A hierarchical exact accelerated stochastic simulation algorithm

David Orendorff and Eric Mjolsness[a)]

Department of Computer Science, University of California, Irvine, California 92697, USA

A new algorithm, "HiER-leap" (hierarchical exact reaction-leaping), is derived which improves on the computational properties of the ER-leap algorithm for exact accelerated simulation of stochastic chemical kinetics. Unlike ER-leap, HiER-leap utilizes a hierarchical or divide-and-conquer organization of reaction channels into tightly coupled "blocks" and is thereby able to speed up systems with many reaction channels. Like ER-leap, HiER-leap is based on the use of upper and lower bounds on the reaction propensities to define a rejection sampling algorithm with inexpensive early rejection and acceptance steps. But in HiER-leap, large portions of intra-block sampling may be done in parallel. An accept/reject step is used to synchronize across blocks. This method scales well when many reaction channels are present and has desirable asymptotic properties. The algorithm is exact, parallelizable and achieves a significant speedup over the stochastic simulation algorithm and ER-leap on certain problems. This algorithm offers a potentially important step towards efficient *in silico* modeling of entire organisms. © *2012 American Institute of Physics*. [http://dx.doi.org/10.1063/1.4766353]

I. INTRODUCTION

Computational biology is moving toward even more complex, comprehensive and detailed biological models. It is becoming increasingly important to simulate and understand these models computationally. The stochastic simulation algorithm[1] (SSA) was introduced to exactly sample the chemical master equation (CME) and has seen widespread adoption.

The original SSA iteratively samples reaction events in a way that requires $O(R)$ computational steps per sampled reaction event, where R is the number of reaction channels. This can be prohibitively slow when there are a large number of reaction channels or reaction events.

This fact together with the importance of the SSA has inspired a slew of SSA acceleration techniques.[2–10] The work of Gillespie[11] and its recent variants[12–14] reduces the total number of reaction events that need to be sampled but does so at the cost of accuracy. Additionally, the work of Gibson and Bruck[2] reduces the amount of work per simulated reaction event to $\log R$. The work of Slepoy *et al.*[15] ups the ante further by finding the next reaction event to sample in $O(1)$ time using rejection sampling under assumptions reasonable for biochemical networks.

There have been recent advances in consumer level multi-core central processing unit (CPU) technology. There are indications that next-generation CPU technology is moving from maximizing single-core speed to increasing the number of cores by orders of magnitude. There has been work on the parallelization of SSA via graphics processing units (GPUs)[6, 16, 17] and multi-core CPUs.[18] However, the parallelization was used to speed up the sampling of *many* trajectories rather than speeding up each trajectory in a large system. Multi-core GPUs and CPUs have not been effectively

used to speed up the sampling of a single chemical master equation trajectory exactly. Arguably this becomes the dominant problem when extremely large systems are being studied. For example, the *E. coli* genome has been estimated to have about 4400 gene products.[19] This fact suggests that tens of thousands of molecular species will needed to be present if an *E. coli* specimen is to ever be comprehensively modeled *in silico*.

Relatively little work has succeeded in reducing the number of reaction events sampled without introducing bias. While the work of Riedel and Bruck[20] is able to skip over cyclic states (e.g. loops), this method of reducing work does not apply to reaction networks with little state cycling. The previous work of the present authors (ER-leap)[10] is a leaping algorithm and was the first known general method to effectively reduce the number of SSA iterations sampled without sacrificing accuracy. This method scales well when reducing the number of SSA iterations. However, this method does not scale well when many reaction channels are present.

The currently proposed work describes a new SSA-equivalent algorithm that can take advantage of parallel hardware, and *additionally* provides an algorithmic speedup for systems with many reaction channels. Like exact reaction-leaping (ER-leap), this "HiER-leap" (hierarchical exact reaction-leaping) algorithm achieves these advances without the loss of accuracy. The HiER-leap algorithm uses a divide-and-conquer strategy to independently sample sparsely connected submodules of the reaction network, in a way somewhat similar to ER-leap. HiER-leap then performs a network-wide synchronization using rejection sampling. As will be shown, this synchronization step is efficient for "reasonably" independent submodules. The acceptance probability associated with synchronization is asymptotically equal to one as the number of reaction channels goes to infinity. This implies that the majority of the work will take place during submodule sampling, which may be performed in parallel.

a)Also at Institute for Genomics and Bioinformatics and Department of Mathematics. Electronic mail: emj@ics.uci.edu.

This work therefore presents a potentially important step towards organism-scale simulation.

II. BACKGROUND THEORY

The new HiER-leap algorithm begins its derivation from the state transition distribution defined by the chemical master equation after $L > 0$ reaction events. We then algebraically manipulate the CME until a distribution suitable for parallel sampling and synchronization is found.

In many ways this derivation closely follows the derivation found in ER-leap. Therefore, this section is dedicated to recalling the notation and key equations from ER-leap[10] that will serve as a starting point for the algorithm derivation in Sec. III.

A. Notation

We define reaction channels, indexed by r, as a set of input and output species, C_a, with corresponding input ($m^{(r_a)}$) and output ($m'^{(r_a)}$) stoichiometries

$$\{m_a^{(r)} C_a\} \to \{m_a'^{(r)} C_a\} \quad \text{with reaction rate } \rho_r \quad (1)$$

and the net stoichiometry for a given species and reaction channel as

$$\Delta m^{(r_a)} = m'^{(r_a)} - m^{(r_a)}.$$

Later, we will show the probabilities of state transitions after L "reaction events" occur.

Under the chemical master equation it is assumed that each reaction channel has a small probability of firing during a small time interval dt with probability equal to $a_r(\mathbf{n})dt$. The vector $\mathbf{n}$, possibly indexed by α for species type α, represents the quantities of the constituent species in terms of raw counts. This $a_r(\mathbf{n})$ term is also called the *propensity* or *rate* of reaction channel r and is defined as

$$a_r(\boldsymbol{n}) \equiv \rho_r F_{\boldsymbol{n}}^{(r)},$$

$$a_0(\boldsymbol{n}) \equiv \sum_{r=1}^{R} a_r(\boldsymbol{n}), \quad (2)$$

where

$$F_{\mathbf{n}}^{(r)} \equiv \prod_{\{a|m_a^{(r)} \neq 0\}} \left(\begin{cases} \frac{n_a!}{(n_a - m_a^{(r)})!} & \text{if } n_a \geqslant m_a^{(r)} \\ 0 & \text{otherwise} \end{cases} \right). \quad (3)$$

Note that superscripts involving "r" and related variables that index reaction numbers occur here and numerous times in the following. These are enclosed in parentheses "(r)" throughout, to indicate that they are not powers but rather indexes.

In this work, it will be notationally convenient for us to keep the propensity term factored out into ρ_r and $F_{\mathbf{n}}^{(r)}$.

As a brief aside, the upcoming derivations in this paper *may* work with other forms for $F_{\mathbf{n}}^{(r)}$. For example, the "umbral transformation" of a Hill function,

$$F_{\mathbf{n}}^{(r)} = Umbral[Hill(n; K)]$$

$$= \frac{n_{(k)}}{(K^n + n_k)},$$

may work as propensity function, where the falling factorial $n_{(k)} \equiv n!/(n-k)!$ replaces n^k in $Hill(n; K)$ (or more generally in a rational function) for each power of any integer-valued molecule number n. This functional form has the advantage of being monotonic and equal to zero for $n < k$, as required for a stochastic version of the Hill function with discrete integer numbers of molecules.

Furthermore, we define the total propensity D_I for some reaction to occur in state I as

$$D_I \equiv \sum_r \rho_r F_I^{(r)}, \quad (4)$$

which is equivalent to Eq. (2).

Bounds for $F^{(r)}$ and D, after L reaction events, are computed by bounding species counts after L reaction events. For each species identifier (ID) a, we bound the number of molecules present, n_a', by

$$n_a + L \min_r\{\Delta m_a^{(r)}\} \leqslant n_a' \leqslant n_a + L \max_r\{\Delta m_a^{(r)}\}. \quad (5)$$

If we introduce the notation that a tilde superscript or subscript, $\tilde{x}$ or $\underline{x}$, represents upper or lower bounding values, respectively, then, we can re-write the above as

$$\underline{n}_a \leq n_a' \leq \tilde{n}.$$

The corresponding propensities calculated from using the upper and lower bounding states, after $L - 1$ reaction events, respectively, are written as

$$\tilde{F}_{K,L-1}^{(r)} \equiv F_{\tilde{\mathbf{n}}}^{(r)},$$

$$\underline{F}_{K,L-1}^{(r)} \equiv \underline{F}_{\mathbf{n}}^{(r)},$$

and therefore

$$\tilde{D}_{K,L-1} = \sum_r \rho_r \tilde{F}_{K,L-1}^{(r)},$$

$$\underline{D}_{K,L-1} = \sum_r \rho_r \underline{F}_{K,L-1}^{(r)}$$

for any state K.

B. Markov process

In the ER-leap paper[10] it was shown that the probability of starting at state I_0 and ending up in state I_L after τ time elapses is

$$P(I_L, \tau | I_0, L) = \sum_{\{R_k | k=1..L-1\}} \left[\prod_{k=L-1 \searrow 0} \rho_{R_k} F_{I_k(\mathbf{R}, I_0)}^{(R_k)} \right.$$

$$\left. \times \exp(-\tau_k(D_{I_k(\mathbf{R}, I_0), I_k(\mathbf{R}, I_0)})) \right] \quad (6)$$

for every ordered vector $\{R_k | k = 1 \ldots L - 1\}$ of reaction channel events. Each state may be uniquely transformed by a reaction as $I_0 \to I_1(R, I_0)$.

Furthermore, it was shown in ER-leap[10] that for any function $e(\mathbf{r})$ summing over all possible orderings of L reaction events is equivalent to summing over all possible counts of reactions (e.g., a multinomial with L draws) and then permuting each of these draws for all unequal reactions, yielding

$$\sum_{\{r_k|k=1..L-1\}} e(\mathbf{r}) = \sum_{\{\mathbf{s}|s_r \in \mathbb{N}, \sum_r s_r = L\}} \sum_{\{\sigma|\sigma \text{ permutes unequal } r\text{'s}|\mathbf{s}\}} e(\sigma(\mathbf{r})),$$
(7)

which, when combined with Eq. (6), and introducing the previously defined bounds, separating out terms in $e(\mathbf{r})$ which are permutation invariant, and after some algebra results in

$P(I_L, \tau | I_0, L)$

$$= \sum_{\{\mathbf{s}|s_r \in \mathbb{N}, \sum_r s_r = L\}} \binom{L}{s_1 \ \cdots \ s_R} \times \left[\prod_{r=1}^{R} \left(\frac{\rho_r \tilde{F}_{I_0,L}^{(r)}}{\sum_r \rho_r \tilde{F}_{I_0,L}^{(r)}} \right)^{s_r} \right]$$

$$\times (\mathcal{Q}_{I_0 L-1})^L \exp\left(-\left(\sum_k \tau_k \right) \mathcal{Q}_{I_0 L-1} \right)$$

$$\times \frac{(\mathcal{Q}_{I_0 L-1})^L}{(\mathcal{Q}_{I_0 L-1})^L} \left\langle \left[\prod_{k=L-1 \searrow 0} \left(\frac{F_{I_k(\sigma(\mathbf{r})),I_0}^{(r_k)}}{\tilde{F}_{I_0,L-1}^{(r_k)}} \right) \right. \right.$$

$$\left. \left. \times \exp(-\tau_k(D_{I_k(\sigma(\mathbf{r})),I_0}, I_k(\sigma(\mathbf{r})),I_0) - \mathcal{Q}_{I_0 L-1})) \right] \right\rangle_{\{\sigma|\mathbf{s}\}}.$$
(8)

This expression can be interpreted as a rejection-sampling algorithm (last line) that corrects a multinomial approximate sampling algorithm (first two lines).

1. Rejection sampling

Through the lens of rejection sampling, Eq. (8) represents an algorithm.

Briefly, rejection sampling is a method to sample x from some distribution, $x \sim P(x)$, by means of an approximate distribution $P'(x)$. This can be expressed algebraically since $P(x)$ can be rewritten as

$$P(x) = P'(x) \frac{P(x)}{M P'(x)} + (1 - 1/M) P(x) \qquad (9)$$

assuming $M \geq 1$. Equation (9) can be viewed as a mixture distribution with the probability of sampling from $P'(x)$ being the "acceptance" $A(x)$

$$A(x) = \frac{P(x)}{M P'(x)} \qquad (10)$$

for some constant M such that $\forall_x A(x) \leq 1$.

It is now possible to see the ER-leap algorithm represented in Eq. (8). If in Eq. (8) we recognize $P(x) = P'(x) M A(x)$ (equivalent to Eq. (9)), with $M = (\tilde{D}_{I_0 L-1})^L / (\mathcal{Q}_{I_0 L-1})^L$, then we will implicitly define a $P'(x)$. This $P'(x)$ has the next L reaction events sampled from a multinomial with the probability

p_r of choosing the rth reaction channel being equal to

$$p_r = \left(\frac{\rho_r \tilde{F}_{I_0,L}^{(r)}}{\sum_r \rho_r \tilde{F}_{I_0,L}^{(r)}} \right).$$

Furthermore, our $P'(x)$ samples τ from an Erlang distribution (equivalent to a Gamma distribution with integer "shape" parameter) with rate parameter being $\mathcal{Q}_{I_0,L}$ and shape parameter being L. Finally, if needed when calculating $A(x)$, a random permutation σ is drawn uniformly and $\{\tau_k|\tau = \sum_{k=0}^{L-1} \tau_k\}$ is sampled from an L-simplex.

The work in Sec. III will similarly arrive at an equation representing an efficient and exact leaping algorithm for sampling L reaction events from an SSA equivalent distribution.

III. THEORY

A. Hierarchical notation

The HiER-leap algorithm uses a divide-and-conquer strategy to accelerate SSA. Evidence suggests that protein-protein interaction (PPI) networks tend to be modular.[21] These networks contain submodule clusters that interact heavily inside the cluster. Interactions with other clusters of proteins are less common. Although still an active area of research, evidence[22] suggests that similar modularity may exist in genetic regulatory networks as well. Additionally, when modeling spatial interactions,[23–28] events spatially distant must interact through sparse intermediate diffusion reaction channels. In this way, it is probably common that many reaction channels are weakly coupled to the majority of other channels. This observation suggests a potential avenue towards algorithm acceleration and parallelization for large biological networks.

Notation is introduced below to describe a hierarchical organization of reaction channels. Table I provides a comprehensive guide to notation used throughout Secs. III B–III D. Next, following and generalizing the strategy of Sec. II, we will derive bounds on propensities and species. The bounds will be essential for deriving an algorithm for exact speedup of SSA for systems amenable to hierarchical organization.

Reaction channels must belong to exactly one *block*. A *block* is defined as a set of reaction channels. If reactions are "connected" by shared reactants, it is preferred that reactions should be more strongly connected within them between

TABLE I. Notation: Accents and meaning.

Symbol	Meaning
$\tilde{x}$	Upper bounding value for x after $L - 1$ reaction events. Calculated by assuming each species type will be maximal.
$\hat{x}$	Upper bounding value for x after $L - 1$ reaction events. May not depend on bounding all species values and therefore may be tighter than $\tilde{x}$.
$\bar{x}$	Upper bounding value given u. Will often involve inner block calculations.
$\underline{x}, \underset{\smile}{x}, \underline{x}$	Lower bounding versions of the above definitions.
x^*	The optimal value of x with respect to some objective function.

blocks. For this work, a two-level hierarchy of reactions and blocks is used. However, it is straightforward to apply this method repeatedly to multiple levels.

Each reaction channel is indexed by its block ID r_1, and its within-block ID r_2, and will be designated as $R = (r_1 r_2)$ for $r_1 \in \{1 \ldots b\}$ and $r_2 \in \{1 \ldots b_{r_1}\}$. The "block propensity" for block r_1 and state I, denoted $D_I^{(r_1)}$ is the sum of propensities of constituent reaction channels. Specifically, similar to Eq. (4) this means

$$D_I^{(r_1)} = \sum_{r_2 \in r_1} \rho_{r_1 r_2} F_I^{(r_1 r_2)}. \qquad (11)$$

Furthermore, we denote the number of reaction events occurring within block r_1 as u_{r_1}. Finally, the number of events for the reaction channel indexed by $R = (r_1 r_2)$ is denoted by $v_{r_1 r_2}$.

B. Bounds on propensities and species counts

Similar to Eq. (5), we now develop bounds on species counts and propensities. This enables us to derive a two-scale rejection sampling algorithm in many ways analogous to ER-leap at each scale. For reasons that will become evident in Sec. III C, we first derive bounds on the block propensities given L and I_0. Afterwards, bounds will be developed on the species molecule counts and reaction channel propensities given u.

First, recall that in Eq. (5) we found bounds on species and propensities after L reaction events. Note that similar to Eq. (5) we can define

$$\tilde{D}_{K,L-1}^{(r_1)} = \sum_{r_2 \in r_1} \rho_{r_1 r_2} \tilde{F}_{K,L-1}^{(r_1 r_2)}$$

and a similar definition for $\underline{D}_{K,L-1}^{(r_1)}$.

1. Optimized block level bounds

If it is the case that we only need bounds on the block propensities, and not individual reaction channels, then we can take advantage of "reaction event exclusion". This means that we only need to consider the sequence of at most length L reaction events which will result in the most extreme value for the *sum* of propensities in block r_1. Therefore, we no longer need to assume that all species counts are at the most extreme value possible after L reaction events.

We want to find a bound closer to the optimal block propensity

$$\widehat{D}_{(I_0 L)}^{(r_1)}{}^* = \max_{v_{r_1} || u_{r_1} = L} \sum_{r_2 \in r_1} \rho_{r_1 r_2} F_{I(I_0, \mathbf{v}_{r_1})}^{(r_1 r_2)}. \qquad (12)$$

Unfortunately, naïvely solving this exactly for r_1 requires enumerating $(b_{r_1})^L$ possible choices for $\mathbf{v}_{r_1}$ upon every iteration. Fortunately the bound we seek, $\widehat{D}_{(I_0 L)}^{(r_1)}$, is not required to be exactly optimal. Instead we only require that

$$\widehat{D}_{(I_0 L)}^{(r_1)}{}^* \leq \widehat{D}_{(I_0 L)}^{(r_1)} \leq \tilde{D}_{(I_0 L)}^{(r_1)}, \qquad (13)$$

such that $\widehat{D}_{(I_0 L)}^{(r_1)}{}^* \leq \widehat{D}_{(I_0 L)}^{(r_1)}$ is required for algorithmic correctness and $\widehat{D}_{(I_0 L)}^{(r_1)} \leq \tilde{D}_{(I_0 L)}^{(r_1)}$ is needed for improved efficiency.

A heuristic algorithm for $\widehat{D}_{(I_0 L)}^{(r_1)}$ is developed. We demonstrate that this falls between the requisite values and has "nice" asymptotic properties that will be discussed later.

a. Derivation   The idea is to find the maximum $\Delta \widehat{D}_{(I_0 L)}^{(r_1)}$ possible resulting from one reaction channel firing sometime during the next L reaction events. If we determine this value, we can upper bound $D_{(I_0 \ldots I_{L-1})}^{(r_1)}$ with

$$D_{(I_0, L)}^{(r_1)} \leq D_{I_0}^{(r_1)} + (L-1)\Delta \widehat{D}_{(I_0 L)}^{(r_1)}{}^*, \qquad (14)$$

where

$$\Delta \widehat{D}_{(I_0 L)}^{(r_1)}{}^* = \max_{R_{r_1 r_2} || I_0 .. I_{L-1}} \Delta \widehat{D}_{(I_0 L)}^{(r_1)}.$$

Note how this is an upper bound on $D_{(I_0, L)}^{(r_1)}$. By construction, $\Delta \widehat{D}_{(I_0 L)}^{(r_1)}{}^*$ is the largest amount that the block propensity may change for any of the upcoming possible $(L-1)$ reaction events in r_1. Since there are $(L-1)$ reaction events, and the most any of them may increase $D_{(I_0, L)}^{(r_1)}$ is $\Delta \widehat{D}_{(I_0 L)}^{(r_1)}{}^*$, Eq. (14) will always bound $D_{(I_0, L)}^{(r_1)}$.

This method improves upon our previous methods, which found the maximum $\tilde{n}_a$ for all species and then calculates the block propensity. Each within-block reaction channel propensity will be larger when $\tilde{n}_a$ rather than n_a is used to calculate the propensity. Therefore, using the increased bound will result in a block's propensity being $O(b_{r_1} * L)$ larger than $D_{I_0}^{(r_1)}$. However, by calculating using Eq. (14) the bound will be just $O(L)$ larger than $D_{I_0}^{(r_1)}$.

Again, naïvely solving for $\Delta \widehat{D}_{(I_0 L)}^{(r_1)}{}^*$ requires an impractical amount of work. But as with our previous argument, we can upper-bound $\Delta \widehat{D}_{(I_0 L)}^{(r_1)}{}^*$ and still achieve an upper bound for $D_{(I_0, L)}^{(r_1)}$. To upper bound $\Delta \widehat{D}_{(I_0 L)}^{(r_1)}{}^*$ we use the monotonic nature of $D_{I_k}^{(r_1)}$. If any species increases to $n_a' \geq n_a$ we know that $D_{n_a'}^{(r_1)} \geq D_{n_a}^{(r_1)}$. Therefore, if we find the reaction channel that increases the block propensity the most when $\tilde{n}_{I_0, L}$ is used for positive $\Delta m_a^{(r_1 r_2)}$, we are guaranteed that there does not exist a larger $\Delta \widehat{D}_{(I_0 L)}^{(r_1)}$.

This yields

$$\Delta \widehat{D}_{(I_0 L)}^{(r_1)} = \max_{r_2 \in r_1}[D^{(r_1)}(\mathbf{q}(\tilde{\mathbf{n}}, r_2)) - D^{(r_1)}(\tilde{\mathbf{n}})], \qquad (15)$$

where

$$\mathbf{q}(\mathbf{n}, r_2)_a = \begin{cases} n_a + \Delta m_a^{(r_1 r_2)} & \text{if } \Delta m_a^{(r_1 r_2)} > 0, \\ n_a & \text{otherwise,} \end{cases} \qquad (16)$$

as our final equation for $\Delta \widehat{D}_{(I_0 L)}^{(r_1)}$. A proof that this bounds the maximum delta possible can be found in the Appendix.

This tighter bound will result in a greater acceptance ratio. The basic reason for this improvement is that we need not overestimate *every* propensity in r_1 by $O(L)$, and add the overestimates up, since only L and not $b_1 L$ reactions will occur.

Naïvely finding the reaction channel $R = (r_1 r_2)$ that will increase $D^{(r_1)}(\tilde{n}_{I_0, L})$ by a maximal amount will cost $O(|R_{r_1}|)$ steps to compute. To accelerate this step further, blockwise priority queues (PQ) are used to find this R efficiently. Nodes in the PQ are reaction channels and values are the $\Delta D_{(I(\tilde{n}, b_{r_1}))}^{(r_1)}$ caused by each reaction channel firing. Upon acceptance

of L reaction events we must update the priority queue for each block. Only nodes that interact with species which have changed, need to be adjusted. This, at worst, will be $O(\log b_{r_1})$ work for each node, although in practice the order rarely needs to change.

2. Propensity bounds given u

If we know u, the number of reaction events for r_1 and adjacent blocks, we can derive even tighter bounds on the reaction channels R_{r_1*}. In fact, these tighter bounds help us to efficiently increase L when larger systems are considered, as will be demonstrated in Sec. III C.

We determine $\bar{F}^{(r_1*)}$ by finding bounds on species counts given u. In other words, we want to find

$$n_A(r, n_0, k) \leq \bar{n}_{Ar_1}(u, n_0), \text{ for}$$

$$k = 0..[(\text{index of final } r_1 \text{ event}) - 1],$$

which is the maximum possible value of n_{Ar_1} prior to the last event in r_1 occurring. In this way $n_A(r, n_0, k)$ will never exceed the propensity calculated from $\bar{n}_{Ar_1}(u, n_0)$.

Finding the optimal value for $\bar{n}_{Ar_1}(u, n_0)$ is straightforward. We first need to consider blocks other than r_1 which may change n_{Ar_1}. Since the order of reactions is unknown, we must assume that all reaction events in blocks except r_1, written as $u \setminus \{u_{r_1}\}$, occur prior to those in u_{r_1}. It is desired that the number of neighbors relative to the total number of blocks will be small. This will decrease $n_A(r, n_0, k)$ and ultimately lead to a more efficient algorithm. Second, we need to consider reactions in r_1. In calculating the bound, it is assumed that all $(u_{r_1} - 1)$ reaction events chosen will behave adversarially. This is analogous to the method considered in Sec. III B, with the modification that we will consider a subset of reaction channels. Thus

$$\bar{n}_{Ar_1}(u, n_0) \equiv n_A + \sum_{r_1'}(u_{r_1'} - \delta_{r_1 r_1'}) \max_{r_2'} \Delta m_{(a_1 a_2)}^{(r_1' r_2')}$$

will bound each n_{Ar_1} with respect to r_1 and u. The Kronecker delta function $\delta(a, b)$ or $\delta(a - b)$ is as usual:

$$\delta(a - b) = \delta_{ab} = \mathbf{1}(a = b) \equiv \begin{cases} 1 & \text{if } a = b \\ 0 & \text{otherwise} \end{cases}.$$

Finally, the propensities of reaction channels inside of block r_1 are bound as,

$$\bar{F}^{(r_1 r_2)}(u, n_0) \equiv F^{(r_1 r_2)}(u, \bar{n}_{Ar_1}).$$

As in ER-leap, lower-bounding the propensities and species is done with the same techniques as that used for upper-bounding with the restriction that propensities and species molecule counts cannot go below zero. These derived bounds are used in Secs. III C–III D.

C. Equivalent Markov process

Similar to Sec. II B, we want to algebraically manipulate the distribution represented by the chemical master equation (a special case of the Kolmogorov-Chapman equation[29]) into

a form suitable for parallelization and acceleration. The hierarchical description from Sec. III A will aid us in this transformation.

First, note that it is possible to rewrite Eq. (7) into a hierarchical version with u's and v's strictly ordered such that

$$\sum_{\{R_k | k=1..L-1\}} e(\mathbf{R}) = \sum_{\{\mathbf{u} | u_R \in \mathbb{N}, \sum_R u_R = L\}} \sum_{\{\mathbf{v_{r_1}} | v_R \in \mathbb{N}, \sum_{r_2'} v_{r_1 r_2'} = u_{r_1}\}}$$
$$\sum_{\{\sigma_1 | \sigma_1 \text{ permutes unequal } R's | \mathbf{u}\}} \sum_{\{\sigma_2 | \sigma_2 \text{ permutes unequal } R's | \mathbf{v_{r_1}}\}} e(\sigma_1(\sigma_2(\mathbf{R}))).$$

By taking an average of $e(\sigma(\mathbf{R}))$ and weighting by the number of ways the selection may occur we get

$$= \sum_{\{\mathbf{u} | u_R \in \mathbb{N}, \sum_R u_R = L\}} \sum_{\{\mathbf{v_{r_1}} | v_R \in \mathbb{N}, \sum_{r_2'} v_{r_1 r_2'} = u_{r_1}\}}$$
$$\times \binom{L}{R_1 R_2 \ldots R_n} \langle\langle e(\sigma_1(\sigma_2(\mathbf{R}))) \rangle_{\sigma_2} \rangle_{\sigma_1}$$

and analogous to the way shuffling a deck of cards is the same as shuffling by suit and then, maintaining that order, shuffling by value independently for each suit, we may write

$$\binom{L}{R_1 R_2 \ldots R_n} = \frac{L!}{R_1! R_2! \ldots R_n!}$$
$$= \frac{L!}{u_{r_1}! u_{r_1'}! \ldots u_{r_1''}!} \prod_{r_1} \frac{u_{r_1}!}{v_{r_1 r_2}! v_{r_1 r_2'}! \ldots v_{r_1 r_2''}!}$$
$$= \binom{L}{u_{r_1} \ldots u_{r_1'}} \prod_{r_1} \binom{u_{r_1}}{v_{r_1 r_2} \ldots v_{r_1 r_2'}}$$

and arrive at a useful form for our distribution, which is already suggestive of a block-parallel algorithm:

$$\sum_{\{r_k | k=1..L-1\}} e(\mathbf{r}) = \sum_{\{\mathbf{u} | u_R \in \mathbb{N}, \sum_R u_R = L\}} \binom{L}{u_{r_1} \ldots u_{r_1'}}$$
$$\times \sum_{\{\mathbf{v_{r_1}} | v_R \in \mathbb{N}, \sum_{r_2'} v_{r_1 r_2'} = u_{r_1}\}} \prod_{r_1} \binom{u_{r_1}}{v_{r_1 r_2} \ldots v_{r_1 r_2'}}$$
$$\times \langle\langle e(\sigma_1(\sigma_2(\mathbf{R}))) \rangle_{\sigma_2} \rangle_{\sigma_1}. \quad (17)$$

To go further, we need to re-examine $e(\ldots)$.

1. Introduction of probability bounds

We now make use of our previously derived propensity bounds to derive a parallel algorithm. From Eq. (6) we have

$$\langle\langle e(\sigma_1(\sigma_2(\mathbf{R}))) \rangle_{\sigma_2} \rangle_{\sigma_1} = \left\langle\left\langle \prod_{k=L-1 \searrow 0} \rho_{R_k} F_{I_k(\mathbf{R}, I_0)}^{(R_k)}\right.\right.$$
$$\left.\left. \times \exp(-\tau_k(D_{I_k(\mathbf{R}, I_0), I_k(\mathbf{R}, I_0)})) \right\rangle_{\sigma_2}\right\rangle_{\sigma_1};$$

with inclusion of derived bounds,

$$\langle\langle e(\sigma_1(\sigma_2(\mathbf{R})))\rangle_{\sigma_2}\rangle_{\sigma_1}$$

$$= \left\langle\left\langle \prod_{k=L-1\searrow 0} \frac{F^{(R_k)}_{I_k(\mathbf{R},I_0)}}{\bar{F}^{(r_1r_2)}(u,I_0)} \frac{\rho_{R_k}\bar{F}^{(r_1r_2)}(u,I_0)}{\bar{D}^{(r_1)}(u,I_0)}\right.\right.$$

$$\times \frac{\bar{D}^{(r_1)}(u,I_0)}{\widehat{D}^{(r_1)}_{(I_0L)}}\widehat{D}^{(r_1)}_{(I_0L)}$$

$$\times \exp(-\tau_k(D_{I_k(\mathbf{R},I_0),I_k(\mathbf{R},I_0)} - \mathcal{Q}_{(I_0L)})$$

$$\left.\left.\times \exp(-\tau_k \mathcal{Q}_{(I_0L)})\right\rangle_\sigma\right\rangle_{\sigma_1}.$$

If we separate out terms based on independence of σ_1, σ_2, and v, then

$$\langle\langle e(\sigma_1(\sigma_2(\mathbf{R})))\rangle_{\sigma_2}\rangle_{\sigma_1}$$

$$= \exp(-\tau \mathcal{Q}_{(I_0L)})\left(\prod_{r_1}\widehat{D}^{(r_1)}_{(I_0L)}{}^{u_{r_1}}\right)$$

$$\times\left(\prod_{r_1}\prod_{r_2}\left(\frac{\rho_{R_k}\bar{F}^{(r_1r_2)}(u,I_0)}{\bar{D}^{(r_1)}(u,I_0)}\right)^{v_{r_2}}\right)$$

$$\times\left(\prod_{r_1}\left(\frac{\bar{D}^{(r_1)}(u,I_0)}{\widehat{D}^{(r_1)}_{(I_0L)}}\right)^{u_{r_1}}\right)$$

$$\times\left\langle\prod_{k=L-1\searrow 0}\frac{F^{(R_k)}_{I_k(\mathbf{R},I_0)}}{\bar{F}^{(r_1r_2)}(u,I_0)}\right.$$

$$\left.\times \langle\exp(-\tau_k(D_{I_k(\mathbf{R},I_0),I_k(\mathbf{R},I_0)} - \widehat{D}_{(I_0L)}))\rangle_{\sigma_2}\right\rangle_{\sigma_1}$$

We now substitute the expression for $\langle\langle e(\sigma_1(\sigma_2(\mathbf{R})))\rangle_{\sigma_2}\rangle_{\sigma_1}$ into Eq. (17), combining terms where appropriate:

$$\left[\prod_{k=l-1\searrow 0}\hat{W}\quad\exp(-\tau_k D)\right]_{I_l,I_0}$$

$$= \left(\frac{\sum_{r_1'}\widehat{D}^{(r_1')}_{(I_0L)}}{\mathcal{Q}_{(I_0,L)}}\right)^L \sum_{\{\mathbf{u}|u_R\in\mathbb{N},\sum_R u_R=L\}}$$

$$\times\left[\binom{L}{u_{r_1}\ldots u_{r_1'}}\prod_{r_1}\left(\frac{\widehat{D}^{(r_1)}_{(I_0L)}}{\sum_{r_1'}\widehat{D}^{(r_1')}_{(I_0L)}}\right)^{u_{r_1}}\right]$$

$$\times \sum_{\{\mathbf{v_{r_1}}|v_R\in\mathbb{N},\sum_{r_2'}v_{r_1r_2'}=u_{r_1}\}}\left[\prod_{r_1}\binom{u_{r_1}}{v_{r_1r_2}\ldots v_{r_1r_2'}}\right.$$

$$\left.\times\prod_{r_2}\left(\frac{\rho_{R_k}\bar{F}^{(r_1r_2)}(u,I_0)}{\bar{D}^{(r_1)}(u,I_0)}\right)^{v_{r_2}}\right]$$

$$\times (\mathcal{Q}_{(I_0,L)})^L\exp(-\tau \mathcal{Q}_{(I_0L)})$$

$$\times AcceptCoarse(u;I_0,L)$$

$$\times\left(\prod_{r_1}AcceptBlock(v_{r_1},\sigma_2;u)\right)$$

$$\times AcceptFine(\sigma_1;u,v,I_0,\sigma_2) \tag{18}$$

The acceptance probabilities are as follows:

$$AcceptCoarse(u;I_0,L) = \prod_{r_1}\left(\frac{\bar{D}^{(r_1)}(u,I_0)}{\widehat{D}^{(r_1)}_{(I_0L)}}\right)^{u_{r_1}}, \tag{19}$$

$$AcceptBlock(v_{r_1},\sigma_2;u,r_1) = \prod_{k\in r_1}\frac{F^{(R_k)}_{I_k(\mathbf{R},I_0)}}{\bar{F}^{(r_1r_2)}(u,I_0)}, \tag{20}$$

$$AcceptFine(\sigma_1;u,v,I_0,\sigma_2)$$

$$= \prod_{k=L-1\searrow 0}\exp(-\tau_k(D_{I_k(\mathbf{R},I_0),I_k(\mathbf{R},I_0)} - \widehat{D}_{(I_0L)})). \tag{21}$$

Furthermore, prior to turning these equations into an algorithm, we note that we can lower-bound these acceptance probabilities. This will enable us to do an early acceptance or rejection without always doing all of the work to calculate these values exactly.

2. Lower bounding acceptance probabilities

We begin by lower-bounding $AcceptFine(\ldots)$. This probability requires the most work to calculate and as we will see may be bound fairly tightly. The bound only requires that τ has been sampled.

The lower bound $\underaccent{\smile}{AcceptFine}(\ldots)$ is sought such that

$$\underaccent{\smile}{AcceptFine}(\ldots)$$

$$\leq \prod_{k=L-1\searrow 0}\exp(-\tau_k(D_{I_k(\mathbf{R},I_0),I_k(\mathbf{R},I_0)} - \mathcal{Q}_{(I_0L)})).$$

for all possible $\{\tau_k\}$ and $\{I_0\ldots I_{L-1}\}$. In the above, note that $\mathcal{Q}_{(I_0L)}$ is constant with respect to k. Therefore, when we also upper bound

$$\widehat{D}_{I_0L} \geq D_{I_k(\mathbf{R},I_0),I_k(\mathbf{R},I_0)},$$

this creates an easily computable expression for the lower bound

$$AcceptFine(\sigma_1;u,v,I_0,\sigma_2,\tau)$$

$$\geq \prod_{k=L-1\searrow 0}\exp(-\tau_k(\widehat{D}_{I_0L} - \mathcal{Q}_{(I_0L)}))$$

so that

$$\underaccent{\smile}{AcceptFine}(\tau;I_0,L) = \exp(-\tau(\widehat{D}_{I_0L} - \mathcal{Q}_{(I_0L)})) \tag{22}$$

Furthermore, recall that $E[\tau] = L/\mathcal{Q}_{(I_0,L)}$. If we assume that $\Delta\widehat{D}_{(I_0L)} \propto L$ when computing $\mathcal{Q}_{(I_0L)}$ and $\widehat{D}_{(I_0L)}$ (see Eq. (14)) in the limit of many non-zero propensity reaction channels

$$\left\langle\lim_{|\mathcal{R}|\to\infty}\exp(-\tau(\widehat{D}_{I_0L} - \mathcal{Q}_{(I_0L)}))\right\rangle \to 1,$$

which implies that both $AcceptFine(\ldots)$ and $\underaccent{\smile}{AcceptFine}(\ldots)$ tend to unity as the number of reaction channels increases.

Next, we set out to lower-bound $AcceptBlock(\ldots)$. This acceptance probability depends on σ_2. Therefore, work will be saved if we can calculate the lower bound without sampling σ_2. This can be accomplished by noting that $AcceptBlock(\ldots)$ is a product of fractions. If we have a numerator and denominator that are independent of σ_2 we can re-write

this equation in terms of r_2. Specifically, using $\underline{F}_{u,I_0}^{(r_1 r_2)}$ allows us to lower-bound the equation,

$$\widehat{Accept\,Block}(v_{r_1}, \sigma_2; u, r_1)$$

$$= \prod_{k \in r_1} \frac{F_{I_k(\mathbf{R}, I_0)}^{(R_k)}}{\bar{F}^{(r_1 r_2)}(u, I_0)} \geq \prod_{k \in r_1} \frac{\underline{F}^{(r_1 r_2)}(u, I_0)}{\bar{F}^{(r_1 r_2)}(u, I_0)}$$

yielding

$$\widehat{Accept\,Block}(v_{r_1}; u) = \prod_{r_2 \in r_1} \left(\frac{\underline{F}^{(r_1 r_2)}(u, I_0)}{\bar{F}^{(r_1 r_2)}(u, I_0)} \right)^{v_{r_2}}. \quad (23)$$

D. Algorithm

The above equations, along with rejection sampling, allow us to create an efficient algorithm that will allow much of the work be done in parallel. From Eq. (18) it is observed that there are two probability mass function expressions for a multinomial distribution. Specifically,

$$Multinomial \left(u; \left\{ p_{r_1} = \frac{\widehat{D}_{(I_0 L)}^{(r_1)}}{\sum_{r_1'} \widehat{D}_{(I_0 L)}^{(r_1')}} \right\}, L \right)$$

$$= \binom{L}{u_{r_1} \dots u_{r_1'}} \prod_{r_1} \left(\frac{\widehat{D}_{(I_0 L)}^{(r_1)}}{\sum_{r_1'} \widehat{D}_{(I_0 L)}^{(r_1')}} \right)^{u_{r_1}}$$

is the multinomial distribution for sampling u. And for each r_1 the vector v_{r_1} is sampled as

$$Multinomial \left(v_{r_1}; \left\{ p_{r_1 r_2} = \frac{\rho_{R_k} \bar{F}^{(r_1 r_2)}(u, I_0)}{\bar{D}^{(r_1)}(u, I_0)} \right\}, u_{r_1} \right)$$

$$= \binom{u_{r_1}}{v_{r_1 r_2} \dots v_{r_1 r_2'}} \prod_{r_2} \left(\frac{\rho_{R_k} \bar{F}^{(r_1 r_2)}(u, I_0)}{\bar{D}^{(r_1)}(u, I_0)} \right)^{v_{r_2}},$$

which is interesting and implies that v_{r_1} is independent of all other block's $v_{r_1'}$ given u.

The multinomials v_{r_1} may be sampled independently for each block, however it may be the case that Eq. (20) needs to be computed by considering multiple blocks simultaneously. Specifically, computing $I_k(\mathbf{R}, I_0)$ for block r_1 may require knowledge about any neighboring blocks, r_1', changing the chemical species counts for reaction channels v_{r_1} reacting on the same species. Since Eq. (23) is independent of $I_k(\dots)$, this "joint" acceptance probability only needs to be calculated when (a) blocks r_1 and r_1' share a chemically reacting species in their respective v_{r_1} and $v_{r_1'}$ sampled reaction channels and (b) block r_1 or r_1' does not pass early block-acceptance. Computing Eq. (20) jointly involves computing a σ_2 for reaction events in r_1 and r_1'. Then, $I_k(\dots)$ may be computed properly. It should be noted that in the pseudocode below some possible optimizations (e.g., independent early block accept and some parallel execution) are not shown for clarity. Instead "connected components", block groups which *may* need to be sampled jointly if conditions (a) and (b) are met, are sampled entirely in "joint" form.

We now present the HiER-leap algorithm, which is a realization of the aforementioned equations, in pseudocode. First,

note that if we have an early global acceptance, then most of the computational effort will be put into line 12 of the following pseudocode. The subroutine from this line will be shown later. Notice that this function is independent for all blocks, with the exception that computing Eq. (20) *may* need to be done jointly for neighboring blocks, and needs to be done for all blocks with at least one reaction event. This is an ideal scheme for parallelization and is done so with good efficacy as will be shown. Furthermore, for the tests in Sec. IV B the full calculation *AcceptFine*(...) was rare because in general: $\widehat{Accept\,Fine}(\dots) \geq 0.995$. The algorithm is as follows:

Require: $\widehat{D}_{(I_0, L)}, \mathcal{D}_{(I_0, L)}, \{\widehat{D}_{(I_0, L)}^{(r_1)}\}$ precomputed for $L \geq 1$ and I_0.
Ensure: Return $(I_L, \Delta t)$ ▷ return updated state and duration of L steps
  function HIER-LEAP(I_0, L)
    $\tau \leftarrow$ ERLANG$(\tau; \mathcal{D}_{(I_0, L)}, L)$
    $u \leftarrow$ MULTINOMIAL$\left(u; \left\{ p_{r_1} = \frac{\widehat{D}_{(I_0 L)}^{(r_1)}}{\sum_{r_1'} \widehat{D}_{(I_0 L)}^{(r_1')}} \right\}, L \right)$
    Compute $\bar{D}$'s   ▷ May be done in parallel.
5:            ▷ In accordance with Eq. (19).
    if UNIFORMRANDOM $(0,1) \geq$ ACCEPTCOARSE$(u; I_0, L)$ **then**
      return HIER-LEAP(I_0, L) ▷ Early Rejection. Try again.
    end if
10:   **for all** $c \in$ CONNECTEDCOMPONENTS(u, R, I_0) **do**
                  ▷ See algorithm below.
      $(v_{r_1}, \sigma_2) \leftarrow$ SAMPLECONNECTEDCOMPONENTS(c, u, I_0)
                  ▷ May be done in parallel.
    end for
15:   $z \leftarrow$ UNIFORMRANDOM $(0,1)$
    if $z \leq \widehat{Accept\,Fine}(\tau; I_0, L)$ **then**   ▷ See Eq. (22).
      return $(I_L(I_0, v, u), \tau)$ ▷ Early Acceptance.
    end if
           ▷ Computation should be rare; see Eq. (21).
20:   **if** $z \leq$ ACCEPTFINE$(\sigma_1; u, v, I_0, \sigma_2, \tau)$ **then**
      return $(I_L(I_0, v, u), \tau)$
    else
      return HIER-LEAP(I_0, L) ▷ Try again.
    end if
25: **end function**
Ensure: Return each connected component of blocks, where two blocks are in the same connected component if they share a reaction species and each blocks has at least one reaction event such that $u_{r_1} \geq 1$.
  function CONNECTEDCOMPONENTS (u, R, I_0)
    $C \leftarrow \{\}$ ▷ C is a set.
    $B \leftarrow R$ ▷ B is the set of blocks.
30:   **while** $B \neq \oslash$ **do**
      $b \leftarrow$ B.CHOOSEELEMENT$()$
      if $u_b \geq 1$ **then**
        $c \leftarrow$ DEPTHFIRSTSEARCH (b, u, B)
         ▷ Blocks share an edge if and only iff for each $u_{r_1} \geq 1$ and they share a species.
35:      $B \leftarrow B \backslash c$ ▷ Set operation subtraction.
        $C \leftarrow$ C.Append(c)
      end if
    end while
    return C
40: **end function**

The pseudocode to sample each connected component is as follows:

Require: The connected component c contains blocks which all have at least one reaction event.

function SAMPLECONNECTEDCOMPONENTS(c, u, I_0)
  $pEarly \leftarrow 1$
  for all $r_1 \in c$ **do**
    $v_{r_1} \leftarrow$ MULTINOMIAL $\left(v_{r_1}; \left\{ p_{r_1 r_2} = \frac{\rho_{R_k} \bar{F}^{(r_1 r_2)}(u, I_0)}{\bar{D}^{(r_1)}(u, I_0)} \right\}, u_{r_1}\right)$
    for $r_2 \in r_1$ **and** $v_{(r_1 r_2)} \geq 1$ **do**
      $pEarly \leftarrow pEarly \times \left(\frac{F^{(r_1 r_2)}(u, I_0)}{\bar{F}^{(r_1 r_2)}(u, I_0)}\right)^{v_{r_2}}$
    end for
  end for
  $z \leftarrow$ UNIFORMRANDOM $(0,1)$
  if $z \leq pBlockEarly$ **then**
    return $(\mathbf{v}, \sigma_2)$ ▷ Early Accept.
  end if
  $pComponent \leftarrow 1$
  $\sigma_2 \leftarrow$ PERMUTATION (u_{r_1}) ▷ Must compute exact acceptance probability.
  for all $k = 1 \ldots |\mathbf{v}|$ **do**
    $r'_2 \leftarrow \sigma_2(\mathbf{v}, k)$
    $pComponent \leftarrow pComponent \times \frac{F^{(r_1 r'_2)}_{I_k (\sigma_2(v_{r_1}), I_0)}}{\bar{F}^{(r_1 r'_2)}(u, I_0)}$ Calculating I_k
    takes the most work.
  end for
  if $z \leq pBlock$ **then**
    return $(\mathbf{v}, \sigma_2)$ ▷ Accept sample.
  else
    return SAMPLECONNECTEDCOMPONENTS (c, u, I_0) ▷ Sample rejected, try again.
  end if
end function

IV. NUMERICAL EXPERIMENTS

A. CaliBayes validation

We check the HiER-leap algorithm correctness numerically with the CaliBayes test suite similar to the work in ER-leap.[10] If is possible to solve analytically for $P(X|t)$, this allows us to compare many simulated trajectories to the true distribution defined by the CME. Since HiER-leap reduces to ER-leap when the number of blocks goes to one, and it has already been shown that ER-leap samples the correct distribution, we wish to test across a variety of reaction channel quantities and organization structure.

The reaction networks in CaliBayes for which we know the analytical solution involves at most two species types. However, simulating many replicates of these networks on a grid, not connected with diffusion, will allow us to treat each block as an independent sample. We can then treat the simulation of many network replicates as many sampled trajectories of a single network.

We perform tests over a number of network replicates $m = 2 \ldots 1000$. The number of blocks range from $b = 1 \ldots m$. The leap is in the range $L = 3 \ldots 18$, where the leap used depends on the specific reaction network, m and b, but is held constant throughout the simulation.
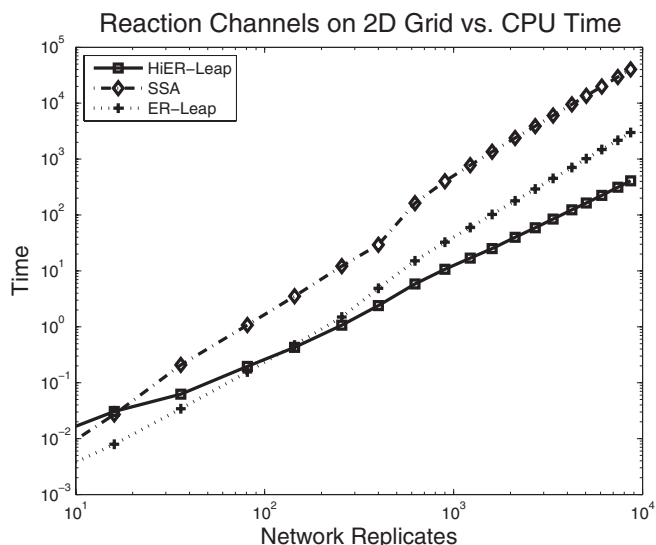

FIG. 1. The Williamowski-Rössler model as seen in Sec. IV B is used for this experiment. There are different number of network replicates on a 2D square grid with diffusion rate of 0.1. The number of replicates ranges from 4 to 8649 which equates to 64 to 189 612 reaction channels.

CaliBayes models 1-01, 1-03, 1-04, 2-01, 2-02, 3-01 and 3-02[30] are tested, on the spaced defined by the Cartesian product of the possible values for the m, b, and L parameters as described above, for parameters which result in an acceptance probability greater than about 0.05. These tests pass on these cases using the criteria of Evans *et al.*[30]

We now turn to a large, spatially coupled system.

B. Acceleration

As an exact algorithm, the key performance metric of relevance to HiER-leap is the amount of acceleration achievable. As discussed earlier, in principle adding more reaction channels and processors should increase the relative speedup over SSA. We can see this trend experimentally in Figures 1 and 2.
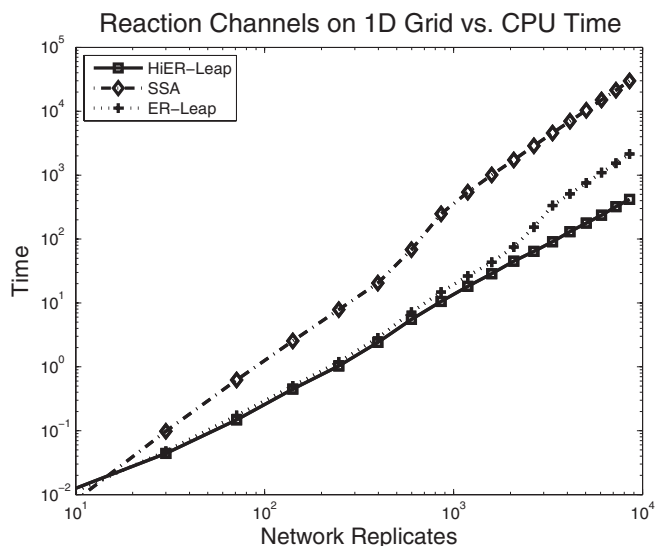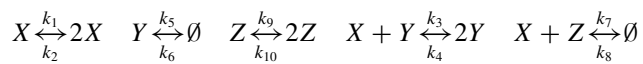

FIG. 2. The same experimental setup as used for Figure 1 except 1D diffusion is used.

We test using a spatially coupled version of the Williamowski-Rössler model[31] defined as

$$X \xrightleftharpoons[k_2]{k_1} 2X \quad Y \xrightleftharpoons[k_6]{k_5} \emptyset \quad Z \xrightleftharpoons[k_{10}]{k_9} 2Z \quad X + Y \xrightleftharpoons[k_4]{k_3} 2Y \quad X + Z \xrightleftharpoons[k_8]{k_7} \emptyset$$

replicated over a d-dimensional grid for $d = 1$ or $d = 2$. Diffusion reaction channels for all species are added between adjacent grid cells with a rate of $k_d = 0.1$. Parameters and initial conditions for each of the replicated Williamowski-Rössler grid cells are as follows: $k_1 = 900$, $k_2 = 8.3 \times 10^{-4}$, $k_3 = 0.00166$, $k_4 = 3.32 \times 10^{-7}$, $k_5 = 100$, $k_6 = 18.06$, $k_7 = 0.00166$, $k_8 = 18.06$, $k_9 = 198$, $k_{10} = 0.00166$. $X(0) = 39570$. $Y(0) = 511470$. $Z(0) = 0$.

The following tests are all run on an Apple Macintosh Pro with a Quad-Core Intel Xeon processes running a total of 8 cores at 2.26 GHz and 13 GB of RAM using OS X 10.6.8. The algorithms are coded in C++ and *Boost.Thread*[32] and the Intel Threading Building Blocks[33] are used for multithreading. Connected components were found using the depth-first search algorithm. We compiled the code using the LLVM compiler 1.0.2. The HiER-leap code may be found at http://computableplant.ics.uci.edu/hierleap/.

Results are shown in Figures 1 and 2. They show a substantial speedup of HiER-leap over SSA and ER-leap, around $100\times$ and $10\times$, respectively, as we increase the number of reaction channels to around 190 000. The spatial nature of this experiment means that blocks are neighbors with relatively few other blocks. This leads to a greater "coarse-scale" acceptance probability and therefore increased efficiency.

Additionally, we see that the slopes of the log-log runtime plots for SSA and ER-leap become nearly equal as the number of reaction channels increase. This is expected, since ER-leap finds bounds on individual reaction channels after L reaction events, and this bound is independent of the number of reaction channels. HiER-leap, however, does not have this shortcoming and has a lower slope (e.g., better asymptotic behavior) as a result.

C. HiER-leap properties

The algorithm parameters, such as leap size and hierarchical organization, require optimization before the fastest possible execution time is achieved. To find the ideal methods with which to optimize our algorithm, we explore various trade-offs here.

In Figure 3 we observe that the optimal b and L are interdependent for a given network. However, it is interesting to note that for this experiment there is a relatively large plateau of nearly equivalent optimal running times. This means that the range of reasonably good parameters is large. Furthermore, the contour plot of Figure 3 indicates that there is only one global optimum. This seemingly convex behavior indicates that finding the optimum requires only a simple hill climbing algorithm.

Thus, the results from Figure 3 indicate that finding the optimal L and hierarchical organization for a spatially distributed system is an easy optimization problem. These results, and those from ER-leap, suggest that L will generally have a local optimum that is also a global optimum. However,
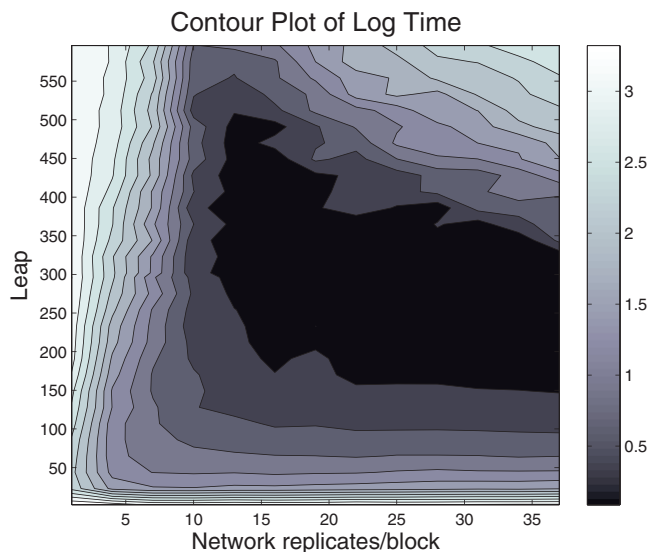

FIG. 3. Log CPU Time vs Leap and Hierarchical Structure. The Williamowski-Rössler model as seen in Sec. IV B is used for this experiment. There are 400 network replicates on a 1D grid with diffusion rate of 0.1. The model execution time depends on leap and hierarchical organization. As leap increases the amount of work per iteration goes up but the acceptance ratio goes down. Furthermore, if there are many reaction channels per block the total acceptance probability of the system goes down. However, in this situation the inner-block acceptance probability goes up. When the number of reaction channels per block goes down, the opposite trends occur. In this way the chosen leap and block organization will determine the total execution time.

the optimal configuration of the blocks and reaction channels for networks not specifically representing a spatially distributed reaction network remains an open problem.

V. SUMMARY

We have presented a novel accelerated stochastic simulation algorithm which has demonstrated an ability to sample from the CME without a loss of accuracy. Due to its hierarchical design, this method (a) scales very well with the number of reaction channels and simultaneously (b) takes advantage of parallel hardware for single trajectory samples. As far as we are aware, this is the first exact accelerated algorithm with either property (a) or (b), and is therefore of potential significance to the computational biology community.

Open questions and future work abound. For example, it is not known how well this method works on "real networks" of substantial complexity taken from biological modeling practice. We believe that modular structure in biological networks will make the method particularly useful. Additionally, it is unknown how substantial increases in the parallel architectures of future computers will increase performance.

ACKNOWLEDGMENTS

APPENDIX: BOUND ON REACTION BLOCK PROPENSITIES

We will show that for $\Delta \widehat{D}_{(I_0 L)}^{(r_1)}{}^*$ from Eq. (14) and $\Delta \widehat{D}_{(I_0 L)}^{(r_1)}$ from Eqs. (15) and (16), it is the case that $\Delta \widehat{D}_{(I_0 L)}^{(r_1)}{}^* \leq \Delta \widehat{D}_{(I_0 L)}^{(r_1)}$.

Proof by contradiction. Assume there is some $r_2 \in r_1$ and state $I' = \mathbf{n}'$ with $\forall_a n_a' \leq \tilde{n}_a$ and $\exists_a n_a' < \tilde{n}_a$, reachable from I_0 in at most $L - 1$ reaction events used to find $\Delta \widehat{D}_{(I_0 L)}^{(r_1)}{}^*$ such that $\Delta \widehat{D}_{(I_0 L)}^{(r_1)}{}^* > \Delta \widehat{D}_{(I_0 L)}^{(r_1)}$. Substituting in our definitions for $\Delta \widehat{D}_{(I_0 L)}^{(r_1)}{}^*$ and $\Delta \widehat{D}_{(I_0 L)}^{(r_1)}$, using Eq. (4), and introducing the notation that $I(r_2)$ will be the result of r_2 applied to I and $I(r_2^+)$ is the result of r_2 applied to I only for species which have net gain ($\Delta m_a^{(r_1 r_2)} > 0$), yields

$$
\begin{aligned}
\Delta \widehat{D}_{(I_0 L)}^{(r_1)}{}^* &= D_{I'(r_2)}^{(r_1)} - D_{I'}^{(r_1)} \\
&= \sum_{r_2'' \in r_1} \rho_{(r_1 r_2'')} F_{I'(r_2)}^{(r_1 r_2'')} - \sum_{r_2'' \in r_1} \rho_{(r_1 r_2'')} F_{I'}^{(r_1 r_2'')} \\
&= \sum_{r_2'' \in r_1} \rho_{(r_1 r_2'')} \left(F_{I'(r_2)}^{(r_1 r_2'')} - F_{I'}^{(r_1 r_2'')} \right)
\end{aligned}
$$

and

$$
\Delta \widehat{D}_{(I_0 L)}^{(r_1)} = \sum_{r_2'' \in r_1} \rho_{(r_1 r_2'')} \left(F_{\tilde{I}(r_2^+)}^{(r_1 r_2'')} - F_{\tilde{I}}^{(r_1 r_2'')} \right).
$$

Therefore, we can equivalently say that we are trying to disprove

$$
\sum_{r_2'' \in r_1} \rho_{(r_1 r_2'')} \left(F^{(r_1 r_2'')}(\mathbf{n}' + \Delta \mathbf{m}^{(r_1 r_2)}) - F^{(r_1 r_2'')}(\mathbf{n}') \right)
$$
$$
> \sum_{r_2'' \in r_1} \rho_{(r_1 r_2'')} \left(F^{(r_1 r_2'')}(\tilde{\mathbf{n}} + \Delta \mathbf{m}^{(r_1 r_2)}) - F^{(r_1 r_2'')}(\tilde{\mathbf{n}}) \right). \quad (A1)
$$

Note that by grouping terms by r_2'', there is a one-to-one correspondence between the summation terms on each side of the inequality.

If true, Eq. (A1) implies that there is at least one reaction channel $r_2' \in r_1$ for $\Delta m_a^{(r_1 r_2)} > 0$ such that

$$
\begin{aligned}
F^{(r_1 r_2')}(\mathbf{n}' + \Delta \mathbf{m}^{(r_1 r_2)}) - F^{(r_1 r_2')}(\mathbf{n}') &> F^{(r_1 r_2')}(\tilde{\mathbf{n}} + \Delta \mathbf{m}^{(r_1 r_2)}) \\
&- F^{(r_1 r_2')}(\tilde{\mathbf{n}}).
\end{aligned} \quad (A2)
$$

But we will show that this is impossible for any $\tilde{n}_a > n_a' \geq 0$. Note that we do not need to consider $\Delta m_a^{(r_1 r_2)} \leq 0$ because F is monotonic, the LHS will be decreased and the RHS will not change as per the definition of $\Delta \widehat{D}_{(I_0 L)}^{(r_1)}$ (negative $\Delta m_a^{(r_1 r_2)}$ are ignored).

Before proceeding we will introduce the forward difference operator, $\Delta_{F(i)}$, such that

$$
\Delta_{F(i)} f(z) \equiv f(z + i) - f(z) \quad (A3)
$$

for any function $f(z)$.

Furthermore, $F^{(r_1 r_2)}(\mathbf{n})$ can be decomposed by species into terms including chemical species C_a and those which do not. Following from Eq. (3), this allows us to rewrite $F^{(r_1 r_2)}(\mathbf{n})$

as

$$
F^{(r_1 r_2')}(\mathbf{n}) = G^{(r_1 r_2')}(\mathbf{n} \backslash \{n_a\}) \times (n_a)_k
$$

for some constant $G^{(r_1 r_2)}(\mathbf{n} \backslash \{n_a\}) \geq 0$ which does not depend on n_a, where $k = m_a^{(r_1 r_2')}$ is the input stoichiometry for reaction r_2' and species C_a, and

$$
(n)_k \equiv \frac{n!}{(n - k)!}.
$$

For Eq. (A2) to be true there must exist a species C_a such that

$$
\begin{aligned}
F^{(r_1 r_2')}(n_a' + \Delta m_a^{(r_1 r_2)}) - F^{(r_1 r_2')}(n_a') &> F^{(r_1 r_2')}(\tilde{n}_a + \Delta m_a^{(r_1 r_2)}) \\
&- F^{(r_1 r_2')}(\tilde{n}_a)
\end{aligned} \quad (A4)
$$

is true. All of the above $F^{(r_1 r_2)}$ are calculated using $n_b = \mathbf{n}' \backslash \{n_a\}$ and $n_a \in \{n_a', \tilde{n}_a\}$. When we show that n_a' will not result in a greater delta than that offered by using $\tilde{n}_a$ instead, this implies that Eq. (A2) may never be true.

Equivalent to Eq. (A4), by dividing out $G^{(r_1 r_2)}(\mathbf{n}') \geq 0$, using Eq. (A3), and setting $m = \Delta m_a^{(r_1 r_2)}$ we arrive at

$$
\Delta_{F(m)}(\tilde{n}_a)_k - \Delta_{F(m)}(n_a')_k < 0. \quad (A5)
$$

However, because $n_a' < \tilde{n}_a$, if it is shown that $\Delta_{F(m)}(n)_k$ is monotonic in n then this will imply Eq. (A5) is false.

Therefore, it just remains to be shown that $\Delta_{F(m)}(n)_k$ is monotonic in n. Consider the following equation which tests for monotonicity:

$$
\begin{aligned}
\Delta_{F(m)}&(n + 1)_k - \Delta_{F(m)}(n)_k \\
&= [\Delta_{F(1)}(n + m)_k + \ldots + \Delta_{F(1)}(n + 1)_k] - \\
&\quad [\Delta_{F(1)}(n + m - 1)_k + \ldots + \Delta_{F(1)}(n)_k] \\
&= \Delta_{F(1)}(n + m)_k - \Delta_{F(1)}(n)_k \\
&= k(n + m)_{k-1} - k(n)_{k-1} \\
&= k \left[\frac{(n + m)!}{(n + m - k + 1)!} - \frac{n!}{(n - k + 1)!} \right] \\
&= k \frac{n!}{(n - k + 1)!} \left[\frac{n + m}{n + m - k + 1} \right. \\
&\quad \times \ldots \times \left. \frac{n + 1}{n - k + 2} - 1 \right] \\
&\geq 0
\end{aligned}
$$

because $k \geq 1$ implies every factor in the long product is ≥ 1. This implies monotonicity. Therefore Eq. (A4) is false for all C_a, implying Eqs. (A2) is false, as was to be proved. $\qquad \square$

[1] D. T. Gillespie, J. Phys. Chem. **81**, 2340 (1977).
[2] M. A. Gibson and J. Bruck, J. Phys. Chem. A **104**, 1876 (2000).
[3] Y. Cao, D. T. Gillespie, and L. R. Petzold, J. Chem. Phys. **123**, 144917 (2005).
[4] H. Lu and P. Li, Comput. Phys. Commun. **183**, 1427 (2012).
[5] D. T. Gillespie and L. R. Petzold, J. Chem. Phys. **119**, 8229 (2003).
[6] D. D. Jenkins and G. D. Peterson, Comput. Phys. Commun. **182**, 2580 (2011).
[7] D. Cangelosi, in *Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques, SIMUTools '10,*

Brussels, Belgium (ICST, Belgium, 2010), pp. 36:1–36:10; see http://dl.acm.org/citation.cfm?id=1808190.

[8]B. Bayati, P. Chatelain, and P. Koumoutsakos, J. Comput. Phys. **228**, 5908 (2009).

[9]W. Zhou, X. Peng, Z. Yan, and Y. Wang, Comput. Biol. Chem. **32**, 240 (2008).

[10]E. Mjolsness, D. Orendorff, P. Chatelain, and P. Koumoutsakos, J. Chem. Phys. **130**, 144110 (2009).

[11]D. T. Gillespie, J. Chem. Phys. **115**, 1716 (2001).

[12]A. Chatterjee, K. Mayawala, J. S. Edwards, and D. G. Vlachos, Bioinformatics **21**, 2136 (2005).

[13]Y. Cao, D. T. Gillespie, and L. R. Petzold, J. Chem. Phys. **123**, 054104 (2005).

[14]Y. Cao, D. T. Gillespie, and L. R. Petzold, J. Chem. Phys. **124**, 044109 (2006).

[15]A. Slepoy, A. P. Thompson, and S. J. Plimpton, J. Chem. Phys. **128**, 205101 (2008).

[16]H. Li and L. Petzold, Int. J. High Perform. Comput. Appl. **24**, 107 (2010).

[17]G. Klingbeil, R. Erban, M. Giles, and P. K. Maini, Bioinformatics **27**, 1170 (2011).

[18]A. Gabrieli, P. Demontis, F. G. Pazzona, and G. B. Suffritti, Phys. Rev. E **83**, 056705 (2011).

[19]M. H. Serres, S. Goswami, and M. Riley, Nucleic Acids Res. **32**, D300 (2004).

[20]M. D. Riedel and J. Bruck, "Exact stochastic simulation of chemical reactions with cycle leaping," Technical Report No. ETR007, California Institute of Technology, 2006.

[21]S. Maslov and K. Sneppen, Science **296**, 910 (2002).

[22]A. Hintze and C. Adami, PLOS Comput Biol **4**, e23 (2008).

[23]T. T. Marquez-Lago and K. Burrage, J. Chem. Phys. **127**, 104101 (2007).

[24]D. Rossinelli, B. Bayati, and P. Koumoutsakos, Chem. Phys. Lett. **451**, 136 (2008).

[25]D. G. Vlachos, Phys. Rev. E **78**, 046713 (2008).

[26]K. A. Iyengar, L. A. Harris, and P. Clancy, J. Chem. Phys. **132**, 094101 (2010).

[27]M. Jeschke, R. Ewald, and A. M. Uhrmacher, J. Comput. Phys. **230**, 2562 (2011).

[28]W. Koh and K. T. Blackwell, J. Chem. Phys. **134**, 154103 (2011).

[29]M. Ullah and O. Wolkenhauer, IET Syst. Biol. **1**, 247 (2007).

[30]T. W. Evans, C. S. Gillespie, and D. J. Wilkinson, Bioinformatics **24**, 285 (2008).

[31]H. Wang and Q. Li, J. Chem. Phys. **108**, 7555 (1998).

[32]B. Kempf, C/C++ Users Journal (2002), Vol. 20.

[33]"Thread building blocks," see http://threadingbuildingblocks.org/ (2012).

A hierarchical exact accelerated stochastic simulation algorithm

David Orendorff and Eric Mjolsness

Additional information on J. Chem. Phys.