symmetry

Article

SMGen: A Generator of Synthetic Models of Biochemical Reaction Networks

Simone G. Riva [1,2,*], Paolo Cazzaniga [3,4,5], Marco S. Nobile [4,5,6], Simone Spolaor [1,4], Leonardo Rundo [7,8], Daniela Besozzi [1,4,5] and Andrea Tangherloni [3,*]

1 Department of Informatics, Systems and Communication, University of Milano-Bicocca, 20126 Milan, Italy; simone.spolaor@unimib.it (S.S.); daniela.besozzi@unimib.it (D.B.)
2 Weatherall Institute of Molecular Medicine, Radcliffe Department of Medicine, University of Oxford, Oxford OX3 9DU, UK
3 Department of Human and Social Sciences, University of Bergamo, 24129 Bergamo, Italy; paolo.cazzaniga@unibg.it
4 Bicocca Bioinformatics Biostatistics and Bioimaging Centre (B4), 20854 Vedano al Lambro, Italy; m.s.nobile@tue.nl
5 SYSBIO/ISBE.IT Centre for Systems Biology, 20126 Milan, Italy
6 Department of Industrial Engineering & Innovation Sciences, Eindhoven University of Technology, 5612 AZ Eindhoven, The Netherlands
7 Department of Radiology, University of Cambridge, Cambridge CB2 0QQ, UK; lr495@cam.ac.uk
8 Cancer Research UK Cambridge Centre, University of Cambridge, Cambridge CB2 0RE, UK
* Correspondence: simone.riva@imm.ox.ac.uk (S.G.R.); andrea.tangherloni@unibg.it (A.T.)

Abstract: Several software tools for the simulation and analysis of biochemical reaction networks have been developed in the last decades; however, assessing and comparing their computational performance in executing the typical tasks of computational systems biology can be limited by the lack of a standardized benchmarking approach. To overcome these limitations, we propose here a novel tool, named SMGen, designed to automatically generate synthetic models of reaction networks that, by construction, are characterized by relevant features (e.g., system connectivity and reaction discreteness) and non-trivial emergent dynamics of real biochemical networks. The generation of synthetic models in SMGen is based on the definition of an undirected graph consisting of a single connected component that, generally, results in a computationally demanding task; to speed up the overall process, SMGen exploits a main–worker paradigm. SMGen is also provided with a user-friendly graphical user interface, which allows the user to easily set up all the parameters required to generate a set of synthetic models with any number of reactions and species. We analysed the computational performance of SMGen by generating batches of symmetric and asymmetric reaction-based models (RBMs) of increasing size, showing how a different number of reactions and/or species affects the generation time. Our results show that when the number of reactions is higher than the number of species, SMGen has to identify and correct a large number of errors during the creation process of the RBMs, a circumstance that increases the running time. Still, SMGen can generate synthetic models with hundreds of species and reactions in less than 7 s.

Keywords: synthetic models; reaction-based models; biochemical networks; systems biology

check for updates

Citation: Riva, S.G.; Cazzaniga, P.; Nobile, M.S.; Spolaor, S.; Rundo, L.; Besozzi, D.; Tangherloni, A. SMGen: A Generator of Synthetic Models of Biochemical Reaction Networks. *Symmetry* **2022**, *14*, 119. https://doi.org/10.3390/sym14010119

Academic Editors: Miroslav Miletín and György Keglevich

Received: 29 July 2021
Accepted: 3 December 2021
Published: 10 January 2022

1. Introduction

Systems biology is a multidisciplinary research field that combines mathematical, computational, and experimental expertise to understand and predict the behaviour of complex biological systems [1,2]. Among the different formalisms that can be used to describe intracellular processes, reaction-based models (RBMs) [3–5] are the most suitable for obtaining a detailed comprehension of the mechanisms that control the emergent behaviour of the system under analysis [4]. The analysis of RBMs can be used to drive the design of focused lab experiments; to this aim, computational tasks such as parameter estimation (PE), sensitivity analysis (SA), and parameter sweep analysis (PSA) are generally applied [1,5–7]. Unfortunately, these computational tasks require the execution of huge amounts of simulations, so that the capabilities of biochemical simulators running on central processing units (CPUs) (see, e.g., [8–10]) can be easily overtaken. Thus, several simulators exploiting graphics processing units (GPUs) have been lately introduced to reduce the running times (see, e.g., [11–19]).

A crucial point, whenever new simulators are designed and implemented, regards the evaluation of their computational performance and their efficiency in executing the aforementioned demanding tasks. In this context, RBMs represent a key means as they can be exploited to run both stochastic simulation algorithms and (deterministic) numerical integration methods. Although there are more than 1000 models of real biochemical systems publicly available on BioModels [20,21], some of which characterized by hundreds of species and reactions, the majority of these models do not follow the law of mass-action. In addition, these models do not have the structural characteristics (e.g., the number of species and the number of reactions) necessary to perform a fair comparison among the simulators under different scenarios. Only a limited number of RBMs is present in the literature (e.g., signal transduction pathways [22–24] or metabolic pathways [25]). Thus, the lack of detailed RBMs, especially those characterized by hundreds or thousands of reactions and molecular species, hampers the possibility of performing a thorough analysis of the performance of these simulators.

The computational performance of several GPU-powered tools has already been assessed using randomly generated synthetic RBMs [13,18,19]. However, only a few generators of biochemical models have been proposed so far, hindering the possibility of having a common and well-defined benchmarking approach. For instance, Komarov et al. [13,14] developed a tool to generate synthetic networks, which was then used to test the performance of their GPU-based simulators. Given the number of reactants, the type of reactions to be included in the RBM, and the total number of reactions, they generated synthetic RBMs by exploiting a hash table to avoid duplicates. The tool was then modified by randomly sampling the values of the initial concentrations of the species from a uniform distribution and the kinetic constants from a logarithmic distribution [18]. Another known and established model generator is the reaction mechanism generator (RMG) [26], which was specifically developed to create synthetic chemical processes. RMG exploits an extensible set of 45 reaction families to generate elementary reactions from chemical species, while the reaction rates are estimated using a database of known rate rules and reaction templates. RMG relies on graphs to represent the chemical structures and trees to represent thermodynamic and kinetic data. Due to its peculiarities, RMG was used to, e.g., automatically create kinetic models for the conversion of bio-oil to syngas through gasification [27]. Finally, other tools, such as Moleculizer [28], were introduced for the generation of reaction systems to obtain a deeper understanding of transduction networks.

Despite the efforts done to automatically define synthetic models, all these generators share a common drawback, that is, they have a limited flexibility and can generate only a restricted set of biochemical networks and processes. As a matter of fact, the existing methods generally do not perform any check on the generated models, while some of them can only be used to create a restricted set of models (e.g., RMG can generate only elementary reactions). Moreover, in the case of the approach proposed by Komarov et al., even though it can generate first- and second-order reactions, including degradation and production reactions—which are the most common reactions in real biochemical networks—the tool is not publicly available.

Considering the impelling necessity of defining a common benchmarking approach that allows for fairly evaluating and comparing different simulation approaches [29], we propose here a novel tool, named SMGen, designed to automatically generate synthetic biological networks codified as RBMs, which allow to obtain non-trivial dynamics. It is worth mentioning that we were mainly interested in the generation of RBMs suitable for the analysis and comparison of the performance of biochemical simulators. An example of such RMBs is a model composed of plausible reactions (e.g., transformation, production, and degradation reactions, which are common in real biochemical networks) leading to non-trivial dynamics. For instance, in the case of a dynamics that instantly exhausts all reactants, some of the most advanced integration algorithms are able to simulate such stable and/or flat dynamics in just one computation step, thus hampering the possibility of a fair comparison among the different simulation approaches.

In addition, SMGen overcomes the existing approaches and tools, which generally do not allow for using different probability distributions for the initialization of the species amounts and the kinetic constants. The possibility of exploiting different probability distributions for the initialization of these parameters is an important feature offered by SMGen, since the effort required for the simulation of a model can drastically vary according to its initial parameterization (i.e., species amounts and kinetic constants).

SMGen adheres to well-defined structural characteristics based on graph theory and linear-algebra properties; in particular, it exploits the definition of an undirected graph with a single connected component, which makes the whole generation process a computationally demanding task. To overcome this limitation, on the one hand, SMGen internally codifies all data structures by means of sparse matrices as well as ad-hoc structures specifically designed to avoid worthless values that would increase the running time required to generate RBMs. On the other hand, SMGen is able to drastically reduce the computational time by exploiting a main–worker paradigm used to distribute the overall generation process of RBMs onto multi-core CPUs. We show that SMGen can create, in less than 7 s, synthetic RBMs with hundreds of chemical species and molecular reactions, whose dynamics exhibit non trivial characteristics of real biochemical networks. Among the different features provided by SMGen, it allows for easily generating both symmetric and asymmetric RBMs: symmetric RBMs are composed of a number of species equal to the number of reactions, while, in asymmetric RBMs, the number of species can be lower than the number of reactions or vice-versa. From a computational point of view, the concept of symmetry is crucial in the analysis of complex networks to measure their information and entropy [30]. In addition, considering that every RBM can be converted into a corresponding system of coupled ordinary differential equations (ODEs), studying the symmetries of the system of ODEs can reveal the intrinsic properties of the system of interest. Ohlsson et al. pointed out that an alternative analysis of the system of ODEs can be carried out by considering the symmetries of the system solutions, aiming at formalizing the structures and behaviour of the underlying dynamics of biological systems [31]. Moreover, the possibility of evaluating GPU-powered simulators using symmetric and asymmetric RBMs is fundamental to understand their performance under different conditions. Indeed, a fair comparison would allow the user to select the best simulator based on characteristics of the RBM that has to be analysed. Thanks to its features and efficiency, SMGen was used to generate the synthetic RBMs necessary to realize a thorough comparison of the performance of different meta-heuristics in solving the PE problem of biochemical networks [3,5], which is one of the most common and difficult computational issues in systems biology. The outcome of such analyses showed that some well-known and widely used meta-heuristics, generally able to outperform all competitors in the optimization of benchmark functions, obtained poor performance when used to solve the PE problem. Moreover, SMGen was exploited to generate symmetric and asymmetric RBMs required for the in-depth analyses and comparisons of the computational performance of different biochemical simulators, and to highlight their peculiarities [19]. These analyses allowed for determining the best simulator to employ under specific conditions, such as the size of the RBM and the total number of simulations to perform. SMGen can also be used for the generation of biochemical networks to investigate the performance of approaches specifically designed to tackle other well-known and computationally demanding tasks in systems biology (e.g., SA and PSA [1,7]).

SMGen allows also for exporting the generated RBMs into the Systems Biology Markup Language (SBML) [32], Version 4 Level 2, and into the BioSimWare standard [33], which is used by different GPU-powered simulators. Thus, we designed and developed SMGen to be a unifying, user-friendly, and standalone tool freely accessible to the systems biology community. The RBMs can be easily generated by using the provided user-friendly graphical user interface (GUI), which was designed to help the users in setting all the parameters required to generate the desired RBMs.

The manuscript is structured as follows. Section 2 describes the mathematical formalism of RBMs, as well as the structural characteristics that must be complied to generate synthetic biological networks. In addition, we provide all the algorithms and details at the basis of SMGen. Section 3 shows the experimental results achieved by SMGen. Finally, a discussion and conclusive remarks are provided in Section 4.

2. Materials and Methods

2.1. Reaction-Based Models

An RBM is defined by specifying the set $\mathcal{S} = \{S_1, \ldots, S_N\}$ of N molecular species, and the set $\mathcal{R} = \{R_1, \ldots, R_M\}$ of M biochemical reactions that describe the interactions among the species appearing in $\mathcal{S}$. Each reaction R_i, with $i = 1, \ldots, M$, is defined as:

$$R_i : \sum_{j=1}^{N} a_{ij} S_j \xrightarrow{k_i} \sum_{j=1}^{N} b_{ij} S_j, \tag{1}$$

where a_{ij} and $b_{ij} \in \mathbb{N}$ are the stoichiometric coefficients, and $k_i \in \mathbb{R}^+$ is the kinetic constant associated with R_i. The stoichiometric coefficients specify how many molecules of species S_j, with $j = 1, \ldots, N$, appear either as reactants or products in reaction R_i. Note that some species might not appear in a reaction, so that the corresponding stoichiometric coefficient will be equal to 0. The order of a reaction is equal to the total number of molecules (of the same or different species) that appear as reactants in that reaction.

Each RBM can be written in the compact matrix-vector form $\mathbf{AS} \xrightarrow{\mathbf{K}} \mathbf{BS}$, where $\mathbf{S} = [S_1 \cdots S_N]^\top$ is the N-dimensional column vector of the molecular species, $\mathbf{K} = [k_1 \cdots k_M]^\top$ is the M-dimensional column vector of the kinetic constants, while $\mathbf{A}, \mathbf{B} \in \mathbb{N}^{M \times N}$ are the stoichiometric matrices, whose non-negative elements $[A]_{i,j}$ and $[B]_{i,j}$ correspond to the stoichiometric coefficients a_{ij} and b_{ij} of the reactants and products of the reactions, respectively.

Starting from an RBM and assuming the law of mass-action [34–36], the system of coupled ODEs corresponding to the RBM can be derived as follows:

$$\frac{d\mathbf{X}}{dt} = (\mathbf{B} - \mathbf{A})^T [\mathbf{K} \circ \mathbf{X}^{\mathbf{A}}], \tag{2}$$

where each ODE describes the variation in time of a species' concentration. In Equation (2), the N-dimensional vector $\mathbf{X} = [X_1 \cdots X_N]$ represents the concentration values of species $S_1, \ldots, S_N$, while $\mathbf{X}^{\mathbf{A}}$ is the vector-matrix exponentiation form [34]; the symbol $\circ$ denotes the entry-by-entry matrix multiplication (Hadamard product).

2.2. SMGen

In order to generate synthetic models of biochemical networks, SMGen complies with specific structural characteristics that the RBMs have to satisfy, that is:

- *System connectivity*: a biochemical network can be represented as an undirected graph with a single connected component, where the nodes represent the molecular species, and the edges correspond to the species interactions. This representation easily allows for ensuring the system connectivity, a property that is strictly required to guarantee that each species $S_j \in \mathcal{S}$, with $j = 1, \ldots, N$, will be involved in at least one reaction $R_i \in \mathcal{R}$, with $i = 1, \ldots, M$.
  To be more precise, as a first step, an undirected graph with a single connected component is built. This undirected graph is randomly generated by using $N - 1$

edges and by taking into account the maximum number of reactants and products, obtaining a connected biochemical reaction network. It is worth noting that a graph with a single connected component can be built if and only if the following condition is met:

$$(max_{num_r} + max_{num_p}) \times M > N,$$

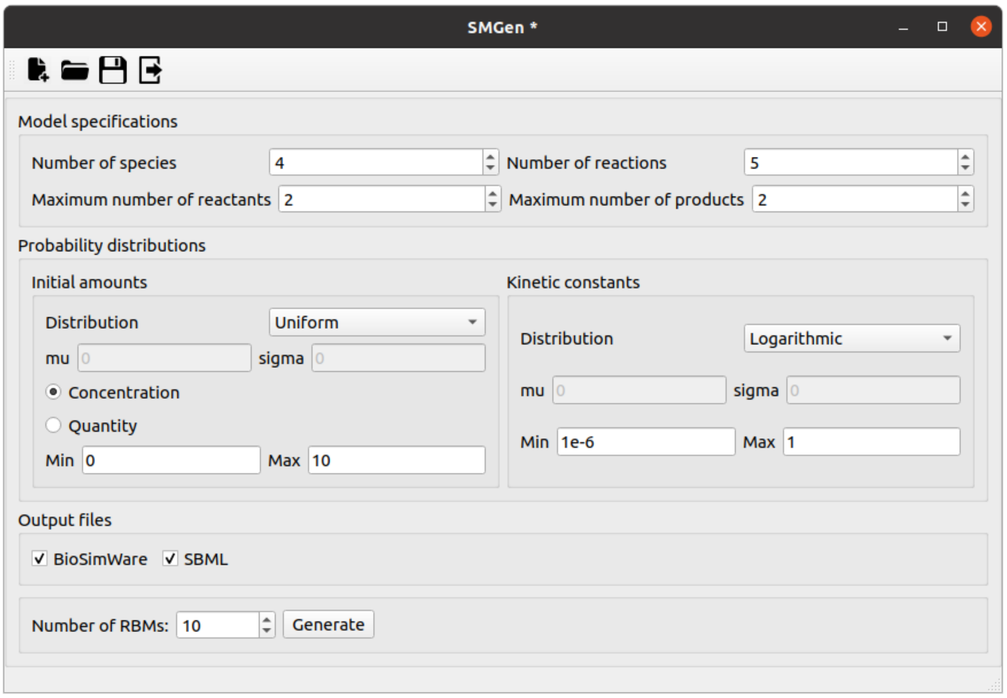
where max_{num_r} and max_{num_p} are the maximum number of reactants and products, respectively. As a second step, starting from the initial undirected graph, the stoichiometric matrices are generated. The stoichiometric matrices can be viewed and treated as a Petri net [37,38] that, in turn, can be considered as a bipartite graph. Then, the stoichiometric matrices are randomly updated by adding and removing connections among the species, always taking into consideration the maximum number of reactants and products. Note that the initial connections, which correspond to the edges of the initial undirected graph, are never removed to ensure that all the species are involved in at least one reaction, maintaining the whole biochemical network connected. We designed an algorithm (see Algorithm A1) that builds the graph, composed of a single connected component, with the minimum number of edges that are needed to connect all the nodes.

- *Maximum number of reactants and products*: for each reaction $R_i \in \mathcal{R}$, with $i = 1, \ldots, M$, the number of reactants and the number of products cannot be arbitrarily large but has to be lower than or equal to a user-defined value (i.e., max_{num_r} and max_{num_p}). Stated otherwise, the maximum order of the generated reactions should be fixed. SMGen does not explicitly account for conservation conditions during the generation of the RBMs; however, it can generate reactions that are akin to, e.g., protein modifications or conformational changes, thus resulting in two biochemical species whose sum is constant during the simulation.
- *Linear independence*: to ensure that each reaction R_i, with $i = 1, \ldots, M$, is endowed with plausible characteristics of a real biochemical reaction, the vectors of the stoichiometric coefficients of the reactants and products involved in R_i must be linearly independent. An example of an unrealistic reaction consists in increasing or decreasing the amount of a species starting from one molecule of the species itself, e.g., $R_i : S_j \xrightarrow{k_i} \alpha S_j$, with $\alpha > 1$. Since the linear independence is evaluated between the reactants and the products of each reaction, it is related to the number of species and reactions. Thus, it could happen that when the number of species is much higher than the number of reactions it is not possible to build a graph with a single connected component. In such a case, duplicated reactions or linear-dependent vectors between reactants and products of some reactions will occur. On the contrary, it is always possible to build a graph, composed of a single connected component, when the number of reactions is much higher than the number of species.
- *Reaction discreteness*: each reaction R_i, with $i = 1, \ldots, M$, must appear only once in the network; that is, duplicated reactions are not allowed.

SMGen is provided with a user-friendly GUI (see Figure 1) that allows the user to easily set up all the parameters required to generate the desired synthetic RBMs:

- the number of species N and the number of reactions M;
- the maximum number of reactants and products max_{num_r} and max_{num_p} that might appear in any reaction;
- the probability distribution $\mathcal{D}_s$ that is used to initialize the species amounts (to be chosen among uniform, normal, logarithmic, or log-normal distributions). Note that all species amounts are initialized using the same distribution probability;
- the minimum and maximum values min_s and max_s for the initial species amounts (to be specified either as number of molecules or concentrations);
- the probability distribution $\mathcal{D}_r$ that is used to set the values of the kinetic constants (to be chosen among uniform, normal, logarithmic, or log-normal distributions). Note that all kinetic constants are generated using the same distribution probability;

- the minimum and maximum values min_r and max_r for the kinetic constants;
- the total number of RBMs that the user wants to generate;
- the output format file to export the generated RBMs (i.e., BioSimWare [33] and SBML [32]);
- the mean and standard deviation values μ_s and σ_s for the initial amounts—as well as the mean and standard deviation values μ_r and σ_r for the kinetic constants—must also be provided if the normal or log-normal distributions are selected.


SMGen *

Model specifications

Number of species: 4
Number of reactions: 5
Maximum number of reactants: 2
Maximum number of products: 2

Probability distributions

Initial amounts
Distribution: Uniform
mu: 0
sigma: 0
● Concentration
○ Quantity
Min: 0
Max: 10

Kinetic constants
Distribution: Logarithmic
mu: 0
sigma: 0
Min: 1e-6
Max: 1

Output files
✓ BioSimWare ✓ SBML

Number of RBMs: 10

Figure 1. Graphical user interface of SMGen. The user can set all the parameters to generate the desired RBMs, i.e., number of species and reactions, maximum number of reactants and products, probability distribution for the initial amounts and kinetic constants, and the output format file (i.e., BioSimWare, SBML).

Figure 2 shows a high-level scheme of the proposed implementation of SMGen, which exploits the main–worker paradigm to speed up the generation of the RBMs [39]. The user can specify the number of processes P—otherwise automatically set to the minimum value 3—which are used as follows:

- Proc$_1$ manages the GUI;
- Proc$_2$ is the main process that orchestrates the computation;
- Proc$_p$, with $p = 3, \ldots, P$, are the worker processes.

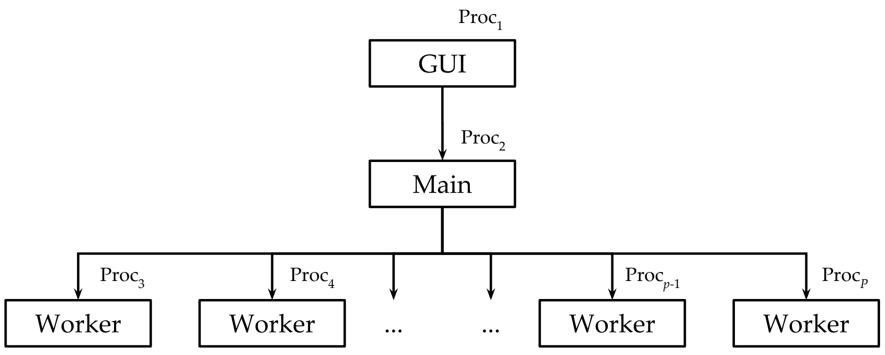


Figure 2. Scheme of the main–worker implementation of SMGen. The main process (Proc$_2$) orchestrates all the available workers (Proc$_p$, with $p = 3, \ldots, P$), which generate the RBMs in a distributed computing fashion.

The whole functioning of SMGen can be summarized as follows:

- the user interacts with the GUI, managed by Proc$_1$, to fill in all the required values for the parameters necessary to create the RBMs;
- Proc$_1$ sends the values of all parameters to the main process (Proc$_2$), which allocates the resources and distributes the work to the workers (Proc$_p$, with $p = 3, \ldots, P$);
- each worker (Proc$_p$, with $p = 3, \ldots, P$) generates an RBM. As soon as a worker terminates its execution, it communicates to the main process that the RBM has been created. If necessary, the main process assigns the generation of other RBMs to idle workers. When all the required RBMs are obtained, the workers enter in the death state, while the main process waits for further instructions from Proc$_1$.

The workflow of each worker consists in 9 different phases, in which a specific algorithm is executed (see Figure 3).

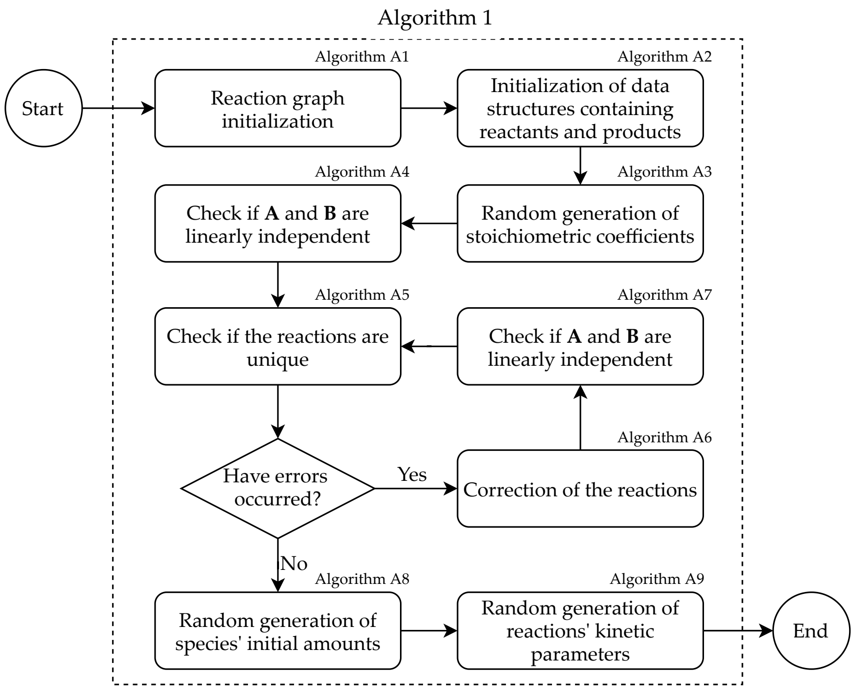

Algorithm 1
Start
Algorithm A1
Reaction graph initialization
Algorithm A2
Initialization of data structures containing reactants and products
Algorithm A3
Random generation of stoichiometric coefficients
Algorithm A4
Check if **A** and **B** are linearly independent
Algorithm A5
Check if the reactions are unique
Algorithm A7
Check if **A** and **B** are linearly independent
Have errors occurred?
Yes
Algorithm A6
Correction of the reactions
No
Algorithm A8
Random generation of species' initial amounts
Algorithm A9
Random generation of reactions' kinetic parameters
End

Figure 3. Workflow of a single worker execution. First, the graph of the reactions is randomly initialized and then converted into the data structures used to store the reactants and products. Second, the stoichiometric coefficients are randomly generated and the consistency of the reactants and products is verified. Third, the initial amounts of the species and the kinetic parameters of the reactions are randomly generated using the probability distributions specified by the user.

The pseudo-code reported in Algorithm 1 briefly summarizes all the steps required to generate a single RBM; the pseudo-code of the procedures invoked within Algorithm 1 are reported in Appendix A. For the sake of clarity, Table 1 lists the symbols used in the following description and in the pseudo-codes.

Algorithm 1 SMGen: workflow of a single worker execution.

```
function GENERATOR($M, N, max_{num_p}, max_{num_r}, \mathcal{D}_s, \mathcal{D}_r, min_s, max_s, min_r, max_r, \mu_s, \sigma_s, \mu_r, \sigma_r$)
    ## Algorithm A1
    $\mathbf{G} \leftarrow$ GRAPH_GEN($N$)
    ## Algorithm A2
    $\mathbf{A}, \mathbf{B} \leftarrow$ STOICH_MATRICES_GEN($M, N, \mathbf{G}$)
    $AIJ[\cdot], BIJ[\cdot] \leftarrow [\,]$                                  ▷
    for $i = 1$ to $M$ do                                                     ▷
        for $j = 1$ to $N$ do                                                 ▷
            if $\mathbf{A}[i, j] == 1$ then                                   ▷
                $AIJ \leftarrow AIJ \odot \langle i, j \rangle$               ▷
            if $\mathbf{B}[i, j] == 1$ then                                   ▷
                $BIJ \leftarrow BIJ \odot \langle i, j \rangle$               ▷
    ## Algorithm A3
    $\mathbf{A}, \mathbf{B} \leftarrow$ STOICH_COEFFICIENTS_GEN($\mathbf{A}, \mathbf{B}, M, N, max_{num_r}, max_{num_p}, AIJ, BIJ$)
    ## Algorithm A4
    $err_{LinDep} \leftarrow$LINEAR_INDEPENDENCE_MATRIX($\mathbf{A}, \mathbf{B}, M$)
    ## Algorithm A5
    $err_{Repeat} \leftarrow$UNIQUE_REACTIONS($\mathbf{A}, \mathbf{B}, M$)
    while $err_{LinDep} \wedge err_{Repeat}$ are not empty do
        $rows_{Err} \leftarrow unique(err_{LinDep} \odot err_{Repeat})$
        ## Algorithm A6
        $\mathbf{A}, \mathbf{B} \leftarrow$ CORRECTION_REACTIONS($\mathbf{A}, \mathbf{B}, rows_{Err}, AIJ, BIJ, max_{num_r}, max_{num_p}$)
        ## Algorithm A7
        $err_{LinDep} \leftarrow$ LINEAR_INDEPENDENCE_REACTION($\mathbf{A}, \mathbf{B}, rows_{Err}$)
        ## Algorithm A5
        $err_{Repeat} \leftarrow$UNIQUE_REACTIONS($\mathbf{A}, \mathbf{B}, M$)
    ## Algorithm A8
    $\mathbf{M_0} \leftarrow$ AMOUNTS_GEN($N, \mathcal{D}_s, min_s, max_s, \mu_s, \sigma_s$)
    ## Algorithm A9
    $\mathbf{K} \leftarrow$ KINETIC_CONSTANTS_GEN($M, \mathcal{D}_r, min_r, max_r, \mu_r, \sigma_r$)
```

▷ $\rightarrow$ the instructions shown in lines 6–12 are required to build the structure of the initial graph of the reactions.

The steps performed by each worker to generate an RBM are the following:

1. Given the parameters provided by the user, the graph representing the species and their interactions is randomly initialized (line 3 of Algorithm 1; see Algorithm A1).
2. The adjacency matrix of the graph generated in Step 1 is converted into the stoichiometric matrices **A** and **B** (line 5 of Algorithm 1; see Algorithm A2). Note that the instructions in lines 6–17 of Algorithm 1 are required to build the data structure of the initial graph, which is then modified.
3. The stoichiometric coefficients are randomly generated (line 19 of Algorithm 1; see Algorithm A3).
4. For each reaction R_i, with $i = 1, \ldots, M$, the linear independence between the reactants and products is verified (line 21 of Algorithm 1; see Algorithm A4).
5. The uniqueness of each reaction in the RBM is verified (line 23 of Algorithm 1; see Algorithm A5).
6. Any error in the RBM identified in the previous steps is corrected (line 27 of Algorithm 1; see Algorithm A6); the linear independence and the uniqueness of the reactions

in the modified RBM are iteratively verified (lines 29 and 31 of Algorithm 1; see Algorithms A5 and A7, respectively).

7. The initial amounts of the species are generated according to the chosen probability distribution (line 33 of Algorithm 1; see Algorithm A8). If a species appears only as a reactant in the whole RBM, its amount is set to remain unaltered. The rationale behind this is double: on the one hand, we avoid the possibility of creating reactions that could be applied at most once, which is a highly improbable situation in biological systems; on the other hand, we mimic the non-limiting availability of some biochemical resources, for instance, it might be used to reproduce the execution of in vitro experiments where some species are continually introduced in the systems to keep their amount constant [40].
8. The kinetic constants of the reactions are generated according to the chosen probability distribution (line 35 of Algorithm 1; see Algorithm A9).

SMGen was developed using the Python programming language and exploiting `mpi4py` [41], which provides bindings of the message passing interface (MPI) specifications for Python to leverage multi-core CPUs [42]. The open-source code of SMGen is available on GitLab (https://gitlab.com/sgr34/smgen, accessed on 2 December 2021) under the GNU GPL-3 license. In addition, it can be easily installed using the Python package installer pip (https://pypi.org/project/smgenerator, accessed on 2 December 2021).

Table 1. List of symbols used in the pseudo-code of algorithms at the basis of SMGen.

Symbol	**Description**
M	Number of reactions composing the RBM
N	Number of species involved in the RBM
max_{num_r}	Maximum number of the reactants
max_{num_p}	Maximum number of the products
$\mathbf{M_0}$	Array of the initial amounts
$\mathbf{K}$	Array of the kinetic constants
$\mathbf{A}$	Stoichiometric matrix of the reagents
$\mathbf{B}$	Stoichiometric matrix of the products
$\mathbf{G}$	Adjacency matrix of the graph of the reactions
$\mathcal{D}_s$	Probability distribution for the initial amounts
min_s	Minimum value of the initial amounts
max_s	Maximum value of the initial amounts
μ_s	Mean of the normal and log-normal distributions for the initial amounts
σ_s	Standard deviation of the normal and log-normal distributions for the initial amounts
$\mathcal{D}_r$	Probability distribution for the kinetic constants
min_r	Minimum value of the kinetic constants
max_r	Maximum value of the kinetic constants
μ_r	Mean of the normal and log-normal distributions for the initial amounts
σ_r	Standard deviation of the normal and log-normal distributions for the kinetic constants amounts
$\odot$	The concatenation operator

3. Results

We analysed the performance of SMGen regarding both its capability of creating RBMs characterized by non-trivial dynamics and the computational time required to generate sets of RBMs of increasing size. All tests were executed on a workstation equipped with an Intel Core i7-8750H CPU (clock 4.1 GHz), 16 GB of RAM, and a Samsung 970 EVO solid-state drive NVMe PCIe (up to 3400 MB/s and 1500 MB/s read and write speeds, respectively), running on Ubuntu 20.04 LTS.

As a first batch of tests, we generated 100 synthetic RBMs characterized by a limited number of reactions and species (4 and 5, respectively), and we analysed their characteristics and dynamics. We set to 3 both the maximum number of reactants max_{num_r} and products max_{num_p}. We sampled the initial amounts of species from a normal distribution with mean $\mu_s = 5$ and standard deviation $\sigma_s = 5$, considering a minimum value $min_s = 0$ and a maximum value $max_s = 10$. The kinetic constants were sampled from a logarithmic distribution with minimum value $min_r = 10^{-16}$ and maximum value $max_r = 10$.

Table 2 shows the list of reactions along with the kinetic constants of one of these 100 synthetic RBMs. Since the species X_0 appears only as a reactant, its amount will be kept constant during the simulation. The initial molecular amounts of all species—given as the number of molecules—are listed in Table 3. This small RBM includes the basic "cascade of reactions" structure typically observed in signalling pathways, starting from the source represented by species X_0 and X_4, toward species X_2 and X_3.

We simulated the dynamics of this RBM for 50 time steps (arbitrary units) using FiCoS [19], and the achieved dynamics are shown in Figure 4. These plots evidence that, although the RBM was randomly generated by SMGen, the simulated behaviour is non-trivial (e.g., the reactants were not instantly exhausted, resulting in flat dynamics). The capability of generating synthetic RBMs that exhibit non-trivial dynamics is fundamental to perform in-depth computational analyses and comparisons among any existing and novel simulators. Indeed, in the case of stable or flat dynamics, or when the overall behaviour of the network is extremely fast and all the reactants are immediately depleted, some of the most advanced integration algorithms can simulate the emergent dynamics in just one computation step [19]. In such a case, the computational performance of the simulation tools is only partially assessed, thus hindering a fair comparison among the tools.

Table 2. List of the reactions of an RBM with 4 reactions and 5 species generated by SMGen.

No.	Reagents	Products	Constant
R_1	$X_0 + X_4$	X_3	4.295×10^{-5}
R_2	X_4	$X_1 + 2X_2$	2.207×10^{-2}
R_3	X_4	$X_2 + X_4$	7.070×10^{-4}
R_4	$X_1 + X_4$	$X_2 + X_3$	4.613×10^{-2}

Table 3. Initial molecular amounts of the RBM generated by SMGen shown in Table 2.

Species	Initial Amount
X_0	4
X_1	8
X_2	7
X_3	8
X_4	1

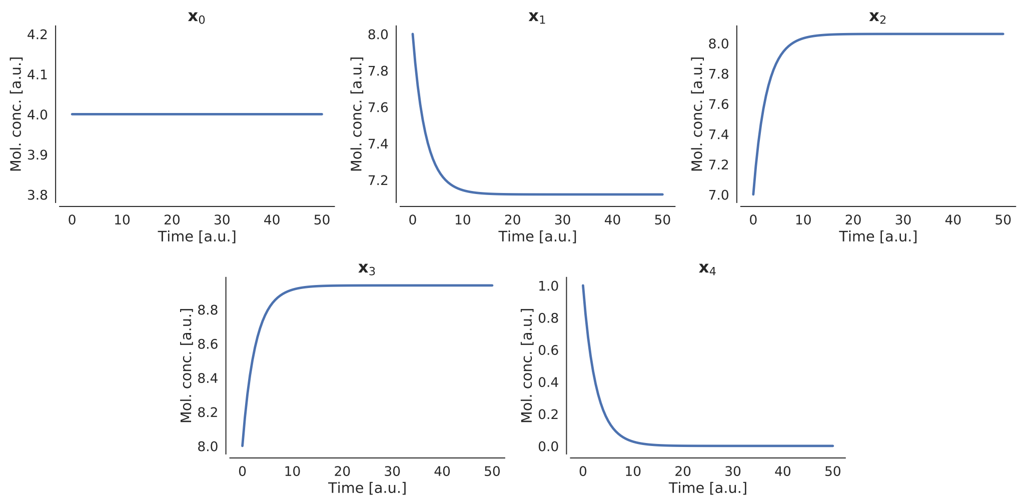


Figure 4. Dynamics of the species of the synthetic RBM generated by SMGen shown in Table 2.

As a second batch of tests, we evaluated the computational performance of SMGen exploiting the main–worker paradigm running on 4 distinct cores of the CPU. First, we considered the generation of symmetric RBMs with an increasing number of species and reactions (i.e., $M = N = 2^x$, with $x = 2, \ldots, 9$). The initial amounts and kinetic constants were randomly sampled from a uniform distribution with minimum values $min_s = min_r = 0$ and maximum values $max_s = max_r = 10$. We also varied the maximum numbers of reactants and products considering the set of values $\{2, 3, 4\}$, and setting $max_{num_r} = max_{num_p}$. For each of the resulting 24 parameters combinations, we created 100 RBMs to collect statistically sound results about the performance of SMGen. As described in Section 2, two kinds of error can occur during the generation of an RBM: a linear dependence between reactants and products, and duplicated reactions. Since the correction of these errors is one of the most time-consuming phases of SMGen, we separately measured the generation time, which indicates the running time spent by SMGen to generate an RBM, and the saving time, which refers to the writing operations on the solid-state drive. Figure 5 shows the average running time required by SMGen to generate and save an RBM. As expected, both the generation and the saving time increase along with the number of species and reactions of the RBM. Moreover, we observed that the maximum number of reactants and products have a slight impact on both the generation and the saving time; in most of the cases, increasing these values resulted in a higher running time.

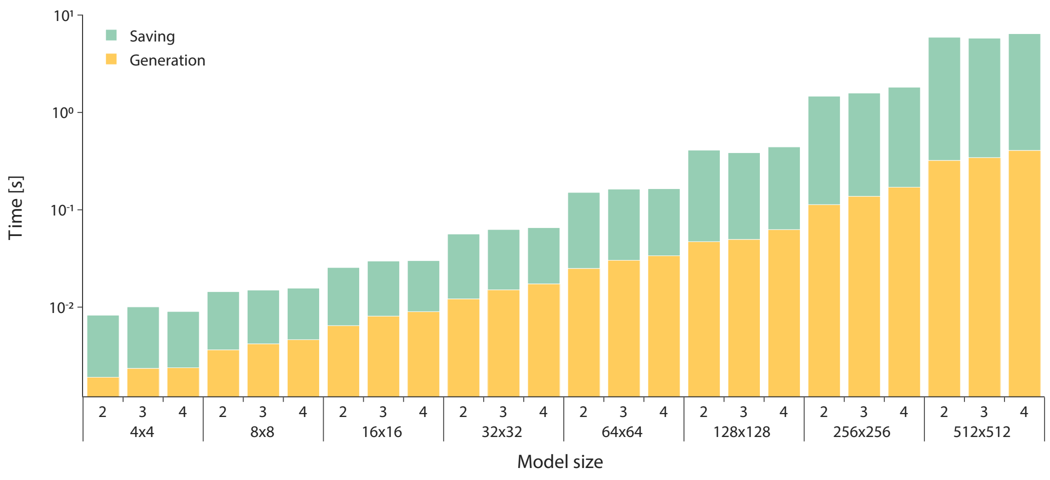
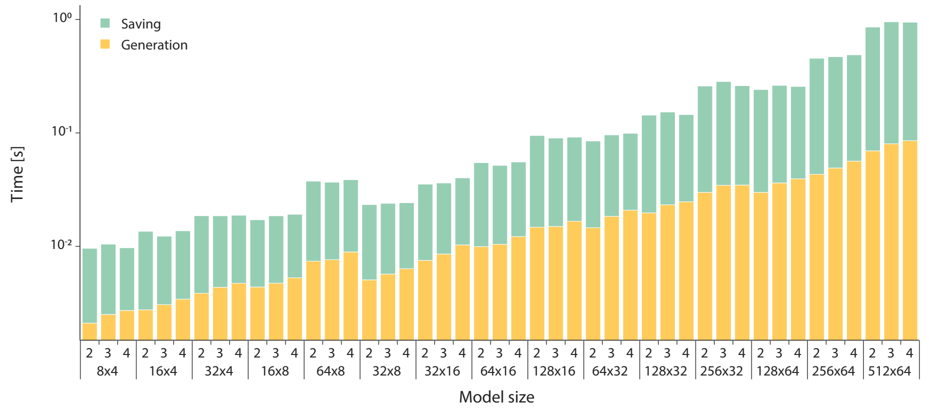


Figure 5. Average generation time (yellow bars) and average saving time (green bars) required by SMGen to generate a symmetric RBM. Note that the y-axis is in logarithmic scale.

Finally, we exploited SMGen for the creation of asymmetric RBMs to evaluate how a different number of species and reactions affects the running time. As in the case of symmetric RBMs, we measured both the generation time and the saving time. The asymmetric RBMs were created as follows:

- we set the number of species $N \in \{4, 8, 16, 32, 64\}$, and then we varied the number of reactions $M \in \{2N, 4N, 8N\}$;
- we set the number of reactions $M \in \{4, 8, 16, 32, 64\}$, and then we varied the number of species $N \in \{2M, 4M, 8M\}$;
- we varied both the maximum numbers of reactants max_{num_r} and products max_{num_p} in $\{2, 3, 4\}$.

In such a way, we obtained a total of 90 different combinations of the parameters to be tested (i.e., number of species, number of reactions, and maximum number of reactants and products); as in the previous tests, for each combination we generated 100 RBMs to collect statistically sound results. Figure 6 shows the average running time required to create RBMs with dimensions $N \times M$, highlighting once again that both the generation time and the saving time increase along with the size of the RBMs. As in the case of symmetric RBMs, we observed the same effect due to the maximum number of reactants and products allowed in the reactions. As expected, when there are more reactions than species (bottom panel in Figure 6), the generation times is higher than the opposite situation (top panel in Figure 6). This circumstance is due to the potential higher number of errors that SMGen has to identify and correct. Indeed, when $M \gg N$, the probability that repeated reactions are randomly generated is higher than the case when $N \gg M$, because the number of admissible reactions strictly depends on the number of species.



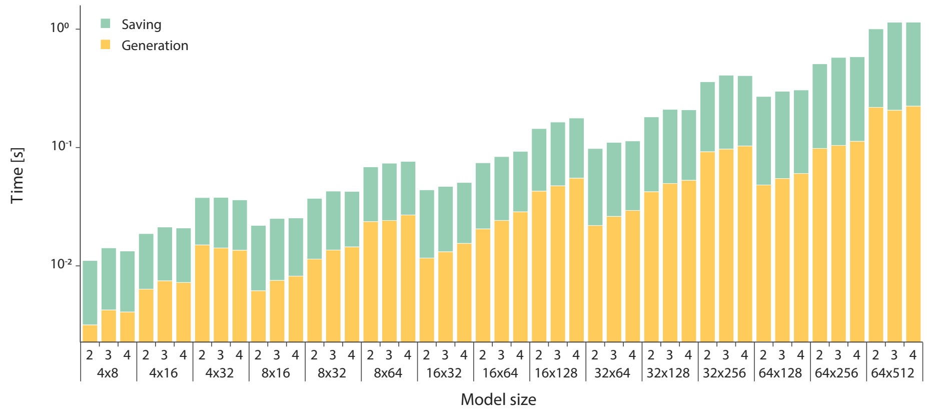


Figure 6. Average generation time (yellow bars) and average saving time (green bars) required by SMGen to generate an asymmetric RBM with more species than reactions (**top**), and with more reactions than species (**bottom**). Note that the y-axes are in logarithmic scale.

4. Conclusions

In this work we presented SMGen, a generator of synthetic RBMs displaying the characteristics of real biochemical networks, which can be exploited to create benchmarks for the evaluation of novel and existing simulators. In particular, SMGen is suitable for GPU-based simulators, since their performance can drastically change with the number of chemical species and reactions composing an RBM. Indeed, given the system of coupled ODEs corresponding to a RBM, the resolution of the system of ODEs can be performed in a parallel fashion, where each ODE is resolved by a thread. Since each ODE is related to a specific chemical species, the higher the number of species the higher the parallelization, which increases the computational performance of the simulator. On the contrary, considering that the number of the reactions composing the biological system is roughly related to the length of each ODE, in terms of the mathematical complexity, the higher the number of reactions the higher the number of operations that must be performed by each thread, leading to a higher running time [18,19].

SMGen was developed in Python and was designed to be a unifying, user-friendly, and standalone tool. In addition, SMGen exploits the main–worker paradigm to speed up the generation of RBMs; this was implemented using the `mpi4py` [41] library, where the first process manages the GUI, the second one is the main process, and all the other processes are the workers that generate the RBMs in a distributed computing fashion. Thanks to the GUI of SMGen, the user can easily set up all the parameters characterizing the required RBMs, e.g., the number of species and reactions, the maximum number of reactants and products per reaction, the probability distributions (uniform, normal, logarithmic, and log-normal) to generate the initial amounts of the species and the values of the kinetic constants associated with the reactions, and the output file format to save the RBMs. It is noteworthy that the performance of SMGen is not affected by the choice of different distributions for the sampling of the species amounts and kinetic constants. On the contrary, different distributions can drastically affect the dynamics of the model (see, e.g., the Brusselator model [38,43]). The analysis of the features of the generated models, according to different distributions, will be addressed in a future work.

We assessed the capabilities of SMGen for the creation of RBMs characterized by non-trivial behaviour, and we presented an example of a synthetic RBM together with the simulated dynamics. We also tested the computational performance of SMGen by generating batches of symmetric and asymmetric RBMs of increasing size, showing the impact of the number of reactions and species, and of the number of reactants and products per reaction, on the generation times. We observed that when the number of reactions is higher than the number of species, SMGen generally identifies and corrects high numbers of errors during the creation process of the RBMs, a circumstance that inevitably increases the overall running time.

As a future extension of this work, we plan to develop an application programming interface (API), so that SMGen can be seamlessly integrated into other processing pipelines, tools, and simulators. We are also developing a well-documented command-line interface, which will be released together with the API, to increase the user experience as well as the integrability with the existing approaches. Another feature that could be introduced is a module for the visualization of the networks, especially those that do not consist in a huge number of entities; indeed, the visualization of networks composed of thousands of species and reactions might not provide relevant information to the user. We will also introduce a new feature specifically developed to generate feedback loops in synthetic RBMs, exploiting the theory of Petri nets [37,38]. Feedback loops are fundamental elements of biological processes that lead to the establishment of oscillatory regimes and non-linear dynamics [22]. Moreover, we plan to develop a function to rescale the kinetic constants by taking into account the number of reactants involved in the reactions, as the order of magnitude and the value of each kinetic constant are related to the number of reactants composing the corresponding reaction. In the current version of SMGen, we assumed that all reactions follow the law of mass-action; we will implement additional kinetics (e.g., Michaelis–Menten and Hill kinetics [44,45]) in the future.

SMGen relies on graph theory and linear-algebra properties to comply with specific structural characteristics that are essential to create synthetic but real RBMs. Nevertheless, the current version of SMGen does not exploit network graphlets and motifs, which characterize families of real biological networks. Graphlets are induced sub-graphs appearing at any frequency because they are independent of the network's null model (i.e., a random graph model defined by a probability distribution), while network motifs are repeated sub-graphs appearing with a frequency higher than in random graphs and depending on the network's null model [46]. Graphlets have been used to analyse local network structures and to cluster different network types [46]. Probabilistic graphlets have also been developed to analyse the local wiring patterns of probabilistic networks. When applied to study biological networks, probabilistic graphlets resulted in a robust tool, which was able to capture the information underlying biological networks thanks to the capabilities of managing the low-signal topology information [47]. Graphlets can also be used to measure the structural similarity among large networks, calculating the graphlet degree distribution [48]. Motifs can be used to study autoregulation, single-input modules, dense overlapping regulons, and feedback loops as well as to reveal answers to many important biological questions [49,50]. The frequencies of the motifs were also exploited as classifiers for the selection of the network models [51]. Studying the motifs is also fundamental to understand the stability and robustness of the biological networks in response to small perturbations [52]. As a matter of fact, Prill et al. showed that the stability and robustness of a network is strictly related to the relative abundance of the motifs [52]. This result suggests that the structural organization of a biological network can be highly related to the dynamic properties of the small network motifs. We plan to extend SMGen to incorporate network graphlets and motifs during the generation of the RBMs, to obtain synthetic models with an improved biological significance.

In addition, we will modify the random generation of the kinetic constants to take into account the fact that different biological processes can operate on different time scales. In order to introduce this modification, as a first step, the reactions will be grouped into different families based on the biological process that they describe (e.g., protein or mRNA degradation, transcription rates, translation rates). Then, for each family of reactions, a different probability distribution will be used to sample the kinetic constants to reflect the time scale of the described biological process. We are also investigating the possibility of generating models characterized by specific emergent dynamical aspects (e.g., stable oscillatory regimes, state switching, and multi-stability [16]). We derived a set of heuristics to detect these conditions from the time-series, based on the combination of SMGen with GPU-powered stochastic simulators [16]. Still, regardless of the acceleration used, robustly detecting these phenomena is computationally challenging because some parameterizations might lead to very stiff models, which, in turn, might require a very long running time. Thus, we are investigating alternative approaches to guide the generation of synthetic models having such features. In particular, we are considering the possibility of providing a set of "functional modules", i.e, a group of already parameterized reactions implementing specific behaviours. These modules could be coerced into the synthetic models generated by SMGen, possibly leading to the emergence of the desired phenomena. Finally, we plan to include an initial check of the parameter values set by the user, based on some heuristics, to verify whether the RBMs can be actually generated as requested. This initial step will allow for avoiding worthless calculations and to suggest useful modifications of parameters to the user.

Author Contributions: S.G.R., P.C., M.S.N. and A.T. conceived and designed the tool. S.G.R. developed the tool. S.G.R., P.C. and A.T. conceived and designed the analyses. S.G.R. performed the analyses. P.C., D.B. and A.T. wrote the manuscript. S.G.R., M.S.N., S.S. and L.R. reviewed the manuscript. D.B., P.C., M.S.N. and A.T. supervised the whole work. All authors have read and agreed to published this version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The open-source code of SMGen is available on GitLab (https://gitlab.com/sgr34/smgen, accessed on 6 December 2021) under the GNU GPL-3 license. SMGen can be also installed using the Python package installer pip (https://pypi.org/project/smgenerator, accessed on 6 December 2021).

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Algorithms

We report here all the algorithms referring to the functions called by Algorithm 1, which represents the workflow of each worker process.

Algorithm A1 Random initialization of the graph of reactions.

```
1:  function GRAPH_GEN($N$)
2:      $\mathbf{G}[\cdot,\cdot], v[\cdot] \leftarrow 0$
3:      for $j = 1$ to $N$ do
4:          $v[j] \leftarrow j$
5:      $ind_1 \leftarrow random(1, len(v))$
6:      $ind_2 \leftarrow random(1, len(v))$
7:      $n_1, n_2 \leftarrow v[ind_1], v[ind_2]$
8:      $v \leftarrow delete(v[ind_1])$
9:      $v \leftarrow delete(v[ind_2])$
10:     $\mathbf{G}[n_1, n_2] \leftarrow 1$
11:     while $v$ is not empty do
12:         if $random \in \{0,1\} == 0$ then
13:             $ind \leftarrow random(1, len(v))$
14:             $k \leftarrow v[ind]$
15:             $v \leftarrow delete(v[ind])$
16:             $\mathbf{G}[n_1, k] \leftarrow 1$
17:             $n_2 \leftarrow k$
18:         else
19:             $ind \leftarrow random(1, len(v))$
20:             $k \leftarrow v[ind]$
21:             $v \leftarrow delete(v[ind])$
22:             $\mathbf{G}[k, n_2] \leftarrow 1$
23:             $n_1 \leftarrow k$
24:     return G
```

Algorithm A2 Conversion of the adjacency matrix **G** into the stoichiometric matrices **A** and **B**.

```
1:  function STOICH_MATRICES_GEN($M, N, \mathbf{G}$)
2:      $i \leftarrow 1$
3:      for $n_1 = 1$ to $N$ do
4:          for $n_2 = 1$ to $N$ do
5:              if $\mathbf{G}[n_1, n_2] == 1$ then
6:                  $\mathbf{A}[i, n_1] \leftarrow 1$
7:                  $\mathbf{B}[i, n_2] \leftarrow 1$
8:                  if $i == M$ then
9:                      $i \leftarrow 1$
10:                 else
11:                     $i \leftarrow i + 1$
12:     return A, B
```

Algorithm A3 Generation of the random stoichiometric coefficients.

```
1: function STOICH_COEFFICIENTS_GEN($\mathbf{A}, \mathbf{B}, M, N, max_{num_r}, max_{num_p}, AIJ, BIJ$)
2:     for $i = 1$ to $M$ do
3:         for $k = 0$ to $max_{num_r}$ do
4:             $coef \leftarrow random(0, max_{num_r})$
5:             $j \leftarrow random(1, N)$
6:             if $coef \neq 0$ & $A[i,\cdot] + coef - A[i,j] \leq max_{num_r}$ then
7:                 $\mathbf{A}[i,j] \leftarrow coef$
8:             else if $coef == 0$ & $\langle i,j \rangle \notin AIJ$ then
9:                 $\mathbf{A}[i,j] \leftarrow coef$
10:    for $i = 1$ to $M$ do
11:        for $k = 0$ to $max_{num_p}$ do
12:            $coef \leftarrow random(0, max_{num_p})$
13:            $j \leftarrow random(1, N)$
14:            if $coef \neq 0$ & $B[i,\cdot] + coef - B[i,j] \leq max_{num_p}$ then
15:                $\mathbf{B}[i,j] \leftarrow coef$
16:            else if $coef == 0$ & $\langle i,j \rangle \notin BIJ$ then
17:                $\mathbf{B}[i,j] \leftarrow coef$
18:    return $\mathbf{A}, \mathbf{B}$
```

Algorithm A4 Checking the linear independence between stoichiometric matrices.

```
1: function LINEAR_INDEPENDENCE_MATRIX($\mathbf{A}, \mathbf{B}, M$)
2:     $err_{LinDep} \leftarrow [\,]$
3:     for $i = 1$ to $M$ do
4:         if $\mathbf{A}[i,\cdot] \wedge \mathbf{B}[i,\cdot]$ are linearly dependent then
5:             $err_{LinDep} \leftarrow err_{LinDep} \odot i$
6:     return $err_{LinDep}$
```

Algorithm A5 Checking if the generated reactions are unique.

```
1: function UNIQUE_REACTIONS($\mathbf{A}, \mathbf{B}, M$)
2:     $\mathbf{AB}, \mathbf{ABs} \leftarrow [\,]$
3:     $err_{Repeat} \leftarrow [\,]$
4:     for $i = 1$ to $M$ do
5:         $\mathbf{AB}[i] \leftarrow \mathbf{A}[i,\cdot] \odot \mathbf{B}[i,\cdot]$
6:     for $i = 1$ to $M$ do
7:         if $\mathbf{AB}[i]$ is in $\mathbf{ABs}$ then
8:             $err_{Repeat} \leftarrow err_{Repeat} \odot i$
9:         else
10:            $\mathbf{ABs} \leftarrow \mathbf{ABs} \odot \mathbf{AB}[i]$
11:    return $err_{Repeat}$
```

Algorithm A6 Random correction of the repeated reactions.

```
1: function CORRECTION_REACTIONS($\mathbf{A}, \mathbf{B}, rows_{Err}, AIJ, BIJ, max_{num_r}, max_{num_p}$)
2:     for $i = 1$ to $len(rows_{Err})$ do
3:         $\mathbf{A}[rows_{Err}[i], \cdot] \leftarrow 0$
4:         $\mathbf{B}[rows_{Err}[i], \cdot] \leftarrow 0$
5:         if $rows_{Err}[i] \in AIJ[\cdot, 1]$ then
6:             for $k = 1$ to $len(AIJ)$ do
7:                 if $AIJ[k, 1] == rows_{Err}[i]$ then
8:                     $\mathbf{A}[AIJ[k, 1], AIJ[k, 2]] \leftarrow 1$
9:             for $c = 0$ to $max_{num_r}$ do
10:                $coef \leftarrow random(0, max_{num_r})$
11:                $col \leftarrow random(1, N)$
12:                if $coef \neq 0$ & $A[rows_{Err}[i], \cdot] + coef - A[rows_{Err}[i], col] \leq max_{num_r}$ then
13:                    $\mathbf{A}[rows_{Err}[i], col] \leftarrow coef$
14:                else if $coef == 0$ & $\langle rows_{Err}[i], col \rangle \notin AIJ$ then
15:                    $\mathbf{A}[rows_{Err}[i], col] \leftarrow coef$
16:        if $rows_{Err}[i] \in BIJ[\cdot, 1]$ then
17:            for $k = 1$ to $len(BIJ)$ do
18:                if $BIJ[k, 1] == rows_{Err}[i]$ then
19:                    $\mathbf{B}[BIJ[k, 1], BIJ[k, 2]] \leftarrow 1$
20:            for $c = 0$ to $max_{num_p}$ do
21:                $coef \leftarrow random(0, max_{num_p})$
22:                $col \leftarrow random(1, N)$
23:                if $coef \neq 0$ & $B[rows_{Err}[i], \cdot] + coef - B[rows_{Err}[i], col] \leq max_{num_p}$ then
24:                    $\mathbf{B}[rows_{Err}[i], col] \leftarrow coef$
25:                else if $coef == 0$ & $\langle rows_{Err}[i], col \rangle \notin BIJ$ then
26:                    $\mathbf{B}[rows_{Err}[i], col] \leftarrow coef$
27:    return $\mathbf{A}, \mathbf{B}$
```

Algorithm A7 Checking the linear independence between reactants or products.

```
1: function LINEAR_INDEPENDENCE_REACTION($\mathbf{A}, \mathbf{B}, rows_{Err}$)
2:     $err_{LinDep} \leftarrow [\,]$
3:     for $i = 1$ to $len(rows_{Err})$ do
4:         if $\mathbf{A}[rows_{Err}[i], \cdot] \wedge \mathbf{B}[rows_{Err}[i], \cdot]$ are linearly dependent then
5:             $err_{LinDep} \leftarrow err_{LinDep} \odot rows_{Err}[i]$
6:     return $err_{LinDep}$
```

Algorithm A8 Random initialization of the amounts of the species.

```
1: function AMOUNTS_GEN($N, \mathcal{D}_s, min_s, max_s, \mu_s, \sigma_s$)
2:     $M_0[\cdot] \leftarrow 0$
3:     if $dist$ is Uniform or Logarithmic then
4:         for $j = 1$ to $N$ do
5:             $\mathbf{M_0}[j] \leftarrow random(\mathcal{D}_s, min_s, max_s)$
6:     else
7:         for $j = 1$ to $N$ do
8:             $\mathbf{M_0}[j] \leftarrow random(\mathcal{D}_s, min_s, max_s, \mu_s, \sigma_s)$
9:     return $\mathbf{M_0}$
```

Algorithm A9 Random generation of kinetic constants of the reactions.

```
1: function KINETIC_CONSTANTS_GEN($M, \mathcal{D}_r, min_r, max_r, \mu_r, \sigma_r$)
2:     $\mathbf{K}[\cdot] \leftarrow 0$
3:     if $dist$ is Uniform or Logarithmic then
4:         for $i = 1$ to $M$ do
5:             $\mathbf{K}[i] \leftarrow random(\mathcal{D}_r, min_r, max_r)$
6:     else
7:         for $i = 1$ to $M$ do
8:             $\mathbf{K}[i] \leftarrow random(\mathcal{D}_r, min_r, max_r, \mu_r, \sigma_r)$
9:     return $\mathbf{K}$
```

Appendix B. Analysis of the Properties of the RBMs Generated by SMGen

The properties of the RBMs generated by SMGen were analysed by considering the network deficiency [53], the scale-freeness of the network [54], and the number of zero complexes, i.e., the null species used to denote degradation reactions or the influx of chemicals.

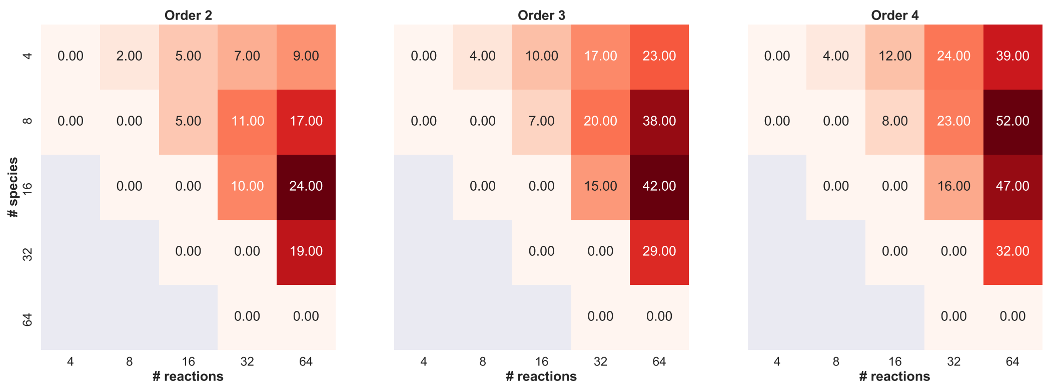
Network deficiency is an important structural attribute of a reaction network; indeed, this measure gives an indication of the independence of the reaction vectors. Specifically, the network deficiency is a non-negative integer index calculated considering three coefficients: the number of distinct complexes n, the number of linkage classes l, and the rank of the network r. In this notation, the complexes of a network are the entities that appear at the head and tail of each reaction (examples of distinct complexes are A, $2A$, $A + B$, etc.), while the linkage classes are the sets of complexes in the various "parts" composing the network, where complexes in a set are linked to each other, directly or indirectly, but they are not linked to any other complex in the network. Finally, the rank of the network is exactly the rank of its set of reaction vectors. To be more precise, the network has a rank r if there is a subset containing r linearly independent reaction vectors, but there is not any subset containing $r + 1$ linearly independent reaction vectors. The network deficiency is equal to $n - l - r$. It is worth noting that networks having a deficiency equal to zero are the ones where the reaction vectors are as independent as the partition of complexes into linkage classes will allow.

To perform such analysis, as a first step, we generated symmetric and asymmetric RBMs characterized by a number of species N and reactions M ranging from 4 to 64, by also varying max_{num_r} and max_{num_p} into $\{2, 3, 4\}$. Specifically, for each condition (i.e., N, M, max_{num_r}, and max_{num_p}), 100 RBMs were randomly created to collect statistically sound results. We limited this analysis to the models' size that can be generated with a single connected component (independently from the maximum number of reactants and products), i.e., with the following constraint:

$$(max_{num_r} + max_{num_p}) \times M > N.$$

Then, we calculated the network deficiency for each generated RBM, and, for each condition, we calculated the median of the network deficiency values. The heatmaps depicted in Figure A1 clearly show that the network deficiency increases when $M > N$. Moreover, the maximum number of reactants and products affects the network deficiency: the higher the maximum number of reactants and products, the higher the network deficiency. These results are coherent with the definition of network deficiency itself. As a matter of fact, the number of possible distinct complexes and the number of possible distinct linkage classes increase along with N and the maximum number of reactants and products, while the number of linearly independent reaction vectors is bounded by $\min(M, N)$. Finally, the values obtained from this analysis are congruent with those of real biochemical models, such as the Brussellator [38,43]) (5 species and 4 reactions) with deficiency $= 0$, the prokaryotes gene expression model [55] (5 species and 8 reactions) with deficiency $= 1$, the heat shock response in eukaryotes model (10 species and 17 reactions) with deficiency

= 5, the Ras/cAMP/PKA model [22] (33 species and 39 reactions) with deficiency = 7, and the human intracellular metabolic pathway in red blood cells [56] (114 species and 226 reactions) with deficiency = 13.



Order 2	4	8	16	32	64
4	0.00	2.00	5.00	7.00	9.00
8	0.00	0.00	5.00	11.00	17.00
16		0.00	0.00	10.00	24.00
32			0.00	0.00	19.00
64				0.00	0.00

Order 3	4	8	16	32	64
4	0.00	4.00	10.00	17.00	23.00
8	0.00	0.00	7.00	20.00	38.00
16		0.00	0.00	15.00	42.00
32			0.00	0.00	29.00
64				0.00	0.00

Order 4	4	8	16	32	64
4	0.00	4.00	12.00	24.00	39.00
8	0.00	0.00	8.00	23.00	52.00
16		0.00	0.00	16.00	47.00
32			0.00	0.00	32.00
64				0.00	0.00

Figure A1. Heatmaps showing the median value of the network deficiency calculated on RBMs generated by varying the number of species and reactions, and the maximum number of reactants and products.

Biological networks, particularly metabolic networks [57], often show scale-freeness properties [54]. In order to prove that SMGen generates biochemical reaction networks that are characterized by this property, we converted the bipartite graph defined by the stoichiometric matrices to a common interaction graph, where the nodes represent the chemical species, and the edges represent reactions involving the nodes either as reactants or products. Then, we analysed the structural properties of the generated network, and, in particular, we investigated whether the degree distribution follows a power law distribution, which is characteristic of scale-free graphs. Since the scale-free properties emerge only for large scale networks, we generated and analysed the interaction network induced by an RBM consisting in 10, 000 chemicals species involved in 10, 000 reactions. According to our results, the degree distribution seems to fit well with a power law (see Figure A2), confirming that SMGen can generate scale-free networks. It is worth noting that curated models of this size are seldom published, because both the initial amounts of the chemical species and the kinetic reaction rates would be difficult to collect, and, in any case, they would represent a challenge for biochemical simulation.

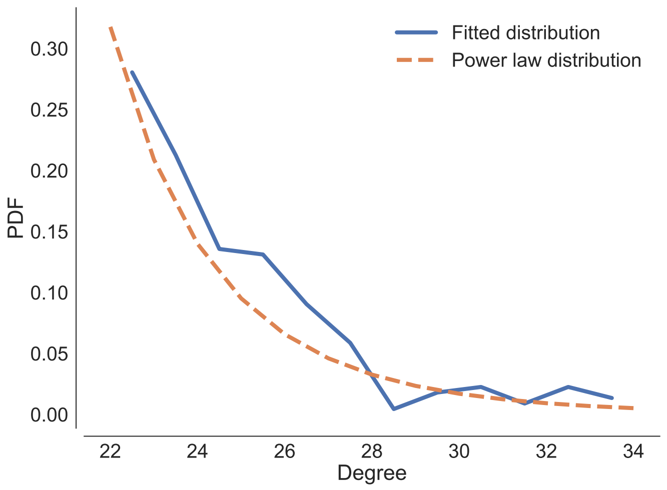


Figure A2. Degree distribution of the interaction network induced by an RBM randomly generated using SMGen. The discrete fitted distribution is shown using linearly spaced bins.

Finally, in order to evaluate the abundance of the zero complex appearing as reactant or product in a reaction, we used the same set of RBMs generated to calculate the network deficiency. As a first step, for each RBM we calculated the percentage of reactions where the zero complex appears either as a reactant or as a product. The heatmaps reported in Figure A3 clearly show that this value increases when $M >> N$. It is worth noting that the higher the maximum number of reactants and products for each reaction, the lower the percentage of reactions involving the zero complex. Overall, we observed that, at most, 2 out of 10 species were involved in one of such reactions. We investigated the abundance of the zero complex in real models. In the prokaryotes gene expression model [55], the zero complex appeared in 25% of the reactions, while it was present in 5.88% of the reactions composing the heat shock response model [58]. Thus, the values calculated on synthetic RBMs generated by SMGen are in agreement with those obtained from models of real biochemical networks.

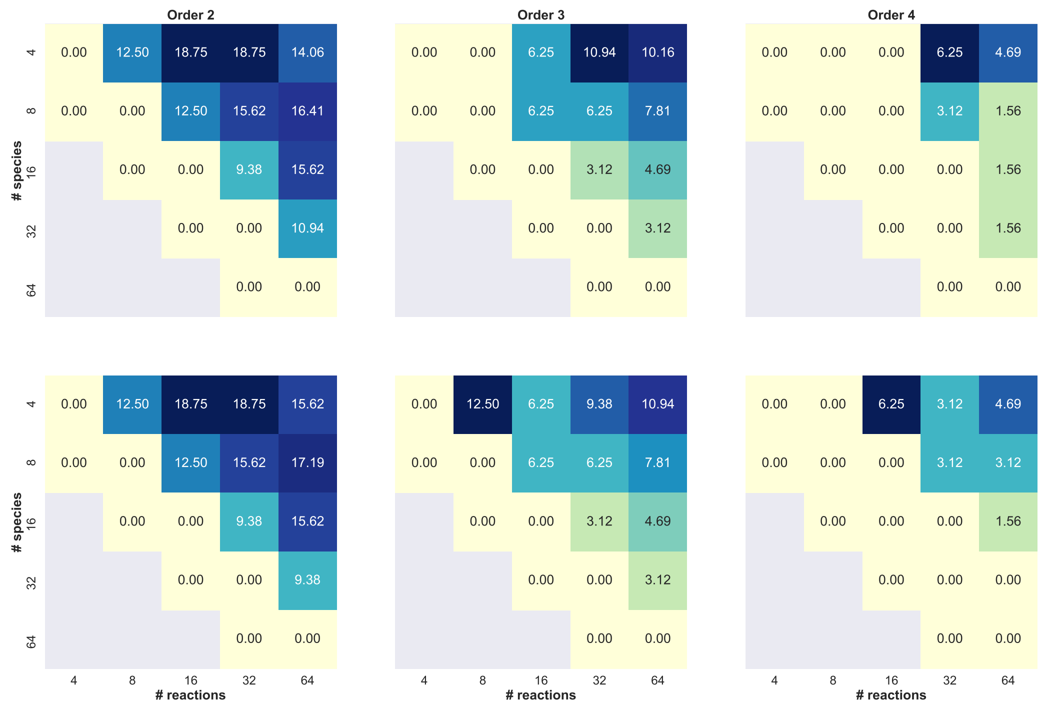


Figure A3. Heatmaps showing the median percentage of reactions where the zero complex appears as reactant (**top**) or product (**bottom**), calculated on RBMs generated by varying the number of species and reactions as well as the maximum number of reactants and products

References

1. Aldridge, B.B.; Burke, J.M.; Lauffenburger, D.A.; Sorger, P.K. Physicochemical modelling of cell signalling pathways. *Nat. Cell Biol.* **2006**, *8*, 1195–1203. [CrossRef]
2. Szallasi, Z.; Stelling, J.; Periwal, V. *System Modeling in Cellular Biology: From Concepts to Nuts and Bolts*; The MIT Press: Cambridge, MA, USA, 2006.
3. Nobile, M.S.; Tangherloni, A.; Rundo, L.; Spolaor, S.; Besozzi, D.; Mauri, G.; Cazzaniga, P. Computational Intelligence for Parameter Estimation of Biochemical Systems. In Proceedings of the 2018 IEEE Congress on Evolutionary Computation (CEC), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8.
4. Munsky, B.; Hlavacek, W.S.; Tsimring, L.S. *Quantitative Biology: Theory, Computational Methods, and Models*; MIT Press: Cambridge, MA, USA, 2018.
5. Tangherloni, A.; Spolaor, S.; Cazzaniga, P.; Besozzi, D.; Rundo, L.; Mauri, G.; Nobile, M.S. Biochemical parameter estimation vs. benchmark functions: A comparative study of optimization performance and representation design. *Appl. Soft Comput.* **2019**, *81*, 105494. [CrossRef]
6. Kitano, H. Systems biology: A brief overview. *Science* **2002**, *295*, 1662–1664. [CrossRef] [PubMed]
7. Chou, I.C.; Voit, E.O. Recent developments in parameter estimation and structure identification of biochemical and genomic systems. *Math. Biosci.* **2009**, *219*, 57–83. [CrossRef] [PubMed]

8. Somogyi, E.T.; Bouteiller, J.M.; Glazier, J.A.; König, M.; Medley, J.K.; Swat, M.H.; Sauro, H.M. libRoadRunner: A high performance SBML simulation and analysis library. *Bioinformatics* **2015**, *31*, 3315–3321. [CrossRef] [PubMed]
9. Hoops, S.; Sahle, S.; Gauges, R.; Lee, C.; Pahle, J.; Simus, N.; Singhal, M.; Xu, L.; Mendes, P.; Kummer, U. COPASI—A complex pathway simulator. *Bioinformatics* **2006**, *22*, 3067–3074. [CrossRef]
10. Moraru, I.I.; Schaff, J.C.; Slepchenko, B.M.; Blinov, M.; Morgan, F.; Lakshminarayana, A.; Gao, F.; Li, Y.; Loew, L.M. Virtual Cell modelling and simulation software environment. *IET Syst. Biol.* **2008**, *2*, 352–362. [CrossRef]
11. Li, H.; Petzold, L. Efficient parallelization of the stochastic simulation algorithm for chemically reacting systems on the graphics processing unit. *Int. J. High Perform. Comput. Appl.* **2010**, *24*, 107–116.
12. Zhou, Y.; Liepe, J.; Sheng, X.; Stumpf, M.P.; Barnes, C. GPU accelerated biochemical network simulation. *Bioinformatics* **2011**, *27*, 874–876. [CrossRef] [PubMed]
13. Komarov, I.; D'Souza, R.M.; Tapia, J. Accelerating the Gillespie τ-leaping method using graphics processing units. *PLoS ONE* **2012**, *7*, e37370.
14. Komarov, I.; D'Souza, R.M. Accelerating the Gillespie exact stochastic simulation algorithm using hybrid parallel execution on graphics processing units. *PLoS ONE* **2012**, *7*, e46693. [CrossRef]
15. Nobile, M.S.; Cazzaniga, P.; Besozzi, D.; Mauri, G. GPU-accelerated simulations of mass-action kinetics models with cupSODA. *J. Supercomput.* **2014**, *69*, 17–24. [CrossRef]
16. Nobile, M.S.; Cazzaniga, P.; Besozzi, D.; Pescini, D.; Mauri, G. cuTauLeaping: A GPU-powered tau-leaping stochastic simulator for massive parallel analyses of biological systems. *PLoS ONE* **2014**, *9*, e91963. [CrossRef] [PubMed]
17. Sumiyoshi, K.; Hirata, K.; Hiroi, N.; Funahashi, A. Acceleration of discrete stochastic biochemical simulation using GPGPU. *Front. Physiol.* **2015**, *6*, 42. [CrossRef] [PubMed]
18. Tangherloni, A.; Nobile, M.; Besozzi, D.; Mauri, G.; Cazzaniga, P. LASSIE: Simulating large-scale models of biochemical systems on GPUs. *BMC Bioinform.* **2017**, *18*, 246. [CrossRef] [PubMed]
19. Tangherloni, A.; Nobile, M.S.; Cazzaniga, P.; Capitoli, G.; Spolaor, S.; Rundo, L.; Mauri, G.; Besozzi, D. FiCoS: A fine-and coarse-grained GPU-powered deterministic simulator for biochemical networks. *PLoS Comput. Biol.* **2021**, *17*, e1009410. [CrossRef]
20. Glont, M.; Nguyen, T.V.N.; Graesslin, M.; Hälke, R.; Ali, R.; Schramm, J.; Wimalaratne, S.M.; Kothamachu, V.B.; Rodriguez, N.; Swat, M.J.; et al. BioModels: Expanding horizons to include more modelling approaches and formats. *Nucleic Acids Res.* **2018**, *46*, D1248–D1253. [CrossRef] [PubMed]
21. Malik-Sheriff, R.S.; Glont, M.; Nguyen, T.V.N.; Tiwari, K.; Roberts, M.G.; Xavier, A.; Vu, M.T.; Men, J.; Maire, M.; Kananathan, S.; et al. BioModels—15 years of sharing computational models in life science. *Nucleic Acids Res.* **2020**, *48*, D407–D415. [CrossRef]
22. Besozzi, D.; Cazzaniga, P.; Pescini, D.; Mauri, G.; Colombo, S.; Martegani, E. The role of feedback control mechanisms on the establishment of oscillatory regimes in the Ras/cAMP/PKA pathway in *S. cerevisiae*. *EURASIP J. Bioinform. Syst. Biol.* **2012**, *2012*, 10. [CrossRef] [PubMed]
23. Cazzaniga, P.; Nobile, M.S.; Besozzi, D.; Bellini, M.; Mauri, G. Massive exploration of perturbed conditions of the blood coagulation cascade through GPU parallelization. *BioMed Res. Int.* **2014**, *2014*, 863298. [CrossRef] [PubMed]
24. Pescini, D.; Cazzaniga, P.; Besozzi, D.; Mauri, G.; Amigoni, L.; Colombo, S.; Martegani, E. Simulation of the Ras/cAMP/PKA pathway in budding yeast highlights the establishment of stable oscillatory states. *Biotechnol. Adv.* **2012**, *30*, 99–107. [CrossRef]
25. Renz, A.; Widerspick, L.; Dräger, A. Genome-Scale Metabolic Model of Infection with SARS-CoV-2 Mutants Confirms Guanylate Kinase as Robust Potential Antiviral Target. *Genes* **2021**, *12*, 796. [CrossRef] [PubMed]
26. Gao, C.W.; Allen, J.W.; Green, W.H.; West, R.H. Reaction Mechanism Generator: Automatic construction of chemical kinetic mechanisms. *Comput. Phys. Commun.* **2016**, *203*, 212–225. [CrossRef]
27. Khanshan, F.S.; West, R.H. Developing detailed kinetic models of syngas production from bio-oil gasification using Reaction Mechanism Generator (RMG). *Fuel* **2016**, *163*, 25–33. [CrossRef]
28. Lok, L.; Brent, R. Automatic generation of cellular reaction networks with Moleculizer 1.0. *Nat. Biotechnol.* **2005**, *23*, 131–136. [CrossRef] [PubMed]
29. Städter, P.; Schälte, Y.; Schmiester, L.; Hasenauer, J.; Stapor, P.L. Benchmarking of numerical integration methods for ODE models of biological systems. *Sci. Rep.* **2021**, *11*, 2696. [CrossRef] [PubMed]
30. Garrido, A. Symmetry in complex networks. *Symmetry* **2011**, *3*, 1–15. [CrossRef]
31. Ohlsson, F.; Borgqvist, J.; Cvijovic, M. Symmetry structures in dynamic models of biochemical systems. *J. R. Soc. Interface* **2020**, *17*, 20200204. [CrossRef]
32. Keating, S.M.; Waltemath, D.; König, M.; Zhang, F.; Dräger, A.; Chaouiya, C.; Bergmann, F.T.; Finney, A.; Gillespie, C.S.; Helikar, T.; et al. SBML Level 3: An extensible format for the exchange and reuse of biological models. *Mol. Syst. Biol.* **2020**, *16*, e9110. [CrossRef]
33. Besozzi, D.; Cazzaniga, P.; Mauri, G.; Pescini, D. BioSimWare: A software for the modeling, simulation and analysis of biological systems. In *International Conference on Membrane Computing*; Springer: Berlin, Germany, 2010; pp. 119–143.
34. Chellaboina, V.; Bhat, S.P.; Haddad, W.M.; Bernstein, D.S. Modeling and analysis of mass-action kinetics. *IEEE Control Syst.* **2009**, *29*, 60–78.
35. Voit, E.O.; Martens, H.A.; Omholt, S.W. 150 years of the mass action law. *PLoS Comput. Biol.* **2015**, *11*, e1004012. [CrossRef]
36. Nelson, D.L.; Lehninger, A.L.; Cox, M.M. *Lehninger Principles of Biochemistry*, 5th ed.; Macmillan: London, UK, 2008.

37. Chaouiya, C.; Remy, E.; Thieffry, D. Petri net modelling of biological regulatory networks. *J. Discrete Algorithms* **2008**, *6*, 165–177. [CrossRef]
38. Davidrajuh, R. Detecting Existence of Cycles in Petri Nets. In Proceedings of the International Joint Conference SOCO'16-CISIS'16-ICEUTE'16, San Sebastián, Spain, 9–21 October 2016; pp. 376–385.
39. Tangherloni, A.; Spolaor, S.; Rundo, L.; Nobile, M.S.; Cazzaniga, P.; Mauri, G.; Liò, P.; Merelli, I.; Besozzi, D. GenHap: A novel computational method based on genetic algorithms for haplotype assembly. *BMC Bioinform.* **2019**, *20*, 172. [CrossRef] [PubMed]
40. Cho, Y.J.; Ramakrishnan, N.; Cao, Y. Reconstructing chemical reaction networks: Data mining meets system identification. In Proceedings of the 14th ACM International Conference on Knowledge Discovery and Data Mining, Las Vegas, NV, USA, 24–27 August 2008; pp. 142–150.
41. Dalcín, L.; Paz, R.; Storti, M. MPI for Python. *J. Parallel Distrib. Comput.* **2005**, *65*, 1108–1115. [CrossRef]
42. Gropp, W.D.; Gropp, W.; Lusk, E.; Skjellum, A. *Using MPI: Portable Parallel Programming With the Message-Passing Interface*; MIT Press: Cambridge, MA, USA, 1999; Volume 1.
43. Vaidyanathan, S. Dynamics and control of Brusselator chemical reaction. *Int. J. ChemTech Res.* **2015**, *8*, 740–749.
44. Cornish-Bowden, A. One hundred years of Michaelis–Menten kinetics. *Perspect. Sci.* **2015**, *4*, 3–9. [CrossRef]
45. Hill, A.V. The possible effects of the aggregation of the molecules of haemoglobin on its dissociation curves. *J. Physiol.* **1910**, *40*, 4–7.
46. Dimitrova, T.; Petrovski, K.; Kocarev, L. Graphlets in multiplex networks. *Sci. Rep.* **2020**, *10*, 1928. [CrossRef]
47. Doria-Belenguer, S.; Youssef, M.K.; Böttcher, R.; Malod-Dognin, N.; Pržulj, N. Probabilistic graphlets capture biological function in probabilistic molecular networks. *Bioinformatics* **2020**, *36*, i804–i812. [CrossRef] [PubMed]
48. Pržulj, N. Biological network comparison using graphlet degree distribution. *Bioinformatics* **2007**, *23*, e177–e183. [CrossRef]
49. Wong, E.; Baur, B.; Quader, S.; Huang, C.H. Biological network motif detection: Principles and practice. *Brief. Bioinform.* **2012**, *13*, 202–215. [CrossRef] [PubMed]
50. Stone, L.; Simberloff, D.; Artzy-Randrup, Y. Network motifs and their origins. *PLoS Comput. Biol.* **2019**, *15*, e1006749. [CrossRef] [PubMed]
51. Kim, W.; Li, M.; Wang, J.; Pan, Y. Biological network motif detection and evaluation. *BMC Syst. Biol.* **2011**, *5*, S5. [CrossRef] [PubMed]
52. Prill, R.J.; Iglesias, P.A.; Levchenko, A. Dynamic properties of network motifs contribute to biological network organization. *PLoS Biol.* **2005**, *3*, e343. [CrossRef]
53. Feinberg, M. *Foundations of Chemical Reaction Network Theory*; Springer: Berlin, Germany, 2019.
54. Arita, M. Scale-freeness and biological networks. *J. Biochem.* **2005**, *138*, 1–4. [CrossRef]
55. Wang, Y.; Christley, S.; Mjolsness, E.; Xie, X. Parameter inference for discretely observed stochastic kinetic models using stochastic gradient descent. *BMC Syst. Biol.* **2010**, *4*, 99. [CrossRef] [PubMed]
56. Totis, N.; Tangherloni, A.; Beccuti, M.; Cazzaniga, P.; Nobile, M.S.; Besozzi, D.; Pennisi, M.; Pappalardo, F. Efficient and settings-free calibration of detailed kinetic metabolic models with enzyme isoforms characterization. In Proceedings of the International Meeting on Computational Intelligence Methods for Bioinformatics and Biostatistics, Caparica, Portugal, 6–8 September 2018; pp. 187–202.
57. Ravasz, E.; Somera, A.L.; Mongru, D.A.; Oltvai, Z.N.; Barabási, A.L. Hierarchical organization of modularity in metabolic networks. *Science* **2002**, *297*, 1551–1555. [CrossRef] [PubMed]
58. Kline, M.P.; Morimoto, R.I. Repression of the heat shock factor 1 transcriptional activation domain is modulated by constitutive phosphorylation. *Mol. Cell. Biol.* **1997**, *17*, 2107–2115. [CrossRef] [PubMed]