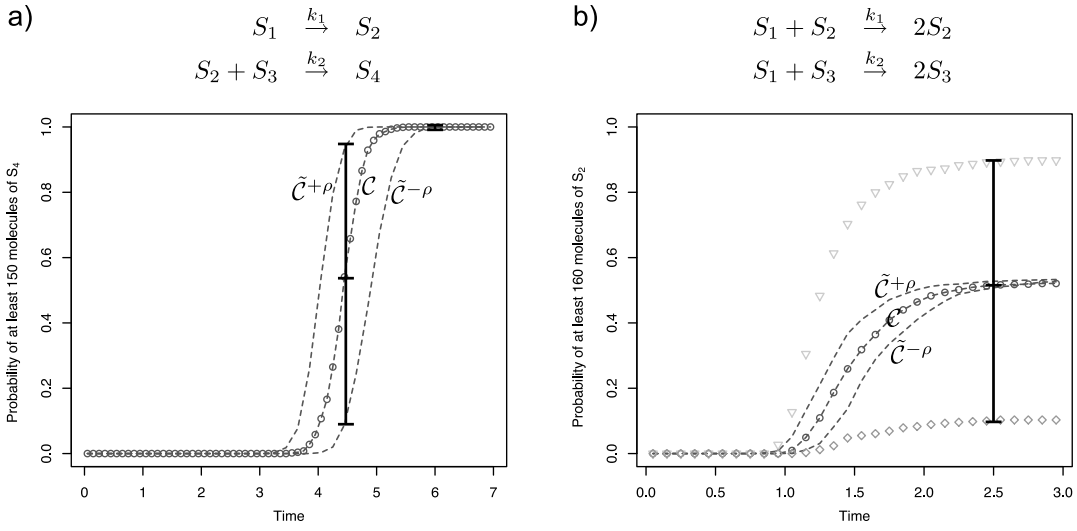

FIG. 1. Examples of SCRNs exhibiting contrasting degrees of robustness. The SSA process $\mathcal{C}$ and outcome Γ are defined for the two systems as follows: **(a)** Rate constants: $k_1 = 1$, $k_2 = 0.001$; start state: $\vec{x_0} = (300, 0, 300, 0)$; outcome Γ: $x_4 \geq 150$. **(b)** Rate constants: $k_1 = 0.01$, $k_2 = 0.01$; start state: $\vec{x_0} = (300, 10, 10)$; outcome Γ: $x_2 \geq 160$. Plots show $F^{\Gamma}(\cdot, t)$ for an SSA process or ρ-perturbation estimated from 10^3 SSA runs. **(Dashed line with circles)** Original SSA process $\mathcal{C}$. **(Dashed lines without circles)** The two extremal ρ-perturbations: $\tilde{\mathcal{C}}^{+\rho}$ with constant $\xi_j(t) = 1 + \rho$, and $\tilde{\mathcal{C}}^{-\rho}$ with constant $\xi_j(t) = 1 - \rho$. For SCRN (b), we also plot $F^{\Gamma}(\cdot, t)$ for a ρ-perturbation with constant $\xi_1(t) = 1 + \rho$, $\xi_2(t) = 1 - \rho$ (triangles), or constant $\xi_1(t) = 1 - \rho$, $\xi_2(t) = 1 + \rho$ (diamonds). Perturbation parameter $\rho = 0.1$ throughout.

To see how the robustness of this system can be quantified using our definition of (ρ, δ)-robustness, first consider two time points $t = 4.5$ and $t = 6$. At $t = 4.5$, the probability that the correct SSA process $\mathcal{C}$ has produced at least 150 molecules of S_4 is slightly more than 0.5. The corresponding probability for ρ-perturbations of $\mathcal{C}$ can be no larger than about 0.95 and no smaller than about 0.1. Thus $\mathcal{C}$ is (ρ, δ)-robust with respect to outcome Γ at time $t = 4.5$ for $\rho = 0.1$ and δ approximately 0.45, but not for smaller δ. On the other hand at $t = 6$, the dashed lines are essentially on top of each other, resulting in a tiny δ. In fact, δ is small for all times less than approximately 3.5 or greater than approximately 5.5.

What information did we need to be able to measure (ρ, δ)-robustness? Processes $\tilde{\mathcal{C}}^{-\rho}$ and $\tilde{\mathcal{C}}^{+\rho}$ are simply $\mathcal{C}$ scaled in time. Thus knowing how $F^{\Gamma}(\mathcal{C}, t)$ varies with t allows one to quantify (ρ, δ)-robustness at the various times; $F^{\Gamma}(\mathcal{C}, t)$ can be estimated from multiple SSA runs of $\mathcal{C}$ as in Figure 1. Intuitively, $\mathcal{C}$ is (ρ, δ)-robust for small δ at all times t when $F^{\Gamma}(\mathcal{C}, t)$ does not change quickly with t (see Appendix A.3). For systems that are not monotonic, knowing how $F^{\Gamma}(\mathcal{C}, t)$ varies with time may not help with evaluating (ρ, δ)-robustness.

Indeed, for a contrasting example, consider the SCRN in Figure 1b. We start with 300 molecules of S_1, 10 molecules of S_2, and 10 molecules of S_3, and we are interested in the outcome of having at least 160 molecules of S_2. Since S_1 is a reactant in both reactions, Lemma A.5 cannot be used. In fact, the figure shows two ρ-perturbations (triangles and diamonds) that clearly escape from the boundaries set by the dashed lines. The triangles show F for the ρ-perturbation where the first reaction is maximally sped up and the second reaction is maximally slowed down. (Vice versa for the diamonds.) For characterization of the robustness of this system via (ρ, δ)-robustness, consider the time point $t = 2.5$. The probability of having at least 160 molecules of S_2 in the correct SSA process $\mathcal{C}$ is around 0.5. However, this probability for ρ-perturbations of $\mathcal{C}$ can deviate by at least approximately 0.4 upward and downward as seen by the two ρ-perturbations (triangles and diamonds). Thus at this time the system is not (ρ, δ)-robust for δ approximately 0.4. What about other ρ-deviations? It turns out that for this particular system, the two ρ-perturbations corresponding to the triangles and diamonds bound F in the same way that $\tilde{\mathcal{C}}^{-\rho}$ and $\tilde{\mathcal{C}}^{+\rho}$ bounded F in the first example (exercise left to the reader). Nonetheless, for general systems that are not monotonic it is not clear how one can find such bounding ρ-perturbation and in fact they likely would not exist.

Of course, there are other types of SSA process that are not like either of the above examples: e.g., systems that are robust at many times but not monotonic. General ways of evaluating robustness of such systems remains an important open problem.

Finally, it is important to note that quantifying the robustness of SSA processes, even monotonic ones, seems to require computing many SSA runs. This is self-defeating when in practice one wants to show that the given SSA process is (ρ, δ)-robust in order to justify the use of an approximate simulation algorithm to quickly simulate it. In these cases, we have to consider (ρ, δ)-robustness a theoretical notion only. Note, however, that it may be much easier to show that a system is *not* robust by comparing the simulation runs of different ρ-perturbations, since the runs can be quickly obtained using fast approximate simulation algorithms such as that presented in the next section.

4. BOUNDED TAU-LEAPING

4.1. The algorithm

We argued in the Introduction that sloppiness can allow for faster simulation. In this section we give a new approximate stochastic simulation algorithm called *bounded tau-leaping* (BTL) that simulates exactly a certain ρ-perturbation rather than the original SSA process. Consequently, the algorithm solves the prediction problem with allowed error δ for (ρ, δ)-robust SSA processes.

The algorithm is a variant of existing tau-leaping algorithms (Gillespie, 2007). However, while other tau-leaping algorithms have an implicit notion of robustness, BTL is formally compatible with our explicit definition. As we'll see below, our algorithm also has certain other advantages over many previous tau-leaping implementations: it naturally disallows negative concentrations and scales to SSA in a manner that there is always at least one reaction per leap. It also seems easier to analyze formally; obtaining a result similar to Theorem 4.1 is an open question for other tau-leaping variants.

BTL has overall form typical of tau-leaping algorithms. Rather than simulating every reaction occurrence explicitly as per the SSA, BTL divides the simulation into leaps which group multiple reaction events. The propensities of all of the reactions are assumed to be fixed throughout the leap. This is obviously an approximation since each reaction event affects molecular counts and therefore the propensities. However, this approximation is useful because simulating the system with the assumption that propensities are fixed turns out to be much easier. Instead of having to draw random variables for each reaction occurrence, the number of random variables drawn to determine how many reaction firings occurred in a leap is independent of the number of reaction firings. Thus we effectively "leap" over all of the reactions within a leap in few computational steps. If molecular counts do not change by much within a leap then the fixed propensities are close to their correct SSA values and the approximation is good.

Our definition of a ρ-perturbation allows us to formally define "good." We want to guarantee that the approximate process that tau-leaping actually simulates is a ρ-perturbation of the exact SSA process. We can achieve this as follows. If $\vec{x}$ is the state on which the leap started, throughout the leap the simulated reaction propensities are fixed at their SSA propensities on $\vec{x}$: $a_j(\vec{x})$. Then for any state $\vec{y}$ within the leap we want the correct SSA propensities $a_j(\vec{y})$ to satisfy the following *ρ-perturbation constraint* ($0 < \rho < 1$): $(1-\rho)a_j(\vec{y}) \leq a_j(\vec{x}) \leq (1+\rho)a_j(\vec{y})$. As soon as we reach a state $\vec{y}$ for which this constraint is violated, we start a new leap at $\vec{y}$ which will use simulated reaction propensities fixed at $a_j(\vec{y})$. This ensures that at any time in the simulation, there is some $(1-\rho) \leq \xi_j(t) \leq (1+\rho)$ such that multiplying the correct SSA propensity of reaction R_j by $\xi_j(t)$ yields the propensity of R_j that the simulation algorithm is actually using. Therefore, we actually simulate a ρ-perturbation, and for (ρ, δ)-robust SSA processes, the algorithm can be used to provably solve the prediction problem with error δ.

Can we implement this simulation quickly, and, as promised, do little computation per leap? Note that in order to limit the maximum propensity deviation in a leap, we need to make the leap duration be a random variable dependent upon the stochastic events in the leap. If we evaluate $a_j(\vec{y})$ after each reaction occurrence in a leap to verify the satisfaction of the ρ-perturbation constraint, we do not save time over SSA. However, we can avoid this by using a stricter constraint we call the *$\{\varepsilon_{ij}\}$-perturbation constraint* ($0 < \varepsilon_{ij} < 1$), defined as follows. If the leap starts in state $\vec{x}$, reaction R_j is allowed to change the molecular count of species S_i by at most plus or minus $\varepsilon_{ij} x_i$ within a leap. Again, as soon as we reach a state $\vec{y}$ where this constraint is violated, we start a new leap at $\vec{y}$.[8]

For any ρ, we can find a set of $\{\varepsilon_{ij}\}$ bounds such that satisfying the $\{\varepsilon_{ij}\}$-perturbation constraint satisfies the ρ-perturbation constraint. In Appendix A.1 we show that for any SCRN, the ρ-perturbation constraint is satisfied if $\varepsilon_{ij} \leq \frac{3}{4M}\left(1 - \sqrt{\frac{1+\rho/9}{1+\rho}}\right)$, where M is the number of reactions in the SCRN.

Simulating a leap such that it satisfies the $\{\varepsilon_{ij}\}$-perturbation constraint is easy and only requires drawing M gamma and $M-1$ binomial random variables. Suppose the leap starts in state $\vec{x}$. For each reaction R_j, let b_j be the number of times R_j needs to fire to cause a violation of the $\{\varepsilon_{ij}\}$ bounds for some species. Thus b_j is the smallest positive integer such that $|b_j \nu_{ij}| > \varepsilon_{ij} x_i$ for some S_i. To determine τ, the duration of the leap, we do the following. First we determine when each reaction R_j would occur b_j times, by drawing from a gamma distribution with shape parameter b_j and rate parameter a_j. This generates a time τ_j for each reaction. The leap ends as soon as some reaction R_j occurs b_j times; thus to determine the duration of the leap τ we take the minimum of the τ_j's. At this point, we know that the first-violating reaction R_{j^*} (the one with the minimum τ_{j^*}) occurred b_{j^*} times. But we also need to know how many times the other reactions occur. Consider any other reaction R_j ($j \neq j^*$). Given that the b_jth occurrence of reaction R_j would have happened at time τ_j had the leap not ended, we need to distribute the other $b_j - 1$ occurrences to determine how many happen before time τ. The number of occurrences at time τ is given by the binomial distribution with number of trials $b_j(\vec{x}) - 1$ and success probability τ/τ_j. This enables us to define BTL as shown in Figure 2.

The algorithm is called "bounded" tau-leaping because the deviations of reaction propensities within a leap are always bounded according to ρ. This is in contrast with other tau-leaping algorithms, such

[8]An added benefit of providing $\{\varepsilon_{ij}\}$ bounds rather than ρ as a parameter to the BTL algorithm is that it allows flexibility on the part of the user to assign less responsibility for a violation to a reaction that is expected to be fast compared to a reaction that is expected to be slow. This may potentially speed up the simulation, while still preserving the ρ-perturbation constraint.

0. Initialize with time $t = t_0$ and the system's state $\vec{x} = \vec{x_0}$.
1. With the system in state $\vec{x}$ at time t, evaluate all the propensities a_j, and determine firing bounds b_j for all possible reactions, where b_j is the smallest positive integer such that $|b_j \nu_{ij}| > \varepsilon_{ij} x_i$ for some S_i.
2. Generate violating times $\tau_j \sim \text{Gamma}(b_j, a_j)$ for all possible reactions.
3. Find the first-violating reaction and set the step size to the time of the first violation: let $j^* = \text{argmin}_j\{\tau_j\}$ and $\tau = \tau_{j^*}$.
4. Determine the number of times each possible reaction occurred in interval τ: for $j \neq j^*$, $n_j \sim \text{Binomial}(b_j - 1, \tau/\tau_j)$; for j^*, $n_{j^*} = b_{j^*}$.
5. Effect the leap by replacing $t \leftarrow t + \tau$ and $\vec{x} \leftarrow \vec{x} + \sum_j \vec{\nu_j} n_j$.
6. Record $(\vec{x}, t)$ as desired. Return to Step 1, or else end the simulation.

FIG. 2. The bounded tau-leaping (BTL) algorithm. The algorithm is given the SCRN, the initial state $\vec{x_0}$, the volume V, and a set of perturbation bounds $\{\varepsilon_{ij}\} > 0$. If the state at a specific time t_f is desired, the algorithm checks if $t + \tau > t_f$ in step (3), and if so uses $\tau = t_f - \tau$, and treats all reactions as not first-violating in step (4). Gamma(n, λ) is a gamma distribution with shape parameter n and rate parameter λ. Binomial(n, p) is a binomial distribution with number of trials n and success probability p.

as Gillespie's (Cao et al., 2006), in which the deviations in reaction propensities are small with high probability, but not always, and in fact can get arbitrarily high if the simulation is long enough. This allows BTL to satisfy our definition of a ρ-perturbation, and permits easier analysis of the behavior of the algorithm.

As any algorithm exactly simulating a ρ-perturbation would, BTL naturally avoids negative concentrations. Negative counts can occur only if an impossible reaction happens—in some state $\vec{x}$ reaction R_j fires for which $a_j(\vec{x}) = 0$. But since in a ρ-perturbation propensity deviations are multiplicative, in state $\vec{x}$, $a'_j(\vec{x}, t) = \xi_j(t) a_j(\vec{x}) = 0$ and so R_j cannot occur. Further, no matter how small the $\{\varepsilon_{ij}\}$ bounds are, there is always at least one reaction per leap and thus BTL cannot take more steps than SSA.

On the negative side, in certain cases the BTL algorithm can take many more leaps than Gillespie's tau-leaping (Cao et al., 2006; Gillespie, 2001, 2003) and other versions. Consider the case where there are two fast reactions that partially undo each others' effect (for example the reactions may be reverses of each other). While both reactions may be occurring very rapidly, their propensities may be very similar (Rathinam and El Samad, 2007). Gillespie's tau-leaping will attempt to leap to a point where the molecular counts have changed enough according to the *averaged* behavior of these reactions. However, our algorithm considers each reaction separately and leaps to the point where the first reaction violates the bound on the change in a species in the absence of the other reactions. Thus in this situation our algorithm would perform unnecessarily many leaps for the desired level of accuracy.

4.2. *Upper bound on the number of leaps*

Suppose we fix some SCRN of interest, and run BTL on different initial states, volumes, and lengths of simulated time. How does varying these parameters change the number of leaps taken by BTL? In this section, we prove that no matter what the SCRN is, we can upper bound the number of leaps as a function of the total simulated time t, the volume V, and the maximum total molecular count m encountered during the simulation. For simplicity, we assume that all the ε_{ij} are equal to some global ε. (Alternatively, the theorem and proof can be easily changed to use min/max $\{\varepsilon_{ij}\}$ values where appropriate.)

Theorem 4.1. *For any SCRN $\mathcal{S}$ with M species, any ε such that $0 < \varepsilon < 1/(12M)$, and any $\delta > 0$, there are constants $c_1, c_2, c_3 > 0$ such that for any bounds on time t and total molecular count m, for any volume V and any starting state, after $c_1 \log m + c_2 t\,(C + c_3)$ leaps where $C = m/V$, either the bound on time or the bound on total molecular count will be exceeded with probability at least $1 - \delta$.*

Proof. The proof is presented in Appendix A.2. ■

Note that the upper bound on ε implies that the algorithm is exactly simulating some ρ-perturbation (see previous section).

Intuitively, a key idea in the proof of the theorem is that the propensity of a reaction decreasing a particular species is linear to the amount of that species (since the species must appear as a reactant). This allows us to bound the decrease of any species if a leap is short. Actually this implies that a short leap probably increases the amount of some species by a lot (some species must cause a violation—if not by a decrease it must be by an increase). This allows us to argue that if we have a lot of long leaps we exceed our time bound t and if we have a lot of short leaps we exceed our bound on total molecular count m. In fact because the effect of leaps is multiplicative, logarithmically many short leaps are enough to exceed m.

It is informative to compare this result with exact SSA, which in the worst case takes $O(1)\,m\,t\,(C+O(1))$ steps, since each reaction occurrence corresponds to an SSA step and the maximum reaction propensity is $k_j m^2/V$ or $k_j m$. Since m can be very large, the speed improvement can be profound.

We believe, although it remains to be proven, that other versions of tau-leaping (Gillespie, 2007) achieve the same asymptotic worst case number of leaps as our algorithm.

How much computation is required per each leap? Each leap involves arithmetic operations on the molecular counts of the species, as well as drawing from a gamma and binomial distributions. Since there are fast algorithms for obtaining instances of gamma and binomial random variables (Ahrens and Dieter, 1978; Kachitvichyanukul and Schmeiser, 1988), we do not expect a leap of BTL to require much more computation than other forms of tau-leaping, and should not be a major contributor to the total running time. Precise bounds are dependent on the model of computation. (In the next section, we state reasonable asymptotic bounds on the computation time per leap for a randomized Turing machine implementation of BTL.)

5. ON THE COMPUTATIONAL COMPLEXITY OF THE PREDICTION PROBLEM FOR ROBUST SSA PROCESSES

What is the computational complexity inherent in the prediction problem for robust SSA processes, and how close does BTL come to the optimum computation time? In order to be able to consider these questions formally, we specify our model of computation as being randomized Turing machines (see below). Then in terms of maximum total molecular count m, $\log m$ computation time is required to simply read in the initial state of the SSA process and target state of the prediction problem. We say that computation time polylogarithmic in m is *efficient in m*. What about the length of simulated time t and maximum concentration C? We have shown that the number of leaps that BTL takes scales at most linearly with t and C. However, for some systems there are analytic shortcuts to determining the probability of being in Γ at time t. For instance the "exponential decay" SCRN consisting of the single reaction $S_1 \to S_2$ is easily solvable analytically (Malek-Mansour and Nicolis, 1975). The calculation of the probability of being in any given state at any given time t (among other questions) can be solved in time that grows minimally with t and C. In this section we prove that despite such examples, for any algorithm solving prediction problems for robust SSA processes, there are prediction problems about such processes that cannot be solved faster than linear in t and C, assuming a reasonable conjecture in computational complexity theory. We prove this result for any algorithm that is efficient in m. We finally argue, with certain caveats regarding implementing BTL on a Turing machine, that as an algorithm for solving prediction problems for robust SSA processes, BTL is asymptotically optimal among algorithms efficient in m because its computation time scales linearly with t and C.

In order to prove formal lower bounds on the computational complexity of the prediction problem, we must be specific about our computation model. We use the standard model of computation which captures stochastic behavior: randomized Turing machines (TM). A randomized TM is a non-deterministic TM[9] allowing multiple possible transitions at a point in a computation. The actual transition taken is uniform

[9]Arbitrary finite number of states and tapes. Without loss of generality, we can assume a binary alphabet.

over the choices (for equivalent formalizations, see Sipser, 1997). We say a given TM on a given input runs in computational time t_{tm} if there is no set of random choices that makes the machine run longer.

We want to show that for some SCRNs, there is no method of solving the prediction problem fast, no matter how clever we are. We also want these stochastic processes to be robust despite having difficult prediction problems. We use the following two ideas. First, a method based on Angluin et al. (2006) shows that predicting the output of given randomized TMs can be done by solving a prediction problem for certain robust SSA processes. Second, an open conjecture, but one that is strongly compatible with the basic beliefs of computational complexity theory, bounds how quickly the output of randomized TMs can be determined.

Computational complexity theory concerns measuring how the computational resources required to solve a given problem scale with input size n (in bits). The two most prevalent efficiency measures are time and space—the number of TM steps and the length of the TM tape required to perform the computation. We say a Boolean function $f(x)$ is *probabilistically computable* by a TM M in time $t(n)$ (where $n = |x|$) and space $s(n)$ if $M(x)$ runs in time $t(n)$ using space at most $s(n)$, and with probability at least $2/3$ outputs $f(x)$.[10] A basic tenet of computational complexity is that allowing asymptotically more computation time $t(n)$ always expands the set of problems that can be solved. Thus it is widely believed that for any (reasonable) $t(n)$, there are "$t(n)$-hard" functions that can be probabilistically computed in $t(n)$ time, but not in asymptotically smaller time.[11] For our argument we will need such a $t(n)$-hard function, but one that does not require too much space. Formally we assume the following *hierarchy conjecture*:

Conjecture 5.1 ([Probabilistic, Space-Limited] Time Hierarchy). *For any $\alpha < 1$, and polynomials $t(n)$ and $s(n)$ such that $t(n)^\alpha$ and $s(n)$ are at least linear, there are Boolean functions that can be probabilistically computed within time and space bounds bounds $t(n)$ and $s(n)$, but not in time $O(1)t(n)^\alpha$ (with unrestricted space usage).*

Intuitively, we take a Boolean function that requires $t(n)$ time and embed it in a chemical system in such a way that solving the prediction problem is equivalent to probabilistically computing the function. The conjecture implies that we cannot solve the prediction problem fast enough to allow us to solve the computational problem faster than $t(n)$. Further, since the resulting SSA process is robust, the result lower-bounds the computational complexity of the prediction problem for robust processes. Note that we need a time hierarchy conjecture that restricts the space usage and talks about probabilistic computation because it is impossible to embed a TM computation in an SCRN such that its computation is error free (Soloveichik et al., 2008), and further such embedding seems to require more time as the space usage increases.

The following theorem lower-bounds the computational complexity of the prediction problem. The bound holds even if we restrict ourselves to robust processes. It shows that this computational complexity is at least linear in t and C, as long as the dependence on m is at most polylogarithmic. It leaves the possibility that there are algorithms for solving the prediction problem that require computation time more than polylogarithmic in m but less than linear in t or C. Let the prediction problem be specified by giving the SSA process (via the initial state and volume), the target time t, and the target outcome Γ in some standard encoding such that whether a state belongs to Γ can be computed in time polylogarithmic in m.

Theorem 5.1. *Fix any perturbation bound $\rho > 0$ and $\delta > 0$. Assuming the hierarchy conjecture (Conjecture 5.1), there is an SCRN $\mathcal{S}$ such that for any prediction algorithm $\mathcal{A}$ and constants $c_1, c_2, \beta, \eta, \gamma > 0$, there is an SSA process $\mathcal{C}$ of $\mathcal{S}$ and a $(m, t, C, 1/3)$-prediction problem $\mathcal{P}$ of $\mathcal{C}$ such that $\mathcal{A}$ cannot solve $\mathcal{P}$ in computational time $c_1 (\log m)^\beta t^\eta (C + c_2)^\gamma$ if $\eta < 1$ or $\gamma < 1$. Further, $\mathcal{C}$ is (ρ, δ)-robust with respect to $\mathcal{P}$.*

[10] Any other constant probability bounded away from $1/2$ will do just as well: to achieve a larger constant probability of being correct, we can repeat the computation a constant number of times and take majority vote.

[11] If we do not allow any chance of error, the corresponding statement is proven as the (deterministic) time hierarchy theorem (Sipser, 1997). Also see Barak (2002) and Fortnow and Santhanam (2006) for progress in proving the probabilistic version.

Proof. The proof is presented in Appendix A.5. ■

With the above theorem demarcating a boundary of what is possible, the natural question is how close to optimal does BTL come? In the previous section, we have derived an upper bound on the number of leaps that our algorithm takes. However, we need to address how the idealized bounded-tau leaping algorithm presented in Section 4.1 can be implemented on a randomized TM which allows only finite precision arithmetic and a restricted model of randomness generation. We have to deal with round-off error and approximate gamma and binomial random number generators, whose effect on the probability of outcome is difficult to track formally. Further, the computational complexity of these operations is a function of the bits of precision and is complicated to rigorously bound.

In Appendix A.6, we argue that BTL on a randomized TM runs in total computation time

$$O(1)((\log(m))^{O(1)} + l)\, t\, (C + O(1)) \tag{1}$$

where, in each leap, polylogarithmic time in m is required for arithmetic manipulation of molecular counts, and l captures the extra computation time required for the real number operations and drawing from the gamma and binomial distributions. Here l is potentially a function of m, V, t, and the bits of precision used. Assuming efficient methods for drawing the random variables, l is likely very small compared to the total number of leaps. So in as far as l in (Equation (1)) can be neglected, and further assuming we can ignore errors introduced due to finite precision arithmetic and approximate random number generation, bounded-tau leaping is asymptotically optimal up to multiplicative constants and powers of $\log m$ among algorithms efficient in m.

6. DISCUSSION

The behavior of many stochastic chemical reaction networks does not depend crucially on getting the reaction propensities exactly right, prompting our definition of ρ-perturbations and (ρ, δ)-robustness. A ρ-perturbation of an SSA process is a stochastic process with stochastic deviations of the reaction propensities from their correct SSA values. These deviations are multiplicative and bounded between $1-\rho$ and $1+\rho$. If we are concerned with how likely it is that the SSA process is in a given state at a given time, then (ρ, δ)-robustness captures how far these probabilities may deviate for a ρ-perturbation.

We formally showed that predicting the behavior of robust processes does not require simulation of every reaction event. Specifically, we described a new approximate simulation algorithm called bounded tau-leaping (BTL) that simulates a certain ρ-perturbation as opposed to the exact SSA process. The accuracy of the algorithm in making predictions about the state of the system at given times is guaranteed for (ρ, δ)-robust processes. We proved an upper bound on the number of leaps of BTL that helps explain the savings over SSA. The bound is a function of the desired length of simulated time t, volume V, and maximum molecular count encountered m. This bound scales linearly with t and $C = m/V$, but only logarithmically with m, while the total number of reactions (and therefore SSA steps) may scale linearly with t, C, and m. When total concentration is limited, but the total molecular count is large, this represents a profound improvement over SSA. Because the number of BTL leaps scales only logarithmically with m, BTL asymptotically nears the speed of mass action ODE solvers—which have no dependence on m. We also argue that asymptotically as a function of t and C our algorithm is optimal in as far as no algorithm can achieve sublinear dependence of the number of leaps on t or C. This result is proven based on a reasonable assumption in computational complexity theory. Unlike Gillespie's tau-leaping (Cao et al., 2006), our algorithm seems better suited to theoretical analysis. Thus while we believe other versions of tau-leaping have similar worst-case running times, the results analogous to those we obtain for BTL remain to be proved.

Our results can also be seen to address the following question. If concerned solely with a particular outcome rather than with the entire process trajectory, can one always find certain shortcuts to determine the probability of the outcome without performing a full simulation? Since our lower bound on computation time scales linearly with t, it could be interpreted to mean that, except in problem-specific cases, there is no shorter route to predicting the outcomes of stochastic chemical processes than via simulation. This negative result holds even restricting to the class of robust SSA processes.

While the notion of robustness is a useful theoretical construct, how practical is our definition in deciding whether a given system is suitable to approximate simulation via BTL or not? We prove that for the class of monotonic SSA processes, robustness is guaranteed at all times when in the SSA process the outcome probability is stable over an interval of time determined by ρ. However, it is not clear how this stability can be determined without SSA simulation. Even worse, few systems of interest are monotonic. Consequently, it is compelling to develop techniques to establish robustness for more general classes of systems. A related question is whether it is possible to connect our notion of robustness to previously studied notions in mass action stability analysis (Horn and Jackson, 1972; Sontag, 2007).

7. APPENDIX

A.1. Enforcing the ρ-perturbation constraint by the $\{\varepsilon_{ij}\}$-perturbation constraint

Recall that in Section 4.1 we introduced two constraints bounding the number of reaction events within a leap. If $\vec{x}$ is the state at the beginning of the leap, the ρ-perturbation constraint is satisfied if for every reaction R_j, $(1-\rho)a_j(\vec{y}) \leq a_j(\vec{x}) \leq (1+\rho)a_j(\vec{y})$ for any state $\vec{y}$ within the leap. The $\{\varepsilon_{ij}\}$-perturbation constraint is satisfied if no reaction R_j changes the molecular count of species S_i by more than plus or minus $\varepsilon_{ij}x_i$ within the leap. For a given ρ, we would like to find an appropriate $\{\varepsilon_{ij}\}$-perturbation constraint to use in the BTL algorithm such that we satisfy the ρ-perturbation constraint, thereby ensuring that we are exactly simulating some ρ-perturbation. To avoid making the $\{\varepsilon_{ij}\}$-perturbation constraint tighter than necessary requires knowledge of the exact reactions in the given SCRN. Nevertheless, worst-case analysis below shows that setting $\varepsilon_{ij} \leq \frac{3}{4M}(1-\sqrt{\frac{1+\rho/9}{1+\rho}})$ works for any SCRN of M reactions.

If $\varepsilon_{ij} = \varepsilon$ then, for any SCRN with M reactions, the maximum change of any species S_i within a leap allowed by the $\{\varepsilon_{ij}\}$-perturbation constraint is plus or minus $M\varepsilon x_i$. We want to find an $\varepsilon > 0$ such that if the changes to all species stay within the $M\varepsilon$ bounds, then no reaction violates the ρ-perturbation constraint. Consider a bimolecular reaction R_j: $2S_i \to \ldots$ first. The algorithm simulates its propensity as $a_j(\vec{x}) = k_j x_i(x_i-1)/V$ throughout the leap. If $x_i = 0$ or 1, then $a_j(\vec{x}) = 0$, and as long as $M\varepsilon < 1$, then still $a_j(\vec{y}) = 0$, satisfying the ρ-perturbation constraint for R_j. Otherwise, if $x_i \geq 2$, then the SSA propensity at state $\vec{y}$ within the leap is $a_j(\vec{y}) = k_j y_i(y_i-1)/V \leq k_j(1+M\varepsilon)x_i((1+M\varepsilon)x_i-1)/V$, and so the left half of the ρ-perturbation constraint $(1-\rho)a_j(\vec{y}) \leq a_j(\vec{x})$ is satisfied if $(1-\rho)(1+M\varepsilon)x_i((1+M\varepsilon)x_i-1) \leq x_i(x_i-1)$. Similarly, the right half of the ρ-perturbation constraint $a_j(\vec{x}) \leq (1+\rho)a_j(\vec{y})$ is satisfied if $(1+\rho)(1-M\varepsilon)x_i((1-M\varepsilon)x_i-1) \geq x_i(x_i-1)$. These inequalities are satisfied for $x_i \geq 2$ when $\varepsilon \leq \frac{3}{4M}(1-\sqrt{\frac{1+\rho/9}{1+\rho}})$ (which also ensures that $M\varepsilon < 1$).

In a likewise manner, for a unimolecular reaction R_j: $S_i \to \ldots$, the ρ-perturbation constraint is satisfied if $(1-\rho)(1+M\varepsilon)x_i \leq x_i$ and $(1+\rho)(1-M\varepsilon)x_i \geq x_i$, and for a bimolecular reaction R_j: $S_i + S_{i'} \to \ldots$, the constraint is satisfied if $(1-\rho)(1+M\varepsilon)^2 x_i x_{i'} \leq x_i x_{i'}$ and $(1+\rho)(1-M\varepsilon)^2 x_i x_{i'} \geq x_i x_{i'}$. It is easy to see that setting ε as above also satisfies the inequalities for these reaction types.

Throughout the paper we assume that ρ, ε or $\{\varepsilon_{ij}\}$ are fixed and most of our asymptotic results do not show dependence on these parameters. Nonetheless, we can show that for a fixed SCRN and for small enough ρ, ε can be within the range $O(1)\rho \leq \varepsilon \leq O(1)\rho$ and thus scales linearly with ρ. Therefore, in asymptotic results, the dependence on ε and ρ can be interchanged. Specifically, the ε dependence explored in Appendix A.2 can be equally well expressed as a dependence on ρ.

A.2. Proof of Theorem 4.1: upper bound on the number of leaps

In this section we prove Theorem 4.1 from the text, which upper-bounds the number of leaps BTL takes as a function of m, t, and C:

Theorem. *For any SCRN $\mathcal{S}$ with M species, any ε such that $0 < \varepsilon < 1/(12M)$, and any $\delta > 0$, there are constants $c_1, c_2, c_3 > 0$ such that for any bounds on time t and total molecular count m, for any volume V and any starting state, after $c_1 \log m + c_2 t\,(C + c_3)$ leaps where $C = m/V$, either the bound on time or the bound on total molecular count will be exceeded with probability at least $1-\delta$.*

We prove a more detailed bound than stated in the theorem above which explicitly shows the dependence on ε hidden in the constants. Also since we introduce the asymptotic results only the end of the argument, the interested reader may easily investigate the dependence of the constants on other parameters of the SCRN such as N, M, ν_{ij}, and k_j. We also show an approach to probability 1 that occurs exponentially fast as the bound increases: if the bound above evaluates to n, then the probability that the algorithm does not exceed m or t in n leaps is at most $2e^{-O(1)n}$.

Our argument starts with a couple of lemmas. Looking within a single leap, the first lemma bounds the decrease in the molecular count of a species due to a given reaction as a function of time. The argument is essentially that for a reaction to decrease the molecular count of a species, that species must be a reactant, and therefore the propensity of the reaction is proportional to its molecular count. Thus we see a similarity to an exponential decay process and use this to bound the decrease. Note that a similar result does not hold for the *increase* in the molecular count of a species, since the molecular count of the increasing species need not be in the propensity function.[12] Then the second lemma uses the upper bound on how fast a species can decrease (the first lemma), together with the fact that in a leap some reaction must change some species by a relatively large amount, to classify leaps into those that either (1) take a long time or (2) increase some species significantly without decreasing any other species by much. Finally we show that this implies that if there are too many leaps we either violate the time bound or the total molecular count bound.

For the following, values f and g will be free parameters to be determined later. It helps to think of them as $0 < f \ll g \ll 1$. How long does it take for a reaction to decrease x_i by gth fraction of the violation bound εx_i? The number of occurrences of R_j to decrease x_i by $g\varepsilon x_i$ or more is at least $g\varepsilon x_i/|\nu_{ij}|$. The following lemma bounds the time required for these many occurrences to happen.

Lemma A.1. *Take any f and g $(0 < f, g < 1)$, any reaction R_j and species S_i such that $\nu_{ij} < 0$, any state $\vec{x}$, and any ε. Assuming that the propensity of R_j is fixed at $a_j(\vec{x})$, with probability at least $1 - f/g$, fewer than $g\varepsilon x_i/|\nu_{ij}|$ occurrences of R_j happen in time $f\varepsilon/(|\nu_{ij}|k_j)$ if R_j is unimolecular, or time $f\varepsilon/(|\nu_{ij}|k_jC)$ if R_j is bimolecular.*

Proof. For reaction R_j to decrease the amount of S_i, it must be that S_i is a reactant, and thus x_i is a factor in the propensity function. Suppose R_j is unimolecular. Then $a_j = k_j x_i$ and the expected number of occurrences of R_j in time $f\frac{\varepsilon}{|\nu_{ij}|k_j}$ is $a_j f\frac{\varepsilon}{|\nu_{ij}|k_j} \le f\frac{\varepsilon x_i}{|\nu_{ij}|}$. The desired result then follows from Markov's inequality. If R_j is bimolecular with $S_i \neq S_{i'}$ being the other reactant then $a_j = k_j\frac{x_i x_{i'}}{V}$; alternatively, $a_j = k_j\frac{x_i(x_i-1)}{V}$ if R_j is bimolecular with identical reactants. In general for bimolecular reactions $a_j \le k_j x_i C$. So the expected number of occurrences of R_j in time $f\frac{\varepsilon}{|\nu_{ij}|k_jC}$ is $a_j f\frac{\varepsilon}{|\nu_{ij}|k_jC} \le f\frac{\varepsilon x_i}{|\nu_{ij}|}$. The desired result follows as before. ∎

Let time $\tilde{\tau}$ be the minimum over all reactions R_j and S_i such that $\nu_{ij} < 0$ of $1/(|\nu_{ij}|k_j)$ if R_j is unimolecular, or $1/(|\nu_{ij}|k_jC)$ if R_j is bimolecular. We can think of $\tilde{\tau}$ setting the units of time for our argument. The above lemma implies that with probability at least $1 - f/g$ no reaction decreases x_i by $g\varepsilon x_i$ or more within time $f\varepsilon\tilde{\tau}$. The following lemma defines typical leaps; they are of two types: long or S_i-increasing. Recall M is the number of reaction channels and N is the number of species.

Lemma A.2 (Typical leaps). *For any f and g $(0 < f, g < 1)$, and for any ε, with probability at least $1 - NMf/g$ one of the following is true of a leap:*

1. *(long leap) $\tau > f\varepsilon\tilde{\tau}$*
2. *(S_i-increasing leap) $\tau \le f\varepsilon\tilde{\tau}$, and the leap increases some species S_i at least as $x_i \mapsto x_i + \lceil\varepsilon x_i\rceil - gM\varepsilon x_i$, while no species $S_{i'}$ decreases as much as $x_{i'} \mapsto x_{i'} - gM\varepsilon x_{i'}$.*

[12] If a reaction is converting a populous species to a rare one, the rate of the increase of the rare species can be proportional to m times its molecular count. The rate of decrease, however, is always proportional to the molecular count of the decreasing species, or proportional to C times the molecular count of the decreasing species (as we will see below).

Proof. By the union bound over the M reaction channels and the N species, Lemma A.1 implies that the probability that *some* reaction decreases the amount of *some* species S_i by $g\varepsilon x_i$ or more in time $f\varepsilon\tilde{\tau}$ is at most NMf/g. Now suppose this unlucky event does not happen. Then if the leap time is $\tau \leq f\varepsilon\tilde{\tau}$, no decrease is enough to cause a violation of the deviation bounds, and thus it must be that some reaction R_j increases some species S_i by more than εx_i. (Since R_j must occur an integer number of times, it actually must increase S_i by $\lceil \varepsilon x_i \rceil$ or more.) Since no reaction decreases S_i by $g\varepsilon x_i$ or more, we can be sure that S_i increases at least by $\lceil \varepsilon x_i \rceil - gM\varepsilon x_i$. ■

Lemma A.3. *For any species S_i, a leap decreases S_i at most as $x_i \mapsto x_i - M\lfloor \varepsilon x_i \rfloor - 2$.*

Proof. At most M reactions may be decreasing S_i. A reaction can decrease S_i by as much as $\lfloor \varepsilon x_i \rfloor$ without causing a violation of the deviation bounds. The last reaction firing that causes the violation of the deviation bounds ending the leap uses up at most 2 molecules of S_i (since reactions are at most bimolecular). ■

Note that a similar lemma does not hold for Gillespie's tau-leaping algorithms (Cao et al., 2006; Gillespie, 2001, 2003) because the number of reaction firings in a leap can be only bounded probabilistically. With some small probability a leap can result in "catastrophic" changes to some molecular counts. Since with enough time such events are certain to occur, the asymptotic analysis must consider them. Consequently, asymptotic results analogous to those we derive in this section remain to be proved for tau-leaping algorithms other than BTL.

Our goal now is to use the above two lemmas to argue that if we have a lot of leaps, we would either violate the molecular count bound (due to many S_i-increasing leaps for the same S_i), or violate the time bound (due to long leaps). Let n be the total number of leaps. By Hoeffding's inequality, with probability at least $1 - 2e^{-2n(NMf/g)^2}$ (i.e., exponentially approaching 1 with n), the total number of atypical steps is bounded as:

$$[\#\text{ of atypical leaps}] < 2nNMf/g. \tag{2}$$

Further, in order not to violate the time bound t, the number of long steps can be bounded as:

$$[\#\text{ of long leaps}] \leq t/(f\varepsilon\tilde{\tau}). \tag{3}$$

How can we bound the number of the other leaps (S_i-increasing, for some species S_i)? Our argument will be that having too many of such leaps results in an excessive increase of a certain species, thus violating the bound on the total molecular count. We start by choosing an S_i for which there is the largest number of S_i-increasing steps. Since there are N species, there must be a species S_i for which

$$[\#\text{ of } S_i\text{-increasing leaps}] > \frac{1}{N} \sum_{S_{i'} \neq S_i} [\#\text{ of } S_{i'}\text{-increasing leaps}]. \tag{4}$$

At this point, it helps to develop an alternative bit of notation labeling the different kinds of leaps with respect to the above-chosen species S_i to indicate how much x_i may change in the leap. Since our goal will be to argue that the molecular count of S_i must be large, we would like to lower-bound the increase in S_i and upper-bound the decrease. An atypical leap or a long leap we label "$\downarrow\downarrow$." By Lemma A.3, these leaps decrease S_i *at most* as $x_i \mapsto x_i - M\lfloor \varepsilon x_i \rfloor - 2$. An S_i-increasing leap we label "$\uparrow$." Finally, an $S_{i'}$-increasing leap for $S_{i'} \neq S_i$ we label "$\downarrow$." By Lemma A.2, $\uparrow$ leaps increase S_i *at least* as $x_i \mapsto x_i + \lceil \varepsilon x_i \rceil - gM\varepsilon x_i$, while $\downarrow$ leaps decrease S_i *by less* than $x_i \mapsto x_i - gM\varepsilon x_i$.

We would like to express these operations purely in a multiplicative way so that they become commutative, allowing for bounding their total effect on x_i independent of the order in which these leaps occurred but solely as a function of the number of each type. Further, the multiplicative representation of the leap effects is important because we want to bound the number of leaps logarithmically in the maximum molecular count. Note that $\downarrow\downarrow$ leaps cause a problem because of the subtractive constant term, and $\uparrow$ leaps cause a problem because if x_i drops to 0 multiplicative increases are futile. Nonetheless, for the sake of argument suppose we knew that throughout the simulation $x_i \geq 3$. Then assuming $\varepsilon \leq 1/(12M)$, we can

bound the largest decrease due to a $\downarrow\downarrow$ leap multiplicatively as $x_i \mapsto (1/4)\,x_i$. Further, we lower-bound the increase due to a $\uparrow$ leap as $x_i \mapsto (1 + (1 - gM)\varepsilon)x_i$. Then the lower bound on the final molecular count of S_i and therefore the total molecular count is

$$3(1 + (1 - gM)\varepsilon)^{n^{\uparrow}}(1 - gM\varepsilon)^{n^{\downarrow}}(1/4)^{n^{\downarrow\downarrow}} \leq m. \tag{5}$$

This implies an upper bound on the number of $\uparrow$ leaps, that together with (Equations (2)–(4)) provides an upper bound on the total number of leaps, as we will see below.

However, x_i might dip below 3 (including at the start of the simulation). We can adjust the effective number of $\uparrow$ leaps to compensate for these dips. We say a leap is in a dip if it starts at $x_i < 3$. Observe that the first leap in a dip starts at $x_i < 3$ while the leap after a dip starts at $x_i \geq 3$. Thus, unless we end in a dip, cutting out the leaps in the dips can only decrease our lower bound on the final x_i. We'll make an even looser bound and modify Equation (5) simply by removing the contribution of the $\uparrow$ leaps that are in dips.[13] How many $\uparrow$ leaps can be in dips? First, let us ensure $g < 1/(3M)$. Then, since a $\downarrow$ leap decreases x_i by less than $gM\varepsilon x_i < x_i/3$, and the decrease amount must be an integer, a $\downarrow$ leap cannot bring x_i below 3 starting at $x_i \geq 3$. Thus, if we start at $x_i \geq 3$ a $\downarrow\downarrow$ leap must occur before we dip below 3. Thus the largest number of dips is $n^{\downarrow\downarrow} + 1$ (adding 1 since we may start the simulation below 3). Let $n_d^{\uparrow}$ and $n_d^{\downarrow\downarrow}$ be the number of $\uparrow$ and $\downarrow\downarrow$ leaps in the dth dip (we don't care about $\downarrow$ leaps in a dip since they must leave x_i unchanged). Then $n_d^{\uparrow} < 2n_d^{\downarrow\downarrow} + 3$ and $\sum_d n_d^{\uparrow} < \sum_d 2n_d^{\downarrow\downarrow} + \sum_d 3 \leq 2n^{\downarrow\downarrow} + 3(n^{\downarrow\downarrow} + 1) = 5n^{\downarrow\downarrow} + 3$. Therefore, the adjusted bound (5) becomes: $3(1 + (1 - gM)\varepsilon)^{n^{\uparrow} - 5n^{\downarrow\downarrow} - 3}(1 - gM\varepsilon)^{n^{\downarrow}}(1/4)^{n^{\downarrow\downarrow}} \leq m$. For simplicity, we use the weaker bound

$$3(1 + (1 - gM)\varepsilon)^{n^{\uparrow}}(1 - gM\varepsilon)^{n^{\downarrow}}(1/4)^{6n^{\downarrow\downarrow}+3} \leq m. \tag{6}$$

In order to argue that this bounds the number of $\uparrow$ leaps, we need to make sure the $\downarrow$ leaps and the $\downarrow\downarrow$ leaps don't cancel out the effect of the $\uparrow$ leaps. By inequality (4) we know that $n^{\downarrow} < Nn^{\uparrow}$. If we can choose g to be a small enough constant such that more than N $\downarrow$ leaps are required to cancel the effect of a $\uparrow$ leap we would be certain the bound increases exponentially with $n^{\uparrow}$ without caring about $\downarrow$ leaps. Specifically, we choose a g small enough such that $(1 + (1 - gM)\varepsilon)(1 - gM\varepsilon)^N \geq 1 + \varepsilon/2$. For example, we can let $g = \frac{1}{M}(1 - (9/10)^{1/N})$.[14] Note that g depends only on constants N and M and is independent of ε. The bound then becomes $3(1 + \varepsilon/2)^{n^{\uparrow}}(1/4)^{6n^{\downarrow\downarrow}+3}$.

Thus, finally we have the following system of inequalities that are satisfied with probability exponentially approaching 1 as $n \to \infty$:

$$n = n^{\uparrow} + n^{\downarrow} + n^{\downarrow\downarrow} \tag{7}$$

$$n^{\downarrow\downarrow} \leq t/(f\varepsilon\tilde{\tau}) + 2nNMf/g \tag{8}$$

$$n^{\downarrow} < Nn^{\uparrow} \tag{9}$$

$$3(1 + \varepsilon/2)^{n^{\uparrow}}(1/4)^{6n^{\downarrow\downarrow}+3} \leq m. \tag{10}$$

Solving for n, we obtain[15]

$$n \leq \frac{h\log(m/3) + (12h + 1)t/(f\varepsilon\tilde{\tau}) + 6h}{(1 - 24hNMf/g)}$$

[13]We know we cannot end in a dip if the resulting bound evaluates to 3 or more. Thus technically we assume $m \geq 3$ for the bound to be always valid.

[14]Since $g \leq 1/(3M)$, make the simplification $(1 + (1 - gM)\varepsilon) \geq (1 + 2\varepsilon/3)$ and solve for g. The solution is minimized when $\varepsilon = 1$.

[15]Logarithms are base-2.

if $(1 - 24hf/g) > 0$ where $h = (N + 1)/\log(1 + \varepsilon/2)$ (also recall $g = \frac{1}{M}(1 - (9/10)^{1/N})$). To ensure this we let $f \leq g/(48hNM)$. Then with probability exponentially approaching 1 as n increases,

$$n \leq 2\log(m/3) + 96(12h + 1)th/(g\varepsilon\tilde{\tau}) + 12h.$$

Asymptotically as $\varepsilon \to 0, m \to \infty, t \to \infty$ with the system of chemical equations being fixed, we have $g = O(1)$, $h \leq O(1)/\varepsilon$, and write the above as $n \leq O(1)(1/\varepsilon)\log m + O(1)(1/\varepsilon)^3 t/\tilde{\tau}$. Recall our unit of time $\tilde{\tau}$ was defined to be the minimum over all reactions R_j and species S_i such that $v_{ij} < 0$ of $1/(|v_{ij}|k_j)$ if R_j is unimolecular, or $1/(|v_{ij}|k_j C)$ if R_j is bimolecular. No matter what C is, we can say $\tilde{\tau} \geq 1/(O(1)C + O(1))$. Thus we can write the above as

$$n \leq O(1)(1/\varepsilon)\log m + O(1)(1/\varepsilon)^3 t(C + O(1)).$$

For any δ, we can find appropriate constants such that the above bound is satisfied with probability at least $1 - \delta$.

This bound on the number of leaps has been optimized for simplicity of proof rather than tightness. A more sophisticated analysis can likely significantly decrease the numerical constants. Further, we believe the cubic dependence on $1/\varepsilon$ in the time term is excessive.[16]

A.3. Proving robustness by monotonicity

In this section, we develop a technique that can be used to prove the robustness of certain SSA processes. We use these results to prove the robustness of the example in Section 3 as well as of the construction of Angluin et al. (2006), simulating a Turing machine in Appendix A.4.

Since ρ-perturbations are not Markovian, it is difficult to think about them. Can we use a property of the original SSA process that would allow us to prove robustness without referring to ρ-perturbations at all?

Some systems have the property that every reaction can only bring the system "closer" to the outcome of interest (or at least "no futher"). Formally, we say an SSA process is *monotonic* for outcome Γ if for all reachable states $\vec{x}, \vec{y}$ such that there is a reaction taking $\vec{x}$ to $\vec{y}$, and for all t, the probability of reaching Γ within time t starting at $\vec{y}$ is at least the probability of reaching Γ within time t starting at $\vec{x}$. Note that by this definition Γ must be absorbing. Intuitively, perturbation of propensities in monotonic systems only change how fast the system approaches the outcome. Indeed, we can bound the deviations in the outcome probability of any ρ-perturbation at any time by two specific ρ-perturbations, which are the maximally slowed down and sped up versions of the original process. This implies that monotonic SSA processes are robust at all times t when the outcome probability does not change quickly with t, and thus slowing down or speeding up the SSA process does not significantly affect the probability of the outcome.

For an SSA process or ρ-perturbation $\mathcal{C}$ and set of states Γ, define $F^{\Gamma}(\mathcal{C}, t)$ to be the probability of being in Γ at time t. For SSA process $\mathcal{C}$, let $\tilde{\mathcal{C}}^{-\rho}$ be the ρ-perturbation defined by the constant deviations $\xi_j(t) = 1 - \rho$. Similarly, let $\tilde{\mathcal{C}}^{+\rho}$ be the ρ-perturbation defined by the constant deviations $\xi_j(t) = 1 + \rho$.

Lemma A.4. *If an SSA process $\mathcal{C}$ is monotonic for outcome Γ, then for any ρ-perturbation $\tilde{\mathcal{C}}$ of $\mathcal{C}$, $F^{\Gamma}(\tilde{\mathcal{C}}^{-\rho}, t) \leq F^{\Gamma}(\tilde{\mathcal{C}}, t) \leq F^{\Gamma}(\tilde{\mathcal{C}}^{+\rho}, t)$.*

Proof. If an SSA process is monotonic, allowing extra "spontaneous" transitions (as long as they are legal according to the SSA process) cannot induce a delay in entering Γ. We can decompose a perturbation with $\xi_j(t) \geq 1$ as the SSA process combined with some extra probability of reaction occurrence in the next interval dt. Thus, for a perturbation $\tilde{\mathcal{C}}$ of a monotonic SSA process $\mathcal{C}$ in which $\xi_j(t) \geq 1$, we have

[16]The cubic dependence on $1/\varepsilon$ in the time term is due to having to decrease the probability of an atypical step as ε decreases. It may be possible to reduce the cubic dependence to a linear one by moving up the boundary between a dip and the multiplicative regime as a function of ε rather than fixing it at 3. The goal is to replace the constant base $(1/4)^{O(1)n^{\downarrow\downarrow}+O(1)}$ term with a $(1 - O(1)\varepsilon)^{O(1)n^{\downarrow\downarrow}+O(1)}$ term. Then the effect of a $\downarrow\downarrow$ leap would scale with ε, as does the effect of an $\uparrow$ leap.

$F^\Gamma(\mathcal{C},t) \leq F^\Gamma(\tilde{\mathcal{C}},t)$. By a similar argument, if $\tilde{\mathcal{C}}$ has $\xi_j(t) \leq 1$, then $F^\Gamma(\tilde{\mathcal{C}},t) \leq F^\Gamma(\mathcal{C},t)$. Now $\tilde{\mathcal{C}}^{-\rho}$ and $\tilde{\mathcal{C}}^{+\rho}$ are themselves monotonic SSA processes ($\mathcal{C}$ scaled in time). Then by the above bounds, for any ρ-perturbation $\tilde{\mathcal{C}}$ of $\mathcal{C}$ we have $F^\Gamma(\tilde{\mathcal{C}}^{-\rho},t) \leq F^\Gamma(\tilde{\mathcal{C}},t) \leq F^\Gamma(\tilde{\mathcal{C}}^{+\rho},t)$. ■

Since $\tilde{\mathcal{C}}^{-\rho}$ and $\tilde{\mathcal{C}}^{+\rho}$ are simply the original SSA process $\mathcal{C}$ scaled in time by a factor of $1/(1+\rho)$ and $1/(1-\rho)$, respectively, we can write the bound of the above lemma as $F^\Gamma(\mathcal{C},t/(1+\rho)) \leq F^\Gamma(\tilde{\mathcal{C}},t) \leq F^\Gamma(\mathcal{C},t/(1-\rho))$. Rephrasing Lemma A.4:

Corollary A.1. *If an SSA process $\mathcal{C}$ is monotonic for outcome Γ then it is (ρ,δ)-robust with respect to Γ at time t where $\delta = F^\Gamma(\tilde{\mathcal{C}}^{+\rho},t) - F^\Gamma(\tilde{\mathcal{C}}^{-\rho},t) = F^\Gamma(\mathcal{C},t/(1-\rho)) - F^\Gamma(\mathcal{C},t/(1+\rho))$.*

For many SSA processes, it may not be obvious whether they are monotonic. We would like a simple "syntactic" property of the SCRN that guarantees monotonicity and can be easily checked. The following lemma makes it easy to prove monotonicity in some simple cases.

Lemma A.5. *Let $\mathcal{C}$ be an SSA process and Γ an outcome of SCRN $\mathcal{S}$. If every species is a reactant in at most one reaction in $\mathcal{S}$, and there is a set $\{n_j\}$ such that outcome Γ occurs as soon as every reaction R_j has fired at least n_j times, then $\mathcal{C}$ is monotonic with respect to Γ.*

Proof. The restriction on Γ allows us phrase everything in terms of counting reaction occurrences. For every reaction R_j, define $F_j(n,t)$ to be the probability that R_j has fired at least n times within time t. Now suppose we induce some reaction to fire by fiat. The only way this can decrease some $F_j(n,t)$ is if it decreases the count of a reactant of R_j or makes it more likely that some reaction $R_{j'}$ ($j' \neq j$) will decrease the count of a reactant of R_j. Either possibility is avoided if the SCRN has the property that any species is a reactant in at most one reaction. Since $F^\Gamma(\mathcal{C},t) = \prod_j F_j(n_j,t)$, this quantity cannot decrease as well, and monotonicity follows. ■

A.4. Robust embedding of a TM in an SCRN

Since we are trying to bound how the complexity of the prediction problem scales with increasing bounds on t and C but not with different SCRNs, we need a method of embedding a TM in an SCRN in which the SCRN is independent of the input length. Among such methods available (Angluin et al., 2006; Soloveichik et al., 2008), asymptotically the most efficient and therefore best for our purposes is the construction of Angluin et al. (2006). This result is stated in the language of distributed multi-agent systems rather than molecular systems; the system is a well-mixed set of "agents" that randomly collide and exchange information. Each agent has a finite state. Agents correspond to molecules (the system preserves a constant molecular count m); states of agents correspond to the species, and interactions between agents correspond to reactions in which both molecules are potentially transformed.

Now for the details of the SCRN implementation of the protocol of Angluin et al. (2006). Suppose we construct an SCRN corresponding to the Angluin et al. system as follows: Agent states correspond to species (i.e., for every agent state i there is a unique species S_i). For every pair of species S_{i_1}, S_{i_2}, $(i_1 \leq i_2)$ we add reaction $S_{i_1} + S_{i_2} \to S_{i_3} + S_{i_4}$ if the population protocol transition function specifies $(i_1,i_2) \mapsto (i_3,i_4)$. Note that we allow null reactions of the form $S_{i_1} + S_{i_2} \to S_{i_1} + S_{i_2}$ including for $i_1 = i_2$. For every reaction R_j, we'll use rate constant $k_j = 1$. The sum of all reaction propensities is $\lambda = \frac{m(m-1)}{2V}$ since every molecule can react with any other molecule.[17] The time until next reaction is an exponential random variable with rate λ. Note that the transition probabilities between SCRN states are the same as the transition probabilities between the corresponding configurations in the population protocol since in the SCRN every two molecules are equally likely to react next. Thus, the SSA process is just a continuous time version of the population protocol process (where unit "time" expires between transitions). Therefore, the SCRN can simulate a TM in the same way as the population protocol.

[17] Just to confirm, splitting the reactions between the same species and between different species, the sum of the propensities is $\sum_i \frac{x_i(x_i-1)}{2V} + \sum_{i<i'} \frac{x_i x_{i'}}{V} = \frac{1}{2V}(\sum_i x_i x_i - \sum_i x_i + 2\sum_{i<i'} x_i x_{i'}) = \frac{1}{2V}(\sum_{i,i'} x_i x_{i'} - \sum_i x_i) = \frac{n(n-1)}{2V}$ using the fact that $2\sum_{i<i'} x_i x_{i'} = \sum_{i\neq i'} x_i x_{i'}$ and $\sum_i x_i x_i + \sum_{i\neq i'} x_i x_{i'} = \sum_{i,i'} x_i x_{i'}$.

But first we need to see how does time measured in the number of interactions correspond to the real-valued time in the language of SCRNs?

Lemma A.6. *If the time between population protocol interactions is an exponential random variable with rate λ, then for any positive constants c, c_1, c_2 such that $c_1 < 1 < c_2$, there is N_0 such that for all $N > N_0$, N interactions occur between time $c_1 N/\lambda$ and $c_2 N/\lambda$ with probability at least $1 - N^{-c}$.*

Proof. The Chernoff bound for the left tail of a gamma random variable T with shape parameter N and rate λ is $\Pr[T \leq t] \leq (\frac{\lambda t}{N})^N e^{N-\lambda t}$ for $t < N/\lambda$. Thus $\Pr[T \leq c_1 N/\lambda] \leq (c_1 e^{1-c_1})^N$. Since $c_1 e^{1-c_1} < 1$ when $c_1 \neq 1$, $\Pr[T \leq c_1 N/\lambda] < N^{-c}$ for large enough N. An identical argument applies to the right tail Chernoff bound $\Pr[T \geq t] \leq (\frac{\lambda t}{N})^N e^{N-\lambda t}$ for $t > N/\lambda$. ■

The following lemma reiterates that an arbitrary computational problem can be embedded in a chemical system, and also shows that the chemical computation is robust with respect to the outcome of the computation. For a given TM and agent count m, let $\vec{x_{f_0}}$ and $\vec{x_{f_1}}$ be SCRN states corresponding to the TM halting with a 0 and 1 output respectively.

Lemma A.7. *Fix a perturbation bound $\rho > 0$, $\delta > 0$, and a randomized TM M with a Boolean output. There is an SCRN implementing Angluin et al.'s population protocol, such that if $M(x)$ halts in no more than t_{tm} steps using no more than s_{tm} time, then starting with the encoding of x and using $m = O(1)2^{s_{tm}}$ molecules, at any time $t \geq t_{ssa} = O(1)V t_{tm} \log^4(m)/m$ the SSA process is in $\vec{x_{f_b}}$ with probability that is within δ of the probability that $M(x) = b$. Further, this SSA process is (ρ, δ)-robust with respect to states $\vec{x_{f_0}}$ and $\vec{x_{f_1}}$ at all times $t \geq t_{ssa}$.*

The first part of the lemma states that we can embed an arbitrary TM computation in an SCRN, such that the TM computation is performed fast and correctly with arbitrarily high probability. The second part states that this method can be made arbitrarily robust to perturbations of reaction propensities. The first part follows directly from the results of Angluin et al. (2006), while the second part requires some additional arguments on our part.

If we only wanted to prove the first part, fix any randomized TM M with a Boolean output and any constant $\delta > 0$. There is a population protocol of Angluin et al. that can simulate the TM's computation on arbitrary inputs as follows: If on some input x, M uses t_{tm} computational time and s_{tm} space, then the protocol uses $m = O(1)2^{s_{tm}}$ agents, and the probability that the simulation is incorrect or takes longer than $N = O(1)t_{tm} m \log^4 m$ interactions is at most $\delta/2$. This is proved by using Theorem 11 of Angluin et al. (2006), combined with the standard way of simulating a TM by a register machine using multiplication by a constant and division by a constant with remainder. The total probability of the computation being incorrect or lasting more than N interactions obtained is at most $O(1)t_{tm} m^{-c}$. Since for any algorithm terminating in t_{tm} steps, $2^{s_{tm}} \geq O(1)\, t_{tm}$, we can make sure this probability is at most $\delta/2$ by using a large enough constant in $m = O(1)2^{s_{tm}}$. By Lemma A.6, the probability that $O(1)N$ interactions take longer than $O(1)N/\lambda$ time to occur is at most $\delta/2$. Thus the total probability of incorrectly simulating M on x or taking longer than $O(1)N/\lambda = O(1)V t_{tm} \log^4(m)/m$ time is at most δ. The Boolean output of M is indicated by whether we end up in state $\vec{x_{f_0}}$ or $\vec{x_{f_0}}$. (If the computation was incorrect or took too long we can be in neither.) This proves the first part of the lemma.

We now sketch out the proof of how the robustness of the system of Angluin et al. (2006) can be established, completing the proof of Lemma A.7. The whole proof requires retracing the argument in the original paper; here, we just outline how this retracing can be done. First, we convert the key lemmas of their paper to use real-valued SCRN time rather than the number of interactions. The consequences of the lemmas (e.g., that something happens before something else) are preserved and thus the lemmas can be still be used as in the original paper to prove the corresponding result for SCRNs. The monotonicity of the processes analyzed in the key lemmas can be used to argue that the overall construction is robust.

We need the following consequence of Lemma A.4:

Corollary A.2. *If an SSA process $\mathcal{C}$ is monotonic for outcome Γ, and with probability p it enters Γ after time t_1 but before time t_2, then for any ρ-perturbation $\tilde{\mathcal{C}}$ of $\mathcal{C}$, the probability of entering Γ after time $t_1/(1+\rho)$ but before time $t_2/(1-\rho)$ is at least p.*

Proof. Let $p_1 = F^\Gamma(\mathcal{C}, t_1)$ and $p_2 = F^\Gamma(\mathcal{C}, t_2)$. Using Lemma A.4 we know that $\forall t,\ F^\Gamma(\mathcal{C}, t/(1-\rho)) \geq F^\Gamma(\tilde{\mathcal{C}}, t)$. Thus, $p_1 = F^\Gamma(\mathcal{C}, t_1) \geq F^\Gamma(\tilde{\mathcal{C}}, (1-\rho)t_1)$. Similarly we obtain $p_2 = F^\Gamma(\mathcal{C}, t_2) \leq F^\Gamma(\tilde{\tilde{\mathcal{C}}}, (1+\rho)t_2)$. Thus $F^\Gamma(\tilde{\tilde{\mathcal{C}}}, (1+\rho)t_2) - F^\Gamma(\tilde{\mathcal{C}}, (1-\rho)t_1) \geq p_2 - p_1 = p$. ■

As an example let us illustrate the conversion of Lemma 2 of Angluin et al. (2006). The Lemma bounds the number of interactions to infect k agents in a "one-way epidemic" starting with a single infected agent. In the one-way epidemic, a non-infected agent becomes infected when it interacts with a previously infected agent. With our notation, this lemma states:

> Let $N(k)$ be the number of interactions before a one-way epidemic starting with a single infected agent infects k agents. Then for any fixed $\varepsilon > 0$ and $c > 0$, there exist positive constants c_1 and c_2 such that for sufficiently large total agent count m and any $k > m^\varepsilon$, $c_1 m \ln k \leq N(k) \leq c_2 m \ln k$ with probability at least $1 - m^{-c}$.

For any m and k we consider the corresponding SSA process $\mathcal{C}$ and outcome Γ in which at least k agents are infected. Since the bounds on $N(k)$ scale at least linearly with m, we can use Lemma A.6 to obtain:

> Let $t(k)$ be the time before a one-way epidemic starting with a single infected agent infects k agents. Then for any fixed $\varepsilon > 0$ and $c > 0$, there exist positive constants c_1 and c_2 such that for sufficiently large total agent count m and any $k > m^\varepsilon$, $c_1 m \ln(k)/\lambda \leq t(k) \leq c_2 m \ln(k)/\lambda$ with probability at least $1 - m^{-c}$.

Finally consider the SSA process of the one-way epidemic spreading. The possible reactions either do nothing (reactants are either both infected or both non-infected), or a new agent becomes infected. It is clear that for any m and k, $\mathcal{C}$ is monotonic with respect to outcome Γ in which at least k agents are infected. This allows us to use Corollary A.2 to obtain:

> Fix any $\rho > 0$, and let $t(k)$ be the time before a one-way epidemic starting with a single infected agent infects k agents in some corresponding ρ-perturbation. Then for any fixed $\varepsilon > 0$, $c > 0$, there exist positive constants c_1 and c_2 such that for sufficiently large total agent count m and any $k > m^\varepsilon$, $c_1 m \ln(k)/(\lambda(1+\rho)) \leq t(k) \leq c_2 m \ln(k)/(\lambda(1-\rho))$ with probability at least $1 - m^{-c}$.

Since ρ is a constant, what we have effectively done is convert the result in terms of interactions to a result in terms of real-valued time that is robust to ρ-perturbations simply by dividing by λ and using different multiplicative constants.

The same process can be followed for the key lemmas of Angluin et al. (2006) (Lemmas 3–8). This allows us to prove a robust version of Theorem 11 of Angluin et al. (2006) by retracing the argument of their paper using the converted lemmas and the real-valued notion of time throughout. Since the only way that time is used is to argue that something occurs before something else, the new notion of time, obtained by dividing by λ with different constants, can always be used in place of the number of interactions.

A.5. Proof of Theorem 5.1: lower bound on the computational complexity of the prediction problem

In this section, we prove Theorem 5.1 from the text which lower-bounds the computational complexity of the prediction problem as a function of m, t, and C. The bound holds even for arbitrarily robust SSA processes. The theorem shows that this computational complexity is at least linear in t and C, as long as the dependence on m is at most polylogarithmic. The result is a consequence of the robust embedding of a TM in an SCRN (Lemma A.7).

Let the prediction problem be specified by giving the SSA process (via the initial state and volume), the target time t, and the target outcome Γ in some standard encoding such that whether a state belongs to Γ can be computed in time polylogarithmic in m.

Theorem. *Fix any perturbation bound $\rho > 0$ and $\delta > 0$. Assuming the hierarchy conjecture (Conjecture 5.1), there is an SCRN $\mathcal{S}$ such that for any prediction algorithm $\mathcal{A}$ and constants $c_1, c_2, \beta, \eta, \gamma > 0$,*

there is an SSA process $\mathcal{C}$ of $\mathcal{S}$ and a $(m, t, C, 1/3)$-prediction problem $\mathcal{P}$ of $\mathcal{C}$ such that $\mathcal{A}$ cannot solve $\mathcal{P}$ in computational time $c_1 (\log m)^\beta t^\eta (C + c_2)^\gamma$ if $\eta < 1$ or $\gamma < 1$. Further, $\mathcal{C}$ is (ρ, δ)-robust with respect to $\mathcal{P}$.

Suppose someone claims that for any fixed SCRN, they can produce an algorithm for solving $(m, t, C, 1/3)$-prediction problems for SSA processes of this SCRN assuming the SSA process is (ρ, δ)-robust with respect to the prediction problem for some fixed ρ and δ, and further they claim the algorithm runs in computation time at most

$$O(1) (\log(m))^\beta t^\eta (C + O(1))^\gamma \tag{11}$$

for some $\eta < 1$ ($\beta, \gamma > 0$). We argue that assuming the hierarchy conjecture is true, such a value of η is impossible.

To achieve a contradiction of the hierarchy conjecture, consider any function probabilistically computable in $t_{tm}(n) = O(1)n^\zeta$ time and $s_{tm}(n) = O(1)n$ space for $\zeta = \frac{\beta+4\eta}{1-\eta} + 1$. Construct a randomized TM having error at most $1/24$ by running the original randomized TM $O(1)$ times and taking the majority vote. Use Lemma A.7 to encode the TM probabilistically computing this function in a (ρ, δ)-robust SSA process such that the error of the TM simulation is at most $1/24$. Then predicting whether the process ends up in state $\vec{x_{f_0}}$ or $\vec{x_{f_1}}$ provides a probabilistic algorithm for computing this function. The resulting error is at most $1/24 + 1/24 + 1/3 = 5/12 < 1/2$, where the first term $1/24$ is the error of the TM, the second term $1/24$ is for the additional error of the TM embedding in the SSA process, and the last term $1/3$ is for the allowed error of the prediction problem. By repeating $O(1)$ times and taking the majority vote, this error can be reduced below $1/3$, thereby satisfying the definition of probabilistic computation. How long does this method take to evaluate the function? We use $V = m$ so that C is a constant, resulting in $t_{ssa} = O(1)t_{tm}(n) \log^4 m = O(1)n^{\zeta+4}$ since $m = O(1)2^n$. Setting up the prediction problem by specifying the SSA process (via the initial state and volume), target final state and time t_{ssa} requires $O(1) \log m = O(1)\, n$ time.[18] Then the prediction problem is solved in computation time $O(1)(\log(m))^\beta t_{ssa}^\eta = O(1)n^{\beta+(\zeta+4)\eta}$. Thus, the total computation time is $O(1)(n^{\beta+(\zeta+4)\eta} + n)$ which, by our choice of ζ, is less than $O(1)n^\zeta$, leading to a contradiction of the hierarchy conjecture.

Is $\gamma < 1$ possible? Observe that if $\gamma < \eta$ then the claimed running time of the algorithm solving the prediction problem (expression (11)) with time $t_{ssa} = O(1)Vt_{tm}(n) \log^4(m)/m$ can be made arbitrarily small by decreasing V. This leads to contradiction of the hierarchy conjecture. Therefore $\gamma \geq \eta \geq 1$.

A.6. On implementing BTL on a randomized TM

The idealized BTL algorithm presented in Section 4.1 relies on infinite precision real-value arithmetic, while only finite precision floating-point arithmetic is possible on a TM. Further, the basic randomness generating operation available to a randomized TM is choosing one of a fixed number of alternatives uniformly, which forces gamma and binomial draws to be approximated. This complicates estimates of the computation time required per leap, and also requires us to ensure that we can ignore round-off errors in floating-point operations and tolerate approximate sampling in random number draws.

Can we implement gamma and binomial random number generators on a randomized TM and how much computational time do they require? It is easy to see that arbitrary precision uniform $[0, 1]$ random variates can be drawn on a randomized TM in time linear in precision. It is likely that approximate gamma and binomial random variables can be drawn using methods available in the numerical algorithms literature which uses uniform variate draws as the essential primitive. Since many existing methods for efficiently drawing (approximate) gamma and binomial random variables involve the rejection method, the computation time for these operations is likely to be an expectation. Specifically, it seems reasonable that drawing gamma and binomial random variables can be approximately implemented on a randomized

[18] By the construction of Angluin et al. (2006), setting up the initial state requires letting the binary expansion of the molecular count of a certain species be equal the input. Since the input is given in binary and all molecular counts are represented in binary, this is a linear time operation. Setting up the final state $\vec{x_{f_0}}$ or $\vec{x_{f_1}}$ is also linear time. Computing the target time for the prediction problem t_{ssa} is asymptotically negligible.

TM such that the expected time of these operations is polynomial in the length of the floating-point representation of the distribution parameters and the resultant random quantity.[19]

The computational complexity of manipulating integer molecular counts on a TM is polylogarithmic in m. Let l be an upper bound on the expected computational time required for drawing the random variables and real number arithmetic; l is potentially a function of m, V, t, and the bits of precision used. Using Markov's inequality and Theorem 4.1 we can then obtain a bound on the total computation time that is true with arbitrarily high probability. We also make the TM keep track of the total number of computational steps it has taken[20] and cut off computation when it exceeds the expectation by some fixed factor. Then we obtain the following bound on the total computation time: $O(1)((\log(m))^{O(1)} + l)\, t\, (C + O(1))$.

We have three sources of error. First, since BTL simulates a ρ-perturbation rather than the original SSA process, the probability of the outcome may be off by δ_1, assuming the SSA process was (ρ, δ_1)-robust. Further, since we are using finite precision arithmetic and only approximate random number generation, the deviation from the correct probability of the outcome may increase by another δ_2. Finally, there is a δ_3 probability that the algorithm cuts off computation before it completes. We have assumed a fixed $\delta_1 < \delta$, where δ is the allowed error of the prediction problem. We can make δ_3 an arbitrarily small constant by increasing the total computation time by a constant factor (using Markov's inequality). Further, let us assume that δ_2 is small enough to ensure that the total error $\delta_1 + \delta_2 + \delta_3 \leq \delta$ fulfills the requirements of solving the (m, t, C, δ)-prediction problem.[21]

ACKNOWLEDGMENTS

I thank Erik Winfree and Matthew Cook for providing invaluable support, technical insight, corrections, and suggestions. This work was supported by the NSF (grant 0523761 to Winfree) and the NIMH (training grant MH19138-15 to the Department of CNS).

DISCLOSURE STATEMENT

No competing financial interests exist.

REFERENCES

Adalsteinsson, D., McMillen, D., and Elston, T.C. 2004. Biochemical network stochastic simulator (BioNetS): software for stochastic modeling of biochemical networks. *BMC Bioinform.* 5, 24–45.

Ahrens, J., and Dieter, U. 1978. Generating gamma variates by a modified rejection technique. *Language* 54, 853–882.

Alon, U. 2007. *An Introduction to Systems Biology: Design Principles of Biological Circuits*. Chapman & Hall/CRC, Boca Raton, FL.

Angluin, D., Aspnes, J., and Eisenstat, D. 2006. Fast computation by population protocols with a leader. Technical Report YALEU/DCS/TR-1358. Department of Computer Science, Yale University, New Haven, CT.

[19]The numerical algorithms literature, which assumes that basic floating point operations take unit time, describes a number of algorithms for drawing from an (approximate) standard gamma distribution (Ahrens and Dieter, 1978), and from a binomial distribution (Kachitvichyanukul and Schmeiser, 1988), such that the expected number of floating-point operations does not grow as a function of distribution parameters (however, some restrictions on the parameters may be required). On a TM basic arithmetic operations take polynomial time in the length of the starting numerical values and the calculated result.

[20]Compute the bound and write this many 1's on a work tape, and after each computational step, count off one of the 1's until no more are left.

[21]We conjecture that for any fixed δ_2, we can find some fixed amount of numerical precision to not exceed δ_2 for (ρ, δ_1)-robust processes. We would like to show that robustness according to our definition implies robustness to round-off errors and approximate random number generation. While this conjecture has strong intuitive appeal, it seems difficult to prove formally, and represents an area for further study.

Arkin, A.P., Ross, J., and McAdams, H.H. 1998. Stochastic kinetic analysis of a developmental pathway bifurcation in phage-l *Escherichia coli*. *Genetics* 149, 1633–1648.
Barak, B. 2002. A probabilistic-time hierarchy theorem for 'slightly non-uniform' algorithms. *Proc. RANDOM*, pgs. 194–208.
Cao, Y., Gillespie, D., and Petzold, L. 2006. Efficient step size selection for the tau-leaping simulation method. *J. Chem. Phys.* 124, 044109.
Elowitz, M.B., Levine, A.J., Siggia, E.D., et al. 2002. Stochastic gene expression in a single cell. *Science* 297, 1183–1185.
Érdi, P., and Tóth, J. 1989. *Mathematical Models of Chemical Reactions : Theory and Applications of Deterministic and Stochastic Models*. Manchester University Press, Manchester, UK.
Ethier, S.N., and Kurtz, T.G. 1986. *Markov Processes: Characterization and Convergence*. John Wiley & Sons, New York.
Fortnow, L., and Santhanam, R. 2006. Recent work on hierarchies for semantic classes. *SIGACT News* 37, 36–54.
Gibson, M., and Bruck, J. 2000. Efficient exact stochastic simulation of chemical systems with many species and many channels. *J. Phys. Chem. A* 104, 1876–1889.
Gillespie, D.T. 1977. Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem.* 81, 2340–2361.
Gillespie, D.T. 1992. A rigorous derivation of the chemical master equation. *Physica A* 188, 404–425.
Gillespie, D.T. 2001. Approximate accelerated stochastic simulation of chemically reacting systems. *J. Chem. Phys.* 115, 1716–1733.
Gillespie, D.T. 2003. Improved leap-size selection for accelerated stochastic simulation. *J. Chem. Phys.* 119, 8229–8234.
Gillespie, D.T. 2007. Stochastic simulation of chemical kinetics. *Ann. Rev. Phys. Chem.* 58, 35–55.
Guptasarma, P. 1995. Does replication-induced transcription regulate synthesis of the myriad low copy number proteins of *Escherichia coli*? *Bioessays* 17, 987–997.
Horn, F., and Jackson, R. 1972. General mass action kinetics. *Arch. Rational Mech. Anal.* 47, 81–116.
Kachitvichyanukul, V., and Schmeiser, B. 1988. Binomial random variate generation. *Commun. ACM* 31, 216–222.
Kierzek, A.M. 2002. STOCKS: STOChastic kinetic simulations of biochemical systems with Gillespie algorithm. *Bioinformatics* 18, 470–481.
Kurtz, T.G. 1972. The relationship between stochastic and deterministic models for chemical reactions. *J. Chem. Phys.* 57, 2976–2978.
Levin, B. 1999. *Genes VII*. Oxford University Press, New York.
Malek-Mansour, M., and Nicolis, G. 1975. A master equation description of local fluctuations. *J. Statist. Phys.* 13, 197–217.
McAdams, H.H., and Arkin, A.P. 1997. Stochastic mechanisms in gene expression. *Proc. Natl. Acad. Sci. USA* 94, 814–819.
McQuarrie, D.A. 1967. Stochastic approach to chemical kinetics. *J. Appl. Probab.* 4, 413–478.
Morohashi, M., Winn, A.E., Borisuk, M.T., et al. 2002. Robustness as a measure of plausibility in models of biochemical networks. *J. Theoret. Biol.* 216, 19–30.
Rathinam, M., and El Samad, H. 2007. Reversible-equivalent-monomolecular tau: a leaping method for "small number and stiff" stochastic chemical systems. *J. Comput. Phys.* 224, 897–923.
Rathinam, M., Petzold, L., Cao, Y., et al. 2003. Stiffness in stochastic chemically reacting systems: the implicit tau-leaping method. *J. Chem. Phys.* 119, 12784.
Sipser, M. 1997. *Introduction to the Theory of Computation*. PWS Publishing, Boston, MA.
Soloveichik, D., Cook, M., Winfree, E., et al. 2008. Computation with finite stochastic chemical reaction networks. *Natural Computing* 7, 615–633.
Sontag, E. 2007. Monotone and near-monotone systems. *Lect. Notes Control Inform. Sci.* 357, 79–122.
Suel, G.M., Garcia-Ojalvo, J., Liberman, L.M., et al. 2006. An excitable gene regulatory circuit induces transient cellular differentiation. *Nature* 440, 545–550.
van Kampen, N. 1997. *Stochastic Processes in Physics and Chemistry*, revised ed. Elsevier, New York.
Vasudeva, K., and Bhalla, U.S. 2004. Adaptive stochastic-deterministic chemical kinetic simulations. *Bioinformatics* 20, 78–84.

Address reprint requests to:
Dr. David Soloveichik
Department of CNS
California Institute of Technology
Mail Code 136-93
Pasadena, CA 91125-9300

E-mail: dsolov@caltech.edu