Block Search Stochastic Simulation Algorithm (*BlSSSA*): A Fast Stochastic Simulation Algorithm for Modeling Large Biochemical Networks

Debraj Ghosh, Rajat K. De, Senior Member IEEE

Abstract—Stochastic simulation algorithms are extensively used for exploring stochastic behavior of biochemical pathways/networks. Computational cost of these algorithms is high in simulating real biochemical systems due to their large size, complex structure and stiffness. In order to reduce the computational cost, several algorithms have been developed. It is observed that these algorithms are basically fast in simulating weakly coupled networks. In case of strongly coupled networks, they become slow as their computational cost become high in maintaining complex data structures. Here, we develop Block Search Stochastic Simulation Algorithm (*BlSSSA*). *BlSSSA* is not only fast in simulating weakly coupled networks but also fast in simulating strongly coupled and stiff networks. We compare its performance with other existing algorithms using two hypothetical networks, viz., linear chain and colloidal aggregation network, and three real biochemical networks, viz., B cell receptor signaling network, FceRI signaling network and a stiff 1,3-Butadiene Oxidation network. It has been shown that *BlSSSA* is faster than other algorithms considered in this study.

Index Terms—Gillespie algorithm, B cell receptor signaling network, FceRI signaling network, Stiff network, Colloidal aggregation network and stochastic modeling.

1 Introduction

Biochemical networks can be modeled using two approaches: deterministic approach (based on ordinary differential equations) and stochastic approach (using stochastic simulation algorithms). The stochastic approach provides more detailed information about reaction diffusion mechanism. It is needed to capture the effect of stochastic fluctuation in biochemical systems [1]. Biochemical networks/pathways involve chemical reactions of various substances/species. Molecular distribution of these chemical substances depends on certain biological phenomena. For example, in glycolysis pathway, a small number of glucose molecules is responsible for respiration, and it produces carbon dioxide through TCA cycle. On the other hand, if the number of glucose molecules increases, it results in fermentation in which the number of molecules of ethanol increases compared to its normal count. This phenomenon is termed as Crabtree effect [2], [3], [4], [5]. This Crabtree effect like phenomenon occurs in tumor cells, and is targeted in cancer therapy. In order to explore and have a deep understanding of this type of phenomenon, *in silico* modeling of biochemical networks/pathways is necessary [6]. Other examples of biological phenomena include identification of various phenotypes of genetically identical cells, which are produced due to stochastic fluctuations in species population of biochemical pathways/networks [7]. In order to develop *in silico* models for studying these biological phenomena, including stochastic fluctuations of species populations, stochastic simulation algorithms (SSAs) are extensively used.

Initially two stochastic simulation algorithms - first reaction method (FRM) and direct method (DM) have been developed [8], [9]. Their computational cost in simulating large biochemical networks is quite high. In order to resolve this issue, various SSAs, including next reaction method (NRM), optimized direct method (ODM), sorting direct method (SDM), partial-propensity direct method (PDM), sorting partial-propensity direct method (SPDM) and partial-propensity method combined with composition rejection PSSA-CR, have been developed [7], [10], [11], [12], [13]. These methods are fast in simulating weakly coupled networks, but their simulation time becomes high in executing strongly coupled networks. In a weakly coupled network, the coupling between reactions and species is low. In other words, the number of reactions is almost equal to that of species. For example, a linear cyclic chain is a weakly coupled network. On the other hand, for a strongly coupled network, the coupling is high. The number of reactions in a strongly coupled network is very high compared to that of species. For example, colloidal aggregation is a strongly coupled network. Both linear chain and colloidal aggregation networks have been discussed later in this study. Detailed description of weakly and strongly coupled networks is given in [14]. An SSA is based on two operations: searching the reaction to be executed and updating the corresponding propensity values after execution of the reaction. In simu-

- *D. Ghosh is with the Machine Intelligence Unit, Indian Statistical Institution, Kolkata, West Bengal, India. E-mail: debrajrock@gmail.com*
- *R. K. De is with the Machine Intelligence Unit, Indian Statistical Institution, Kolkata, West Bengal, India. E-mail: rajat@isical.ac.in*

lating strongly coupled networks, the update operation cost in these algorithms is high, which eventually increases their total simulation cost.

Several algorithms, including rejection based stochastic simulation algorithm (RSSA), rejection based stochastic simulation algorithm-Binary (RSSA-Binary), rejection based stochastic simulation algorithm-Lookup (RSSA-Lookup) and rejection based stochastic simulation algorithm-composition rejection (RSSA-CR), have been introduced [15], [16], [17]. The update operation in these methods is fast. All these algorithms, except RSSA, use complex data structures, and the maintenance cost of the data structure during update operation are still high in executing strongly coupled networks. Total simulation cost in RSSA is high because of its high search operation (linear search) cost. Two methods, including slow update exact stochastic simulation algorithm (SUESSA) and slow update exact sorting stochastic simulation algorithm (SUESSSA), have been developed to reduce propensity update cost [18]. SUESSA and SUESSSA use cache-based linear search (CBLS) for search operation. CBLS based data structure makes the propensity update operation very fast in case of both weakly and strongly coupled networks. On the other hand, search operation using CBLS for weakly coupled networks is slow.

In this paper, we have developed block search stochastic simulation algorithm (*BlSSSA*) for reducing both search and update operation costs, and thereby, the total simulation cost. *BlSSSA* is equipped with fast search operation as well as fast update operation. It employs multilevel search technique in order to speed up the search operation. Multilevel search has also been applied on NRM [19], in which the number of items in each level is of the order of square root of the number of reactions, and therefore, in case of strongly coupled networks, the corresponding cost becomes high. On the other hand, in *BlSSSA*, at first, blocks are searched, then groups, and finally group elements are searched for selecting reactions to be executed. The number of blocks as well as groups is of the order of square root of the number of species. In this data structure, the search cost is low in case of strongly coupled networks. It is shown in the result section that *BlSSSA* is faster than the existing algorithms considered in this study in simulating both weakly and strongly coupled networks. This paper is organized as follows. In Section 2, we discuss the proposed method *BlSSSA*. We compare the performances of *BlSSSA*, and validate its results with others in Section 3. Section 4 further discusses the methodology and results, while Section 5 concludes the article.

2 METHODS

In this section, the key definitions and a brief description of stochastic simulation algorithms (SSAs) are, first of all, provided. Next, the proposed block search stochastic simulation algorithm (*BlSSSA*) is described. Then, the correctness and computational cost of *BlSSSA* are analyzed.

2.1 Stochastic Simulation Algorithms (SSAs)

Let us consider a biochemical network with M chemical species (substances) C_i, $i = 0, \cdots, M-1$, and N chemical reactions R_j, $j = 0, \cdots, N-1$. The population of the species at time t is a random variable $\boldsymbol{X}(t) = [X_0(t), \cdots, X_{M-1}(t)]^T$, as the biochemical reactions occur due to the random interactions among the molecules [20]. A vector $\boldsymbol{v}_j = [v_{0,j}, \cdots, v_{M-1,j}]^T$ represents the stoichiometry corresponding to the reaction R_j. In case of mass action kinetics, propensity $a_j(\boldsymbol{X}(t))$ for a bimolecular reaction R_j between two species C_1 and C_2, having k_j as the reaction parameter, is defined as $a_j(\boldsymbol{X}(t)) = X_1(t) \times X_2(t) \times k_j$. The term $a_j(\boldsymbol{X})\tau$ defines the probability of occurrence of the reaction R_j within a short interval of time $[t, t+\tau)$, where τ is given by

$$\tau = \frac{1}{a(\boldsymbol{X})} \ln\left(\frac{1}{RN_1}\right) \tag{1}$$

Here, RN_1 is a random number, which is uniformly distributed in (0,1). The term $a(\boldsymbol{X})$ is the propensity sum and is defined as

$$a(\boldsymbol{X}) = \sum_{j=1}^{N} a_j(\boldsymbol{X}) \tag{2}$$

The probability that a reaction R_μ is chosen to be executed at time $t+\tau$, is

$$Pr(R_\mu) = \frac{a_\mu(\boldsymbol{X})}{a(\boldsymbol{X})} \tag{3}$$

After the occurrence of R_μ, the population of the species is updated as

$$\boldsymbol{X}(t+\tau) = \boldsymbol{X}(t) + \boldsymbol{v}_\mu \tag{4}$$

where $\boldsymbol{v}_\mu$ is the stoichiometric coefficient of reaction R_μ.

An SSA, like direct method (DM), follows Equations 1 to 4 to simulate a biochemical network and generates trajectory of its executed reactions. Here, we explain the mechanism of direct method (Algorithm 1). It calculates the propensity a_j of each reaction R_j and the total propensity a using Equation 2. Next, it generates the reaction occurrence time τ using Equation 1. Then, it searches for the reaction R_μ to be executed such that μ is the lowest index which satisfies the criteria $\sum_{j=1}^{\mu} a_j > (RN_2 \times a)$. Here, RN_2 is a random number uniformly distributed in (0,1). After the selection of R_μ, it is executed and the system state is updated using Equation 4. If it does not terminate, it recalculates the propensities in order to reflect the state change.

Algorithm 1 Direct Method (*DM*)

1: **STEP 1. (Initialize)**
2: $TIME \leftarrow 0$; $N \leftarrow$ Number of reactions; $M \leftarrow$ Number of species
3: $\mathbf{X} \leftarrow$ (Initial population of species);
4: **STEP 2. (Calculate propensity)**
5: Calculate propensity a_j for each reaction R_j and total propensity $a = \sum_{j=1}^{N} a_j$
6: **STEP 3. (Calculate reaction occurrence time)**
7: Calculate $\tau = \frac{1}{a} \ln\left(\frac{1}{RN_1}\right)$
8: $TIME \leftarrow TIME + \tau$
9: **STEP 4. (Search the reaction R_μ)**
10: Search the reaction R_μ for execution where
11: the index $\mu =$ lowest index such that $\sum_{j=1}^{\mu} a_j > (RN_2 \times a)$
12: **STEP 5. (Execute the reaction R_μ)**
13: $\boldsymbol{X}(t+\tau) = \boldsymbol{X}(t) + \boldsymbol{v}_\mu$
14: **STEP 6. (Terminate)**
15: **if** $(TIME < ENDINGTIME)$ **then**
16: Goto **STEP 2.**
17: **end if**

DM uses linear search for finding reactions to be executed, and the complexity is of the order of reactions.

After each reaction execution, it updates the propensities of all reactions and the corresponding cost is as high as $O(N)$. Due to this large computational cost of search and propensity update, DM is suitable for modeling small networks. Biochemical networks in real organisms are large and complex. Hence, primary goal of developing SSAs is to reduce search operation cost while simulating these large networks. Secondary goal is to reduce the cost of propensity update operation.

ODM has been developed to reduce the propensity update cost. It uses a dependency graph in which for each reaction R_j, it keeps a group of dependent reaction propensities such that these propensities are derived from substrates and products of R_j. Instead of updating all propensities, ODM updates a_j and its dependent reaction propensities. This technique reduces computational cost significantly.

PDM has been developed with improved search operation which maintains reaction propensities in groups. Here, each group G_i corresponds to each reactant species C_i, which stores the partial propensities of the participant reactions. In order to select reactions for execution, a linear search is performed through the groups. Once a group is selected, another linear search is performed through the elements of this group for selecting the reaction index. Thus, the search operation cost in PDM is $O(M)$ as only a few reaction elements are traversed after the selection of a group. The partial propensity based data structure has improved the search operation in case of strongly coupled networks in which the number of reactions are higher than that of species.

SSA-CR has reduced search operation cost by using composition-rejection technique in which reaction propensities are kept in groups. In SSA-CR, a group G_i contains propensity a_j of R_j if a_j falls in $2^{q_i-1} \leq a_j \leq 2^{q_i}$ where q_i is the index value of group G_i and $q_i = \lfloor (log(a_j)) \rfloor$. SSA-CR selects a reaction for execution in two steps. It selects the group G_i using a linear search and then it finds reaction index μ of R_μ to be executed using acceptance-rejection technique based on hat function 2^{q_i}. Since $a_\mu/2^{q_i} \geq 1/2$, only two rejections are performed on an average. The number of groups is a small number and hence the search operation cost is $O(1)$ [17].

Methods including DM, ODM, SDM, PDM and SSA-CR perform propensity updates after each reaction execution, and the corresponding cost is high. Rejection based methods including RSSA, RSSA-Binary, RSSA-Lookup and RSSA-CR skip propensity updates by using propensity bound (lower bound aL_μ and upper bound aU_μ of reaction R_μ to be executed) infrequent propensity update technique. They calculate these propensity bounds from the fluctuation interval $[Lx_i, Ux_i]$ of each species (C_i) population X_i. After the execution of R_μ, propensity updates are skipped for reactions whose reactant species populations fall in their fluctuation intervals. Propensity update cost has been reduced drastically using this technique. Despite skipping propensity updates, the rejection based methods are exact SSAs because they perform a correction mechanism which includes a rejection test after selecting a candidate reaction for execution. RSSA maintains a linear array for storing the reaction propensities and it selects a candidate reaction index μ using a linear search making the search operation expensive.

RSSA-Binary uses a binary tree for storing the reaction propensities, and search operation cost is of the order of the height of the tree, which is faster than RSSA. During propensity update, a reaction propensity traverses from the leaf to the root of the tree to reflect the change of the update. Due to this reason, propensity update in RSSA-Binary is computationally expensive. RSSA-Lookup distributes the reaction occurrence probabilities in a lookup table which helps in selecting a candidate reaction index with $O(1)$ complexity. During propensity update, the lookup table is rebuilt for reflecting the change in probabilities, the time complexity of which is as high as $O(N)$. RSSA-CR stores the reaction propensities in groups, and it uses composition rejection technique for selecting a candidate reaction index μ with $O(1)$ complexity. During propensity update, each propensity moves to another group if it does not fall in the condition of being in the present group. It increases the propensity update cost of RSSA-CR.

The slow update methods SUESSSA and SUESSA, recently developed by the authors, are equipped with lower propensity update cost as well as total simulation cost compared to the SSAs including RSSA-Binary, RSSA-Lookup and RSSA-CR [18]. Total simulation cost is composed of search operation cost and propensity update cost. In total simulation cost of strongly coupled networks, percentage of update cost is higher than that of search operation cost. On the other hand, in case of weakly coupled networks, percentage of search operation cost is higher than that of propensity update cost. Search operation in SUESSSA and SUESSA follows CBLS and the corresponding cost is high. Search operation cost in RSSA-CR, RSSA-Lookup and RSSA-Binary is lower than that in SUESSSA and SUESSA. In order to simulate weakly coupled networks including the linear chain network, total simulation cost in RSSA-CR, RSSA-Lookup and RSSA-Binary is lower than that in SUESSSA and SUESSA, due to their lower search operation cost.

2.2 Block Search Stochastic Simulation Algorithm (*BlSSSA*)

BlSSSA uses block based multilevel search which has improved the search operation cost in simulating large weakly and strongly coupled networks. It uses propensity bound infrequent propensity update technique for reducing propensity update cost in simulating large complex biochemical networks. *BlSSSA* stores reaction propensity upper bounds in groups. Each group corresponds to a species. The ith group contains propensity upper bounds of such reactions where species C_i is a reactant. Let us consider that C_i is a reactant of reactions R_j and R_k. Then, the ith group contains the propensity upper bounds of R_j and R_k only. Therefore, the total number of groups is M, where M is the number of species. *BlSSSA* distributes these groups into blocks. The number of blocks changes with network size. The simulation time varies with the number of blocks for a fixed network size. In [19], [21], [22], it has been proved theoretically that the optimal number of blocks is the square root of the number of species, and hence, *BlSSSA* has chosen the number of blocks as $\left\lfloor \sqrt{M} \right\rfloor$ and each block contains

$\lfloor \sqrt{M} \rfloor$ number of groups. Symbols used in *BlSSSA* are given in Table 1.

BlSSSA performs a multilevel search in which at the first level the algorithm finds block index s^* such that

$$s^* = \min \{ s \in \mathbb{W} | \sum_{j=0}^{s} BLOCKPROP_j > (RN_1 \times TP) \} \quad (5)$$

Here, RN_1 is a random number uniformly distributed in (0,1). Then, *BlSSSA* selects the group Ω_{IVAL^*} in $BLOCKPROP_{s^*}$ such that

$$IVAL^* = \min \{ IVAL \in \mathbb{W} | (\sum_{i=0}^{IVAL} \Omega_i + \sum_{j=0}^{s^*-1} BLOCKPROP_j) > (RN_1 \times TP) \} \quad (6)$$

Finally in Ω_{IVAL^*}, it selects the index $\mu = \Upsilon_{IVAL^*,JVAL^*}$ of the candidate reaction R_μ by finding

$$JVAL^* = \min \{ JVAL \in \mathbb{W} | (\sum_{k=0}^{JVAL} \Gamma U_{IVAL^*,JVAL} + \sum_{i=0}^{IVAL^*-1} \Omega_i + \sum_{j=0}^{s^*-1} BLOCKPROP_j) > (RN_1 \times TP) \} \quad (7)$$

After the selection of R_μ, *BlSSSA* performs an acceptance test to ensure its exactness. R_μ is accepted for execution if $RN_2 \leq \Gamma L_{\mathrm{IVAL,JVAL}} / \Gamma U_{\mathrm{IVAL,JVAL}}$. Otherwise, full propensity $\Gamma_{\mathrm{IVAL,JVAL}}$ of R_μ is calculated and the condition $RN_2 \leq \Gamma_{\mathrm{IVAL,JVAL}} / \Gamma U_{\mathrm{IVAL,JVAL}}$ is checked. Here, RN_2 is a random number uniformly distributed in (0,1). If $RN_2 \leq \Gamma_{\mathrm{IVAL,JVAL}} / \Gamma U_{\mathrm{IVAL,JVAL}}$ is true, R_μ is accepted; otherwise, a new block is searched to find a new candidate reaction R_μ. The reaction occurrence time τ is calculated using Erlang distribution and R_μ is executed [17].

BlSSSA performs propensity update after the execution of R_μ. Propensity update of each group Ω_l is skipped if C_l is either a reactant or product of R_μ, and its population X_l falls in the fluctuation interval $[Lx_l, Ux_l]$. Otherwise, a new fluctuation interval of C_l is calculated; Ω_l is updated; corresponding block propensity upper bound $BLOCKPROP_{\mathrm{SPECIES\_BLOCK}_l}$ is updated, and so is TP. Here, $SPECIES\_BLOCK_l$ provides the index of the block containing group propensity upper bound sum Ω_l. For the best performance, **Lx** and **Ux** are kept at (**X** $-$ (10 to 20)% of **X**) and (**X** $+$ (10 to 20)% of **X**) respectively [17].

BlSSSA simulates a biochemical network from $TIME = 0$ to $ENDINGTIME$, and generates trajectory of its executed reactions. In order to do this, it performs the steps given in Algorithm 2. In STEPs 1 and 2, it declares the necessary data structures for storing species populations, reaction propensities and their corresponding bounds. It calculates propensity bounds for each reaction and group propensity upper bounds. *BlSSSA* selects the number of blocks; assigns group propensity upper bounds to each block; and calculates block propensity upper bounds and total propensity upper bound. It selects the candidate reaction R_μ in STEP 3. The algorithm performs multilevel search and selects a block $BLOCKPROP_i$ (lines 30 to 38). In $BLOCKPROP_i$, it selects a group Ω_{IVAL} (lines 40 to 47). After that in Ω_{IVAL}, it selects the candidate reaction R_μ such that $\mu = \Upsilon_{\mathrm{IVAL,JVAL}}$ (lines 48 to 55). *BlSSSA* performs the acceptance testing for R_μ to be executed in STEP 4. R_μ is either accepted based on the steps given in lines 57 to 65 or it is rejected, and the algorithm jumps to STEP

Algorithm 2 Block Search Stochastic Simulation Algorithm (*BlSSSA*)

1: **STEP 1. (Initialize)**
2: $TIME \leftarrow 0$; $N \leftarrow$ Number of reactions; $M \leftarrow$ Number of species
3: **X** $\leftarrow$ (Population of species); (**Lx**, **Ux**) $\leftarrow$ (Fluctuation interval of **X** where **Lx** is lower bound and **Ux** is upper bound)
4: $\mathbf{\Gamma} \leftarrow$ A matrix in which each element $\Gamma_{i,j}$ is the full propensity of reaction R_j provided C_i is a reactant of R_j and R_j is not included by any preceding $\Gamma_{k,j}$ where $(k < i)$;
5: $\mathbf{\Gamma L} \leftarrow$ Lower bound of $\mathbf{\Gamma}$; $\mathbf{\Gamma U} \leftarrow$ Upper bound of $\mathbf{\Gamma}$;
6: $\mathbf{\Omega} \leftarrow$ An array where each element Ω_i is a group such that $\Omega_i \leftarrow \sum_j \Gamma U_{i,j}$;
7: **BLOCKPROP** $\leftarrow$ It is an array where an element $BLOCKPROP_s$ is the sth block which contains sum of the groups (Ω_i values) belong to it;
8: $TP \leftarrow \sum_s BLOCLPROP_s = \sum_{i=0}^{M-1} \Omega_i$ (Total propensity upper bound);
9: **BLOCK** $\leftarrow$ An array where an element $BLOCK_i$ holds the index of the first group Ω_x contained by ith block;
10: **SPECIES_BLOCK** $\leftarrow$ An array where an element $SPECIES\_BLOCK_i$ holds the index of the block containing Ω_i.
11: $\mathbf{\Upsilon} \leftarrow$ A matrix which is used to keep track of the indexes of the reactions associated with each reactant species such that $\Upsilon_{i,j}$ contains index value of reaction R_j provided C_i is a reactant of R_j;
12: **E** $\leftarrow$ A matrix in which an element $E_{j,i}$ keeps index value of each species C_i such that C_i is either a reactant or a product of R_j;
13: **G** $\leftarrow$ A stoichiometry matrix in which each element $G_{j,i}$ is $-n$ if C_i is a reactant of R_j and it is $+n$ if C_i is a product;
14: $IVAL \leftarrow 0$ (Group index of reaction to be executed), $JVAL \leftarrow 0$ (Element index of reaction to be executed)
15: **STEP 2. (Selection of blocks)**
16: $NOFBL \leftarrow \lfloor \sqrt{M} \rfloor$ (Number of blocks)
17: $s \leftarrow 0$, $BLOCK_0 \leftarrow 0$
18: **for** $i \leftarrow 0$ TO $M-1$ **do**
19: **if** $(i < (s+1)\frac{M}{NOFBL})$ **then**
20: $BLOCKPROP_s \leftarrow BLOCKPROP_s + \Omega_i$
21: **else**
22: $s \leftarrow s+1$
23: $BLOCKPROP_s \leftarrow BLOCKPROP_s + \Omega_i$
24: $BLOCK_s \leftarrow i$
25: **end if**
26: $SPECIES\_BLOCK_i \leftarrow s$
27: **end for**
28: **STEP 3. (Select the index μ of the candidate reaction R_μ)**
29: $ACCPT \leftarrow FALSE$; $TM \leftarrow 1$
30: $RN1 \leftarrow RAND()$; $SAM \leftarrow TP \times RN1$;
31: $SEL \leftarrow 0$; $ABC \leftarrow 0$
32: **for** $i \leftarrow 0$ TO $(NOFBL-1)$ **do**
33: $SEL \leftarrow SEL + BLOCKPROP_i$
34: **if** $(SEL > SAM)$ **then**
35: $SEL \leftarrow SEL - BLOCKPROP_i$
36: $ABC \leftarrow BLOCK_i$
37: $break$
38: **end if**
39: **end for**
40: **for** $i \leftarrow ABC$ TO $(M-1)$ **do**
41: $SEL \leftarrow SEL + \Omega_i$
42: **if** $(SEL > SAM)$ **then**
43: $SEL \leftarrow SEL - \Omega_i$
44: $IVAL \leftarrow i$ (**Calculate the group index**)
45: $break$
46: **end if**
47: **end for**
48: **for all** j in $\mathbf{\Gamma U}_{\mathrm{IVAL}}$ **do**
49: $SEL \leftarrow SEL + \Gamma U_{\mathrm{IVAL,j}}$
50: **if** $(SEL > SAM)$ **then**
51: $JVAL \leftarrow j$ (**Calculate the element index**)
52: $break$
53: **end if**
54: **end for**
55: $\mu \leftarrow (\Upsilon_{\mathrm{IVAL,JVAL}})$
56: **STEP 4. (Acceptance of R_μ for the execution)**
57: $RN2 \leftarrow RAND()$
58: **if** $(RN2 \leq \Gamma L_{\mathrm{IVAL,JVAL}} / \Gamma U_{\mathrm{IVAL,JVAL}})$ **then**
59: $ACCPT \leftarrow TRUE$
60: **else**
61: $\Gamma_{\mathrm{IVAL,JVAL}} \leftarrow$ Calculation of the full propensity of R_μ with present state
62: **if** $(RN2 \leq \Gamma_{\mathrm{IVAL,JVAL}} / \Gamma U_{\mathrm{IVAL,JVAL}})$ **then**
63: $ACCPT \leftarrow TRUE$
64: **end if**
65: **end if**
66: $RN3 \leftarrow RAND()$; $TM \leftarrow TM \times RN3$
67: **if** $(ACCPT \neq TRUE)$ **then**
68: Goto **STEP 3.**
69: **end if**

TABLE 1
Descriptions of the symbols used in *BlSSSA*.

Symbols	Description
M	Number of species
N	Number of reactions
X	Population of species
(**Lx**, **Ux**)	Fluctuation interval of **X** where **Lx** is lower bound and **Ux** is upper bound
$\mathbf{\Gamma}$	A matrix in which each element $\Gamma_{i,j}$ is the full propensity of reaction R_j provided C_i is a reactant of R_j and R_j is not included by any preceding $\Gamma_{k,j}$ where $k < i$
($\mathbf{\Gamma L}$, $\mathbf{\Gamma U}$)	Lower and upper bounds of $\mathbf{\Gamma}$
$\mathbf{\Omega}$	An array where each element Ω_i is a group such that $\Omega_i \leftarrow \sum_j \Gamma \mathrm{U}_{i,j}$
BLOCKPROP	It is an array where an element $BLOCKPROP_s$ is the sth block which contains sum of the groups (Ω_i values) belonging to it
TP	$\sum_s BLOCLPROP_s = \sum_{i=0}^{M-1} \Omega_i$ (Total propensity upper bound)
BLOCK	An array where an element $BLOCK_i$ holds the index of the first group Ω_x contained by ith block
SPECIES_BLOCK	An array where an element $SPECIES\_BLOCK_i$ holds the index of the block containing Ω_i
$\mathbf{\Upsilon}$	A matrix which is used to keep track of the indexes of the reactions associated with each reactant species such that $\Upsilon_{i,j}$ contains index value of reaction R_j provided C_i is a reactant of R_j
E	A matrix in which an element $E_{j,i}$ keeps index value of each species C_i such that C_i is either a reactant or a product of R_j
G	A stoichiometry matrix in which each element $G_{j,i}$ is '$-n$' if C_i is a reactant of R_j and it is '$+n$' if C_i is a product
H	A matrix in which an element $H_{j,i}$ keeps index value of each reactant C_i such that C_i is a reactant of R_j

70: $\tau \leftarrow -ln(TM)/TP$ **(Calculate time τ)**
71: For each index k of $\mathbf{E}_\mu$, $l \leftarrow E_{\mu,k}$ and $X_l \leftarrow X_l + G_{\mu,k}$ **(Execute R_μ)**
72: **STEP 5. (Update $\mathbf{\Gamma}$, $\mathbf{\Omega}$, BLOCKPROP** and TP**)**
73: **for all** k in $\mathbf{E}_\mu$ **do**
74: $l \leftarrow E_{\mu,k}$
75: **if** ($X_l \geq Lx_l$ AND $X_l \leq Ux_l$) **then**
76: Continue the loop
77: **end if**
78: Calculate the new fluctuation interval for X_l
79: $TP \leftarrow TP - \Omega_l$
80: $BLOCKPROP_{\text{SPECIES\_BLOCK}_l} \leftarrow BLOCKPROP_{\text{SPECIES\_BLOCK}_l} - \Omega_l$
81: $\Omega_l \leftarrow 0$
82: **for all** m in $\mathbf{\Gamma L}_l$ and $\mathbf{\Gamma U}_l$ **do**
83: Calculate $\Gamma \mathrm{L}_{l,m}$ and $\Gamma \mathrm{U}_{l,m}$ with present state
84: $\Omega_l \leftarrow \Omega_l + \Gamma \mathrm{U}_{l,m}$
85: **end for**
86: $TP \leftarrow TP + \Omega_l$
87: $BLOCKPROP_{\text{SPECIES\_BLOCK}_l} \leftarrow BLOCKPROP_{\text{SPECIES\_BLOCK}_l} + \Omega_l$
88: **if** $IVAL \neq l$ AND $l \in \mathbf{H}_\mu$ **then**
89: $TP \leftarrow TP - \Gamma \mathrm{U}_{\mathrm{IVAL,JVAL}}$
90: $BLOCKPROP_{\text{SPECIES\_BLOCK}_{\mathrm{IVAL}}} \leftarrow$
91: $BLOCKPROP_{\text{SPECIES\_BLOCK}_{\mathrm{IVAL}}} - \Gamma \mathrm{U}_{\mathrm{IVAL,JVAL}}$
92: $\Omega_{\mathrm{IVAL}} \leftarrow \Omega_{\mathrm{IVAL}}$ - $\Gamma \mathrm{U}_{\mathrm{IVAL,JVAL}}$
93: Calculate $\Gamma \mathrm{L}_{\mathrm{IVAL,JVAL}}$ and $\Gamma \mathrm{U}_{\mathrm{IVAL,JVAL}}$ with present state
94: $TP \leftarrow TP + \Gamma \mathrm{U}_{\mathrm{IVAL,JVAL}}$
95: $BLOCKPROP_{\text{SPECIES\_BLOCK}_{\mathrm{IVAL}}} \leftarrow$
96: $BLOCKPROP_{\text{SPECIES\_BLOCK}_{\mathrm{IVAL}}} + \Gamma \mathrm{U}_{\mathrm{IVAL,JVAL}}$
97: $\Omega_{\mathrm{IVAL}} \leftarrow \Omega_{\mathrm{IVAL}} + \Gamma \mathrm{U}_{\mathrm{IVAL,JVAL}}$
98: **end if**
99: **end for**
100: **STEP 6. Terminate**
101: $TIME \leftarrow TIME + \tau$
102: **if** ($TIME < ENDINGTIME$) **then**
103: Goto **STEP 3.**
104: **end if**

3 for selecting another candidate reaction R_μ (lines 67 to 69). Once R_μ is accepted, the reaction occurrence time τ is calculated in line 70. Next, R_μ is executed by updating the population of its reactants and products. As a result, n molecules are consumed from each reactant, and at the same time, n molecules are generated in each product using the step given in line 71. It updates propensities of all dependent reactions of R_μ in STEP 5. It skips propensity update for each group Ω_l where population X_l is within its fluctuation interval [Lx_l, Ux_l] (lines 73 to 77). Otherwise, *BlSSSA* calculates a new fluctuation interval for X_l. Then, it updates reaction propensity bounds $\mathbf{\Gamma L}_l$ and $\mathbf{\Gamma U}_l$, group propensity upper bound Ω_l, block propensity upper bound $BLOCKPROP_{\text{SPECIES\_BLOCK}_l}$, and total propensity upper bound TP (lines 78 to 87). It also updates $\Gamma \mathrm{L}_{\mathrm{IVAL,JVAL}}$, $\Gamma \mathrm{U}_{\mathrm{IVAL,JVAL}}$, Ω_{IVAL}, $BLOCKPROP_{\text{SPECIES\_BLOCK}_{\mathrm{IVAL}}}$ and TP if C_l is not C_{IVAL} and C_l is a reactant to R_μ (lines 88 to 98). In STEP 6, it checks whether to terminate or not. It updates the $TIME$ at line 101. If $Time < ENDINGTIME$ then it goes to STEP 3, othewise, it terminates (lines 102 to 104).

2.3 Proof of Correctness and Computational Cost of *BlSSSA*

In *BlSSSA*, computational cost has been reduced from that in other SSAs where full propensity update is performed, by skipping a large number of propensity updates. Minimization of the number of propensity updates introduces errors which have been corrected by employing a rejection test. This ensures that *BlSSSA* is exact and executes a reaction with the same probability as any standard SSA does. Here, we provide the proof of exactness followed by the computational complexity of the present algorithm.

2.3.1 Proof of Exactness of BlSSSA

Proof of exactness of the algorithms RSSA and RSSA-CR has been derived mathematically in [17], [23]. Following this way, we have proved the exactness of *BlSSSA*. $BLOCKPROP_i$ is the sum of propensity upper bounds of ith block. Therefore, the probability of selecting ith block is

$$Pr(\text{Selecting } i\text{th block}) = \frac{BLOCKPROP_i}{TP} \tag{8}$$

Here, TP is the total propensity upper bound. In ith block, a candidate reaction R_μ is selected by performing two consecutive linear searches, one through the major index i and another through the minor index j such that $\mu = \Upsilon_{i,j}$. The probability of selecting R_μ in ith block is

$$Pr(\text{Selecting } R_\mu \text{ in } i\text{th block}) = \frac{\Gamma \mathrm{U}_{i,j}}{BLOCKPROP_i} \tag{9}$$

Here, $\Gamma \mathrm{U}_{i,j}$ is the propensity upper bound of R_μ. Thus, the probability of selecting a candidate reaction R_μ is

$$\begin{aligned} Pr(\text{Selecting } R_\mu) &= \frac{BLOCKPROP_i}{TP} \times \frac{\Gamma \mathrm{U}_{i,j}}{BLOCKPROP_i} \\ &= \frac{\Gamma \mathrm{U}_{i,j}}{TP} \end{aligned} \tag{10}$$

The probability of accepting the candidate reaction R_μ is $\Gamma_{i,j}/\Gamma \mathrm{U}_{i,j}$. Therefore, the probability of selecting as well as accepting the candidate reaction R_μ (where $\mu = \Upsilon_{i,j}$) to fire is

$$Pr(\text{Selecting and accepting } R_\mu) = \frac{\Gamma \mathrm{U}_{i,j}}{TP} \times \frac{\Gamma_{i,j}}{\Gamma \mathrm{U}_{i,j}} = \frac{\Gamma_{i,j}}{TP} \quad (11)$$

On the other hand, the probability of selecting as well as accepting an arbitrary reaction R to fire can be written as

$$Pr(\text{Selecting and accepting } R) = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}\Gamma_{i,j}}{TP} = \frac{a}{TP} \quad (12)$$

Here, a is the total propensity. Thus, the probability of selecting as well as accepting R_μ to fire provided an arbitrary reaction R is selected as well as accepted to fire, is

$$Pr(\text{Selecting and accepting } R_\mu | R \text{ is selected and accepted}) = \frac{\Gamma_{i,j}}{TP} \Big/ \frac{a}{TP} = \frac{\Gamma_{i,j}}{a} \quad (13)$$

This is the probability with which a candidate reaction occurs in an exact SSA.

BlSSSA generates reaction firing time τ using Erlang distribution. In RSSA, generation of reaction firing time τ is consistent with that of a standard SSA. **Corresponding proof has been provided in** [15], [17]. Following this way, we have proved that generation of reaction firing time τ in *BlSSSA* is also consistent with that of SSAs, and its probability density function (PDF) is exponentially distributed with total propensity a. **In other words, PDF is $f_\tau(x) = ae^{-ax}$. Hence the PDF of reaction firing time τ is exponentially distributed with total propensity a. The proof has been given in supplementary material.**

2.3.2 Time Complexity

The computational cost of *BlSSSA* includes two parts: (a) cost of selecting and executing the candidate reaction R_μ, and (b) cost of propensity update after the execution of R_μ. For selecting R_μ, a linear search is performed through the block propensities to obtain the block containing R_μ. Once the block is obtained, another linear search is performed through the group propensities within this block to obtain the group index of the reaction R_μ. Then, one more linear search is performed to obtain the element index of the reaction R_μ. The number of blocks in *BlSSSA* is the square root of M. Hence, the time complexity of block search operation is O($\sqrt{M}$). Each block contains approximately $\sqrt{M}$ groups. Therefore, cost of obtaining the group index is O($\sqrt{M}$). If we assume that there is an average of f reactions contained in each group then, time complexity to obtain the element index is O(f). The value of f is generally very small except the case of strongly coupled networks, and therefore, the computational cost of selecting the candidate reaction R_μ is O($\sqrt{M}$). After the selection procedure, a validation test is performed on the candidate reaction R_μ. The average number of validation tests is bounded by a very small number aU_μ/a_μ [16]. Here, aU_μ and a_μ are, respectively, the propensity upper bound and propensity of R_μ. After the acceptance, R_μ is executed. The cost of reaction execution is very small, system dependent and is O(1). Therefore, overall cost of selection, acceptance and execution of the candidate reaction R_μ is O($\sqrt{M}$).

After the execution of R_μ, a new value of Ω_i (group propensity upper bound) corresponding to each C_i is calculated where C_i is either a reactant or a product of R_μ. Then, Ω_i is used to update the block propensity upper bound $BLOCKPROP_{SPECIESBLOCK_i}$ corresponding to C_i and TP. **Unlike RSSA-CR, reaction propensity upper bounds in the groups are static for *BlSSSA*, and they do not move from one group to another.** The cost of propensity update is O(f) as each group contains an average of f reactions. Propensity update is performed for only those C_is for which population crosses the specified fluctuation intervals. Due to this reason, a very few propensity update operation is performed, and therefore, the cost of propensity update can be considered as O(1). Hence, the computational cost of *BlSSSA* is O($\sqrt{M}$). **Propensity update cost in RSSA-Binary, RSSA-Lookup and RSSA-CR is higher compared to that in *BlSSSA*, SUESSA, SUESSSA and RSSA as cost of updating the data structure in RSSA-Binary, RSSA-Lookup and RSSA-CR is high.**

Let us compare the propensity update operations between *BlSSSA* and RSSA. RSSA maintains a species-reaction dependency graph which stores species wise dependent reaction propensities. Consider a reaction R_1 : $C_1 + C_2 \longrightarrow C_3 + C_4$ in a strongly coupled network, in which the number of dependent reactions on each of C_1, C_2, C_3 and C_4 are 100. Therefore, total number of dependent reactions on R_1 is 400. Let us also assume that after the execution of R_1, populations of all four species have exceeded their fluctuation intervals and all the 400 reactions need propensity updates. At the beginning of propensity update operation, a new fluctuation interval for the population of C_1 is calculated. Then, the old propensities of all the 100 dependent reactions are subtracted from the total propensity value. It involves 100 subtractions. Next, the new propensities for all the 100 reactions are calculated. It performs 100 multiplication operations. After that it performs 100 additions to update the total propensity value with the newly calculated 100 reaction propensities. Therefore, C_1 is responsible for 100 subtractions, 100 multiplications and 100 additions. Similarly, for all the above four species, RSSA performs 400 subtractions, 400 multiplications and 400 additions.

Now we discuss the propensity update operation in *BlSSSA* with the same example. Let us consider that Ω_i is the group propensity of C_i, and the propensity of R_1 is contained in Ω_1. All the four group propensities Ω_1 through Ω_4 can belong to the same single block or can be distributed among multiple blocks. The group distribution of propensities among blocks does not create any impact on the complexity of the propensity update operation in *BlSSSA*. Each of Ω_1 through Ω_4 contains 100 reaction propensities and in total 400 reaction propensities are dependent on R_1. During the propensity update for the third group, new fluctuation interval of population for C_3 is calculated. Then, the group propensity Ω_3 is subtracted from total propensity as well as from block propensity and that involves 2 subtraction operations. Then, the new propensities of all the 100 dependent reactions are calculated, which involves 100 multiplica-

tions, and each propensity is cumulatively updated in Ω_3 and it involves 100 additions. Then, Ω_3 is added to the total propensity and block propensity which involves 2 additions. Therefore, update of Ω_3 involves 2 subtractions, 100 multiplications and 102 additions. The same trend is followed in the case of Ω_2 and Ω_4. Propensity of R_1 is contained in Ω_1. For Ω_1, one additional access is needed to recalculate the propensity of R_1 with updated population of C_2. Therefore, Ω_1 performs 4 subtractions, 101 multiplications and 105 additions. Hence, R_1 involves in total 10 subtractions, 401 multiplications and 411 additions. If R_1 is a first order reaction (C_1 being the only reactant), then it involves 6 subtractions, 300 multiplications and 306 additions. In this case, RSSA involves 300 subtractions, 300 multiplications and 300 additions. Therefore, in strongly coupled networks, the update operation in *BlSSSA* is faster than that in RSSA. In case of weakly coupled networks too propensity update in *BlSSSA* is either the same or faster than that in RSSA. Propensity update operation in *BlSSSA* is independent of the number of blocks. For the colloidal aggregation network, if each species depends on an average of f reactions, then it is O(f). For the other networks, f is a small number and hence it is O(1).

3 BENCHMARKS

The performance of *BlSSSA* is compared with that of other existing algorithms in simulating large biochemical networks. For this purpose, the methods, viz., PDM, SPDM, PSSA-CR, SSA-CR, RSSA, RSSA-Binary, RSSA-Lookup, RSSA-CR, SUESSA and SUESSSA [12], [13], [15], [16], [17], [18], [24], have been considered. Two theoretical networks − cyclic linear chain and colloidal aggregation network, and three real systems − B cell receptor signaling network, FcεRI signaling network and 1,3-Butadiene Oxidation network, have been considered here. Here 10^7 reaction executions have been performed for each method in case of each network for the purpose of comparison. Each plot depicting performances of the various methods is generated by considering the mean of 100 independent simulations. Plots of simulation times in case of real systems are shown in supplementary material. Codes of PDM, SPDM, PSSA-CR, SSA-CR and RSSA are implemented by us using java. Java versions of RSSA-Binary, RSSA-Lookup and RSSA-CR have been obtained from [17]. *BlSSSA* is also developed using java. The species fluctuation level is kept at 10% during the simulation of *BlSSSA* and the other rejection based algorithms (RSSA, RSSA-Binary, RSSA-Lookup and RSSA-CR).

3.1 Colloidal Aggregation Network

Colloidal aggregation refers to creating large clusters of nanoparticles. It is an extensively investigated natural phenomenon [25], [26]. Colloidal aggregation forms a strongly coupled biochemical network with M species and N reactions, and is given by

$$\begin{aligned} &C_i + C_j \xrightarrow{k_{i,j}} C_{i+j}, \qquad \text{for} \quad i = 0, ..., \left\lfloor \frac{M-1}{2} \right\rfloor; \quad j = i, ..., M-i, \\ &C_m \xrightarrow{k_{m,n}} C_n + C_{m+n}, \qquad \text{for} \quad m = 0, ..., M-1; \\ &n = 0, ..., \left\lfloor \frac{m-1}{2} \right\rfloor. \end{aligned} \tag{14}$$

Here, $k_{i,j}$ and $k_{m,n}$ are reaction rate constants which have been set randomly between 0 and 100. Initial population of all the species C_is have been kept between 100 and 1000. In this article each method has been executed for this network with number of species (M) being 100, 500 and 1000.

For a biochemical network, total simulation time required by an algorithm is composed of searching time of the reactions to be executed, reaction execution time and the time taken for propensity update. The searching times, propensity update times and total simulation times of *BlSSSA* are compared with those of the others. Reaction execution time being very small has not been taken into consideration.

SPDM, SSA-CR and RSSA-Lookup are not executed for $M = 1000$ due to their large execution times. Plot of the searching times of various methods is shown in Figure 1(a). Searching time in RSSA is quite large compared to that in others as the search operation in RSSA is O(N). Here we have considered $M = 1000$ and $N = 500000$. The computational cost of searching in *BlSSSA* is O($\sqrt{M}$) and it is 1052.12 times lower than that in RSSA for $M = 1000$. *BlSSSA* is also faster in search operation than PDM, SPDM, SUESSA and SUESSSA as search operation cost in these algorithms is O(M). For $M = 1000$, *BlSSSA* selects only 31 blocks, and therefore, the search operation is performed only through these 31 blocks. This is why searching in *BlSSSA* is almost as fast as that in PSSA-CR, SSA-CR, RSSA-Binary, RSSA-Lookup and RSSA-CR.

Propensity update time is plotted in Figure 1(b). Propensity update operation in *BlSSSA* is, respectively, 189.34 and 396.58 times faster than that in PDM and PSSA-CR for $M = 1000$. *BlSSSA* is 1155.89, 994.23 and 1376.67 times faster in propensity update operation than SPDM, SSA-CR and RSSA-Lookup, respectively, for $M = 500$. Number of propensity updates in PDM, SPDM, PSSA-CR and SSA-CR is 10^7, which results in slower propensity update operation compared to that in *BlSSSA*, where the number of propensity updates is $\sim 6.5 \times 10^5$ due to its infrequent propensity updates. Maintenance of the lookup table during propensity update operation increases the propensity update time in RSSA-Lookup despite having small number of propensity updates ($\sim 6.5 \times 10^5$). Propensity update cost in the other rejection based algorithms, including RSSA, RSSA-Binary and RSSA-CR, is lower due to infrequent propensity updates which bounds the number of propensity updates to $\sim 6.5 \times 10^5$. Cost of propensity updates in RSSA-Binary and RSSA-CR is still higher compared to that in *BlSSSA*, as RSSA-Binary needs to maintain binary tree during propensity updates and RSSA-CR uses groups for storing the reaction propensities. During propensity update, some reaction propensities move within these groups, which increases the propensity update cost in RSSA-CR. On the other hand, propensity update operation in *BlSSSA* is almost as fast as that in RSSA, SUESSA and SUESSSA. For each reaction

whose propensity value needs to be updated, *BlSSSA* updates the total propensity value and the corresponding block propensity value. Propensity update operation in *BlSSSA* is, respectively, 31.04 and 39.12 times faster than that in RSSA-Binary and RSSA-CR.

Total simulation time is plotted in Figure 1(c). Due to the strong coupling of the colloidal aggregation network, a substantial part of its total simulation cost involves the propensity update cost. *BlSSSA* has a low propensity update cost, which helps in reducing its total simulation cost. *BlSSSA* is 78.64, 164.24 and 531.47 times faster than PDM, PSSA-CR and RSSA, respectively, for $M = 1000$. High total simulation cost in RSSA is due to its high search operation cost. *BlSSSA* is, respectively, 461.43, 389.57 and 546.78 times faster than SPDM, SSA-CR and RSSA-Lookup for $M = 500$. Simulation in RSSA-Binary and RSSA-CR is comparatively faster due to their lower search and propensity update costs. *BlSSSA* is, respectively, 15.46 and 17.19 times faster than RSSA-Binary and RSSA-CR for $M = 1000$. *BlSSSA* is faster than SUESSA and SUESSSA, due to its fast searching time.

3.2 Cyclic Linear Chain Network

It is a weakly coupled biochemical network with M species $C_0, \cdots, C_{M-1}$ such that

$$C_i \xrightarrow{k_i} \begin{cases} C_{i+1}, & \text{for } (0 \leq i \leq M-2), \\ C_0, & \text{for } i = M-1 \end{cases} \tag{15}$$

Here, k_i is the reaction rate constant for the reaction with C_i as its reactant. The population of each C_i has been considered between 100 and 1000 randomly, and k_i has been set randomly between 0 and 100 for the purpose of simulation. **All the methods are simulated for the cyclic linear chain network with number of species (M) being 1000, 10000, 50000 and 100000**. RSSA-Lookup has not been simulated with $M > 10000$ due to its large simulation time. Plot of searching time of various algorithms is shown in Figure 1(d). Search operation in *BlSSSA* is, respectively, 68.88, 68.76, 71.65, 51.40 and 53.61 times faster than that in PDM, SPDM, RSSA, SUESSA and SUESSSA for $M = 50000$. This is because *BlSSSA* uses block based search which is faster than ordinary linear search used in PDM, SPDM and RSSA, and cache-based linear search used in SUESSA and SUESSSA. Search operation cost in *BlSSSA* is O($\sqrt{M}$), and it selects $\sqrt{M}$ blocks to simulate a network of M species. Therefore, in order to simulate even larger networks ($M = 50000$), only 223 blocks are selected. Hence, the operation of block selection as well as candidate reaction selection within blocks do not involve much computational cost. It is observed that the search operation in *BlSSSA* even for $M = 50000$, is almost as fast as that in the methods including PSSA-CR, RSSA-Binary and RSSA-CR. SSA-CR and RSSA-Lookup are slightly faster in search operation than *BlSSSA*. **Search operation in *BlSSSA* is, respectively, 85.45, 84.56, 103.74, 66.17 and 80.81 times faster than that in PDM, SPDM, RSSA, SUESSA and SUESSSA for $M = 100000$. Search operation in *BlSSSA* is, respectively, 1.41, 1.82, 1.46 and 1.45 times slower than that in PSSA-CR, SSA-CR, RSSA-Binary and RSSA-CR for $M = 100000$.**

Propensity update operation in *BlSSSA* is, respectively, 8.93, 20.37, 9.79, 17.02, 6.43 and 6.95 times faster than that in PDM, SPDM, PSSA-CR, SSA-CR, RSSA-Binary and RSSA-CR for $M = 50000$ (Figure 1(e)). ***BlSSSA* in propensity update operation is, respectively, 8.60, 20.14, 9.52, 16.13, 6.24 and 7.41 times faster than that in PDM, SPDM, PSSA-CR, SSA-CR, RSSA-Binary and RSSA-CR for $M = 100000$.** Propensity update cost in these methods is not much high for this network due to its weak coupling, except the case of RSSA-Lookup. Propensity update operation in *BlSSSA* is 1845.87 times faster than that in RSSA-Lookup for $M = 10000$, while this is almost the same as that in RSSA, SUESSA and SUESSSA.

Simulation in *BlSSSA* is fast due to its efficient search and propensity update operations. Plot of total simulation time of the methods is shown in Figure 1(f). *BlSSSA* is, respectively, 54.45, 54.87, 3.53, 3.46, 54.73, 2.71, 2.63, 38.34 and 39.81 times faster than PDM, SPDM, PSSA-CR, SSA-CR, RSSA, RSSA-Binary, RSSA-CR, SUESSA and SUESSSA for $M = 50000$. **In case of $M = 100000$, BlSSSA is, respectively, 62.06, 65.67, 2.28, 2.20, 74.90, 1.72, 1.75, 46.48 and 56.80 times faster than that in PDM, SPDM, PSSA-CR, SSA-CR, RSSA, RSSA-Binary, RSSA-CR, SUESSA and SUESSSA.** Due to higher propensity update time in RSSA-Lookup, it is 213.51 times slower than *BlSSSA* for $M = 10000$.

3.3 B Cell Receptor Signaling Network

B cells are differentiated and proliferated by the activities of a multimeric protein, called B cell receptor [27]. B cell receptor signaling pathway has two feedback loops which are initiated by two proteins Fyn and Lyn. This pathway controls fates and activities of B cells. This is a large biochemical network consisting of 1122 species and 24388 reactions [28]. This network has been simulated by various SSAs for their performance evaluations. Searching time of the SSAs are given in Table 2. Search operation in RSSA is expensive, and it is 291.22 times faster in *BlSSSA* than RSSA. Search operation in *BlSSSA* is, respectively, 3.05, 2.39, 2.93 and 2.36 times faster than that in PDM, SPDM, SUESSA and SUESSSA. However, search operation cost in *BlSSSA* is comparable to that in PSSA-CR, SSA-CR, RSSA-Binary and RSSA-CR. Searching time in RSSA-Lookup is the lowest for this network.

The propensity update times of various methods are given in Table 2. Propensity update cost in PDM, SPDM, PSSA-CR and SSA-CR is high due to their large number of propensity updates (10^7). Propensity update operation in *BlSSSA* is, respectively, 77.34, 273.23, 205.46, 860.73 and 199.51 times faster than that in PDM, SPDM, PSSA-CR, SSA-CR and RSSA-Lookup. Number of propensity updates in the rejection based algorithms is small and is about 10000. Despite small number of propensity updates, the cost of updating the look up table in RSSA-Lookup is much high. Propensity update cost in RSSA-Binary and RSSA-CR is also high due to maintenance of their own respective data structures. Propensity update operation in *BlSSSA* is, respectively, 7.43 and 7.12 times faster than that in RSSA-Binary and RSSA-CR. Propensity update cost in *BlSSSA* is almost the same as that in RSSA, SUESSA and SUESSSA.

Total simulation time of the SSAs are given in Table 2. *BlSSSA* is, respectively, 15.14, 43.56, 36.56, 145.15, 174.35

and 42.47 times faster than PDM, SPDM, PSSA-CR, SSA-CR, RSSA and RSSA-Lookup. Large simulation cost in these algorithms is basically contributed by their high propensity update cost, except the case in RSSA, in which the simulation cost is contributed mostly by its search operation cost. Simulation in *BlSSSA* is, respectively, 2.63 and 2.18 times faster than that in SUESSA and SUESSSA, due to its fast searching time. *BlSSSA* is, respectively, 3.32 and 3.25 times faster than RSSA-Binary and RSSA-CR. High propensity update cost in RSSA-Binary and RSSA-CR increases their total simulation cost.

3.4 FcϵRI Signaling Network

FcϵRI signaling network is responsible for allergic diseases. Allergic reactions in RBL-2H3 cells are triggered by this signaling pathway. This biochemical network consists of 380 chemical substances and 3862 chemical reactions [29]. Searching time of the various algorithms in simulating the FcϵRI signaling network, are given in Table 2. Search operation cost in RSSA is high. Search operation in *BlSSSA* is, respectively, 2.53, 48.23, 2.34 and 1.91 times faster than that in PDM, RSSA, SUESSA and SUESSSA. As the number of species ($M = 380$) in this network is small, searching time in SPDM is quite low. Search operation in *BlSSSA* is performed only through 19 blocks. This is why searching in *BlSSSA* is quite fast and comparable to that in PSSA-CR, SSA-CR, RSSA-Binary, RSSA-Lookup and RSSA-CR.

The propensity update time of various SSAs are given in Table 2. Propensity update cost in PDM, SPDM, PSSA-CR and SSA-CR is much high due to their large number of propensity updates (10^7). Propensity update in *BlSSSA* is, respectively, 266.13, 324.24, 773.45 and 1372.70 times faster than that in PDM, SPDM, PSSA-CR and SSA-CR. The number of propensity updates in the rejection based algorithms including RSSA, RSSA-Binary, RSSA-Lookup, RSSA-CR, and *BlSSSA* is smaller ($\sim$190000). Propensity update operation in *BlSSSA* is, respectively, 6.15, 236.10 and 7.35 times faster than that in RSSA-Binary, RSSA-Lookup and RSSA-CR. Despite small number of propensity updates, its cost in RSSA-Lookup is much high. Propensity update operation in *BlSSSA* is almost the same as that in RSSA, SUESSA and SUESSSA.

Total simulation times of various SSAs are given in Table 2. SSAs, including PDM, SPDM, PSSA-CR, SSA-CR and RSSA-Lookup, are slow due to their large propensity update times. RSSA is also slow due to its large searching time. *BlSSSA* is, respectively, 43.89, 52.17, 121.20, 214.28, 28.08 and 38.51 times faster than PDM, SPDM, PSSA-CR, SSA-CR, RSSA and RSSA-Lookup. Simulation in *BlSSSA* is, respectively, 1.78 and 1.61 times faster than that in SUESSA and SUESSSA, due to its fast searching time. *BlSSSA* is, respectively, 2.69 and 2.98 times faster than RSSA-Binary and RSSA-CR. Simulation costs in RSSA-Binary and RSSA-CR are higher compared to that in *BlSSSA* due to their large propensity update costs.

3.5 Kinetic Model of 1,3-Butadiene Oxidation

1,3-Butadiene Oxidation network consists of 92 species and 613 reactions. This stiff network plays an important role in the study of hydrocarbon combustion [30]. Due to small size of this network, except the case of RSSA, search operation cost of the algorithms is low (Table 2). In RSSA, search operation is of the order of the number of reactions, and is the highest among all the methods. Search operation of *BlSSSA* is 31.83 times faster than that in RSSA.

PDM, SPDM, PSSA-CR and SSA-CR have large number of propensity updates and their corresponding cost is high (Table 2). Propensity update in *BlSSSA* is, respectively, 197.64, 325.59, 401.24 and 734.10 times faster than that in PDM, SPDM, PSSA-CR and SSA-CR. Among the partial propensity update methods, propensity update cost in RSSA-Lookup is the highest due to maintenance of its lookup table. Propensity update cost in RSSA-Binary and RSSA-CR is high due to maintenance of their data structures. Propensity update in *BlSSSA* is, respectively, 6.97, 222.46 and 7.23 times faster than that in RSSA-Binary, RSSA-Lookup and RSSA-CR. Like SUESSA and SUESSSA, *BlSSSA* keeps its propensity update cost as low as RSSA.

PDM, SPDM, PSSA-CR, SSA-CR, RSSA-Binary, RSSA-Lookup and RSSA-CR are slow due to their slow propensity update (Table 2). RSSA is slow due to its slow search operation. *BlSSSA* is, respectively, 31.92, 51.51, 63.09, 114.33, 16.25, 2.17, 35.64, 2.37, 1.10 and 1.06 times faster than PDM, SPDM, PSSA-CR, SSA-CR, RSSA, RSSA-Binary, RSSA-Lookup, RSSA-CR, SUESSA and SUESSSA. Due to small size of this network, simulation time of SUESSA and SUESSSA is close to that of *BlSSSA*.

3.6 Validation

The results obtained by *BlSSSA* have been validated with those of some other benchmark algorithms including ODM, SDM, PDM and SPDM. Here, each algorithm has been simulated for 100 times in case of each network. The algorithms have been executed upto 10^7 reactions in each simulation. Final species populations have been noted after the completion of each simulation. Mean of each species population is calculated after the completion of 100 simulations. Absolute deviations of the populations from their corresponding means are depicted using box plots, in case of 1000th species of B cell receptor signaling network (Figure 2(f)), 300th species of FcϵRI signaling network (Figure 2(g)), 500th species of colloidal aggregation network (Figure 2(h)) and 1000th species of cyclic linear chain (Figure 2(i)). In case of each network, populations are almost equally deviated from their means. ODM has the minimum inter-quartile range in case of B cell receptor signaling network, which ensures that its population dispersion is minimum compared to that of others (Figure 2(f)). SDM has the minimum population dispersion compared to that of the others in case of cyclic linear chain (Figure 2(i)). Populations are almost equally dispersed in case of the other networks. Plots of absolute deviations of populations in case of other species of these networks are given in supplementary material. Absolute deviations of species populations from their corresponding means are plotted for 200th species (Figure S2(a)), 400th species (Figure S2(b)), 600th species (Figure S2(c)), 800th species (Figure S2(d)) in case of B cell receptor signaling network; 50th species (Figure S2(e)), 100th species (Figure S2(f)), 150th species (Figure S2(g)), 200th species (Figure S2(h)) in case of FcϵRI signaling network; 100th species (Figure S3(a)),

200th species (Figure S3(b)), 300th species (Figure S3(c)), 400th species (Figure S3(d)) in case of colloidal aggregation network; 200th species (Figure S3(e)), 400th species (Figure S3(f)), 600th species (Figure S3(g)) and 800th species (Figure S3(h)) in case of cyclic linear chain network. In case of each network, population deviations are not significantly varying if we compare them among the algorithms. There is a little variation in dispersion of the populations. Populations in SPDM in Figure (S2(f) and Figure S3(a)), are having the minimum dispersion compared to that in the other algorithms. Populations are almost equally dispersed in case of the other networks.

4 Discussion

Plots of propensity update time versus searching time of various methods have been shown in Figure 2(a) for colloidal aggregation network, Figure 2(b) for cyclic linear chain network, Figure 2(c) for B cell receptor signaling network, Figure 2(d) for FcϵRI signaling network and Figure 2(e) for stiff 1,3-Butadiene Oxidation network. *BlSSSA* is developed for reducing the propensity update time as well as search operation time. Data structure for maintaining the blocks used in *BlSSSA* is different from that in rejection-based algorithms including RSSA-CR [17], [31]. During propensity update in *BlSSSA*, the reactions do not move between the blocks. In RSSA-CR, the reactions move from one group to another during propensity update, due to which, unlike *BlSSSA*, the propensity update time increases in RSSA-CR. Since the reactions are static in the blocks, only one random number is used for selecting the candidate reaction index. In RSSA-CR, the reactions are dynamic in the groups, and two random numbers are used for selecting candidate reaction index.

5 Conclusion

Propensity update cost in PDM, SPDM and SSA-CR is high due to their large number of propensity updates. The rejection based algorithms (RSSA, RSSA-Binary, RSSA-Lookup and RSSA-CR) use infrequent propensity update technique in order to reduce propensity update cost. Propensity update cost in RSSA-Binary, RSSA-Lookup and RSSA-CR is high in simulating strongly coupled networks. In order to reduce propensity update cost in case of strongly coupled networks, the slow update methods (SUESSA and SUESSSA) have been developed. Search operation cost in the slow update method is high in simulating weakly coupled networks.

In this article, *BlSSSA* has been developed in order to reduce the search as well as propensity update costs in simulating both weakly and strongly coupled networks. *BlSSSA* is an exact SSA. The algorithm maintains blocks for storing the reaction propensities. *BlSSSA* supports only mass action kinetics. For simulating a network, *BlSSSA* selects the number of blocks, for storing the reaction propensity values, as square root of the number of species of the network. In RSSA-CR, reaction propensities are stored in groups. The grouping structure is dynamic, i.e., the reaction propensities move from one group to another during simulation. For this reason, two random numbers are needed for selecting a candidate reaction. Third one is needed for the acceptance of the candidate reaction to be executed. Reaction propensities in *BlSSSA* are maintained in blocks. They do not move between blocks during simulation. For this reason, only one random number is needed to select a candidate reaction for execution. Another one is needed for the acceptance of the candidate reaction to be executed. That is why the searching in *BlSSSA* is as fast as that in rejection based algorithms.

Unlike RSSA-Binary, RSSA-Lookup and RSSA-CR, *BlSSSA* maintains a linear data structure for storing the reaction propensity values. Due to this reason, the propensity update operation in *BlSSSA* is faster than that in others except RSSA, SUESSA and SUESSSA. Propensity update operation in *BlSSSA* is comparable with that in RSSA, SUESSA and SUESSSA. In simulating strongly coupled networks, total simulation cost in RSSA-Binary, RSSA-Lookup and RSSA-CR are expensive because of their slow propensity update operations. The total simulation time in RSSA is high because of its slow search operation. SUESSA and SUESSSA use cache-based linear search which makes them slow in case of large weakly coupled networks. Due to this reason, total simulation time in SUESSA and SUESSSA becomes high in simulating weakly coupled network. Total simulation time in *BlSSSA* is lower compared to the others in case of both strongly and weakly coupled networks due to its fast search as well as propensity update operations.

6 Code Availability

The codes for *BlSSSA* and other algorithms along with all the networks considered in this study are available at https://github.com/debraj86/Algorithms.

Acknowledgments

Mr. Debraj Ghosh, one of the authors, gratefully acknowledges CSIR, India for providing him a Senior Research Fellowship (9/93 (0150)/2013, EMR-I). He also gratefully acknowledges University of Calcutta and Indian Statistical Institute for allowing him to pursue his Ph.D. Rajat K. De gratefully acknowledges SERB, DST, Government of India for granting him a project (MSC/2020/000350) under the scheme of MATRICS. The additional page charge has been provided from the grant.

References

[1] R. Erban, J. Chapman, and P. Maini, "A practical guide to stochastic simulations of reaction-diffusion processes," *arXiv preprint arXiv:0704.1908*, 2007.

[2] K. H. Ibsen, "The crabtree effect: a review," *Cancer research*, vol. 21, no. 7, pp. 829–841, 1961.

[3] D. M. Hockenbery, M. Tom, C. Abikoff, and D. Margineantu, "The warburg effect and beyond: metabolic dependencies for cancer cells," in *Cell Death Signaling in Cancer Biology and Treatment*, pp. 35–51, Springer, 2013.

[4] K. Otterstedt, C. Larsson, R. M. Bill, A. Ståhlberg, E. Boles, S. Hohmann, and L. Gustafsson, "Switching the mode of metabolism in the yeast saccharomyces cerevisiae," *EMBO reports*, vol. 5, no. 5, pp. 532–537, 2004.

[5] E. Postma, C. Verduyn, W. A. Scheffers, and J. P. Van Dijken, "Enzymic analysis of the crabtree effect in glucose-limited chemostat cultures of saccharomyces cerevisiae.," *Applied and environmental microbiology*, vol. 55, no. 2, pp. 468–477, 1989.

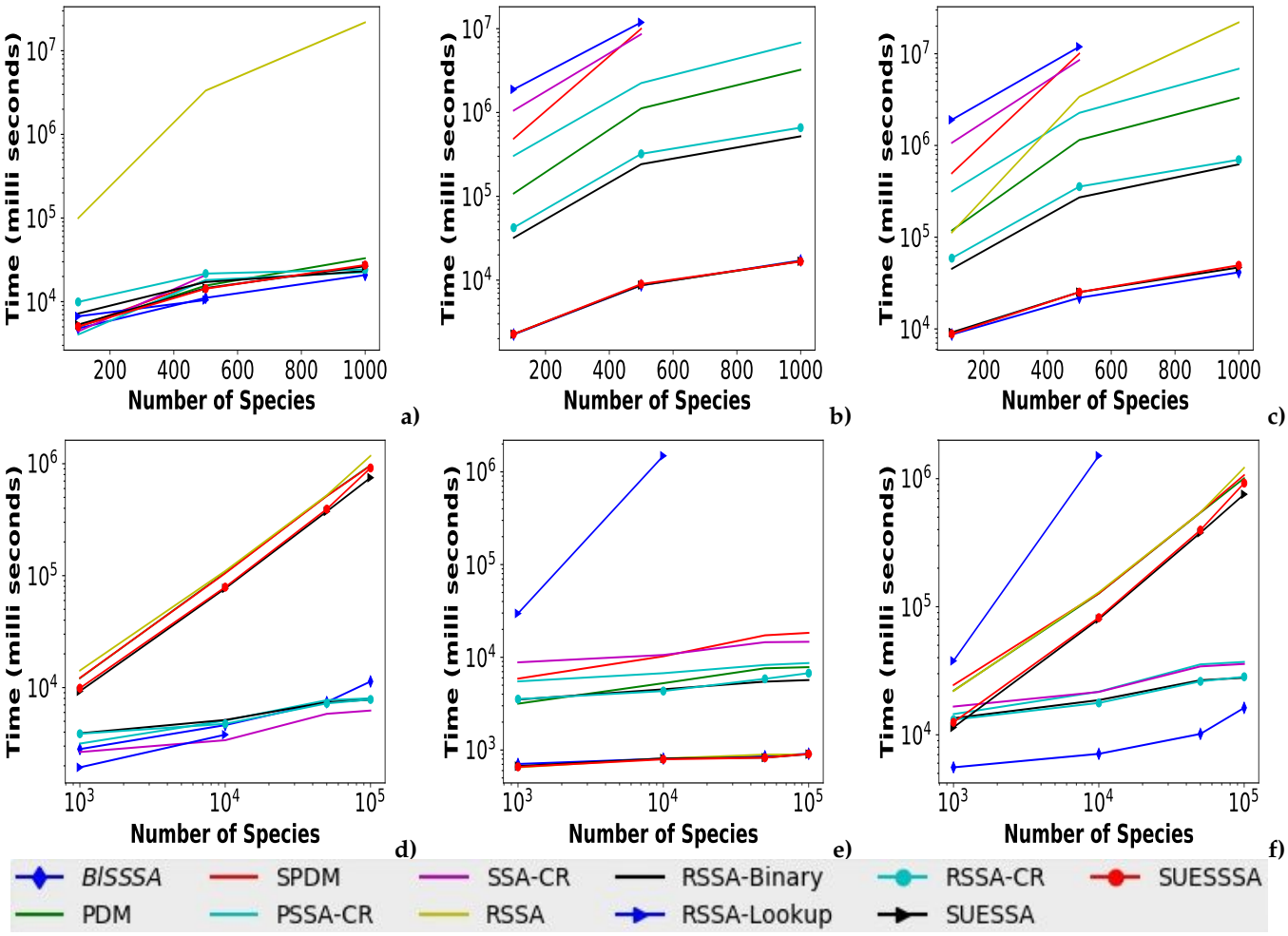

Fig. 1. **Performances of the various algorithms.** a) Searching times, b) propensity update times and c) total simulation times for colloidal aggregation network; d) Searching times, e) propensity update times and f) total simulation times for cyclic linear chain network. Y-axes of a), b) and c) are considered in logarithmic scale. Both the axes of d), e) and f) are considered in logarithmic scale.

[6] D. Ghosh and R. K De, "In silico modeling of crabtree effect," *Endocrine, Metabolic & Immune Disorders-Drug Targets (Formerly Current Drug Targets-Immune, Endocrine & Metabolic Disorders)*, vol. 17, no. 3, pp. 182–188, 2017.

[7] J. M. McCollum, G. D. Peterson, C. D. Cox, M. L. Simpson, and N. F. Samatova, "The sorting direct method for stochastic simulation of biochemical systems with varying reaction execution behavior," *Computational biology and chemistry*, vol. 30, no. 1, pp. 39–49, 2006.

[8] D. T. Gillespie, "A general method for numerically simulating the stochastic time evolution of coupled chemical reactions," *Journal of computational physics*, vol. 22, no. 4, pp. 403–434, 1976.

[9] D. T. Gillespie, "Exact stochastic simulation of coupled chemical reactions," *The journal of physical chemistry*, vol. 81, no. 25, pp. 2340–2361, 1977.

[10] M. A. Gibson and J. Bruck, "Efficient exact stochastic simulation of chemical systems with many species and many channels," *The journal of physical chemistry A*, vol. 104, no. 9, pp. 1876–1889, 2000.

[11] Y. Cao, H. Li, and L. Petzold, "Efficient formulation of the stochastic simulation algorithm for chemically reacting systems," *The journal of chemical physics*, vol. 121, no. 9, pp. 4059–4067, 2004.

[12] R. Ramaswamy, N. González-Segredo, and I. F. Sbalzarini, "A new class of highly efficient exact stochastic simulation algorithms for chemical reaction networks," *The Journal of chemical physics*, vol. 130, no. 24, p. 244104, 2009.

[13] R. Ramaswamy and I. F. Sbalzarini, "A partial-propensity variant of the composition-rejection stochastic simulation algorithm for chemical reaction networks," *The Journal of chemical physics*, vol. 132, no. 4, p. 044102, 2010.

[14] M. Dickison, S. Havlin, and H. E. Stanley, "Epidemics on interconnected networks," *Physical Review E*, vol. 85, no. 6, p. 066109, 2012.

[15] V. H. Thanh, C. Priami, and R. Zunino, "Efficient rejection-based simulation of biochemical reactions with stochastic noise and delays," *The Journal of chemical physics*, vol. 141, no. 13, p. 134116, 2014.

[16] V. H. Thanh, R. Zunino, and C. Priami, "On the rejection-based algorithm for simulation and analysis of large-scale reaction networks," *The Journal of chemical physics*, vol. 142, no. 24, p. 244106, 2015.

[17] V. H. Thanh, R. Zunino, and C. Priami, "Efficient constant-time complexity algorithm for stochastic simulation of large reaction networks," *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 14, no. 3, pp. 657–667, 2016.

[18] D. Ghosh and R. K. De, "Slow update stochastic simulation algorithms for modeling complex biochemical networks," *Biosystems*, vol. 162, pp. 135 – 146, 2017.

[19] S. Mauch and M. Stalzer, "Efficient formulations for exact stochastic simulation of chemical systems," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 8, no. 1, pp. 27–35, 2009.

[20] A. Auger, P. Chatelain, and P. Koumoutsakos, "R-leaping: Accelerating the stochastic simulation algorithm by reaction leaps," *The Journal of chemical physics*, vol. 125, no. 8, p. 084103, 2006.

[21] P. A. Maksym, "Fast monte carlo simulation of MBE growth," *Semiconductor Science and Technology*, vol. 3, pp. 594–596, jun 1988.

[22] J. L. Blue, I. Beichl, and F. Sullivan, "Faster monte carlo simulations," *Phys. Rev. E*, vol. 51, pp. R867–R868, Feb 1995.

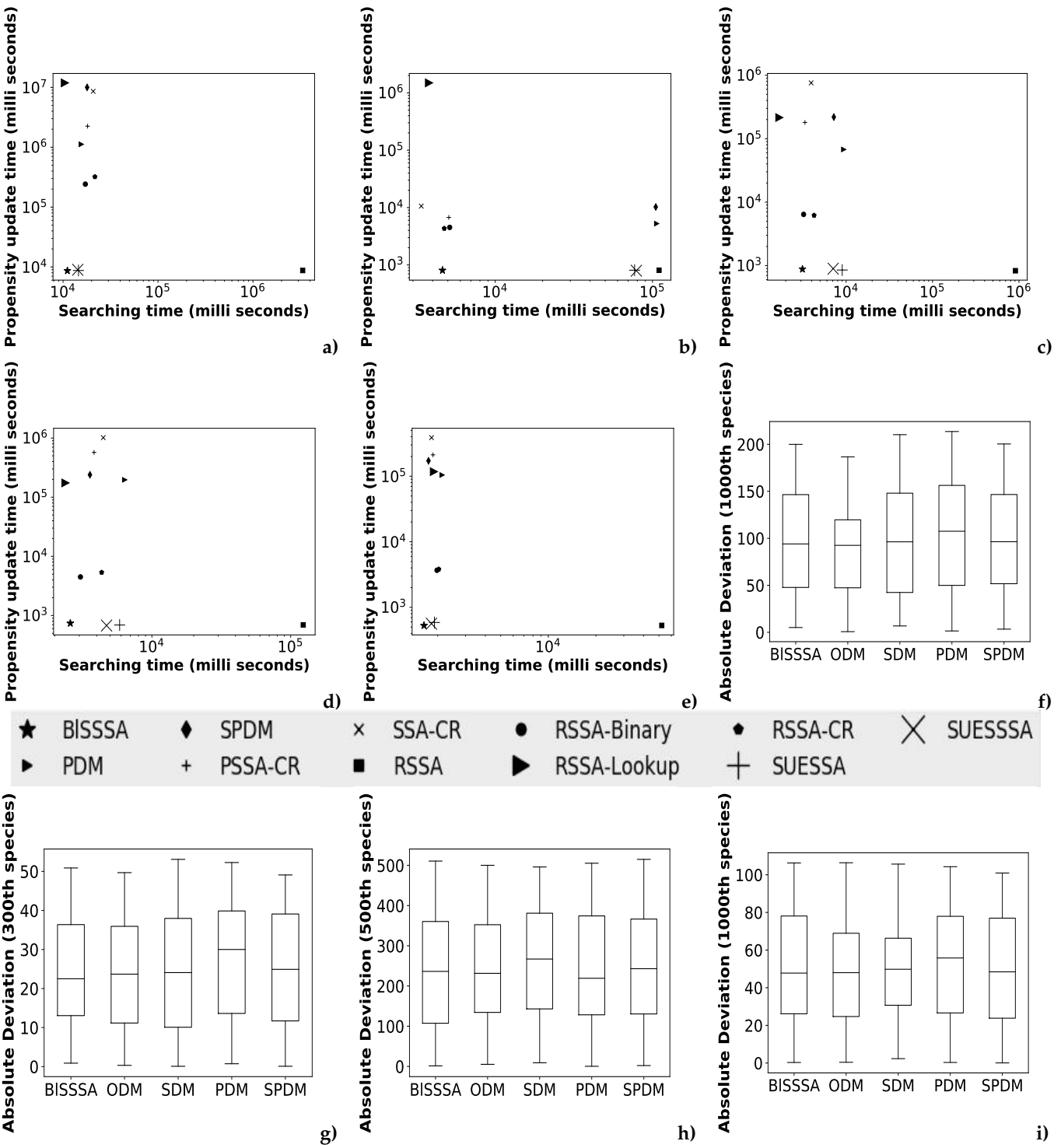

Fig. 2. **Performances of the various algorithms and validation of BISSSA with other benchmark algorithms.** Propensity update time vs searching times for a) colloidal aggregation network, b) cyclic linear chain network, c) B cell receptor signaling network, d) FcϵRI signaling network and e) 1,3-Butadiene Oxidation network. Box plots of absolute deviations of species populations from their corresponding means in case of f) 1000th species of B cell receptor signaling network, g) 300th species of FcϵRI signaling network, h) 500th species of colloidal aggregation network, and i) 1000th species of cyclic linear chain network. Both axes of a) to e) are considered in logarithmic scale.

[23] V. H. Thanh, *On Efficient Algorithms for Stochastic Simulation of Biochemical Reaction Systems*. PhD thesis, University of Trento, 2013.
[24] A. Slepoy, A. P. Thompson, and S. J. Plimpton, "A constant-time kinetic monte carlo algorithm for simulation of large biochemical reaction networks," *The journal of chemical physics*, vol. 128, no. 20, p. 205101, 2008.
[25] P. Meakin, "Models for colloidal aggregation," *Annual Review of Physical Chemistry*, vol. 39, no. 1, pp. 237–267, 1988.
[26] R. Klein and P. Meakin, "Universality in colloid aggregation," *Nature*, vol. 339, no. 3, 1989.
[27] T. Kurosaki, H. Shinohara, and Y. Baba, "B cell signaling and fate decision," *Annual Review of Immunology*, vol. 28, pp. 21–55, 2009.
[28] D. Barua, W. S. Hlavacek, and T. Lipniacki, "A computational model for early events in b cell antigen receptor signaling: analysis of the roles of lyn and fyn," *The Journal of Immunology*, vol. 189,

TABLE 2
Performances of various SSAs. Search, propensity update and total simulation times of B cell receptor signaling, FcεRI signaling and stiff 1,3-Butadiene oxidation networks. The simulation times are calculated in milli seconds.

Networks	Operation	Simulation times (milli seconds)										
		BlSSSA	PDM	SPDM	PSSA-CR	SSA-CR	RSSA	RSSA-Binary	RSSA-Lookup	RSSA-CR	SUESSA	SUESSSA
B cell receptor signaling	Search	3126	9378	7210	3341	3938	909073	3224	1688	4261	9014	7085
	Propensity update	879	67576	218935	179999	757535	830	6407	215189	6186	851	894
	Total	5322	79884	227654	189663	767562	923865	17118	220911	16684	13544	10912
FcεRI signaling	Search	2585	6372	3580	3832	4466	123091	3058	2385	4349	5858	4716
	Propensity update	740	196739	239474	571942	1015253	694	4487	174422	5351	691	677
	Total	4818	207897	246157	579945	1026325	131161	12557	181712	14099	8067	7310
1,3-Butadiene oxidation	Search	1638	2133	1749	1867	1828	52137	1978	1893	2031	1906	1826
	Propensity update	528	104356	171912	211853	387605	530	3678	117459	3816	588	563
	Total	3433	109578	176818	216598	392502	55790	7456	122364	8147	3781	3655

no. 2, pp. 646–658, 2012.
[29] Y. Liu, D. Barua, P. Liu, B. S. Wilson, J. M. Oliver, W. S. Hlavacek, and A. K. Singh, "Single-cell measurements of ige-mediated fcεri signaling using an integrated microfluidic platform," *PloS one*, vol. 8, no. 3, p. e60159, 2013.
[30] A. Laskin, H. Wang, and C. K. Law, "Detailed kinetic modeling of 1, 3-butadiene oxidation at high temperatures," *International Journal of Chemical Kinetics*, vol. 32, no. 10, pp. 589–614, 2000.
[31] L. Marchetti, C. Priami, and V. H. Thanh, *Simulation algorithms for computational systems biology*. Springer, 2017.

Mr. Debraj Ghosh has obtained his M.Tech. degree in Information Technology from University of Calcutta, Kolkata, India, in the year 2012. He has published 4 research papers in international journals and conference proceedings. Mr. Ghosh has 8 years of research experience. His research interest includes computational biology, bioinformatics and stochastic simulation algorithms. Mr. Ghosh is also working in the area of computer vision and naural language processing using statistical modeling, machine learning and deep learning techniques.

Dr. Rajat K. De is a Professor of the Indian Statistical Institute, Kolkata, India. He obtained his Ph.D. degree from the same Institute in 2000. He was a Distinguished Postdoctoral Fellow at the Whitaker Biomedical Engineering Institute, Johns Hopkins University, USA, during 2002-2003. During the last 18 years or so, Professor De has been working in the area of computational biology, bioinformatics and systems biology. He is also working on Big Data Analytics and Deep Learning in the domain of bioinformatics, systems biology and healthcare. Professor De visited the Department of Medicine, University of California, San Diego, in 2017 and 2018, with a Fulbright-Nehru Academic and Professional Excellence Fellowship. He has published about 100 research papers in international journals, conference proceedings and in edited books, and co-edited three books.