Simulation of biochemical reactions with time-dependent rates by the rejection-based algorithm

Article *in* The Journal of Chemical Physics · August 2015

DOI: 10.1063/1.4927916 · Source: PubMed

2 authors:

Vo Hong Thanh
Aalto University
26 PUBLICATIONS **237** CITATIONS
SEE PROFILE

Corrado Priami
Università di Pisa
372 PUBLICATIONS **3,842** CITATIONS
SEE PROFILE

Simulation of Biochemical Reactions with Time-Dependent Rates by the Rejection-based Algorithm

Vo Hong Thanh[1] and Corrado Priami[2]

[1)] *The Microsoft Research - University of Trento Centre for Computational and Systems Biology, Piazza Manifattura 1, Rovereto 38068, Italy.*[a)]

[2)] *Department of Mathematics, University of Trento, Italy and The Microsoft Research - University of Trento Centre for Computational and Systems Biology, Piazza Manifattura 1, Rovereto 38068, Italy.*[b)]

(Dated: 22 July 2015)

We address the problem of simulating biochemical reaction networks with time-dependent rates and propose a new algorithm based on our rejection-based stochastic simulation algorithm (RSSA)[J. Chem. Phys. 141(13):134116 (2014)]. The computation for selecting next reaction firings by our time-dependent RSSA (tRSSA) is computationally efficient. Furthermore, the generated trajectory is exact by exploiting the rejection-based mechanism. We benchmark tRSSA on different biological systems with varying forms of reaction rates to demonstrate its applicability and efficiency. We reveal that for nontrivial cases the selection of reaction firings in existing algorithms introduces approximations because the integration of reaction rates is very computationally demanding and simplifying assumptions are introduced. The selection of the next reaction firing by our approach is easier while preserving the exactness.

Keywords: Computational biology, Stochastic simulation, Rejection-based stochastic simulation algorithm, Time-dependent rejection-based stochastic simulation algorithm.

I. INTRODUCTION

The stochastic nature of biochemical systems at the molecular level arises from the randomness in reactions between molecular species. The fluctuation in population of species may result in significant effects to the genetic regulation, stochastic decision and ultimately the genetic variation[1–7]. Stochastic chemical kinetics is thus an indispensable framework for a quantitative study of stochastic noise in biochemical systems. For well-mixed environment, the biochemical system is modeled by reactions between species. The system state is modeled by a vector of population of each species. The probability that a reaction fires and moves the system to a new state is characterized by a *propensity* function. The reaction propensity is parametrized by the system state and a reaction rate determined from the system being modelled. The temporal dynamics of reactions and the distribution of system state are derived by applying the Gillespie stochastic simulation algorithm (SSA)[8,9]. There are two different formulations, but mathematically equivalent, of SSA: the direct method (DM)[8,9] and the first reaction method (FRM)[8]. DM selects a reaction to fire proportionally to its propensity. The firing time of the selected reaction is then generated following an exponential distribution. FRM generates the putative waiting time to the firing of each reaction, and selects the reaction having the smallest waiting time to be the next reaction firing. Many improvements have been introduced to accelerate both of these algorithms including the next reaction method (NRM)[10], the optimized direct method (ODM)[11], the sorting direct method (SDM)[12], the composition-rejection SSA (CR-SSA)[13,14], the tree-based search SSA[15–18], the partial-propensity SSA[19,20], the rejection-based SSA (RSSA)[21–23] and other improvements including approximate and parallel algorithms[24–30].

SSA assumes reaction rates to be constant. However, changes in the cell like volume size or temperature may alter the rate of a reaction and this must be modelled explicitly as a time-dependent function[31–34]. For example, Lu *et al.*[31] generalizes DM to take the cell growth and division into account in the simulation. The computation, however, is complex, time-consuming and limited to special forms of reaction rates (i.e., exponential forms). Anderson[34] recently introduced the modified next reaction method (MNRM) that mitigates the computation by explicitly modelling reactions as independent time-inhomogeneous Poisson processes. The number of times that an individual reaction fires in a time interval is modeled as a Poisson process with parameter equal to the integration of its propensity. Thus, to compute the waiting time to the firing of each reaction, MNRM integrates the reaction rate and then solves the corresponding *inverse problem* of the waiting time. The next reaction firing will be the one having the smallest waiting time. The generation of the waiting time of each reaction in MNRM is relying on the tractable calculation of the integration of the reaction rates and the solution of the inverse problem. The selection of next reaction firings, otherwise, introduces approximation errors if the integration of the reaction rates are difficult to derive.

In this paper we study the problem of simulating biochemical reactions where the reaction rates are expressed explicitly in a time-dependent form. We present a novel formulation by extending the rejection-based stochastic simulation algorithm (RSSA)[21]. Our time-dependent RSSA (tRSSA) exploits the rejection-based mechanism to select next reaction firings. tRSSA uses propensity bounds of reactions to select a candidate reaction, and then validates the candidate with a rejection-based test by its exact propensity value. The propensity of the candidate reaction with its time-dependent rate is

[a)] Electronic mail: vo@cosbi.eu

[b)] Electronic mail: priami@cosbi.eu

evaluated exactly at the current time. Therefore, the selection of the next reaction firing does not require to integrate reaction rates over time. The propensity bounds of a reaction are defined by specifying arbitrary bounds on both the population of species and the reaction rate. The bound on the reaction rate is computed by discretizing simulation time into intervals. The time discretization is chosen to ensure that the ratio of the lower bound reaction rate over the upper bound reaction rate in the time interval is bound by a constant, so that the acceptance probability of a candidate reaction is also bound. We remark that the selection of propensity bounds does not affect the exactness of tRSSA. The next reaction firing is always selected with a correct probability, hence the trajectories generated by tRSSA are exact.

The paper is organized as follows. Section II reviews the stochastic simulation algorithms for biochemical reactions and their extensions for time-dependent reaction rates. Section III reviews the theoretical background and presents our new tRSSA algorithm. We describe in detail how tRSSA selects the next reaction firing with time-dependent reaction rates. Section IV presents the experimental results in applying our approach to concrete models acting as benchmarks. The concluding remarks are in section V.

II. STOCHASTIC REACTION KINETICS

We consider a well-mixed reactor volume consisting of n molecular species denoted by S_i for $i = 1 \ldots n$. At time t, the n-vector $X(t) = (X_1(t), ..., X_n(t))$ represents the system state in which $X_i(t)$ denotes the number of molecules of species S_i in the system at that time. Species can interact through m reactions. A reaction R_j for $j = 1 \ldots m$ models a possible combination of species in a unidirectional reaction to produce other species.

$$R_j : v_{1j}S_1 + ... + v_{nj}S_n \xrightarrow{c_j} v'_{1j}S_1 + ... + v'_{nj}S_n \tag{1}$$

where c_j is called the (stochastic) reaction *rate*. The species on the left side of the arrow are called reactants while the ones on the right side are called products. A species that appears in both side of a reaction is called a catalyst. The non-negative integer v_{ij} and v'_{ij}, called *stoichiometric coefficients*, denote the number of molecules of a reactant consumed and the number of a product produced by firing R_j, respectively. Thus, the n-vector v_j, called *state change vector*, where ith element is $v'_{ij} - v_{ij}$ denotes the change caused by reaction R_j.

The stochastic chemical kinetics framework models the biochemical reaction system as a (continuous time) jump Markov process. Given state $X(t)$, the system may jump to one of the m possible states $X(t+\tau) = X(t) + v_j$ by firing a reaction R_j at time $t + \tau$. The probability of a reaction R_j firing in the next infinitesimal time dt is proportional to $a_j(X(t))dt$ where propensity a_j is a function of state X and the reaction rate c_j. If the reaction rate c_j is a constant, such a propensity is called *time-homogeneous* and is written as $a_j(X(t))$. The time-homogeneous propensity $a_j(X(t))$ changes only when the state changes. In case the reaction rate c_j is a time-dependent function, the propensity of a reaction R_j is denoted explicitly as $a_j(X(t), t)$, and it is called *time-inhomogeneous* because it depends not only on the state, but also on the time. In the following, we first consider the time-homogeneous case and return to the time-inhomogeneous case in next sections.

For time-homogeneous propensity, $a_j(X(t))$ is defined as:

$$a_j(X(t)) = c_j h_j(X(t)) \tag{2}$$

in which $h_j(X(t))$ counts the possible combinations of reactants involved in R_j, given the state $X(t)$ at time t. For standard mass action models, the propensity has a concrete form

$$a_j(X(t)) = c_j \prod_i \binom{X_i(t)}{v_{ij}} = c_j \prod_i \frac{X_i(t)!}{v_{ij}!(X_i(t) - v_{ij})!} \tag{3}$$

We remark that for the *synthesis reaction*, whose products are produced from external sources, we set $a_j(X(t)) = c_j$.

The probability distribution of the system state is completely described by the chemical master equation (CME)[35]. CME is a collection of differential equations which shows the probability of all possible states in the system. The solution of CME thus gives the distribution of the state $X(t)$ at any time t; however, an analytic solution of CME is hard to find in general due to the high dimensional state space. Recent work[36–38] numerically solves CME by constraining the state space exploration with a small tolerant error.

Instead of solving CME, the exact stochastic simulation algorithm (SSA) gives possible realizations of CME by sampling the joint next reaction probability density function (pdf) $p(\tau, j|x, t)$, which denotes the probability that reaction R_j fires in the next time interval $[t+\tau, t+\tau+d\tau]$ given $X(t) = x$. The pdf $p(\tau, j|x, t)$ is given by:

$$p(\tau, j|x, t) = a_j(x)exp(-a_0(x)\tau) \tag{4}$$

where

$$a_0(x) = \sum_{j=1}^{m} a_j(x) \tag{5}$$

The pdf $p(\tau, j|x, t)$ shows that the next reaction R_j fires with a discrete probability a_j/a_0 and its firing time τ follows an exponential distribution $\mathsf{Exp}(a_0)$. The direct method (DM) samples $p(\tau, j|x, t)$ by directly applying the inverse transformation which yields:

$$\tau = \frac{1}{a_0(x)} ln\left(\frac{1}{r_1}\right) \text{, and} \tag{6}$$

$$j = \text{the smallest index } j \text{ s.t. } \sum_{k=1}^{j} a_k(x) > r_2\ a_0(x) \tag{7}$$

where r_1 and r_2 are two random numbers drawn from a uniform distribution $\mathsf{U}(0, 1)$.

The first reaction method (FRM) also samples the same $p(\tau, j|x, t)$ to generate an exact simulation trajectory, but in a different way. It generates the putative time τ_j to the firing of a reaction R_j by inverse transforming the exponential distribution with rate a_j (i.e., $\tau_j = \ln(1/r_j)/a_j(x)$), and selects the reaction to fire with the smallest putative time $\tau = \min_{j=1}^{m}\{\tau_j\}$. FRM is less efficient than DM when the number of reactions m is large because it requires m random numbers for each selection of the next reaction firing. The next reaction method (NRM)[10] and its modified version[34] improve FRM by recycling the random numbers and updating the waiting times of only those reactions that are affected by the current firing. Thus, after the initialization, NRM consumes only one random number for each simulation step. Moreover, NRM speeds up the extraction of the smallest waiting time by storing waiting times of reactions in a priority queue (i.e., a binary heap).

A. Stochastic Simulation for Reactions with Time-Dependent Rates

To model time-dependent changes due to, e.g, temperature or size of the cell, reaction rates are modelled as depending on the time t explicitly. Let $c_j(t)$ be the time-dependent rate of a reaction R_j. The formula for the time-inhomogeneous propensity $a_j(X(t), t)$ of reaction R_j is represented as:

$$a_j(X(t), t) = c_j(t)h_j(X(t)) \tag{8}$$

Finding the firing time τ of time-dependent reaction rates by a generalization of the DM representation (see Eqs. 6-7) requires to solve the equation

$$\sum_{j=1}^{m} \int_{t}^{t+\tau} a_j(X(s), s)ds = \ln\left(\frac{1}{r}\right) \tag{9}$$

where r is a uniformly distributed random number. Solving Eq. 9 requires to integrate and sum m propensities at the same time. The computational cost of the solution is thus increasing with the number of reactions m. Therefore, the simulation of biochemical reactions with time-dependent reaction rates by this representation is extremely difficult and time-consuming, especially in case of complex reaction rates.

Instead of solving Eq. 9, the modified next reaction method (MNRM) solves simpler equations by explicitly representing the number of times a reaction occurs up to time t by an independent, time-inhomogeneous Poisson process[39]. More specifically, let P_j be an unit-rate Poisson process associated with reaction R_j with rate a_j. Thus, $P_j(\int_0^t a_j(X(s), s)ds)$ will count the number of times the reaction R_j occurs up to time t.

Let T_j (called the *internal time*) be the clock that determines the starting time for the next occurrence of R_j in the time frame of P_j. At time t, we have $T_j = \int_0^t a_j(X(s), s)ds$. Let $S_j > T_j$ be the first firing time of R_j in the time frame of P_j. The waiting time τ_j to the first firing of reaction R_j in this time frame will be the solution of Eq. 10.

$$\int_{t}^{t+\tau_j} a_j(X(s), s)ds = S_j - T_j \tag{10}$$

By substituting the definition of $a_j(X(t), t)$ from Eq. 8 into Eq. 10 and noting that the state $X(t)$ does not change in the time interval $[t, t + \tau_j]$, we derive that

$$\int_{t}^{t+\tau_j} c_j(s)ds = \frac{S_j - T_j}{h_j(X(t))} \tag{11}$$

The waiting time τ_j to the next firing of reaction R_j is therefore obtained by solving the inverse problem in Eq. 11. A numerical method must be applied to solve this equation if an analytical solution for τ_j does not exist.

To simulate biochemical reactions with time-dependent rates, MNRM integrates the reaction rate of each individual reaction, then solves Eq. 11 to have its waiting time τ_j. Knowing the waiting times of all reactions, the next reaction firing is selected to be the reaction R_j having smallest waiting time $\tau = \min_{j=1}^{m}\{\tau_j\}$. Furthermore, during the simulation, MNRM requires only one random number for each simulation step because it tracks both T_j and S_j. We outline the MNRM algorithm in Alg. 1 to conclude this section.

Algorithm 1: Modified Next Reaction Method (MNRM)

```
*procedure:* mnrm
*output:* a trajectory of the reaction network
initialize time $t = 0$ with state $X = x$
set $T_j = 0$ for $j = 1 \dots m$
generate $m$ random numbers $r_j \sim U(0, 1)$ and
set $S_j = \ln(1/r_j)$ for $j = 1 \dots m$ ‘
**while** $(t < t_{max})$ **do**
    compute waiting time $\tau_j$ for $j = 1 \dots m$ by solving $\int_t^{t+\tau_j} a_j(X(s), s)ds = S_j - T_j$
    select reaction $R_j$ having $\tau = \min_{j=1}^{m}\{\tau_j\}$
    set $T_j = T_j + \int_t^{t+\tau} a_j(X(s), s)ds$ for all $j = 1 \dots m$
    update time $t = t + \tau$
    update state $X = X + v_j$
    generate a random number $r \sim U(0, 1)$
    set time of the reaction firing $S_j = S_j + \ln(1/r)$
**end while**
```

III. REJECTION-BASED SIMULATION FOR REACTIONS WITH TIME-DEPENDENT RATES

As discussed in previous sections, finding the next reaction firing where reaction rates are time-dependent is relying on a tractable calculation of integration of reaction rates and solving the inverse problem to obtain the waiting times of reactions. In some cases (e.g., the sigmoidal reaction rates where such a calculation is difficult) the simulation has to approximate the reaction rates, hence introducing approximation errors in the generated trajectories. In this section we briefly review the theoretical background of the rejection-based stochastic simulation algorithm (RSSA). Then we present our new time-dependent RSSA (tRSSA) algorithm for simulating biochemical reactions with time-dependent rates. The advantage of tRSSA is that it does not require to integrate reaction rates. Thus, it allows exact simulation also in the cases in which the calculation of integration of reaction rates is difficult.

A. Theoretical Background of RSSA

RSSA is an exact and computationally efficient simulation algorithm. RSSA accelerates the simulation by reducing the average number of propensity calculations. The theoretical framework for the selection of reaction firings in RSSA is a rejection-based sampling technique. By exploiting such a rejection-based mechanism, RSSA exactly simulates the pdf $p(\tau, j|x, t)$ in Eq. 4. In other words, RSSA selects reaction R_j to fire with probability a_j/a_0 and its firing time τ is drawn from an exponential distribution $\texttt{Exp}(a_0)$ (see Thanh *et al.*[21] for a formal proof of the exactness of RSSA).

RSSA uses propensity bounds $[\underline{a_j}, \overline{a_j}]$, encompassing the exact value of the propensity $a_j(X(t))$, to select the next reaction firing. The propensity bounds are derived by bounding the state $X(t)$ to an interval $[\underline{X}, \overline{X}]$. Having propensity bounds, RSSA selects the next reaction firing in two steps as follows. First, a candidate reaction R_j is selected proportional to its propensity upper bound $\overline{a_j}$. RSSA samples the candidate reaction R_j by linearly accumulating propensity upper bounds until it finds the smallest reaction index j satisfying the inequality: $\sum_{k=1}^{j} \overline{a_k} > r_1 \cdot \overline{a_0}$ where $\overline{a_0} = \sum_{j=1}^{m} \overline{a_j}$ and r_1 is a random number in $U(0, 1)$. For large reaction networks, an efficient search can be applied to improve performance of this step[22]. Then, RSSA validates the candidate R_j through a rejection test with success probability $a_j(X(t))/\overline{a_j}$. To do that, RSSA generates a random number $r_2 \sim \texttt{U}(0, 1)$ and checks whether $r_2 \leq a_j(X(t))/\overline{a_j}$. If the check returns true, R_j is accepted to fire. In the other case, a new candidate is selected to test. The rejection test postpones computing the exact propensity $a_j(X(t))$ by noting that if $r_2 \leq \underline{a_j}/\overline{a_j}$, then $r_2 \leq \underline{a_j}/\overline{a_j} \leq a_j(X(t))/\overline{a_j}$ and RSSA will accept R_j without evaluating $a_j(X(t))$.

The firing time τ of the accepted candidate R_j is generated following an $\texttt{Erlang}(k, \overline{a_0})$ distribution in which parameter k is the number of consecutive trials with $k - 1$ rejections and R_j is accepted at the kth trial, and rate parameter $\overline{a_0} = \sum_{j=1}^{m} \overline{a_j}$. The firing time τ of an accepted candidate is in fact the sum of k independent exponentially distributed numbers with the same rate $\overline{a}_0$. This fact will be used to decompose the selection process in tRSSA for time-dependent reaction rates.

B. RSSA for Time-Dependent Reaction Rates

The time-dependent RSSA (tRSSA) uses the propensity lower bound $\underline{a_j}$ and upper bound $\overline{a_j}$ of reaction R_j for $j = 1 \dots m$ to select the next reaction firing. The derivation of the propensity bounds of the time-dependent propensity $a_j(X(t), t)$ has to consider both the state $X(t)$ and reaction rate $c_j(t)$. tRSSA defines the propensity bounds for a reaction R_j by bounding both the state and its reaction rate.

To derive the bound for the state, we bound the population of species in the state $X(t)$. For species S_i, we set an arbitrary lower bound $\underline{X}_i$ and upper bound $\overline{X}_i$ around its current population $X_i(t)$ (for typical models, $\pm 10\%$ to $\pm 20\%$ of its current population $X_i(t)$ give better performance as shown in Section IV). The state $X(t)$ therefore is bound by the *fluctuation interval* $[\underline{X}, \overline{X}]$ such that the inequality $\underline{X} \leq X(t) \leq \overline{X}$ holds for each species.

To bound the reaction rate $c_j(t)$, we can use its global minimum and maximum values over the whole simulation time interval $[0, t_{max}]$. However, the ratio of the global minimum over maximum could be very small, resulting in a low acceptance probability of the candidate reaction, hence decreasing simulation performance. Therefore our strategy is to discretize the simulation time $[0, t_{max}]$ into k time points $0 < t_1 < \ldots < t_k = t_{max}$. The time discretization is decided so that the lower bound rate $\underline{c_j}$ and upper bound rate $\overline{c_j}$ of the reaction rate c_j in a time interval $[t_i; t_{i+1}]$ satisfy the condition $\underline{c_j}/\overline{c_j} \geq \sigma$ where σ is a predefined constant between $65\% \leq \sigma \leq 90\%$. We remark that both the fluctuation interval $[\underline{X}, \overline{X}]$ and the time discretization scheme can be chosen arbitrarily without affecting the simulation result, but only the simulation performance.

We derive the propensity bounds $\underline{a_j}$ and $\overline{a_j}$ for reaction R_j as follows. Let $\underline{h_j}$ and $\overline{h_j}$ be the minimum and maximum of h_j over the fluctuation interval $[\underline{X}, \overline{X}]$, respectively. For mass action kinetics, we can compute these extreme values easily by $\underline{h_j} = h_j(\underline{X})$ and $\overline{h_j} = h_j(\overline{X})$ since h_j is an monotonic function. Let $\underline{c_j}$ and $\overline{c_j}$ be the minimum and maximum of the rate function $c_j(t)$ over the time interval $[t_i, t_{i+1}]$. By using the definition of a_j in Eq. 8 and applying interval arithmetic[40] we can compute the propensity bounds as:

$$\begin{aligned}
[\underline{a_j}, \overline{a_j}] &= [\underline{h_j}, \overline{h_j}] \cdot [\underline{c_j}, \overline{c_j}] \\
&= [\underline{h_j}\underline{c_j}, \overline{h_j}\overline{c_j}]
\end{aligned} \tag{12}$$

Our tRSSA selects a candidate reaction R_j with probability $\overline{a_j}/\overline{a_0}$, and validates its acceptance for firing with success probability $a_j(X(t), t)/\overline{a_j}$. The rejection test requires to evaluate $a_j(X(t), t)$. However, tRSSA is still able to quickly accept R_j and avoid computing $a_j(X(t), t)$ by using the lower bound propensity $\underline{a_j}$. The selection of reaction firing R_j in tRSSA is exact because this reaction is selected with probability proportional to its exact propensity $a_j(X(t), t)$ at the current time t. Furthermore, tRSSA is computationally efficient because the evaluation of $a_j(X(t), t)$ is easier than the calculation of its integration and solving its inverse problem to find the waiting time.

The firing time of the accepted reaction R_j is the sum of exponentially distributed numbers with rate $\overline{a_0}$ until it is accepted. However, the total propensity upper bound $\overline{a_0}$ may change depending on which time interval $[t_i, t_{i+1}]$ the current time t is residing in. tRSSA thus has to update $\overline{a_0}$ to reflect the change anytime the current time t moves out of the current time interval $[t_i, t_{i+1}]$. To identify the time interval of the current time, tRSSA has to advance the time t by the waiting time τ of each candidate reaction regardless it is accepted or rejected. First, the waiting time τ of a candidate reaction is generated from an exponential distribution $\mathsf{Exp}(\overline{a_0})$. Then, tRSSA advances the current time by this amount to $t = t + \tau$. At this point, if the time t is confined in its current time interval $[t_i, t_{i+1}]$, the candidate reaction will be selected and validated to fire. Otherwise, if the time t jumps out of its current time interval, the next time interval is loaded. The new reaction rate bounds $\underline{c_j}$, $\overline{c_j}$ as well as propensity bounds $\underline{a_j}$, $\overline{a_j}$ are updated, and then a new selection step is performed.

C. The tRSSA algorithm

Alg. 2 outlines the details of tRSSA for simulating biochemical reactions with time-dependent reaction rates. The result of a tRSSA simulation is a trajectory of the reaction network starting at time $t = 0$ with an initial state $\mathbf{x}$ and finishing at time t_{max}.

The lines 2-8 set up the propensity bounds $\underline{a_j}$ and $\overline{a_j}$ for $j = 1 \dots m$ which are used by tRSSA for the selection of next reaction firings. The initialization steps are composed of three steps: 1) specifying the bound $[\underline{X}, \overline{X}]$ around the current state $X(t)$, 2) discretizing the simulation time $[0, t_{max}]$ into intervals $0 < t_1 < \dots < t_k = t_{max}$ and finally 3) computing the corresponding bounds. tRSSA then moves to the main simulation loop which is outlined in lines 9-39.

For each simulation step, tRSSA generates the waiting time τ of candidate reaction from an exponential distribution with rate $\overline{a_0}$. This step (line 11) requires a random number $r_1 \sim \mathsf{U}(0, 1)$. The time is advanced to a new time $t = t + \tau$. tRSSA then checks whether the current time t is bounded by the time point t_i. If it is indeed the case that $t \leq t_i$, the candidate reaction will be selected and validated to fire through a rejection-based selection. If $t > t_i$, we have to load a new time interval and update the propensity bounds. Note that since the reaction rate bounds and the state bounds are independent, we only need to update rate bounds while reusing the state bounds. Thus, we compute new bounds for reaction rate $c_j(t)$ for $j = 1 \dots m$ in this new time interval. The corresponding propensity bounds for reactions are updated as well to reflect changes in the reaction rate bounds and a new simulation loop will be performed. The steps for checking the current time t are implemented in lines 13-19.

The rejection-based selection of tRSSA is implemented in lines 20-38. A candidate R_j is selected so that its index j is the smallest one satisfying: $\sum_{k=1}^{j} \overline{a_k} > r_2 \overline{a_0}$ where $r_2 \sim \mathsf{U}(0, 1)$ (line 21). The candidate R_j is then validated to ensure if it is accepted with success probability $a_j / \overline{a_j}$ (lines 22-30). The validation of candidate requires a random number $r_3 \sim \mathsf{U}(0, 1)$.

If the candidate R_j is accepted, the state is updated to move to a new state $X = X + v_j$. tRSSA then checks whether the new state is in its current bound $[\underline{X}, \overline{X}]$. If this is the case, the next simulation step is performed without changing the propensity bounds. In an uncommon case where the state $X(t) \notin [\underline{X}, \overline{X}]$, a new fluctuation interval should be derived. The new bounds $\underline{h_j}$, $\overline{h_j}$ as well as propensity bounds $\underline{a_j}$, $\overline{a_j}$ are updated to reflect the changes in state. The update of propensity bounds can be performed locally by applying a Species-Reaction (SR) dependency graph[21]. The SR dependency graph is a directed bipartite graph which shows the dependency of reactions on species. A directed edge from a species S_i to a reaction R_j is in the graph if a change in the population of species S_i requires reaction R_j to recompute its propensity. Thus, when a species S_i moves out of its fluctuation interval, only the reactions R_j which are affected by this species need to update their propensity bounds.

Algorithm 2: Time-dependent Rejection-based SSA (tRSSA)

```
    procedure: tRSSA
    output: a trajectory of the reaction network
 1: initialize time $t = 0$ and state $X = \mathbf{x}$
 2: define the bound $[\underline{X}, \overline{X}]$ for state $X$
 3: discretize $[0, t_{max}]$ to $k$ intervals $0 < t_1 < \dots < t_k = t_{max}$
 4: set $i = 1$
 5: compute $\underline{c_j}$ and $\overline{c_j}$ over time interval $[t_{i-1}, t_i]$ for $j = 1 \dots m$
 6: compute $\underline{h_j}$ and $\overline{h_j}$ over abstract state $[\underline{X}, \overline{X}]$ for $j = 1 \dots m$
 7: derive propensity bounds $\underline{a_j}$ and $\overline{a_j}$ for $j = 1 \dots m$
 8: compute $\overline{a_0} = \sum_{j=1}^{m} \overline{a_j}$
 9: while $(t < t_{max})$ do
10:   generate a random number $r_1 \sim \mathsf{U}(0, 1)$
11:   compute $\tau = (-1/\overline{a_0}) \ln(r_1)$
12:   update time $t = t + \tau$
13:   if $(t > t_i)$ then
14:     set $t = t_i$
15:     update $i = i + 1$
16:     compute $\underline{c_j}$ and $\overline{c_j}$ over interval $[t_{i-1}, t_i]$ for $j = 1 \dots m$
17:     update propensity bounds $\underline{a_j}$ and $\overline{a_j}$ for $j = 1 \dots m$
18:     go to 9
19:   end if
20:   generate two random numbers $r_2, r_3 \sim U(0, 1)$
21:   select minimum index $j$ s.t. $\sum_{k=1}^{j} \overline{a_k} > r_2 \overline{a_0}$
22:   set $accept =$ false
23:   if $(r_3 \leq (\underline{a_j}/\overline{a_j}))$ then
24:     set $accepted =$ true
25:   else
26:     evaluate $a_j$ with current state $X$
27:     if $(r_3 \leq (a_j/\overline{a_j}))$ then
28:       $accepted =$ true
29:     end if
30:   end if
31:   if $(accepted)$ then
32:     update state $X = X + v_j$
33:     if $(X \notin [\underline{X}, \overline{X}])$ then
34:       define a new $[\underline{X}, \overline{X}]$ around $X$
35:       compute $\underline{h_j}$ and $\overline{h_j}$ for interval $[\underline{X}, \overline{X}]$ for $j = 1 \dots m$
36:       update propensity bounds $\underline{a_j}$ and $\overline{a_j}$ for $j = 1 \dots m$
37:     end if
38:   end if
39: end while
```

IV. NUMERICAL EXAMPLES

In this section we report the simulation results by our tRSSA algorithm in comparison with MNRM. All of these algorithms were implemented in Java and run on a Intel i5-540M processor. The implementation of the algorithms as well as the benchmark models are freely available at the url `http://www.cosbi.eu/research/prototypes/rssa`.

We report numerical simulation results on three models acting as a benchmark: 1) time-dependent transcription regulation, 2) epidemic model with periodic contract rate and 3) birth-process with sigmoidal birth rate. The reaction rates in the three models have different time-dependent forms. In the transcriptional regulatory model, the rates of reactions are modelled as an exponential function. The analytic formulas for both the integration of these rates and the solution of the inverse problem for computing the waiting times are available. Thus, in this example, the exact value of the firing time of the next reaction firing can be computed. The other two models do not allow the integration of reaction rates and/or the analytical solution of the inverse of the integration. For the epidemic model, we are able to integrate reaction rates; however, finding the reaction firing time requires to solve a non-linear equation and a numerical root finding method must be applied. In the last model, the birth rate has a steep sigmoidal form. The integration of the reaction rate in this example is extremely difficult and inconvenient to implement. Thus, MNRM has to approximate the sigmoidal birth rate by assuming it is piecewise constant during a time interval.

A. Time-Dependent Transcription Regulation

We consider a transcriptional regulatory model listed in Table I[41]. This model has 10 reactions which represents the translation of mRNA into protein M (monomer) with the transcription factor D (dimer). The dimerization of monomer M to produce dimer D is modelled by reactions R_9 and R_{10}, respectively. In this model, the transcription (reaction R_3) and then the translation (reaction R_1) occur after the transcription factor D binds to DNA (reaction R_5). The DNA can be bound at two binding regions denoted by DNA·D and DNA·2D, respectively. Reactions R_5, R_7 and R_9 have a general exponential rate form $c(t) = c_0 e^{kt}$ where c_0 is a constant which is different for each reaction and the parameter k is the same for all reactions. In this example we consider $k = -\ln(2)/P$ where the time period $P = 30$. To simplify the simulation, we do not consider the cell division process, which splits the population of species by a half after time period P.

In the following, we derive the analytical formulas for the integration of the exponential reaction rate $c_j(t) = c_0 e^{kt}$ and then the computation of the waiting time τ_j of reaction R_j, which is used in the simulation by MNRM. More specifically, we calculate the integration of $c_j(t)$ over the time interval $[t, t + \tau_j]$ by:

TABLE I: Time-Dependent Transcription Regulation

Reaction	Rate
R_1: RNA → RNA + M	0.043
R_2: M → ∅	0.0007
R_3: DNA·D → RNA + DNA·D	0.0715
R_4: RNA → ∅	0.0039
R_5: DNA + D → DNA·D	$0.0199e^{kt}$
R_6: DNA·D → DNA + D	0.4791
R_7: DNA·D + D → DNA·2D	$0.00019e^{kt}$
R_8: DNA·2D → DNA·D + D	0.8765e-11
R_9: 2M → D	$0.083e^{kt}$
R_{10}: D → 2M	0.5

$$\begin{aligned}\int_t^{t+\tau_j} c_j(s)ds &= c_0 \int_t^{t+\tau_j} e^{ks}ds \\ &= \left[\frac{c_0}{k}e^{ks}\right]_t^{t+\tau_j} \\ &= \frac{c_0}{k}e^{kt}(e^{k\tau_j} - 1) \qquad (13)\end{aligned}$$

The waiting time τ_j of reaction R_j is derived by plugging the integration of $c_j(t)$ in Eq. 13 into Eq. 11. We have:

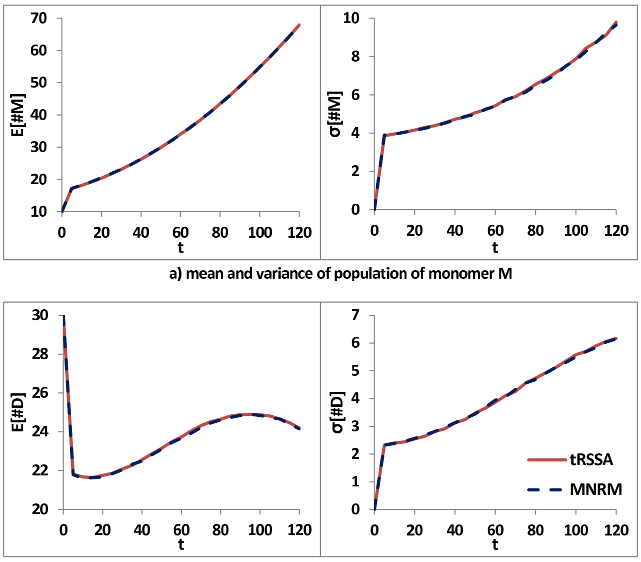
$$\tau_j = \frac{1}{k}\ln\left(\frac{k(S_j - T_j)}{c_0 e^{kt} h_j(X(t))} + 1\right) \qquad (14)$$

We thus use Eq. 14 to compute the waiting times of reactions with exponential rate form in MNRM.

tRSSA discretizes the simulation time into intervals with length $\Delta t = P/2 = 15$. Thus, for time-dependent rate, the ratio of the lower bound rate $\underline{c_j}$ and its upper bound rate $\overline{c_j}$ is $e^{-\ln 2/2} = 0.71$. The fluctuation interval $[\underline{X}, \overline{X}]$ is defined around $\pm 10\%$ of the state $X(t)$.

We simulated the time-dependent transcription regulation model to a time $T_{max} = 120$ with the initial condition #DNA $= 10$, #M $= 10$ and #D $= 30$. We performed 10000 independent simulation runs of this model and averaged the results to benchmark the algorithms. Figure 1 shows the mean (left) and variance (right) of population of monomer M and dimer D by tRSSA and MNRM. The figure shows that both the mean and variance of population of species predicted by tRSSA and MNRM strongly agree with each other.

Figure 2 compares the performances of tRSSA and MNRM in simulating the time-dependent transcription regulation. The figure shows that tRSSA outperforms MNRM in simulating this model by avoiding propensity updates. More in details, there are in average 3150 reaction firings in the simulation. MNRM thus has to perform 3150 propensity updates, while tRSSA only performs about 435 propensity updates (about 14% in comparison with the total number of propensity updates performed by MNRM). The acceptance probability of a candidate reaction in tRSSA is kept around 69.7%. Thus, tRSSA is about 22% faster than MNRM.



a) mean and variance of population of monomer M

b) mean and variance of population of dimer D

FIG. 1: Population of monomer M and dimer D in time-dependent transcription regulation predicted by tRSSA and MNRM.

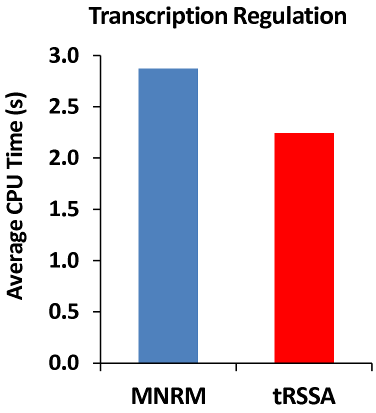


FIG. 2: Performance of tRSSA and MNRM in simulating of the time-dependent transcription regulation model.

B. Epidemic Model with Periodic Contact Rate

The reactions of the epidemic model are listed in Table II. This model contains 3 individual species: susceptible S, infected I and recovered R involved in 6 reactions to represent a simple transmission of an infectious disease[42,43]. The spreading of the disease is modelled as follows. A susceptible species S becomes infected when it contacts with an infected species I by R_1. The infected species I can recover to become disease-free R due to immunity which is modelled by R_2. In this model, a recovered species can be infected again as shown in R_3. The reactions R_4, R_5 and R_6 model the natural degradation of S, I and R, respectively. The contact rate of the disease in R_1 is modelled as a periodic function[44] and chosen to be $c(t) = c_0(1 + \epsilon\sin(\omega t))$ where constant $c_0 = 0.003$ and $\omega = 2\pi/T$ in which the period of the sine function is $T = 6$.

TABLE II: Epidemic Model with Periodic Contact Rate

Reaction	Rate
R_1: S + I → 2I	$c_0(1 + \epsilon\sin(\omega t))$
R_2: I → R	0.02
R_3: R → S	0.007
R_4: S → ∅	0.002
R_5: I → ∅	0.05
R_6: R → ∅	0.002

The integration of reaction rate of a general reaction R_j with sinusoidal form $c_j(t) = c_0(1 + \epsilon\sin(\omega t))$ in an interval $[t, t + \tau_j]$ is derived as follows:

$$\begin{aligned}\int_t^{t+\tau_j} c_j(s)ds &= c_0\int_t^{t+\tau_j}(1+\epsilon\sin(\omega s))ds\\ &= c_0\left[s-\frac{\epsilon}{\omega}\cos(\omega s)\right]_t^{t+\tau_j}\\ &= c_0(\tau_j-\frac{\epsilon}{\omega}(\cos(\omega(t+\tau_j))-\cos(\omega t)))\end{aligned} \tag{15}$$

For MNRM, by substituting Eq. 15 into Eq. 11, the waiting time τ_j of reaction R_j is thus the solution of a non-linear equation:

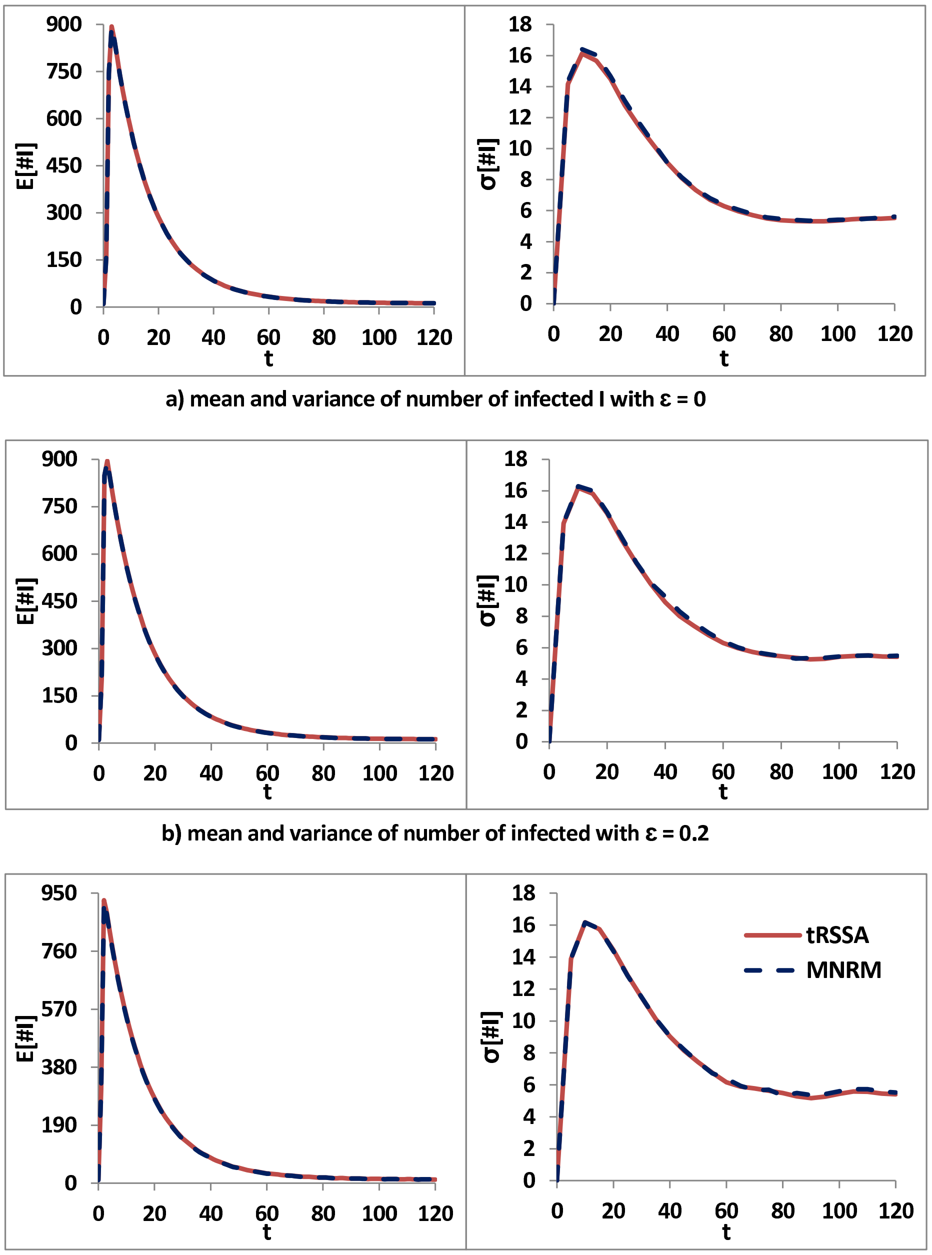
$$f(\tau_j) = \tau_j - \frac{\epsilon}{\omega}\cos(\omega(t+\tau_j)) + \frac{\epsilon}{\omega}\cos(\omega t) - \frac{S_j - T_j}{c_0 h_j(X(t))} = 0 \tag{16}$$

Because an analytic solution for the waiting time τ_j in Eq. 16 is not trivial, we numerically approximate it by applying the Newton-Raphson method. In our implementation, the criteria for stopping the Newton-Raphson search is the relative error smaller than 1.0e-7. Furthermore, we constrain the search for the minimal τ_j in the current period of the sine function that is the time interval $[t, (\underline{t/T} + 1)T]$.

The fluctuation interval $[\underline{X}, \overline{X}]$ for tRSSA is defined to be $\pm 10\%$ of the current state $X(t)$. We compute the bounds of the reaction rate $c_j(t)$ by discretizing the period $[0, T]$ into time intervals so that the ratio of minimum and maximum of the sine function in an interval is $1/2$. Specifically, we discretize the period $[0, T]$ into 6 subintervals and compute the rate bounds such that:

$$[\underline{c_j}, \overline{c_j}] = \begin{cases} [c_0, c_0(1+\epsilon/2)], & \textbf{if } t \in [0, \frac{T}{12}] \cup [\frac{5T}{12}, \frac{T}{2}] \\ [c_0(1+\epsilon/2), c_0(1+\epsilon)], & \textbf{if } t \in [\frac{T}{12}, \frac{5T}{12}] \\ [c_0(1-\epsilon/2), c_0], & \textbf{if } t \in [\frac{T}{2}, \frac{7T}{12}] \cup [\frac{11T}{12}, T] \\ [c_0(1-\epsilon), c_0(1-\epsilon/2)], & \textbf{if } t \in [\frac{7T}{12}, \frac{11T}{12}] \end{cases} \tag{17}$$

The epidemic model is simulated by performing 10000 independent runs with the initial condition $\#S = 1000$, $\#I = 10$ and $\#R = 0$ and ending time of simulation $T_{max} = 120$. Figure 3 plots the mean and variance of number of infected species I with the value of ϵ taken from the set: 0, 0.2 and 0.6. The general behaviour of the system shown in Figure 3 is that the number of infected I approaches the peak, then decreases to steady value. However, the peak value and time to approach the peak are different when increasing ϵ. For example, the



a) mean and variance of number of infected I with ε = 0

b) mean and variance of number of infected with ε = 0.2

c) mean and variance of number of infected with ε = 0.6

FIG. 3: The number of infected I predicted by tRSSA and MNRM with different settings of parameter ϵ.

peak of the number of infected I in case $\epsilon = 0.6$ is around 950 while in case of $\epsilon = 0$ is around 895. Figure 3 shows a strong agreement between tRSSA and MNRM in simulating the epidemic model for all cases of ϵ.

Figure 4 compares the performance of tRSSA and MNRM with different values of ϵ. A general conclusion from Figure 4 is that tRSSA outperforms MNRM in all cases. The speed up gain of tRSSA increases from 1.19 to 1.9 in comparison with MNRM when increasing ϵ from 0 to 0.6. In case $\epsilon = 0$, the contact rate is constant so MNRM can compute the waiting time of corresponding reactions without the need to numerically solving Eq. 16. However, when increasing ϵ MNRM has to spent time to numerically solve Eq. 16 which increases total simulation time. As shown in the figure, the simulation time of MNRM in case $\epsilon = 0.6$ is almost double its simulation time in case $\epsilon = 0$. tRSSA, however, does not need to solve any equation. It only requires to evaluate the propensity which is much more computationally efficient. Furthermore, tRSSA avoids the propensity evaluation as much as possible. Thus, the computational time of tRSSA only slightly increases when ϵ increases from 0 to 0.6. The increasing in the simulation time of tRSSA is due to the decreasing of acceptance probability which decreases from 82% with $\epsilon = 0$ to 78% with $\epsilon = 0.6$.

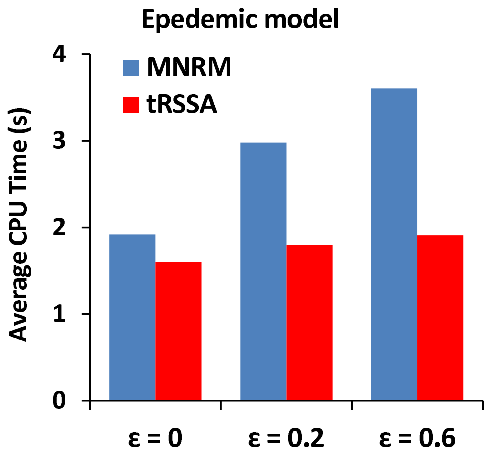


FIG. 4: Performance of tRSSA and MNRM in simulating Epidemic model with different setting of parameter ϵ.

C. Birth Process with Sigmoidal Birth Rate

In this example, we use a simple birth process with a steep sigmoidal birth rate to demonstrate the ability of tRSSA to cope with a very complex reaction rate. The birth process is a simple birth reaction: $\varnothing \overset{c(t)}{\rightarrow}$ S in which species S is continuously produced with a time-dependent birth rate $c(t)$. The birth rate has general form $c(t) = c_0\phi(t)$ where $\phi(t)$ is a steep sigmoidal equation:

$$\phi(t) = \frac{1}{1 + (\frac{t}{K})^m} \tag{18}$$

In this simulation we consider $c_0 = 1$, $K = 20$ and $m = 5$. Since the integration of $\phi(t)$ is difficult to implement[45], MNRM computes the waiting time τ of the reaction by assuming that the rate $c(t)$ is constant during the time interval $[t, t+\tau]$ (i.e., $c(t') = c(t)$ for all $t' \in [t, t+\tau]$). tRSSA discretizes the simulation time T_{max} into intervals with length $\Delta t = K/4$ and compute the bounds of the reaction rate $c(t)$ by using its monotonic decreasing property.

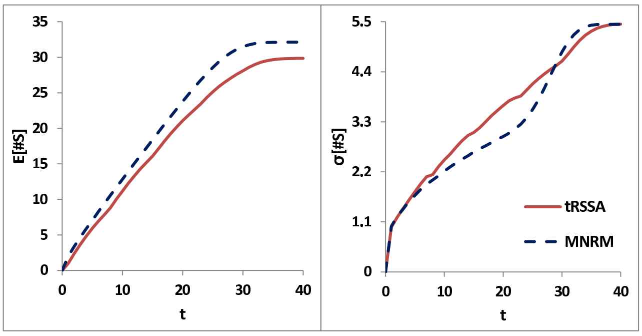


FIG. 5: Prediction of the birth process with one species S by tRSSA and MNRM.

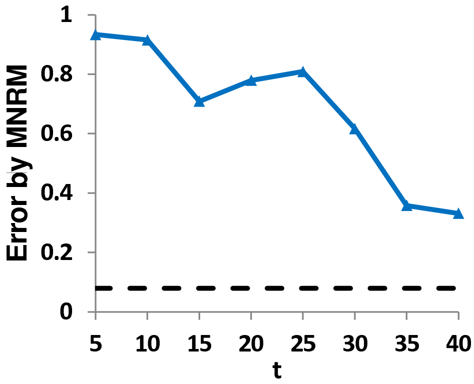


FIG. 6: The approximation error made by MNRM in simulating the birth process with one species S by assuming the birth rate is constant during a time interval. The solid line with filled triangle is the histogram distance between MNRM and tRSSA. The dash line denotes the bound of the histogram self distance $\sqrt{4B/\pi N} \approx 0.0798$

We compare simulation results by tRSSA with MNRM and quantify the approximation error introduced by MNRM in simulating this birth process in Figs. 5 - 6. We simulated the birth process with initial condition $\#S = 0$. The simulation stops at time $T_{max} = 40$. The plot in Fig. 5 shows the mean and variance of species S by tRSSA and MNRM averaging from 10000 independent runs. We further derive the histograms of species S predicted by tRSSA and MNRM and use the *histogram distance*[46] to measure the approximation error introduced by MNRM with the assumption that the sigmoidal rate $c(t)$ is piecewise constant over time interval $[t, t+\tau]$. According to Cao and Petzold[46], two different simulation algorithms are considered to have the same accuracy if the histogram distance between these simulation algorithms is less than the *self distance*, which is the histogram distance computes from two independent ensembles produced from the same simulation algorithm. The self distance is bound by $\sqrt{4B/\pi N}$ where B is the number of bins that is used to divide the entire range of population of species S into a series of small intervals in order to compute the histogram and N is the total number of simulation runs. Thus, a larger histogram distance between an approximate simulation and an exact simulation in comparison with the self distance will quantify the approximation error introduced by the approximate algorithm. In our experiment, we derived the histograms of $\#S(t)$ by tRSSA and MNRM, respectively, and computed their histogram distance by fixing $N = 10000$ runs for each algorithm and total $B = 50$ bins. The bound of the histogram self distance in this setting is thus $\sqrt{4B/\pi N} \approx 0.0798$. Fig 6 plots the approximation error made by MRNM by comparing the histogram distance between it and tRSSA and the bound of self distance. As shown in Fig. 6, the approximation error made by MNRM is very significant especially when the population of S is low. For example, at time $t = 10$ with the average population of species S around 10, the histogram distance is 0.91 which is 11.4 times larger than the bound of the self distance 0.0798.

We compare the performance of tRSSA and MNRM on two models. The first model is composed of one birth process where only one species is produced. The second model has 10 birth processes which produce 10 different species. The

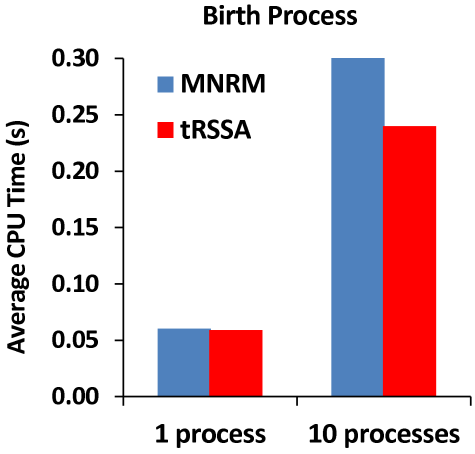


FIG. 7: Performance of tRSSA and MNRM in simulating the birth process with different settings.

second model is created by replicating 10 times the birth process. The initial population is set to zero for all of the species in both of these models. Fig. 7 plots in the left performances of tRSSA and MNRM for one birth process model, while the right is for the ten birth process model. The performance of tRSSA and MNRM for the one birth process model is comparable. However, for the ten-process model tRSSA is roughly 30% faster than MNRM. Such a significant speed-up gain is achieved due to reducing the number of propensity updates. In the ten-process model, tRSSA just performs 7 propensity updates instead of 230 as done by MNRM. We remark that evaluating the sigmoidal rate is time-consuming. Furthermore, the acceptance probability of tRSSA is kept at 90%.

V. CONCLUSIONS

In this paper we studied the problem of simulating biochemical reactions with time-dependent rates. We proposed an exact and efficient simulation algorithm, called tRSSA, to cope with time-dependent rates. Our tRSSA algorithm exploits the propensity bounds of reactions and the rejection-based mechanism to select the next reaction firings. The propensity bounds are derived by bounding both the state and reaction rates. By using these bounds, tRSSA selects the reaction firings without the need to integrate the reaction rates, while preserving the exactness of the selection. Thus, the simulated trajectories by our algorithm are exact. The experiments on the benchmark has shown that our new approach is not only exact but also computationally efficient with respect to existing approaches. In the sense of exactness, tRSSA is significantly better for simulating models where reaction rates are complex and difficult to integrate. In the sense of performance, tRSSA significantly improves simulation performance for large reaction networks where the number of propensity updates is large.

[1] Juan M. Pedraza and Alexander van Oudenaarden. Noise propagation in gene networks. *Science*, 307(5717):1965–1969, 2005.

[2] Harley H. McAdams and Adam Arkin. It's a noisy business! genetic regulation at the nanomolar scale. *Trends in Genetics*, 15(2), 1999.

[3] Harley H. McAdams and Adam Arkin. Stochastic mechanisms in gene expression. *PNAS*, 94(3):814–9, 1997.

[4] J. Vilar, H. Kueh, N. Barkai, and S. Leibler. Mechanisms of noise-resistance in genetic oscillators. *PNAS*, 99(9):5988–5992, 2002.

[5] Adam Arkin, John Ross, and Harley H. McAdams. Stochastic kinetic analysis of developmental pathway bifurcation in phage lambda-infected escherichia coli cells. *Genetics*, 149(4):1633–1648, 1998.

[6] Ertugrul M. Ozbudak, Mukund Thattai, Iren Kurtser, Alan D. Grossman, and Alexander van Oudenaarden. Regulation of noise in the expression of a single gene. *Nature Genetics*, 31:69–73, 2002.

[7] Jonathan M. Raser and Erin K. O'Shea. Noise in gene expression: Origins, consequences, and control. *Science*, 309(5743):2010–2013, 2005.

[8] Daniel Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *J. Comp. Phys.*, 22(4):403–434, 1976.

[9] Daniel Gillespie. Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem.*, 81(25):2340–2361, 1977.

[10] Michael Gibson and Jehoshua Bruck. Efficient exact stochastic simulation of chemical systems with many species and many channels. *J. Phys. Chem. A*, 104(9):1876–1889, 2000.

[11] Yang Cao, Hong Li, and Linda Petzold. Efficient formulation of the stochastic simulation algorithm for chemically reacting systems. *J. Chem. Phys.*, 121(9):4059, 2004.

[12] James McCollum and *et al.* The sorting direct method for stochastic simulation of biochemical systems with varying reaction execution behavior. *Comp. Bio. Chem.*, 30(1):39–49, 2006.

[13] Tim Schulze. Efficient kinetic monte carlo simulation. *J. Comp. Phys.*, 227(4):2455–2462, 2008.

[14] Alexander Slepoy, Aidan P. Thompson, and Steven J. Plimpton. A constant-time kinetic monte carlo algorithm for simulation of large biochemical reaction networks. *J. Chem. Phys.*, 128(20):205101, 2008.

[15] Vo H. Thanh and Roberto Zunino. Tree-based search for stochastic simulation algorithm. In *Proc. of ACM-SAC*, pages 1415–1416, 2012.

[16] Vo H. Thanh and Roberto Zunino. Adaptive tree-based search for stochastic simulation algorithm. *Int. J. of Computational Biology and Drug Design*, 7(4):341–57, 2014.

[17] James Blue, Isabel Beichl, and Francis Sullivan. Faster monte carlo simulations. *Phys. Rev. E*, 51(2):867–868, 1995.

[18] Hong Li and Linda Petzold. Logarithmic direct method for discrete stochastic simulation of chemically reacting systems. Technical Report, 2006.

[19] Sagar Indurkhya and Jacob Beal. Reaction factoring and bipartite update graphs accelerate the gillespie algorithm for large-scale biochemical systems. *PLoS ONE*, 5(1):8125, 2010.

[20] Rajesh Ramaswamy, Nlido Gonzlez-Segredo, and Ivo F. Sbalzarini. A new class of highly efficient exact stochastic simulation algorithms for chemical reaction networks. *J. Chem. Phys.*, 130(24):244104, 2009.

[21] Vo H. Thanh, Corrado Priami, and Roberto Zunino. Efficient rejection-based simulation of biochemical reactions with stochastic noise and delays. *J. Chem. Phys.*, 141(13):134116, 2014.

[22] Vo H. Thanh, Roberto Zunino and Corrado Priami. On the Rejection-based Algorithm for Simulation and Analysis of Large-Scale Reaction Networks. *J. Chem. Phys.*, 142(24):244106, 2015.

[23] Vo H. Thanh. *On Efficient Algorithms for Stochastic Simulation of Biochemical Reaction Systems*. PhD thesis, University of Trento, Italy. `http://eprints-phd.biblio.unitn.it/1070/`, 2013.

[24] S. Mauch and M. Stalzer. Efficient formulations for exact stochastic simulation of chemical systems. *IEEE/ACM Trans. on Computational Biology and Bioinformatics*, 8(1):27–35, 2011.

[25] Werner Sandmann. Discrete-time stochastic modeling and simulation of biochemical networks. *Comput. Biol. Chem.*, 32(4):292, 2008.

[26] Daniel Gillespie. Approximate accelerated stochastic simulation of chemically reacting. *J. Chem. Phys.*, 115:1716–1733, 2001.

[27] Yang Cao, Daniel Gillespie, and Linda Petzold. Efficient step size selection for the tau-leaping simulation method. *J. Chem. Phys.*, 124(4):44109, 2006.

[28] Anne Auger, Philippe Chatelain, and Petros Koumoutsakos. R-leaping: Accelerating the stochastic simulation algorithm by reaction leaps. *J. Chem. Phys.*, 125(8):84103, 2006.

[29] Hong Li and Linda Petzold. Efficient parallelization of the stochastic simulation algorithm for chemically reacting systems on the graphics processing unit. *Intl. J. of High Performance Computing Applications*, 24(2):107–116, 2010.

[30] Vo Hong Thanh and Roberto Zunino. Parallel stochastic simulation of biochemical reaction systems on multi-core processors. In *Proc. of CSSim*, pages 162–170, 2011.

[31] Ting Lu, Dmitri Volfson, Lev Tsimring, and Jeff Hasty. Cellular growth and division in the gillespie algorithm. *IEE Systems Biology*, 1(1):121–128, 2004.

[32] Paola Lecca. A time-dependent extension of Gillespie algorithm for biochemical stochastic π-calculus. In *Proc. of ACM-SAC*, pages 137–144, 2006.

[33] Raluca R. P. Purtan and Andreea Udrea. A modified stochastic simulation algorithm for time-dependent intensity rates. In *Proc. of 19th International Conference on Control Systems and Computer Science (CSCS), 2013* , pages 365–369, 2013.

[34] David F. Anderson. A modified next reaction method for simulating chemical systems with time dependent propensities and delays. *J. Chem. Phys.*, 127(21):214107, 2007.

[35] Daniel Gillespie. A rigorous derivation of the chemical master equation. *Physica A*, 188(1-3):404–425, 2007.

[36] Brian Munsky and Mustafa Khammash. The finite state projection algorithm for the solution of the chemical master equation. *J. Chem. Phys.*, 124(044104), 2006.

[37] K. Burrage, M. Hegland, S. Macnamara, and R. Sidje. A krylov-based finite state projection algorithm for solving the chemical master equation arising in the discrete modelling of biological systems. In *Proc. of The AA Markov 150th Anniversary Meeting*, pages 21–37, 2006.

[38] M. Mateescu, V. Wolf, F. Didier, and T.A. Henzinger. Fast adaptive uniformisation of the chemical master equation. *IET Syst. Biol.*, 4(6):441–452, 2010.

[39] David F. Anderson and Thomas G. Kurtz. Continuous time markov chain models for chemical reaction networks. In H. Koeppl et al, editor, *Design and Analysis of Biomolecular Circuits: Engineering Approaches to Systems and Synthetic Biology*. Springer, 2011.

[40] Ramon E. Moore, R. Baker Kearfott, and Michael J. Cloud Introduction to Interval Analysis. *SIAM*, 2009.

[41] John Goutsias. Quasiequilibrium approximation of fast reaction kinetics in stochastic biochemical systems. *J. Chem. Phys.*, 122(18):184102, 2005.

[42] Roy M. Anderson. *Population Dynamics of Infectious Diseases: Theory and Applications*. Chapman and Hall, London-New York, 1982.

[43] D. J. Daley and J. Gani. *Epidemic Modeling: An Introduction*. Cambridge University Press, 2005.

[44] Nicolas Bacaër. On the stochastic sis epidemic model in a periodic environment. *J. Math. Biol.*, pages 1–21, 2014.

[45] Shev MacNamara and Kevin Burrage. Stochastic modelling of t cell homeostasis for two competing clonotypes via the master equation. In Carmen Molina-Pars and Grant Lythe, editors, *Mathematical Models and Immune Cell Biology*, pages 207–225. Springer, 2011.

[46] Yang Cao and Linda Petzold. Accuracy limitations and the measurement of errors in the stochastic simulation of chemically reacting systems. *J. Comp. Phys.*, 212(1):6–26, 2006.