Efficient Constant-Time Complexity Algorithm for Stochastic Simulation of Large Reaction Networks

Vo Hong Thanh, Roberto Zunino, and Corrado Priami

Abstract—Exact stochastic simulation is an indispensable tool for a quantitative study of biochemical reaction networks. The simulation realizes the time evolution of the model by randomly choosing a reaction to fire and update the system state according to a probability that is proportional to the reaction propensity. Two computationally expensive tasks in simulating large biochemical networks are the selection of next reaction firings and the update of reaction propensities due to state changes. We present in this work a new exact algorithm to optimize both of these simulation bottlenecks. Our algorithm employs the composition-rejection on the propensity bounds of reactions to select the next reaction firing. The selection of next reaction firings is independent of the number reactions while the update of propensities is skipped and performed only when necessary. It therefore provides a favorable scaling for the computational complexity in simulating large reaction networks. We benchmark our new algorithm with the state of the art algorithms available in literature to demonstrate its applicability and efficiency.

Index Terms—Computational biology, Stochastic simulation, Rejection-based stochastic simulation algorithm.

1 Introduction

Biological processes at molecular level are noisy due to the discreteness of species and the randomness of reaction firings [1], [2], [3], [4]. The effects of noise may lead to significant changes in cellular behavior and ultimately in biological response [5], [6]. Stochastic modeling and simulation of biological networks provide a framework for a quantitative study of biological systems by taking biological noise into account.

In the stochastic chemical kinetics framework, the state of the system is modeled as a vector of population of each molecular species. The interactions of species to produce necessary substances for cells are encoded by chemical reactions between species. The occurrence of a reaction event is associated with a probability that is proportional to a *propensity*, which depends on the reaction kinetics. The dynamic behavior of the biochemical network is fully described by the chemical master equation (CME) [7] and its solution can be realized by an exact simulation procedure called the stochastic simulation algorithm (SSA) [8], also known as the direct method (DM) [9]. SSA is exact in the sense that it selects a reaction firing and moves the system to a new state according to a probability distribution that is derived under the same hypothesis as CME. Recently, extensions of SSA have been introduced for considering environmental effects such as biochemical reactions with time-dependent rates [10], [11], [12], [13], reactions with delay times [12], [14], [15], [16], and inhomogeneous space [17], [18].

The performance of SSA for large reaction networks is prohibitively expensive due to two main factors: searching for next reaction events, and updating propensities of reactions. In addition many simulation runs must be performed in order to obtain a reasonable statistical estimation of the system behavior that further pronounce the performance problem of SSA. Different formulations to accelerate these simulation steps of SSA are introduced to improve its performance in simulating large, complex biochemical reaction networks. The next reaction method (NRM) [19] uses a binary heap to extract the reaction firing. It also employs a *reaction dependency graph* to decide which reactions update their propensities after a reaction firing. The optimized direct method (ODM) [20], [21] improves the search for next reaction firings by sorting reactions in descending order of propensities. The multi-dimensional search accelerates the search for next reaction firings by dividing reactions into groups [22]. The selection of the next reaction firing by the multi-dimensional search is composed of selecting a group and locating the next reaction within that group. The finest strategy for grouping of reactions is when each group contains only two reactions which is equivalent to a tree structure where reactions are stored on its leaves. The selection in this case is a tree traversal procedure [23], [24], [25], [26]. The SSA with composition rejection search strategy (SSA-CR) [27], [28] also groups reactions into groups, but the selection of the next reaction firing in a group employs a rejection-based sampling instead. The search cost of the SSA-CR is only depending on the number of groups. So, the performance of SSA-CR is contributed mostly by the propensity update cost. The partial-propensity direct Method (PDM) [29], [30], [31] is a special formulation of DM for improving the simulation performance by exploiting the

• *Vo Hong Thanh is with The Microsoft Research - University of Trento Centre for Computational and Systems Biology, Piazza Manifattura 1, Rovereto 38068, Italy.*
E-mail: vo@cosbi.eu

• *Roberto Zunino is with Department of Mathematics, University of Trento, Italy.*
E-mail: roberto.zunino@unitn.it

• *Corrado Priami is with 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.*
E-mail: priami@cosbi.eu

special form of the *mass-action* propensity function of *elementary reactions* (reactions with at most two reactants). PDM factorizes propensities of reactions into partial propensities and groups these partial propensities by their common reactants. Thus, after a reaction firing, PDM can update propensities of reactions with the shared reactant collectively in one operation. The partial-propensity approach, however, is limited to class of reactions involving at most two reactants and their propensities must be factorizable [30]. PDM does not apply if the model has reactions having more than two reactants (e.g., trimolecular reactions [32]) or a complex propensity function applied (e.g., Michaelis-Menten kinetics [33]).

The rejection-based stochastic simulation algorithm (RSSA) [16], [35] is introduced recently to accelerate the exact simulation. It is specifically tailored for reaction networks where complex propensity functions are applied. For instance, RSSA [13] is able to generate exact trajectory for nontrivial cases where the reaction propensity takes a very complex function (e.g., steep sigmoidal form), while existing algorithms introduces approximations because the computation by these algorithms is very demanding and simplifying assumptions are introduced. The principle of RSSA is using propensity bounds of reactions to select next reaction firings. The propensity bound of a reaction is an interval bounding all possible concrete propensity values of the reaction. The propensity bounds of reactions are derived by specifying an arbitrary bound on the population of each species, which is called *fluctuation interval* or *abstract state*. RSSA updates the propensity bounds infrequently, only when the state moves out of its fluctuation interval. We remark that the choice of the fluctuation interval and propensity bounds does not affect the exactness of RSSA, but only its efficiency. RSSA selects the next reaction firings in two steps. First, a candidate reaction is randomly selected proportionally to its propensity upper bound. Then, a rejection-based test is performed to ensure that the selected reaction fires with the same probability determined by SSA. The evaluation of the exact propensity of the candidate reaction which is required by the rejection test in RSSA is postponed by exploiting its propensity lower bound. The exact propensity is evaluated only if needed. If a reaction is accepted to fire, only the state is updated and the next simulation step is performed without recomputing the propensity bounds. Only in uncommon cases when the population of a species exits its fluctuation interval, a new fluctuation interval for this species is defined. The propensity bounds of reactions as well have to be updated to reflect the changes. Improvements of RSSA [13], [34] have been introduced to improve its efficiency and applicability. For instance, the simultaneous RSSA (SRSSA) is an efficient formulation of RSSA to support the simulation analysis. SRSSA is able to generate many trajectories simultaneously in a single simulation run. It utilizes a single data structure across all simulations to select the reaction firings to form a trajectory of each simulation. The memory requirement for SRSSA is thus independent of the number of generated trajectories.

We present in this paper a new algorithm, called RSSA-CR, to efficiently simulate large-scale biochemical reaction networks. RSSA-CR exploits the composition-rejection search on the propensity bounds to select next reaction firings. The reactions in RSSA-CR are grouped by their propensity upper bounds. Therefore, the search cost for the selection of reaction firings in RSSA-CR is only proportional to the number of groups and independent of the number of reactions. After a reaction firing, RSSA-CR only updates the system state, while the propensity bounds as well as underlying data structure do not need to be updated. The propensity updates are performed only when necessary. Furthermore, RSSA-CR recomputes propensity bounds locally for only reactions affected by the species whose population exits the fluctuation interval by using a *species-reaction (SR) dependency graph* [16]. RSSA-CR is thus providing a significant improvement for both the search and propensity updates, and makes it suitable for simulation of large, complex networks. For models in which the ratio between the largest propensity and the smallest one is bounded and the SR dependency graph is sparse, the computational time complexity of RSSA-CR is constant.

The paper is organized as follows. Section 2 provides the background of SSA for simulating biochemical reactions. Section 3 presents our new RSSA-CR algorithm. We describe in detail how to employ the composition-rejection search on propensity bounds to select the next reaction firing in order to improve both the search and propensity updates of the simulation. Section 4 shows the numerical results of our new algorithm on concrete models acting as benchmarks to demonstrate the applicability and efficiency with respect to the state of the art of SSA optimization. The concluding remarks are in section 5.

2 Stochastic simulation algorithm

We consider a well-mixed biochemical reaction network consisting of N molecular species labeled S_i for $i = 1 \dots N$. The state $X(t)$ of the system at a time t is a N-vector $X(t) = (X_1(t), ..., X_N(t))$ where $X_i(t)$ is the absolute number of molecules of species S_i in the system at the time. Species interacts with each other to produce other species through M reactions. Each reaction R_j for $j = 1 \dots M$ has a general form.

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

where c_j is the stochastic *rate constant*. The species on the left side of the arrow are called *reactants*, while the ones on the right side are called *products*. The non-negative integer v_{ij} and v'_{ij}, respectively, called *stoichiometric coefficients*, denote how many molecules of a reactant are consumed and how many molecules of a product are produced.

A reaction R_j in the stochastic chemical kinetics is characterized by two measurements that are: a *propensity* a_j and a state change vector v_j. The propensity a_j is a state-dependent function defined so that $a_j(X(t))dt$ gives the probability of reaction R_j occurring in the the next time $t + dt$ given the system state $X(t)$ at time t. The state change vector v_j characterizes how many molecules of each species in the state $X(t)$ changes due to an occurrence of R_j. The ith element of the state change vector v_j is equal to $v'_{ij} - v_{ij}$. Thus, an occurrence of R_j moves the system from state $X(t)$ at time t to a new state $X(t + \tau) = X(t) + v_j$ given that R_j is selected to fire at time $t + \tau$.

An exact formula for the propensity of a reaction is depending on the chemical kinetics applied for the system under study. This is referred to as the *fundamental hypothesis* of the stochastic chemical kinetics [7], [8]. For mass action kinetics, propensity a_j of a reaction R_j exists and is defined by:

$$a_j(X(t)) = c_j h_j(X(t)) \qquad (2)$$

where c_j is the stochastic rate constant and $h_j(X(t))$ counts the number of distinct combinations of reactants involved in R_j, given the state $X(t)$ at time t. In case of the *synthesis reaction* (or source reaction) where species are produced from an external source, the number of combinations of reactants is $h_j(X(t)) = 1$.

The probability distribution of the system state, by the stochastic chemical kinetics formulation, is completely described by the chemical master equation (CME) [7]. The solution of CME, however, is hard to find analytically or numerically due to the high dimensional state space, even though recent work [36], [37] tries to numerically solve CME with a predefined tolerant error. The stochastic simulation algorithm (SSA) [8], [9] is an alternative approach for solving CME. SSA does not explore the whole state space, but only realizes a possible new state by firing one reaction at a time. SSA realizes a trajectory of the reaction network by sampling the joint probability density function (pdf) $p(\tau, \mu)$ with $p(\tau, \mu)d\tau$ defining the probability that reaction R_μ fires in the next infinitesimal time $t + \tau + d\tau$, given the system having state $X(t)$ at time t. The analytical form of $p(\tau, \mu)$ is given as:

$$p(\tau, \mu) = a_\mu exp(-a_0\tau) \tag{3}$$

where $a_0 = \sum_{j=1}^{M} a_j$. Integrating $p(\tau, \mu)$ over τ from 0 to ∞ gives that the probability that reaction R_μ occurring in the next time is a discrete probability a_μ/a_0. Summing $p(\tau, \mu)$ over all possible reaction index from 1 to M gives that the probability distribution of the firing time τ is an exponential distribution $\mathsf{Exp}(a_0)$. The behavior of the system given a sufficient number of SSA realizations is ensured to converge to the result of CME.

SSA samples $p(\tau, \mu)$ and constructs a simulation trajectory as follows (see Algorithm 1). It computes propensities a_j for $j = 1 \ldots M$ at the beginning. Then, each SSA simulation step selects the next reaction firing R_μ and its firing time τ by:

$$\tau = \frac{1}{a_0} ln\left(\frac{1}{r_1}\right) \tag{4}$$

$$\mu = \text{smallest reaction index such that: } \sum_{j=1}^{\mu} a_j > r_2 a_0 \tag{5}$$

where r_1 and r_2 are random numbers from a uniform distribution $\mathsf{U}(0, 1)$. Knowing the next reaction firing and its firing time, SSA advances time to $t + \tau$ and updates the state according to the selected reaction R_μ to a new state $X(t + \tau) = X(t) + v_\mu$. It then updates propensities of reactions to reflect the changes in the system state.

Algorithm 1 SSA
```
1: time $t = 0$ with state vector $X = \mathbf{x}$
2: compute propensity $a_j$ for $j = 1 \ldots M$ and $a_0 = \sum_{j=1}^{M} a_j$
3: while ($t < T_{max}$) do
4:     compute $\tau = (1/a_0)\ln(1/r_1)$ with $r_1 \sim \mathsf{U}(0, 1)$
5:     select minimum reaction index $\mu$ s.t. $\sum_{j=1}^{\mu} a_j > r_2 a_0$ with
       $r_2 \sim \mathsf{U}(0, 1)$
6:     advance time $t = t + \tau$ and state $X = X + v_\mu$
7:     update propensity $a_j$ for $j = 1 \ldots M$ and total sum $a_0$
8: end while
```

For simulation of large reaction networks, the computational cost of SSA in Alg. 1 is largely dominated by the cost of the search for next reaction firings (line 5) and the cost of propensity updates after each reaction firing (line 7).

The selection of the next reaction firing in SSA is inefficient because it increases linearly with the number of reactions. The search time complexity can be improved by the composition-rejection search (SSA-CR). SSA-CR reduces the search by grouping reactions and applying the acceptance-rejection for selecting the next reaction in the group. SSA-CR groups a reaction R_j into a group G_i if its propensity a_j satisfies $2^{q_i - 1} \leq a_j \leq 2^{q_i}$. The index q_i of the group G_i is thus computed by $q_i = [\log(a_j)]$ where the truncation operator $[x]$ returns the largest integer not greater than x. The selection of the next reaction firing in SSA-CR is composed of two steps. First, it selects the group G_l, which contains the next reaction firing. Then, the next reaction firing R_μ in the group G_l is located by applying the acceptance-rejection with hat function 2^{q_l}. The average number of rejection tests to select the next reaction firing R_μ is bounded by 2 because of $a_\mu/2^{q_l} \geq 1/2$. Therefore, the selection of next reaction firings of SSA-CR depends only on the number of groups. If the number of groups is bounded by a small constant, the search time complexity of SSA-CR is constant.

A naive implementation for propensity updates after a reaction firing is to recompute all propensities of reactions. This approach has linear time complexity with the number of reactions. An advanced approach will update only propensities of reactions affected by the reaction firing. This is done by employing the reaction dependency graph [19] that is a directed graph showing the dependency of reactions in the network. A directed edge from reaction R_j to reaction R_i exists if firing R_j affects the propensity of reaction R_i and urges R_i to recompute its propensity. The dependency graph reduces the number of propensity updates to model-dependent. Thus, if 1) the number of reactions which requires to update their propensities after a reaction firing is bounded by a small constant (i.e., the dependency graph is sparse) and 2) the propensities do not vary significantly, then the computational cost of SSA-CR is constant time complexity. The dependency graph of practical models, however, is often dense and highly connected. The propensity update cost is high, which often contributes about 65% to 85% to the total simulation time. Especially, for some special models where the number of affected reactions by a reaction firings is $O(M)$, the propensity update cost contributes even up to 99% of the simulation time.

3 Rejection-based Algorithm with Composition-Rejection Search

We present in this section our new algorithm RSSA-CR to accelerate stochastic simulation of large reaction networks by employing the composition-rejection on the propensity bounds. We first review the principle of the rejection-based stochastic simulation algorithm (RSSA) for selection of reaction firings by using propensity bounds. Then, we present the details of data structures and procedure of RSSA-CR to optimize both of the computationally expensive steps of the simulation discussed in the previous section. The selection of reaction firings in RSSA-CR is independent of the number of reactions and bounded only by the number of groups. The propensity updates of RSSA-CR are avoided and performed infrequently. Furthermore, by using a Species-Reaction (SR) dependency graph, propensity updates in RSSA-CR can be preformed locally.

3.1 Background on RSSA

The rejection-based stochastic simulation algorithm (RSSA) [16], [34], [35] is an exact simulation algorithm. RSSA correctly selects the next reaction R_μ to fire with probability a_μ/a_0 and its firing time τ is drawn from an exponential distribution $\texttt{Exp}(a_0)$ (see Thanh *et al.* [16] for a complete proof of its correctness). RSSA accelerates the simulation by reducing average number of propensity updates during the simulation. The propensity updates are avoided and collapsed as much as possible by making use of propensity lower bound $\underline{a_j}$ and upper bound $\overline{a_j}$ of each reaction R_j to select reaction firings. The propensity bounds of reactions are derived by bounding the population $X_i(t)$ of each species S_i to an arbitrary *fluctuation interval* $[\underline{X}_i, \overline{X}_i]$. The constraint $X(t) \in [\underline{X}, \overline{X}]$ holds for each species on the state $X(t)$. The propensity bounds for reactions are computed by applying an interval analysis or an optimization technique [39]. For mass-action kinetics, the computation of $\underline{a_j}$ and $\overline{a_j}$ is easy by making use of its monotonic property.

The selection of reaction firings by using propensity bounds is performed in two steps. First, a candidate reaction R_μ is selected with probability $\overline{a_\mu}/\overline{a_0}$ where $\overline{a_0} = \sum_{j=1}^{M} \overline{a_j}$. The candidate R_μ then enters a rejection test for validation with success probability $a_\mu/\overline{a}_\mu$. The validation requires computing the exact propensity a_μ, but it is postponed by using the fact that if the candidate R_μ is accepted with probability $\underline{a}_\mu/\overline{a}_\mu$ then it is also accepted with probability $a_\mu/\overline{a}_\mu$ because of the inequality $\underline{a}_\mu/\overline{a}_\mu \le a_\mu/\overline{a}_\mu$. If R_μ is accepted through the rejection test, its firing time is generated. The firing time of an accepted candidate R_μ is generated following an $\texttt{Erlang}(k, \overline{a_0})$ distribution where k is the number of trials until it is accepted. In case R_μ is rejected, a new candidate reaction will be selected.

A simple strategy for realizing the candidate reaction R_μ is to represent the M propensity upper bounds $\overline{a_j}$ for $j = 1 \dots M$ by an array of size M and linearly accumulates propensity upper bounds until a smallest reaction index μ satisfying $\sum_{j=1}^{\mu} \overline{a_j} > r \cdot \overline{a_0}$ where $r \sim \texttt{U}(0,1)$. This search strategy is efficient for small models because it does not require to build any complex data structure. However, the search will become very computational expensive for large models because its computational cost is linearly increasing with the number of reactions, i.e., $O(M)$. Improvements for the search of the candidate reaction have been introduced to improve the performance of RSSA [34]. The tree-based search reduces the computational cost to logarithmic time complexity by employing a binary tree structure where the M propensity upper bounds $\overline{a_j}$ are stored in the leaves and the inner nodes store the sums of values of child nodes. A candidate reaction is realized by traversing the tree from the root to a leaf that holds the reaction. The computational time complexity for selection of the candidate reaction by the tree-based search is equal to the height of the tree that is $O(\log M)$. However, if the propensity upper bound of a reaction at a leaf changes, the change must be propagated from the leaf to the tree root which also takes $O(\log M)$. The computational cost for selection of a candidate reaction can be further reduced to constant time, i.e., $O(1)$, by applying the table lookup search at the cost of pre-processing for building lookup tables. The principle of the table lookup search is to distribute M probabilities $\overline{a_j}/\overline{a_0}$ for $j = 1 \dots M$ into lookup tables so that the selection of a candidate reaction given its probability only takes one comparison and (at most) two memory accesses to the lookup tables. The disadvantage of the table lookup search is that if the probability of a candidate reaction changes, the whole lookup tables must be rebuilt which takes $O(M)$ time complexity.

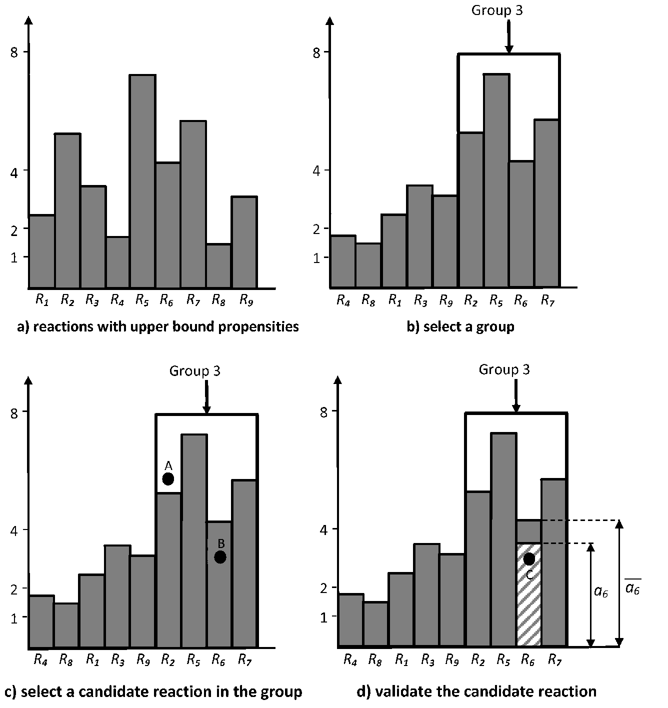


Fig. 1. Steps for the selection of the next reaction firing in RSSA-CR. a) There are 9 reactions. The bars represent the values of propensity upper bounds of reactions varying from 1 to 8. b) Reactions are grouped into $K = 3$ groups by their propensity upper bounds. Group 3 is selected. c) A candidate reaction in the selected group is randomly and uniformly selected by a rejection test. First, reaction R_2 (point A) is randomly selected but is rejected. Reaction R_6 is then selected and accepted (point B). d) Reaction R_6 is validated through a second rejection test and is accepted because the random value (point C) is less than the exact propensity a_6. Note that since the grouping of reactions in RSSA-CR is using the propensity upper bounds, the exact propensity of a reaction may not satisfy the condition of the group.

3.2 RSSA with Composition-Rejection Search

RSSA-CR employs the composition-rejection search on the propensity bounds to improve both the search and propensity update cost in simulating large reaction networks. It groups reactions into K groups labeled $G_1 \dots G_K$. The condition to put a reaction R_j into a group G_i depends on upper bound propensity $\overline{a_j}$. Specifically, G_i contains reaction R_j if the condition $2^{q_i - 1} \le \overline{a_j} \le 2^{q_i}$ holds. In other words, reaction R_j will be put into the group G_i with index $q_i = [\log(\overline{a_j})]$ in which $[-]$ is the truncation operator. Let $p_i = \sum_{R_j \in G_i} \overline{a_j}$ be the sum of the propensity upper bounds of reactions in group G_i and let $p_0 = \sum_{i=1}^{K} p_i = \sum_{j=1}^{M} \overline{a_j}$ be their total sum. The selection of the next reaction firing by RSSA-CR is a two-step search composed of selecting the group and then locating the next reaction firing within that group. Figure 1 depicts the steps for the selection of the next reaction firing in RSSA-CR.

First, RSSA-CR selects a candidate group G_l with probability p_l/p_0. The selection of the candidate group G_l is done by simply linearly summing p_i until a minimum index l such that $\sum_{i=1}^{l} p_i > r_1 \cdot p_0$ is found where $r_1 \sim \texttt{U}(0,1)$. A binary tree-based search [25] can be applied to reduce the time complexity if the number of groups K is large.

Knowing the group G_l, a reaction R_μ is randomly and uniformly selected and goes through a validation test to be accepted for firing. A random number $r_2 \sim \texttt{U}(0,1)$ is drawn and used to compute a random reaction index $\mu = [r_2 \cdot |G_l|]$ in which

$|G_l|$ returns the cardinality of group G_l and $[-]$ denotes the truncation operator. The validation of the candidate R_μ requires two consecutive rejection tests as follows. The first rejection test will accept reaction R_μ as a candidate with acceptance probability $\overline{a_\mu}/2^{q_l}$. This test does not need to generate a new random number by noting that $r_3 = r_2 \cdot |G_l| - \mu$ is randomly distributed in $(0, 1)$. RSSA-CR checks whether $r_3 \leq \overline{a_\mu}/2^{q_l}$. If the condition is true, R_μ will go through the second rejection test to decide whether it is accepted to fire. In case the condition $r_3 \leq \overline{a_\mu}/2^{q_l}$ is false, a new R_μ in G_l is selected until it is accepted. For the second rejection test, RSSA-CR generates a random number r_4 from $\mathtt{U}(0, 1)$ and checks whether $r_4 \leq a_\mu/\overline{a_\mu}$ which requires to compute exact propensity a_μ. If this check returns true, reaction R_μ is accepted to fire and its firing time is generated. Note that the propensity lower bound $\underline{a_\mu}$ can be used in this second rejection test to postpone computing of a_μ as much as possible. If the condition $r_4 \leq a_\mu/\overline{a_\mu}$ is false, the candidate is rejected. At this point, RSSA-CR has to reject also the candidate group G_l. In other words, RSSA-CR repeats the whole selection of a new group and then a candidate reaction in the group for validation. The search of next reaction firings in RSSA-CR is therefore sometimes requiring more computational effort; however, this additional computational cost is small because it is only proportional to the number of groups K, which is often bound by a small constant and independent of the number of reactions.

If reaction R_μ is accepted, its firing time τ is generated from an $\mathtt{Erlang}(k, p_0)$ distribution. However, the difference between RSSA and RSSA-CR is that the number of trials k in RSSA-CR counts only for the second rejection test (i.e., the number of times performs the second rejection test on a candidate reaction after it is accepted by the first rejection test).

3.3 The RSSA-CR Algorithm

The complete RSSA-CR algorithm for exact stochastic simulation of large reaction networks is outlined in Alg. 2. The output of an RSSA-CR simulation run is a trajectory showing the temporal dynamics of the biochemical reaction network starting at time $t = 0$ with an initial state $\mathbf{x}$ and ending at time T_{max}.

Line 3 defines for each species S_i, for $i = 1 \ldots N$, a fluctuation interval $[\underline{X}_i, \overline{X}_i]$ around its current population $X_i(t)$. The fluctuation interval is defined as $[\underline{X}_i, \overline{X}_i] = [(1-\delta_i)X_i, (1+\delta_i)X_i]$ where δ_i is a parameter. For typical models, the parameter δ_i chosen around 10% to 20% of current population $X_i(t)$ gives better performance (see Sec. 4). In case $X_i(t)$ is small, an absolute interval size Δ is used instead [34]. RSSA-CR then computes for each reaction R_j, $j = 1 \ldots M$, a propensity lower bound $\underline{a_j}$ and a propensity upper bound $\overline{a_j}$ (line 4). The needed data structure for the simulation are set up in lines 5 - 6 where reactions are grouped into K groups G_i with $i = 1 \ldots K$ based on their propensity upper bounds $\overline{a_j}$. We remark that the base 2 in the condition for grouping reactions can be chosen arbitrarily. The algorithm would work as well with any other base > 1. If it is a small number, then we have more groups which increases the cost for selecting a group. In the other case, if the base is a large number, we have less groups but the number of rejections of a reaction is high. The base 2 is often chosen because it can be done by a single $\log$ operation of a programming language [28]. The groups in our implementation are maintained dynamically by using a dequeue and are arranged in descending order of p_i to speed up the selection of the group [22]. To decide which reactions should update their propensity bounds if a species exits its fluctuation interval, RSSA-CR employs the Species-Reaction (SR) dependency graph $\mathcal{G}$ which is a directed bipartite graph showing the dependency of reactions on species [16]. The SR dependency graph $\mathcal{G}$ contains a directed edge from a species S_i to a reaction R_j if a change in the population of species S_i requires reaction R_j to recompute its propensity. The SR dependency graph $\mathcal{G}$ is built once at the beginning of the simulation at line 2.

Algorithm 2 RSSA-CR

```
 1: initialize time $t = 0$ and state vector $X = \mathbf{x}$
 2: build the species-reaction (SR) dependency graph $\mathcal{G}$
 3: define a bound $[\underline{X_i}, \overline{X_i}]$ for each $X_i$ in $X$ with $i = 1 \ldots N$
 4: compute an upper bound $\overline{a_j}$ and a lower bound $\underline{a_j}$ for $R_j$,
    $j = 1 \ldots M$
 5: group $M$ reactions into $K$ groups $G_1, \ldots, G_K$ so that group
    $G_i$ contains $R_j$ with $2^{q_i - 1} \leq \overline{a_j} \leq 2^{q_i}$ for $j = 1 \ldots M$
 6: compute for group $G_i$ the $p_i = \sum_{R_j \in G_i} \overline{a_j}$ with $i = 1 \ldots K$
    and sum $p_0 = \sum_{i=1}^{K} p_i = \sum_{j=1}^{M} \overline{a_j}$
 7: while ($t < T_{max}$) do
 8:   repeat
 9:     set $accepted$ = false
10:     set $u = 1$
11:     repeat
12:       select minimum group index $l$ s.t. $\sum_{i=1}^{l} p_l > r_1 \cdot p_0$
          with $r_1 \sim U(0, 1)$
13:       repeat
14:         compute index $\mu = [r_2 \cdot |G_l|]$ with $r_2 \sim U(0, 1)$
15:         set $r_3 = r_2 \cdot |G_l| - \mu$
16:       until ($r_3 \leq \overline{a_\mu}/2^{q_l}$)
17:       generate $r_4 \sim U(0, 1)$
18:       if ($r_4 \leq \underline{a_\mu}/\overline{a_\mu}$) then
19:         set $accepted$ = true
20:       else
21:         evaluate $a_\mu$ with state $X$
22:         if ($r_4 \leq a_\mu/\overline{a_\mu}$) then
23:           set $accepted$ = true
24:         end if
25:       end if
26:       set $u = u \cdot r_5$ with $r_5 \sim U(0, 1)$
27:     until $accepted$
28:     compute firing time $\tau = (-1/p_0)\ln(u)$
29:     update time $t = t + \tau$ and state $X = X + v_\mu$
30:   until (exists $X_i \notin [\underline{X_i}, \overline{X_i}]$)
31:   for all ($X_i \notin [\underline{X_i}, \overline{X_i}]$) do
32:     define a new $[\underline{X_i}, \overline{X_i}]$ around $X_i$
33:     for all ($R_j \in$ ReactionsAffectedBy($S_i$)) do
34:       compute bounds $\overline{a_j}$ and $\underline{a_j}$
35:       update group $G_i$ with its $p_i$ for $i = 1 \ldots K$ and sum
          $p_0$
36:     end for
37:   end for
38: end while
```

The selection of the next reaction firing by the composition-rejection search on the propensity bounds in RSSA-CR are implemented in lines 11 - 27. The search is repeated until a reaction R_μ is accepted to fire. In line 28, RSSA-CR generates the reaction firing time τ. RSSA-CR maintains a variable u, initialized to 1 in line 10, by multiplying it with a random number r_5 (line 26) each time the second rejection test performed.

Knowing the reaction firing R_μ and its firing time τ, the time is advanced to $t = t + \tau$ and the state is updated to a new state $X = X + v_\mu$. Line 30 checks whether the species population is confined in its fluctuation interval. If this is the case, a new search for the next reaction firing is performed without the need for updating the propensity bounds as well as the groups. In the uncommon case there exists a species S_i whose population $X_i \notin [\underline{X_i}, \overline{X_i}]$ and RSSA-CR has to define a new fluctuation interval around the current population of this species and update propensity bounds of reactions. Let ReactionsAffectedBy(S_i) be set of reactions which should update their propensity bounds if species S_i moves out of its fluctuation interval extracted from the SR dependency graph $\mathcal{G}$. RSSA-CR will recompute the propensity bounds for each reaction R_j in ReactionsAffectedBy(S_i). The corresponding group G_i holding R_j as well has to update its p_i and the total sum p_0. In case the upper bound propensity of reaction R_j does not satisfy the constraint of the group G_i, it has to be moved to another group. These steps are implemented in lines 31 - 37.

3.3.1 Correctness of RSSA-CR

RSSA-CR is an exact algorithm, in that the distribution of the generated trajectories is precisely the one given by the pdf $p(\tau, \mu)$ in Eq. 3. Hence, RSSA-CR is equivalent to other exact stochastic simulation algorithms such as DM or RSSA. We now provide a sketch of the correctness argument.

We start by analyzing the group selection step in Alg. 2, which is performed in line 12. There, we use linear search to select a candidate group G_l with probability

$$\mathbb{P}(\text{candidate group } G_l) = \frac{p_l}{p_0} \tag{6}$$

After that, we select a candidate reaction R_μ within group G_l. This is done using a rejection mechanism in the loop at lines 13-16. Inside such loop, reaction index μ is generated according to a discrete uniform distribution, while r_3 follows a continuous uniform distribution, independent from μ. Outside the rejection loop, the distribution of μ becomes conditioned by the $r_3 \le \overline{a_\mu}/2^{q_l}$ test. Since 2^{q_l} is an upper bound for each $\overline{a_\mu}$, the probability mass of each possible μ outcome is proportional to $\overline{a_\mu}$. Hence,

$$\begin{aligned}\mathbb{P}(\text{candidate reaction } R_\mu \mid \text{candidate group } G_l) &= \frac{\overline{a_\mu}}{\sum_{R_j \in G_l} \overline{a_j}} \\ &= \frac{\overline{a_\mu}}{p_l}\end{aligned} \tag{7}$$

Combining Eq. 6 and Eq. 7 and applying the probability chain rule, we get the probability of the candidate reaction R_μ as

$$\mathbb{P}(\text{candidate reaction } R_\mu) = \frac{\overline{a_\mu}}{p_0} \tag{8}$$

which is the same probability used in RSSA for candidate reactions. From this point, the correctness argument of RSSA can be adapted. The computed firing time τ can be shown to follow the `Erlang`(k, p_0) distribution where k is the number of rejections in the outermost loop (11-27) before a candidate reaction R_μ is accepted. Indeed, a convolution method is used in line 28 to sample τ from such distribution. This ensures that the final trajectory distribution is the wanted one [16].

3.3.2 Complexity of RSSA-CR

We now analyze the time complexity for each simulation iteration of RSSA-CR in Alg. 2 assuming basic mathematical operations (such as $+$, $-$, $\times$, $/$, $[-]$, $\log$) to be taken in constant time. The computational time of RSSA-CR is composed of two parts that are: 1) the cost for selection of a reaction firing in lines 11 - 27 and 2) the cost for updating reaction propensity bounds of reactions in lines 31 - 37. Note that the update is performed infrequently and only when there exists a species exiting its population interval.

For selection of a reaction firing, RSSA-CR selects group G_l by a linear search (line 12) which takes $O(K)$ time complexity where K is the number of groups. It then selects a candidate reaction R_μ by the first rejection (lines 13 - 16) in which the acceptance probability is $\overline{a_\mu}/2^{q_l} \ge 1/2$ because of $\overline{a_\mu} \ge 2^{q_l - 1}$. The last step validates the candidate reaction by the second rejection (lines 17 - 25) whose acceptance probability is $a_\mu/\overline{a_\mu} \ge \underline{a_\mu}/\overline{a_\mu}$. The acceptance probability of the next reaction firing R_μ is thus bounded by $\underline{a_\mu}/(2\overline{a_\mu})$. In other words, the average number of times that the validation test is performed to accept the reaction is $\alpha = (2\overline{a_\mu})/\underline{a_\mu}$. The number of tests α is depending only on the ratio of the propensity upper bound and lower bound of the reaction which can be tuned through the fluctuation interval $[\underline{X}, \overline{X}]$. Specifically, let R_μ be a unimolecular reaction of reactant S_i. The mass-action propensity of the reaction has form $a_\mu = c_\mu X_i$ and the fluctuation interval of the species is defined $[\underline{X_i}, \overline{X_i}] = [(1 - \delta_i)X_i, (1 + \delta_i)X_i]$. The ratio of the propensity upper bound over its lower bound is equal to $\overline{a_\mu}/\underline{a_\mu} = (1 + \delta_i)/(1 - \delta_i)$. Therefore, with δ_i is chosen around 10% to 20%, the average number of tests is bounded between $2.44 \le \alpha \le 3$. For a bimolecular reaction R_μ, a similar derivation gives $2.98 \le \alpha \le 4.5$. To conclude, the total computational cost for the selection of a reaction firing is $O(K)$.

For the propensity update cost, let D be the average number of reactions affected by a species in the set ReactionsAffectedBy(S_i). The cost for updating propensity bounds and groups affected by a species in lines 33 - 36 is $O(D)$. Because the number of species involved and caused by the reaction firing to move out of their fluctuation intervals is a small number, the total update cost for reactions which affected by those species is $O(D)$.

Summing up, the cost of simulating one reaction in RSSA-CR is $O(K + D)$. For models where the number of groups K and the number of reactions affected by a species D are bounded by a small constant, the computational time complexity of RSSA-CR is $O(1)$.

4 Benchmark

We report in this section the performance study of our new RSSA-CR algorithm in simulating large models. The benchmark consists of three biological models: the B cell receptor signaling, the Linear chain and the Colloidal aggregation network. The B cell receptor signaling model is a real biological model having a large number of reactions which we use to demonstrate the applicability of RSSA-CR on a large-scale reaction network. The Linear chain and the Colloidal aggregation network are artificial models which are used to demonstrate the scalability of our algorithm in different contexts. For the Linear chain model, the number of reactions increases linearly with the number of species and the number reactions needs updating its propensity after a reaction firing is fixed by a constant. In contrast, the number of reactions in the Colloidal aggregation model increases as the square of the number of the species and the number of reactions needs updating its propensity after a reaction firing is increasing linearly with the number of species. All the simulation algorithms in this section were implemented in Java and run on an Intel i5-540M processor. The implementation of the algorithms as well as the benchmark models are freely available at http://www.cosbi.eu/research/prototypes/rssa.

4.1 B cell receptor signaling model

The B cell receptor (BCR) is an antigen (Ag) receptor located on the B cell's outer surface. It is composed of a membrane-bound immunoglobulin molecule (Ig) and a transmembrane protein CD79, which is composed of two disulfide-linked chains called CD79A (Ig-α) and CD79B (Ig-β). The binding of Ags to the membrane Ig subunit stimulates the receptor aggregation and transmits the signals to the cell interior through the Ig-α/β subunits. BCR aggregation activates Lyn and Fyn of the Src family protein tyrosine kinases (SFKs) as well as other tyrosine kinases and initiates the BCR signaling pathway. The BCR signaling in turn activates multiple signaling cascades which results in many possible effects to the fates of B cells including proliferation, differentiation and apoptosis [40], [41], [42], [43]. The BCR receptor signaling pathway is thus therapeutic target in various neoplasms in cancer [44].

We consider the BCR signaling model developed in Barua *et al.* [45] to study the effects of Lyn and Fyn redundancy to the pathway. The model includes two feedback loops. The first loop is a positive feedback loop that emanates upon the SFK-mediated phosphorylation of BCR and receptor-bound Lyn and Fyn. The positive feedback loop increases the kinase activities of Lyn and Fyn. The second one is a negative feedback loop arising from SFK-mediated phosphorylation of the transmembrane adapter protein PAG1 (phosphoprotein associated with glycosphingolipid-enriched microdomains) which in turn decreases the kinase activities of Lyn and Fyn. The BCR signaling model [45] consists of 1122 species and 24388 reactions. The average number of reactions updating their propensities after a reaction firing is about 546.

The performance of RSSA-CR in simulating the BCR signaling model is compared with SSA with composition-rejection search (SSA-CR), Partial-propensity SSA with composition-rejection search (PSSA-CR). PSSA-CR is an efficient variant of the stochastic simulation algorithm with composition-rejection search [31]. PSSA-CR uses two composition-rejection search to select the next reaction firing. We also compares RSSA-CR with other RSSA variants that are: RSSA with tree-based search (RSSA-Binary) and RSSA with table lookup search (RSSA-Lookup) [34]. The fluctuation interval of the species used by RSSA is $\pm 10\%$ of current state. If the population of a species is less than 25, the absolute interval size $\Delta = 5$ is used to define the fluctuation interval of this species.

Figure 2 shows performances of simulation algorithms on the BCR signaling model. For each algorithm, the performance of a simulation run is recorded after 10^7 steps. The state are written to file after each 10^5 steps, thus having in total 100 time points. The performance result is averaged by 100 independent simulation runs. Three measurements are considered for each algorithm: 1) the search time which is the CPU time spent for the selection of reaction firings, 2) the propensity update time which is CPU time spent for the update of propensities of reactions after each

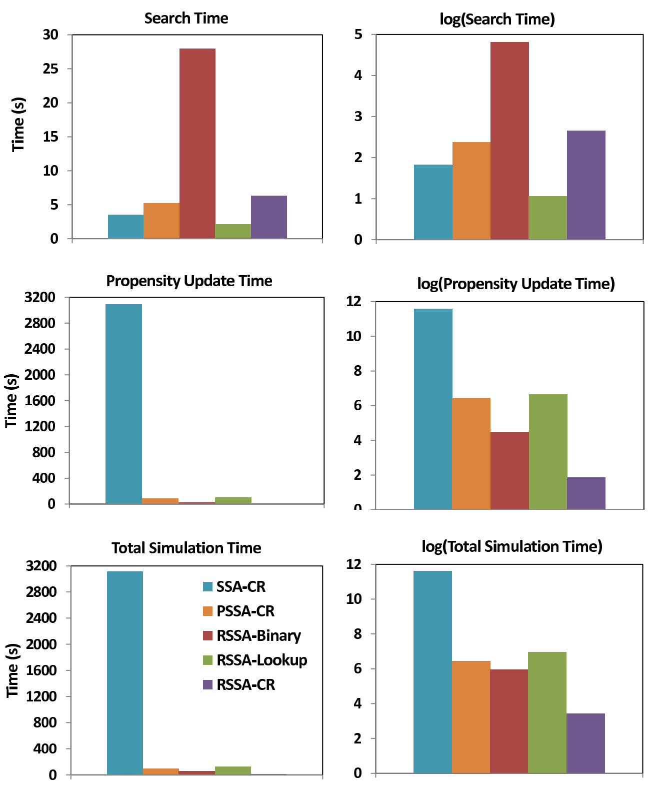


Fig. 2. Performance of SSA-CR, PSSA-CR, RSSA-Binary, RSSA-Lookup and RSSA-CR on simulating the B cell receptor signaling model. The figures on the left show the times spent for the search of next reaction firings (top plot), the propensity update (middle plot) and the total simulation time (bottom plot) which is the sum of search time, propensity update time and all other tasks (e.g., recording state and writing result to file). The figures on the right show the corresponding values in logarithmic scale.

reaction firing and 3) the total simulation time which is the sum of the search time, the propensity update time and the time spent for all other tasks (e.g., recording state and writing into external files).

The overall conclusion from bottom plot of Fig. 2 is that RSSA-CR has the best performance in comparison with all other algorithms. For the composition-rejection approach, RSSA-CR algorithm is especially better in comparison with SSA-CR and PSSA-CR in simulating the B cell receptor signaling model. Specifically, RSSA-CR is 8.5 times faster than PSSA-CR and respectively, 200 times faster than SSA-CR. The significant speed up of RSSA-CR in comparison with SSA-CR and PSSA-CR comes from the significant reduction in the cost of propensity updates while still have comparable search time. The detailed comparison of algorithms follows.

The top plot in Fig. 2 shows the search time of RSSA-CR which is slightly slower than SSA-CR and PSSA-CR. This is because of more effort spent by RSSA-CR for selecting next reaction firings. The acceptance probability of a reaction in RSSA-CR is around 65%, while this value in RSSA-Binary and RSSA-Lookup is 80%. However, the search time of RSSA-CR is still better in comparison with RSSA-Binary. The speed up gain in the search of RSSA-CR in comparison with RSSA-Binary is about 3 times. By employing the lookup table search, the search of RSSA-Lookup achieves the best performance.

The more effort spent for the search of RSSA-CR is compensated by a huge improvement in propensity update time. The number of propensity updates of PSSA-CR and SSA-CR is 10^7 since they have to update propensities of reactions after each reaction firing. The update cost of PSSA-CR is smaller than SSA-CR because the update of propensities with shared reactants in PSSA-CR is performed collectively in single operation. The number of propensity updates for RSSA-CR is only 900 times, which is significantly reduced in comparison with 10^7 times by PSSA-CR and SSA-CR. Thus, the update cost of RSSA-CR is nearly 20 times faster than PSSA-CR, and around 700 times faster than SSA-CR. For all RSSA variants, the propensity update time of RSSA-CR is the best, while this cost for RSSA-Lookup is the worst. Note that SR dependency graph cannot be applied for RSSA-Lookup. The lookup tables used in RSSA-Lookup has to be rebuilt anytime the propensity bound of a reaction is changed, although the update cost of RSSA-Lookup is still better than SSA-CR. The propensity update time of RSSA-Binary is slower than RSSA-CR because each time propensity bound of a reaction changes, it has to be propagated along the tree through the leaf holding the reaction to tree root to reflect the change. While RSSA-CR only has to update the groups and move reactions between groups when necessary. The propensity update cost of RSSA-CR is nearly 10 and 30 times faster than RSSA-Binary and RSSA-Lookup, respectively.

4.2 Linear chain system

The linear chain model is an artificial model that is used to measure the scalability of RSSA-CR when the propensity updates contributed a small percentage in the total simulation time. The search for next reaction firings contributes most to the simulation.

The model consists of N species S_i with $i = 1 \dots N$ in which a species is transformed to the species S_j such that

$$S_i \xrightarrow{c_i} S_j, \text{ for } i = 1 \dots N \text{ and } j = (i+1) \mod N \quad (9)$$

where c_i is the rate constant of the transformation. The number of reactions M in the linear chain is equal to the number of species, i.e., $M = N$. The number of affected reactions which needs to update their propensities in the Linear chain model is fixed by 2. In this experiment, the kinetics rate of all reactions is set to $c_i = 1$ for $i = 1 \dots N$. The initial population of each species S_i for $i = 1 \dots N$ is randomly taken from 0 and 10000. The fluctuation interval of species used by all RSSA variants is $\pm 10\%$ of current state.

The Linear chain model is simulated by increasing the values of N (100, 500, 1000, 5000, 10000 and 50000) to observe the scaling characteristic of the algorithms. The simulation of all algorithms is run 100 times for each value of N to average the results which are collected after 10^7 simulation steps. The performances of SSA-CR, PSSA-CR, RSSA-Binary, RSSA-Lookup and RSSA-CR are depicted in Fig. 3.

We have conclusions from Fig. 3. First, RSSA-Lookup has the best search for all values of N; however, the high update cost negates its total simulation time. The result is the performance of RSSA-lookup is the worst when N is large. Second, the search cost of RSSA-Binary for large N is the worst in comparison with all other algorithms because it increases logarithmic with N. The search cost of RSSA-CR is better than RSSA-Binary because it is independent of N. For example, the search cost of RSSA-CR for $N = 50000$ is 5 times faster than RSSA-Binary. Third, the search of both SSA-CR is slightly faster than both PSSA-CR and RSSA-CR. This is because SSA-CR uses only one rejection-test for selecting the next reaction, while both PSSA-CR and RSSA-CR employs two rejection tests. Finally, the update of RSSA-CR has the best performance in comparison with SSA-CR and PSSA-CR. The number of propensity updates of RSSA-CR is reduced to around 2.4×10^4 and the acceptance probability of a candidate reaction is kept around 65%. The number of propensity updates for both SSA-CR and PSSA-CR is 10^7 because they have to update propensities of reactions after each reaction firing. The result is RSSA-CR is around 30% and 10% faster than SSA-CR and PSSA-CR, respectively.

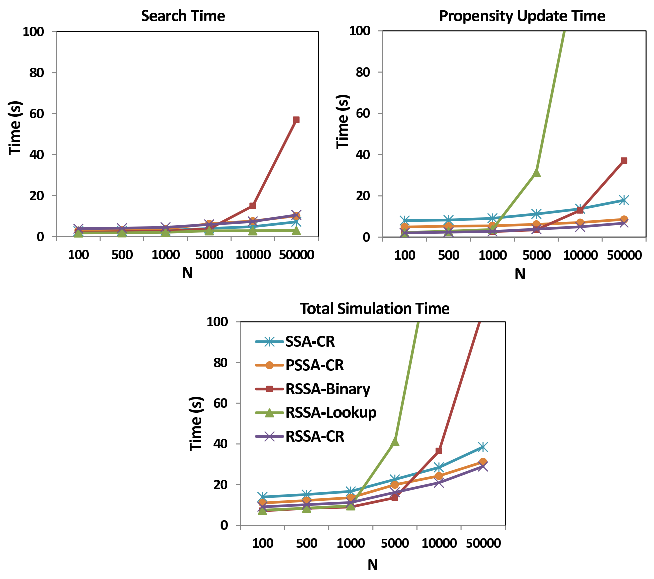


Fig. 3. Performance of SSA-CR, PSSA-CR, RSSA-Binary, RSSA-Lookup and RSSA-CR on the Linear chain model by increasing values of N (100, 500, 1000, 5000, 10000, 50000). The top-left figure shows the time spent for the search of next reaction firings. The top-right figure shows the propensity update time. The bottom figure shows the total simulation time which is the sum of search time, propensity update time and all other tasks (e.g., recording state and writing result to file).

4.3 Colloidal aggregation network

In the final example, the colloidal aggregation is used to demonstrate the significant improvements of RSSA-CR in comparison with PSSA-CR and SSA-CR where both the search for reaction firings and the propensity updates contribute are simulation bottlenecks of these simulation approaches.

The colloidal aggregation is a process that forms big clusters from colloidal particles, e.g., proteins, nanobeads. It is a ubiquitous phenomenon occurring in nature and widely explored in manufacturing [46], [47]. The aggregation of particle is an irreversible process. In the forward process, the particle aggregates on and grow in size. The reverse process disrupts particle aggregates into individual particles. The colloidal aggregation is thus a metastable system where a variety of different aggregation states are achievable.

The model contains N colloidal particles which can interact with other species to form big clusters. The reaction network that models the colloidal aggregation process is defined by:

$$\begin{aligned} S_n + S_m &\xrightarrow{c_{n,m}} S_{n+m}, n = 1 \dots [N/2], m = n \dots N - m \\ S_p &\xrightarrow{c_{p,q}} S_q + S_{pq}, p = 1 \dots N, q = 1 \dots [p/2] \quad (10) \end{aligned}$$

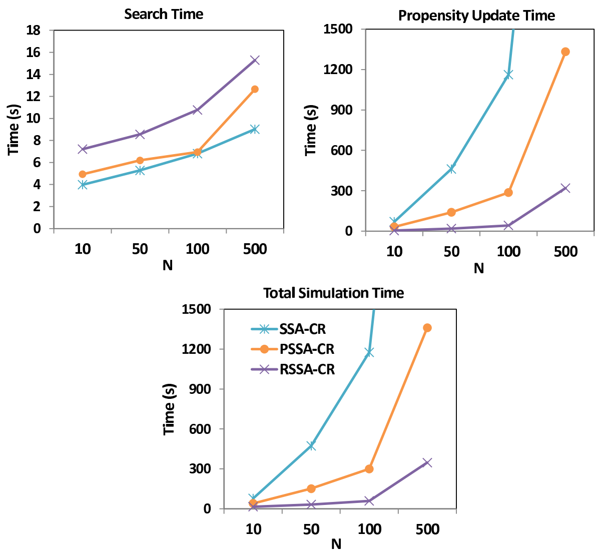


Fig. 4. Performance of SSA-CR, PSSA-CR and RSSA-CR on the Colloidal aggregation model by increasing values of N (10, 50, 100, 500). The top-left figure shows the time spent for the search of next reaction firings. The top-right figure shows the propensity update time. The bottom figure shows the total simulation time which is the sum of search time, propensity update time and all other tasks (e.g., recording state and writing result to file).

where $c_{n,m}$ and $c_{p,q}$ are rate constant of forward and reverse processes, respectively. The number of reactions M in the colloidal aggregation network is increasing in quadratically in the number of particles. Specifically, the number of reactions is $M = [N^2/2]$. The average number of affected reactions which needs to update their propensities is linear with the number of particles N. The initial population of all species in this experiment is set to $\#S_i = 1000$ for $i = 1 \dots N$. The kinetics rate of all reaction is set to $c_{n,m} = c_{p,q} = 1$. The fluctuation interval of species used by RSSA-CR is $\pm 10\%$ of current state.

We simulate the Colloidal aggregation model with different values of N (10, 50, 100 and 500) to observe the scaling characteristic of the simulation algorithms: SSA-CR, PSSA-CR and RSSA-CR. The algorithms are run 100 times for each value of N to average the results which are collected after 10^7 simulation steps. Figure 4 shows the performances of SSA-CR, PSSA-CR and RSSA-CR.

The search of SSA-CR achieves the best performance for all values of N; however; its update cost is many times slower than PSSA-CR and RSSA-CR. For example, in case $N = 100$, the propensity update time of SSA-CR is roughly 20 and 4 times slower than RSSA-CR and PSSA-CR, respectively. The result is the performance of SSA-CR is the worst in simulating the Colloidal aggregation model, especially for large N. The search of RSSA-CR is slightly slower than PSSA-CR. For example, in case $N = 500$, the search time of PSSA-CR is around 20% faster than RSSA-CR. Although the update cost of PSSA-CR is reduced by updating propensities of reactions having common reactant collectively in one operation. PSSA-CR still requires to perform propensity updates after each reaction firing. Note that in this model, the number of reactions needs updating their propensity each reaction firing is linearly increasing with N. This number in RSSA-CR is reduced to around 9×10^5 (corresponding with 9% in comparison with 10^7 times of PSSA-CR). The update cost RSSA-CR is 4 times faster than PSSA-CR. Thus, the total simulation time of RSSA-CR is 3.8 times faster than PSSA-CR.

5 Conclusions

We presented a new algorithm, called RSSA-CR, which combines composition-rejection search and propensity bounds for improving the search for next reaction firings and reducing the number of propensity updates in simulating large reaction networks. The time complexity of the search of RSSA-CR is independent of the number of reactions, but is only depending on the number of groups. By using the propensity bounds for selecting reaction firings, RSSA-CR does not need to recompute these values in most of its simulation steps. The propensity updates in RSSA-CR are performed infrequently and only when necessary. In addition, the recomputing of propensity bounds are performed locally for the affected reactions only. These features of RSSA-CR makes it suitable for the simulation of large reaction networks. For models where the reactions are split among groups so that the search for the candidate group has an average constant bound and the dependencies between species and reactions are bounded, RSSA-CR has constant time complexity. The benchmark confirmed that RSSA-CR is efficient for simulating large and complex models.

References

[1] Adam P. Arkin, John Ross, and Harley H. McAdams. Stochastic kinetic analysis of developmental pathway bifurcation in phage λ-infected escherichia coli cells. *Genetics*, 149:16331648, 1998.
[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. In *PNAS*, 1997.
[4] Nina Fedoroff and Walter Fontana. Small numbers of big molecules. *Science*, 297(5584):1129–1131, 2002.
[5] Jonathan M. Raser and Erin K. O'Shea. Noise in gene expression: Origins, consequences, and control. *Science*, 309:2010–2013, 2005.
[6] Mads Kærn, Timothy C. Elston, William J. Blake, and James J. Collins. Stochasticity in gene expression: from theories to phenotypes. *Nature Reviews Genetics*, 6:451–464, 2005.
[7] Daniel Gillespie. A rigorous derivation of the chemical master equation. *Physica A*, 188(1-3):404–425, 1992.
[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] 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.
[11] Paola Lecca. A time-dependent extension of Gillespie algorithm for biochemical stochastic π-calculus. In *Proc. of ACM-SAC*, pages 137–144, 2006.
[12] 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.
[13] Vo Hong Thanh and Corrado Priami. Simulation of biochemical reactions with time-dependent rates by the rejection-based algorithm. *J. Chem. Phys.*, 143(5):054104, 2015.
[14] Dmitri Bratsun, Dmitri Volfson, Lev S. Tsimring, and Jeff Hasty. Delay-induced stochastic oscillations in gene regulation. *PNAS*, 102:14593–14598, 2005.
[15] Xiaodong Cai. Exact stochastic simulation of coupled chemical reactions with delays. *J. Phys. Chem.*, 126(12):124108, 2007.
[16] Vo Hong 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.
[17] Audrius B. Stundzia and Charles J. Lumsden. Stochastic simulation of coupled reactiondiffusion processes. *J. Comp. Phys.*, 127(168):196207, 1996.
[18] David Bernstein. Simulating mesoscopic reaction-diffusion systems using the gillespie algorithm. *Phys. Rev. E*, 71(4):041103, 2005.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

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

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

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

[26] Vo Hong 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.

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

[28] 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.

[29] 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.

[30] 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.

[31] Rajesh Ramaswamy and Ivo F. Sbalzarini. A partial-propensity variant of the composition-rejection stochastic simulation algorithm for chemical reaction networks. *J. Chem. Phys.*, 132(4):044102, 2010.

[32] F. Schl*ö*gl. Chemical reaction models for non-equilibrium phase transitions. *Zeitschrift fr Physik*, 253(2):147–161, 1972.

[33] E. Crampina, S. Schnella, and P. McSharry. Mathematical and computational techniques to deduce complex biochemical reaction mechanisms. *Progress in Biophysics and Molecular Biology*, 86(1):77–112, 2004.

[34] Vo Hong 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.

[35] Vo Hong 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.

[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] M. Mateescu, V. Wolf, F. Didier, and T.A. Henzinger. Fast adaptive uniformisation of the chemical master equation. *IET Syst. Biol.*, 4(6):441452, 2010.

[38] David Huffman. A method for the construction of minimum-redundancy codes. In *Proc. of the IRE*, volume 40, pages 1098–1101, 1952.

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

[40] Joseph M. D. Porto, Stephen B. Gauld, Kevin T. Merrell, David Mills, Aimee E. Pugh-Bernard, and John Cambier. B cell antigen receptor signaling 101. *Molecular Immunology*, 41(6-7):599613, 2004.

[41] Christopher C. Goodnow, Carola G. Vinuesa, Katrina L. Randall, Fabienne Mackay, and Robert Brink. Control systems and decision making for antibody production. *Nat. Immunol.*, 11(8):6818, 2010.

[42] Naomi E. Harwood and Facundo D. Batista. Early events in b cell activation. *Annu. Rev. Immunol.*, 28:185–210, 2009.

[43] Tomohiro Kurosaki, Hisaaki Shinohara, and Yoshihiro Baba. B cell signaling and fate decision. *Annu. Rev. Immunol.*, 28:21–55, 2009.

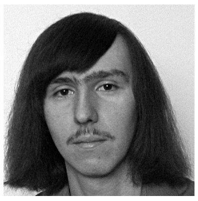
[44] Vaclav Seda and Marek Mraz. B cell receptor signalling and its crosstalk with other pathways in normal and malignant cells. *European Journal of Haematology*, 94(3):193205, 2014.

[45] Dipak Barua, William S. Hlavacek, and Tomasz Lipniacki. A computational model for early events in b cell antigen receptor signaling: Analysis of the roles of lyn and fyn. *J. Immunol.*, 189:646–658, 2012.

[46] Paul Meakin. Models for colloidal aggregation. *Annual Review of Physical Chemistry*, 39:237–267, 1988.

[47] M. Y. Lin, H. M. Lindsay, D. A. Weitz, R. C. Ball, R. Klein, and P. Meakin. Universality in colloid aggregation. *Nature*, 339:360–362, 1989.

Vo Hong Thanh is a research associate at COSBI. He received a PhD in Computer Science from the University of Trento, Italy in 2013. His research focuses on stochastic simulation of biological systems, designing and developing algorithms to efficiently simulate the biochemical reaction networks.



Roberto Zunino is an Assistant Professor in Computer Science at the Department of Mathematics of the University of Trento, Italy. His research topics include stochastic simulation, programming languages, and theoretical computer science.

Corrado Priami is a Professor of Computer Science at the University of Trento and is the President and CEO of the Microsoft Research - University of Trento Centre for Computational and Systems Biology (COSBI). He was member of the expert group on the EU 7th FP of the CRUI and has participated in many EU projects (also as coordinator) for the advancement of emerging areas of research. He is a member of the Scientific committee of Fondazione Veronesi. His research covers computational methods for the modeling, analysis, and simulation of biological systems and programming languages.