SystemC Implementation of Stochastic Petri Nets for Simulation and Parameterization of Biological Networks

NICOLA BOMBIERI, SILVIA SCAFFEO, ANTONIO MASTRANDREA, and SIMONE CALIGOLA, Dept. Computer Science - Univ. Verona, Italy
TOMMASO CARLUCCI, Dept. Medicine - Univ. Verona, Italy
FRANCO FUMMI, Dept. Computer Science - Univ. Verona, Italy
CARLO LAUDANNA and GABRIELA CONSTANTIN, Dept. Medicine - Univ. Verona, Italy
ROSALBA GIUGNO, Dept. Computer Science - Univ. Verona, Italy

Model development and simulation of biological networks is recognized as a key task in Systems Biology. Integrated with in vitro and in vivo experimental data, network simulation allows for the discovery of the dynamics that regulate biological systems. Stochastic Petri Nets (SPNs) have become a widespread and reference formalism to model metabolic networks thanks to their natural expressiveness to represent metabolites, reactions, molecule interactions, and simulation randomness due to system fluctuations and environmental noise. In the literature, starting from the network model and the complete set of system parameters, there exist frameworks that allow for dynamic system simulation. Nevertheless, they do not allow for automatic model parameterization, which is a crucial task to identify, in silico, the network configurations that lead the model to satisfy specific temporal properties. To cover such a gap, this work first presents a framework to implement SPN models into SystemC code. Then, it shows how the framework allows for automatic parameterization of the networks. The user formally defines the network properties to be observed and the framework automatically extrapolates, through Assertion-based Verification (ABV), the parameter configurations that satisfy such properties. We present the results obtained by applying the proposed framework to model the complex metabolic network of the purine metabolism. We show how the automatic extrapolation of the system parameters allowed us to simulate the model under different conditions, which led to the understanding of behavioral differences in the regulation of the entire purine network. We also show the scalability of the approach through the modeling and simulation of four biological networks, each one with different structural characteristics.

CCS Concepts: • **Hardware → Modeling and parameter extraction**; • **Computing methodologies;**

Additional Key Words and Phrases: Stochastic Petri Net, metabolic networks, electronic design automation, T cells, autoimmunity

This work has been partially supported by the European Research Council (ERC) grants IMMUNOALZHEIMER (nr. 695714, ERC advanced grant) and IMPEDE (nr. 693606, ERC Proof of Concept grant).
Authors' addresses: N. Bombieri, S. Scaffeo, A. Mastrandrea, S. Caligola, F. Fummi, and R. Giugno, Dept. Computer Science - Univ. Verona, Italy; emails: {nicola.bombieri, silvia.scaffeo, antonio.mastrandrea, simone.caligola, franco.fummi, rosalba.giugno}@univr.it; T. Carlucci, C. Laudanna, and G. Constantin, Dept. Medicine - Univ. Verona, Italy; emails: {tommaso.carlucci, carlo.laudanna, gabriela.constantin}@univr.it.

1539-9087/2021/05-ART31 $15.00
https://doi.org/10.1145/3427091

ACM Reference format:
Nicola Bombieri, Silvia Scaffeo, Antonio Mastrandrea, Simone Caligola, Tommaso Carlucci, Franco Fummi, Carlo Laudanna, Gabriela Constantin, and Rosalba Giugno. 2021. SystemC Implementation of Stochastic Petri Nets for Simulation and Parameterization of Biological Networks. *ACM Trans. Embed. Comput. Syst.* 20, 4, Article 31 (May 2021), 20 pages.
https://doi.org/10.1145/3427091

1 INTRODUCTION

The purpose of Systems Biology is to support biologists' analysis through mathematical and computational models capable of representing biological systems and predicting, in silico, their complex behaviors. Biological models can be *quantitative* or *qualitative* [44]. Quantitative models such as kinetic models rely on mathematical formalism (e.g., **ordinary differential equations (ODEs)**) and allow for a more accurate modeling through actual molecular concentrations and time scales [44]. These are dynamic models and can be deterministic or stochastic [40]. Deterministic models capture the global behavior of the network elements, while stochastic models incorporate randomness to take into account possible fluctuations and noise due to interacting molecules in the environment [11]. Nevertheless, the applicability of quantitative models is limited to small and well-studied systems. The large number of independent variables, the lack of quantitative information, and the relationships strongly depending on qualitative events make mathematical models difficult or even prohibitive to obtain and analyze [30, 44]. In contrast, qualitative models (e.g., topological models, Boolean networks) or *semi-quantitative* models require fewer configuration parameters. They are more focused to discover network properties and to reproduce the system behavior at a higher level of abstraction [3, 33, 34, 39].

In this context, **Petri Nets (PNs)** have been adopted for both qualitative and quantitative modeling [12]. They provide an intuitive graphical representation, well-founded mathematical properties for qualitative analysis, and dynamic simulation [35]. Thanks to their extension to model more advanced and detailed behaviors like transition delays, inhibition arcs, and colored tokens, they have become one of the reference formalisms to model biological systems [11, 35].

In [23], the authors show the benefits of using PN instead of ODEs in terms of flexibility and understanding, to simulate the metabolic pathways of the glycolysis. In [21], the authors address the synthesis problem, treating synthesis as an automated process that, given behavioral specifications or partial specifications of a system to be realized, decides whether the specifications are feasible, and then produces a Petri net realizing them exactly, or if this is not possible produces a Petri net realizing an optimal approximation of the specifications.

In [37], the authors introduce the ***hybrid functional* Petri Nets (HFPNs)** to solve problems associated with the representation and simulation of bio-pathways. Other approaches and software applications based on ***continuous timed* Petri Nets (CTPNs)** have been applied to model and simulate biological systems through a semi-quantitative paradigm [4, 41]. Different *model checking* techniques and *controllability* evaluation methods have been also proposed to verify, through formal methods, PNs correctness and soundness (see [50] and [43] for the corresponding surveys). PNs have been applied to model *place invariants* and *transition invariants* to validate complex processes such as apoptosis [29] and to understand the processes at the basis of the iron homeostasis [42].

***Stochastic* Petri Nets (SPNs)** are a form of PN that allows the model to support randomness, which is fundamental to take into account possible fluctuations and noise due to molecules interacting in the environment [11]. For example, SPNs have been successfully applied to obtain new insights in the development of hepatic granuloma throughout the course of infection [1], and to model signal transduction pathways in the process of angiogenesis [36].

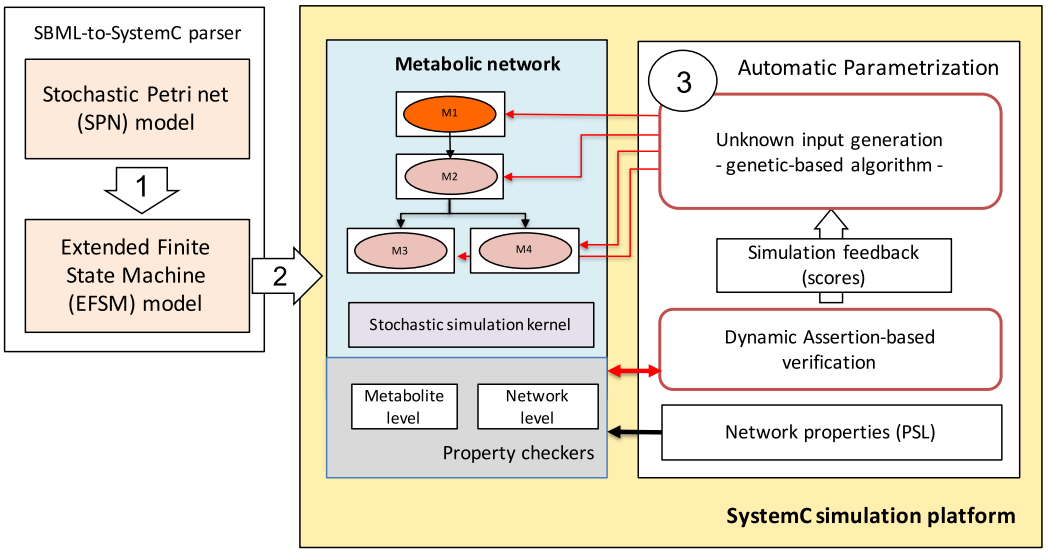

Fig. 1. The framework overview.

Software applications have been developed to implement and simulate both stochastic and deterministic PN models. The best known are Snoopy [28], Monalisa [2], and iBioSim [49]. They allow for the simulation of complex biochemical systems including metabolic pathways, signal transduction pathways, and gene expression networks. Nevertheless, a recurrent issue of these software applications is related to the concept of *parameterization*. Very often, due to the lack of quantitative information, it is necessary to explore the *solution space* of the parameters to extrapolate the network configurations that lead the model to satisfy certain biological properties. In the tools at the state of the art, this process requires manual tuning of all the unknown parameters combined with in vitro experimental data to obtain model behaviors matching the biological knowledge. Since the solution space to explore grows exponentially with the network size, such a manual parameterization task becomes prohibitive when applied to realistically large networks [14, 26, 48, 52].

SystemC has been used in the past to implement signal transduction models for deterministic simulation [5, 15, 17, 18]. This work targets metabolic networks represented by PNs and their stochastic simulation for which, to the best of our knowledge, there is no prior work.

In this work, we propose a framework that applies languages, techniques, and tools well established in the field of **electronic design automation (EDA)** to model and simulate metabolic networks. Since biological systems and electronics systems share several characteristics like concurrency, reactivity, abstraction levels, and issues like reverse engineering and design space exploration [5, 22], we show how the proposed EDA-based framework can introduce automation and flexibility to model, simulate, and help in the analysis of metabolic networks.

Figure 1 shows an overview of the framework (which is available for download at github.com/InfOmics/NAPS). Starting from the PN model of a metabolic network, the framework generates SystemC code that implements the behavior of each single metabolite and reaction. Since metabolic networks are widely available on many biological databases [46] in the **System Biology Markup Language (SBML)** standard format [31], we developed a parser to translate models described in such a format into SystemC [9].

We propose a translation of PN models of biological networks into concurrent **extended finite state machines (EFSMs)** to take advantage of existing and well-consolidated methodologies and tools to modularly and automatically synthesize EFSM models into executable SystemC code. In addition, such a translation will allow us (in our current and future work) to evaluate the robustness and sensitivity of the metabolic networks by applying mutation models for EFSMs, which are well investigated in system-level electronic design, by automatically injecting perturbations in the model.

The framework completes the SystemC implementation of the network with a stochastic simulation kernel. The user defines network properties through a formal specification language (i.e., PSL), which are automatically synthesized and plugged into the network code. The framework then applies **assertion-based verification (ABV)** combined to an automatic parameter generation based on a genetic algorithm to extrapolate the parameter configurations that satisfy the defined network properties.

It is important to note that this work does not target formal verification of biological networks through model checking, for which several existing approaches could be applied directly to the PN representation. This work targets the automatic parameterization of the networks through dynamic event-driven simulation, since the complexity and the design space size of the problem do not allow us to apply formal methods.

We present the results obtained by applying the proposed framework to model the complex metabolic network of the purine metabolism starting from the metabolomics data obtained from naive lymphocytes and autoreactive T cells implicated in the induction of experimental autoimmune disorders. We show that the framework automatically extrapolates system parameterizations that reproduce the in vitro experimental results and that allow us to simulate the model under different conditions. Finally, in order to evaluate the scalability of the approach, we applied the framework to translate and simulate real and synthetic metabolic networks of different sizes and characteristics.

2 BACKGROUND ON STOCHASTIC PETRI NETS

SPN is a class of Petri nets in which at every transition k of the network state is associated a delay τ_k, which is determined by a random variable. Formally, an SPN is a five-tuple $SPN = \{P, T, F, M_0, \Lambda\}$, where P is the set of the places, T is the set of transitions, $F \subset (P \times T) \cup (T \times P)$ is the set of relations between places and transitions, M_0 is the initial configuration of the net (marking), and Λ is the set of exponentially distributed firing rates λ_k associated with the transitions. The firing rates are defined through *propensity functions* (hazard) a_k that have the pre-places of the transition k as domain, and it gives the probability that a reaction will occur in the next infinitesimal time interval.

To compare different behaviors of the network, colored Petri nets could be adopted to model different metabolites. However, in this work we chose to adopt standard PNs as, for the model dynamics we have to observe and simulate, all metabolites are at the same level and have the same importance when involved in a reaction. In our simulations, we are mainly interested in the amount of tokens of each place (i.e., the concentration of each metabolite) and how fast they are consumed or produced. In this perspective, a distinction between tokens would not help us to extrapolate such a speed rate.

To simulate biochemical systems, specific types of propensity functions are used, such as *mass-action propensity functions* and *Michaelis-Menten propensity functions*. A well-established method to perform a simulation of an SPN is the Gillespie algorithm, called **Stochastic Simulation Algorithm (SSA)**. This method is a Monte Carlo procedure that calculates a possible trajectory of the system simulating one chemical reaction at each step, and that chooses the firing time τ.

Gillespie proposed two equivalent variants of his algorithm: the **Direct Method (DM)** [25] and the **First Reaction Method (FRM)** [24]. Several algorithms have been proposed to improve the efficiency of these algorithms [10, 38, 47]. In general, DM is more efficient in terms of computational time and space, while FRM is well suited for a concurrent and parallel implementation [19]. Our methodology is based on the FRM variant.

3 METHODOLOGY

Figure 1 shows an overview of the simulation and parameterization framework. It relies on three main phases:

(1) Modeling of PNs through EFSMs as explained in Section 3.1
(2) Synthesis of EFSMs into SystemC with the support of a kernel for stochastic simulation, as explained in Section 3.2
(3) Automatic parameterization of the network through a genetic-based input generation guided by dynamic assertion-based verification, as explained in Section 3.3

3.1 Modeling PN through EFSM

The proposed methodology considers a PN model of the system under analysis as a starting point. Figure 2 shows an overview of the different possible reactions in metabolic networks [27], which represent the basic blocks of the PN models. The figure reports the PN models and the corresponding representations in **Systems Biology Graphical Notation (SBGN-PD)**, which is the standard representation in the Systems Biology community.

For the sake of clarity, we refer in the following to the simplest and most standard reaction between metabolites of biological networks, whose PN representation is shown on top of Figure 2. The description of such a reaction is represented in Figure 4. The other reactions (i.e., irreversible, inhibition of irreversible, inhibition of reversible, etc.) are similarly translated.

To perform the event-driven simulation of the PN, we first translate the model of each PN place and each PN transition into the equivalent EFSM model [13]. This allows us to modularly synthesize the components of the PN network into executable and concurrent SystemC processes. An additional SystemC module implements communication and synchronization among processes (as explained in Section 3.2).

Figure 3 shows such an EFSM modeling. We assume one process per *reaction* R_i and one process per metabolite, being *reactant* (M_i) or *reaction product* (M_j). We represent each process through a two-state EFSM. All the EFSMs are concurrent and synchronized as follows. Each reaction process R_i starts in a `Ready` state, and it moves to the `In_reaction` state as soon as it receives the `Available` signal *from all* the pre-place machines, i.e., the EFSMs of all the upstream reactants involved and necessary for the reaction R_i (M_i in Figure 3). The reaction process holds in the `In_reaction` state until the reaction delay has expired. The delay value of each reaction (τ_R) is set according to the propensity function adopted to represent the system (see Section 3.2). After such an event, the process sends a `Done` signal to all the metabolites involved in the reaction, both reactants and reaction products, and the machine moves back to the `Ready` state. On the other hand, if the delay has not expired and at least one of the reactants is not available anymore (i.e., the concentration of the metabolite has dropped down under a certain threshold), then nothing happens as the reaction is no longer feasible and the process shifts back to the `Ready` state.

The `Not_available` state of a metabolite represents the condition by which the metabolite cannot be involved in any reaction, either because of the low concentration (below a user-defined threshold) or because the metabolite is already involved in a reaction process. When a process representing a reaction product M_j is in such a state, it can receive a `Done` signal from any reaction

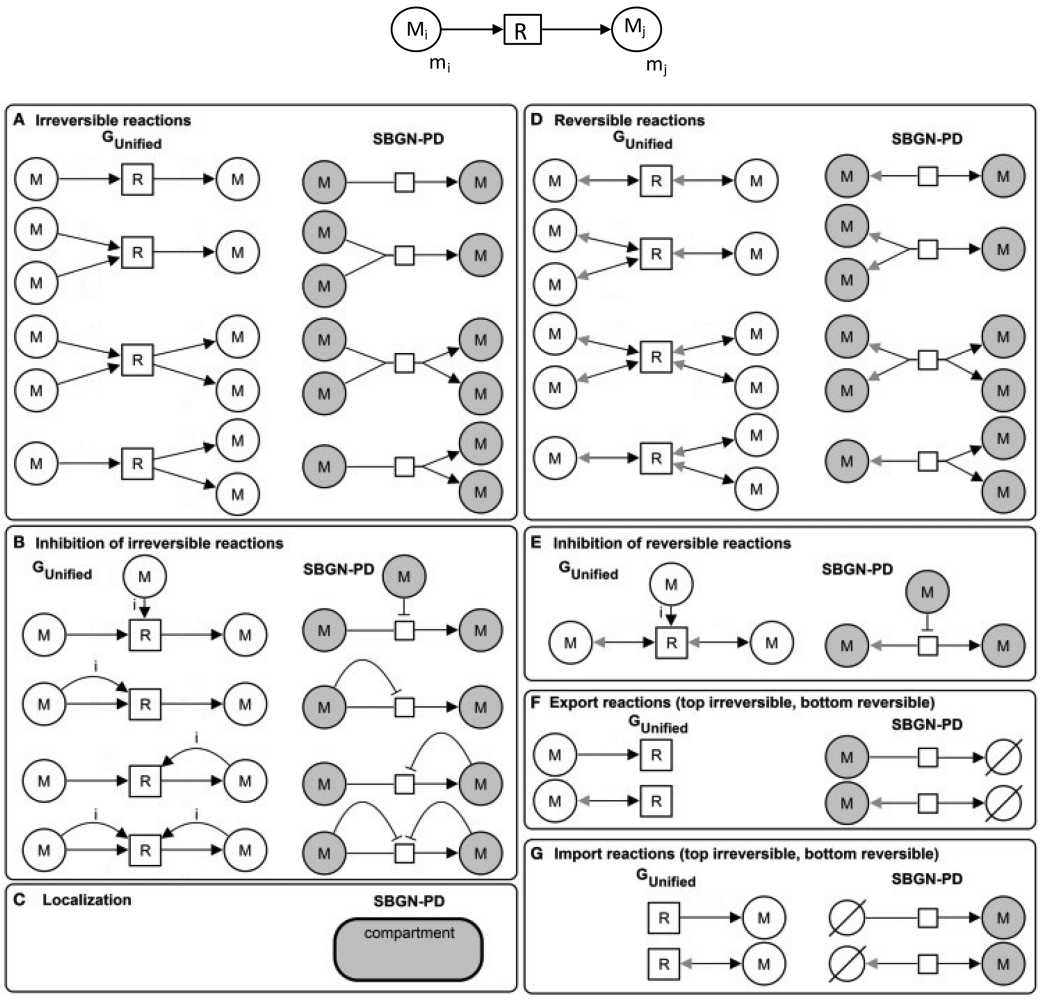

Fig. 2. Basic blocks (reactions) of a metabolic network in PN and the corresponding representation in SBGN-PD.

process in which it is involved (R_i in Figure 3). In this case, the concentration of M_j is incremented by a user-defined constant m_0. If the new concentration goes over a given threshold, then the machine moves to the `Available` state. In the `Available` state, the metabolite concentration can be further increased by any other *upstream* reaction through a *Done* signal.

On the other hand, if the *Done* signal comes from a *downstream* reaction (R_j in Figure 3), then the metabolite is seen as a reactant and its concentration decreases. If the concentration decreases under a given threshold, the model moves back to the `Not_available` state.

Representing each place (metabolite) and each transition (reaction) through single concurrent EFSMs allows us to modularly implement the reaction and the metabolite behaviors. A metabolite process is aware of what is happening in a reaction process only when the reaction delay of the latter has successfully expired. Another advantage of this separation is the automatic translation of the model from the SBML format to SystemC. This way of representing the PN components through EFSMs and corresponding processes reflects the structure of the SBML architecture, en-

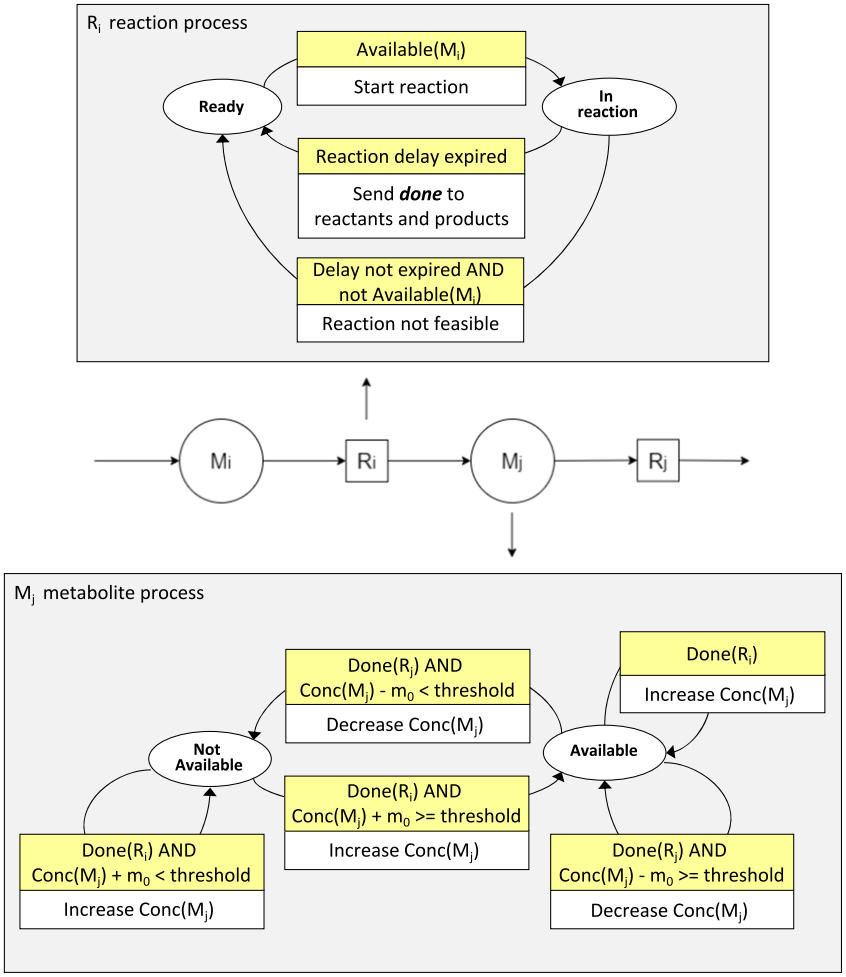

Fig. 3. The EFSM representation of the reaction processes R_i and the metabolite process M_j.

suring an easier and compatible translation, with the possibility of including in the SystemC model all the additional information and annotations present in the original SBML file. However, in this first release of the framework, all the additional biological information and annotations present in the original SBML file (e.g., names, descriptions, links to pathway explanation) have not been considered since they are useless for generating the *internal* SystemC representation. In any case, they can be modularly supported if necessary as an extension of the proposed framework front-end.

3.2 EFSM-SystemC Synthesis with Stochastic Simulation Support

In general, SystemC allows designers to implement systems at different levels of abstraction and with different levels of detail. It provides modeling features such as structural hierarchy and connectivity, communication abstraction, dynamic processes, and timed event notifications. The key SystemC language features used in the framework for modeling and simulating metabolic networks are the following (see Figure 4):

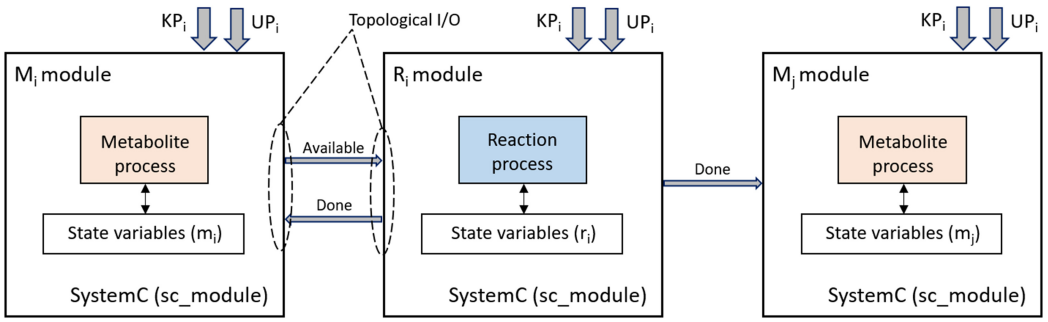

Fig. 4. Implementation of reaction and metabolite processes in SystemC.

- *Modules.* Metabolites (M) and reactions (R) of the network are modeled as SystemC modules, and all the network components are hierarchically organized into a module representing the whole metabolic network. Being the basic building blocks of a SystemC design hierarchy, they are used for hierarchical organization of networks and sub-networks.
- *Ports and signals.* They implement communication and synchronization between modules (e.g., `Available` and `Done` in Figure 4).
- *Processes.* Each metabolite and each reaction behavior has been modeled through a SystemC process, which reacts at each simulated instant of time or at each specific event to update the network state. They are synchronized to update the metabolite concentrations by considering both upstream and downstream reactions. The processes update the metabolite and reaction state variables, which hold dynamic information like state, concentration, reaction rate, reaction starting time, etc.
- *Events.* They allow for synchronization between processes. Events are the key objects in SystemC models to provide event-driven simulation. Each firing of EFSM transition of Figure 4 or each time instant of Figure 6, as explained in the following section, is an example of an event. In this work, we only considered implicit events as changes of signal values and timed events.

Figure 4 shows the proposed hierarchical template, which distinguishes different sets of input/output data for both metabolites and reactions that can affect the model behavior:

- *Topological inputs and outputs (Topological I/O)*: They are inputs and outputs whose value is calculated at simulation time and depends on the topological interaction of the modeled metabolite with reactants and the reaction products.
- *Known parameters (KPi)*: They are inputs whose value depends on the environment characteristics and status and are known at modeling time. They are generally extrapolated through in vitro or in vivo experimental analysis or from available databases of the literature. Examples are the initial concentration of metabolites, reactant coefficients, or the range of temporal delay of reactions.
- *Unknown parameters (UPj)*: They are inputs whose value depends on the environment characteristics and status and which are *unknown* at modeling time. Examples are the parameters affecting the reaction rate (e.g. mass-action rate constants) or any other information not available in the literature and necessary for the dynamic simulation of the system. For each parameter, the platform generates different values with the aim of observing, via simulation, how such values affect the system dynamics (see Section 3.3).

In a stochastic simulation of metabolic networks, each reaction R_i between a metabolite (M_j) and the reactant (M_i) fires:

(1) if the reactant (M_i) is available as it satisfies a concentration constraint ($m_i > m_i^0$), where m_i is the reactant concentration (i.e., number of tokens of the PN place) and m_i^0 is a reaction coefficient (i.e., a constant), and
(2) after a specific time delay τ_R, which is defined as follows:

$$\tau_R = \left\lfloor \left(\frac{1}{a_R(m_i)} \right) \cdot ln \left(\frac{1}{rand_k} \right) \right\rfloor + 1,$$

where $rand_k \in (rand_1, \ldots, rand_n)$ is a random number from the uniform distribution $U(0, 1)$. $a_R(m_i) = c_i m_i$ is the mass-action propensity function, where c_i is the reaction rate and m_i is the number of tokens of the reactant M_i. We consider a lower-bound value of each reaction delay (i.e., 1), which is associated to the minimum delay in the discrete simulation.

According to the FRM model (see Section 2), the stochastic simulation of the system evolves along discrete steps, where each next time step corresponds to the expiration time of the immediately next reaction. In particular, considering t_i as the current simulation time of the system:

$$t_{i+1} = t_i + \tau_{R_i}, \text{ s.t. } \tau_{R_i} = min(\tau_{R_1}, .., \tau_{R_k}),$$

At each simulation step, a *stochastic simulation kernel* updates all the reaction delays by considering the elapsed time and a random component. Figure 5 shows a summarizing overview of such a stochastic simulation paradigm. It is important to note that the updating phase at each simulation step of all the reaction delays $\tau_{R_i} \forall i$, which include a random component ($rand_k$), is the necessary condition to perform the stochastic simulation of the PN model. The simulation ends after a given number (N) of simulation steps.

The stochastic kernel layer was implemented on top of the SystemC native kernel in order to make stochastic simulation possible without altering the SystemC simulation kernel. This is important since, in our current and future work, we also apply deterministic simulation of biological networks, which is an important alternative to the stochastic paradigm for many different analyses.

3.3 Parameter Estimation through Dynamic Assertion-Based Verification (ABV)

Functional verification based on assertions represents one of the main applied and investigated techniques that combines simulation-based (i.e., dynamic) and formal (i.e., static) verification [7, 16, 45]. Assertions are formal descriptions that allow system designers to detect functional errors in the model and in the model evolution over time. They are also combined with techniques of automatic input pattern generation [8] that extrapolate *system configurations* to prove the satisfiability or unsatisfiability of the system properties. The proposed methodology applies simulation-based ABV, by which assertions are defined in a formal language (i.e., PSL [32]); they are automatically synthesized into *checkers*[1] and plugged into the SystemC model representing the network [6]. In our context, checkers aim at monitoring the concentration of the metabolites and giving a *score* (i.e., fitness) to an input generation module, which implements a genetic algorithm to generate good configurations of the kinetic parameters.

The module generates a configuration of parameters and runs a dynamic simulation of the network for such a set of input values for a given simulation time. Then, the module generates a new different configuration for a new simulation. A new run is needed for each new possible configuration of parameters since we always have to start from the same initial conditions of the model,

[1]The framework relies on the IBM FoCs synthesizer [51] for the automatic synthesis of assertions.

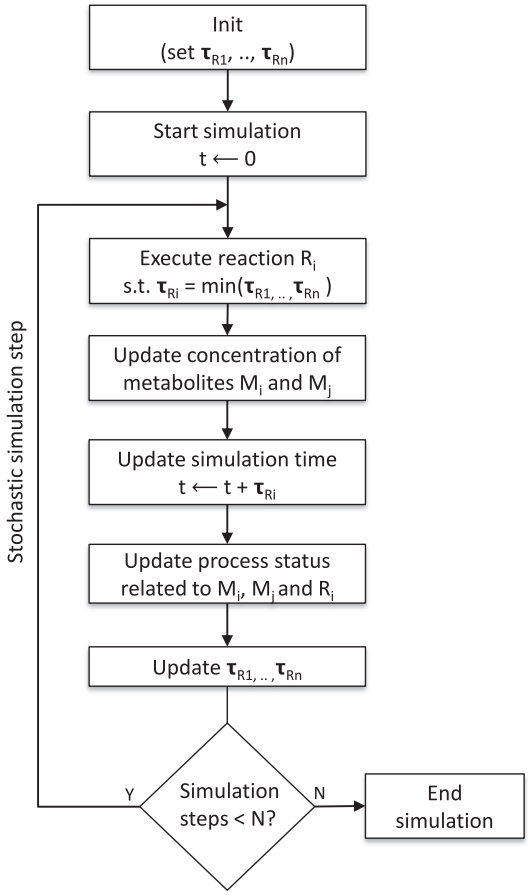

Fig. 5. Overview of the stochastic discrete simulation.

which are declared as part of the case study. A proper fitness function evaluates the goodness of each potential solution estimated through simulation. The run ends when the module finds the parameter configuration that allows the system properties to be satisfied. The definition of the fitness function depends on the property to be checked and it is formulated as the distance between *state vectors* representing the *simulation trend* $s_i(t)$ in comparison to a defined *reference trend* $r_i(t)$ (i.e., the target behavior of the system).

The ABV checks the simulation trend $s_i(t)$ and, eventually, it stops the simulation to provide a score compared to $r_i(t)$. Given the state vectors $S = [s_1, s_2, \ldots, s_n]$ and $R = [r_1, r_2, \ldots, r_n]$ representing, respectively, the simulation and the reference trend, the score is defined as follows:

$$score_c(t) = d(S, R),$$

where d is a distance function (e.g., Euclidean distance) between the state vectors. The formulation of state vectors is general and allows us to model biological behaviors such as the stability in concentration and the change of concentrations between time points.

Figure 6 shows some examples of the state dynamics of a metabolite to be observed and for which assertions can be defined. In particular, the first assertion checks the concentration stability of a given metabolite over time by considering a user-defined tolerance ($\pm\sigma$). The second, more

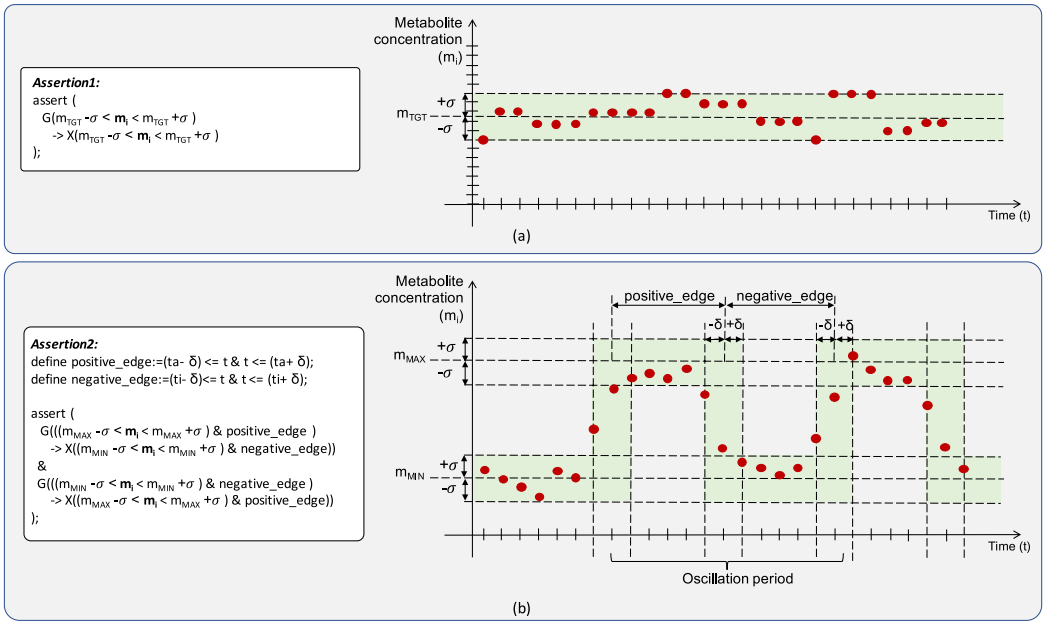

Fig. 6. Examples of assertion templates to verify a metabolite steady state: metabolite concentration stability (a) and the oscillation of a metabolite concentration (b). σ and δ are user-defined tolerance constants. m_{TGT}, m_{MAX}, and m_{MIN} are user-defined concentration values of the observed metabolite. ta and ti are temporal counters initialized at the first oscillation, and that hold the time elapsed from the first state transition ($m_{MIN} \rightarrow m_{MAX}$ and $m_{MAX} \rightarrow m_{MIN}$, respectively). They are used to measure the positive edge and negative edge values. t is the counter, which is set, from the second oscillation on, at each state transition, and it is used to measure the oscillation period.

complex assertion checks the periodic oscillations of the metabolite concentration by considering user-defined tolerances ($\pm\sigma$, $\pm\delta$). For both the examples, if the value of the state variable m_i, during simulation, remains in the green zone, the assertion is satisfied. Otherwise, the assertion fails and the system raises an error signal. We defined assertion templates, which have to be filled out by the user by indicating the metabolite to be monitored and the constants (target concentrations and tolerances).

In the release of our SW (github.com/InfOmics/NAPS), we propose a set of general (yet configurable) assertions. In any case, user-defined assertions can be added ex novo.

The proposed framework applies ABV for the *parameter estimation* phase, which aims at identifying the parameter settings that lead the network to satisfy the properties formalized by the assertions. This verification is delegated to the ABV, which is responsible for property checking during simulation. For each simulation of the model, the input stimuli generator receives a score (see Figure 1) from ABV and it applies this value to perform an evolutionary step of the genetic algorithm used to generate new configurations.

It is important to note that the model could be synthesized to an FPGA for faster simulation (also in different configurations). On the other hand, the application of PSL/ABV is mandatory for the automatic extrapolation of parameters, which requires an iterative automatic generation of parameters and the corresponding verification based on assertions. Such an automatic step has to be transparent to the final user (the biologist) who, for the tool applicability, has not to learn EDA languages or notions.

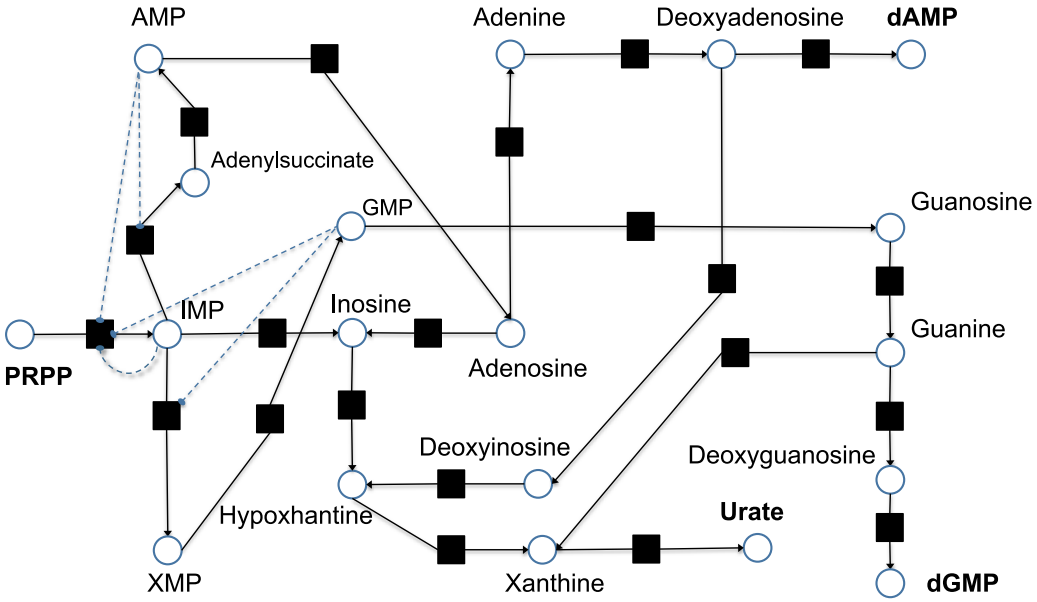

Fig. 7. Example of PN model: the purine pathway case study. The arrow of each arc appears only at the end of each reaction, where each reaction starts from a metabolite, is represented by a square on the arc, and ends into a target metabolite.

4 RESULTS AND DISCUSSION

We applied the proposed framework to understand how the dynamics of the purine pathway changes between normal and autoreactive conditions. We started from the PN model of such a metabolic network described in SBML, which is depicted in Figure 7.

We simulated the system under two different conditions: naive and PLP, which represent the standard condition and the proliferating of lymphocytes, correspondingly. The two conditions differ from the metabolite concentration and production of urate, dGMP, and dAMP (see Figures 8 and 10).

We considered metabolomics data obtained in vitro as initial metabolite concentrations for both conditions. Figure 9 shows such values.

The simulation and parameterization goal was to extrapolate the mass-action parameters of all network reactions that lead the system to a *steady state* of each metabolite except dAMP, dGMP, and urate for each condition. In metabolic pathways, the concept of steady state refers to the stability of the concentration levels of metabolites, and it is crucial to keep homeostasis inside the cell. Despite the lack of experimental data in the transformation rate from PRPP to IMP, we could easily set the system with two different flow speeds, *slow* and *fast*.

We simulated our model by generating reaction delays in the range $[1 - 1,000]$ of all network reactions. We targeted the steady state of each metabolite except dAMP, dGMP, and urate, which are the natural incremental product of the network. We assumed that the pathway is considered at steady state if the concentration of each element does not differ by more than ±30% from the initial concentration and it is maintained stable throughout a simulation time of 10^5 simulation cycles (see Figure 10). We formally specified this property through PSL assertions.

The choice of considering a +/− 30% range is an assumption that has been suggested by the research group that defined the network. It is a variable that can be changed and adapted to the

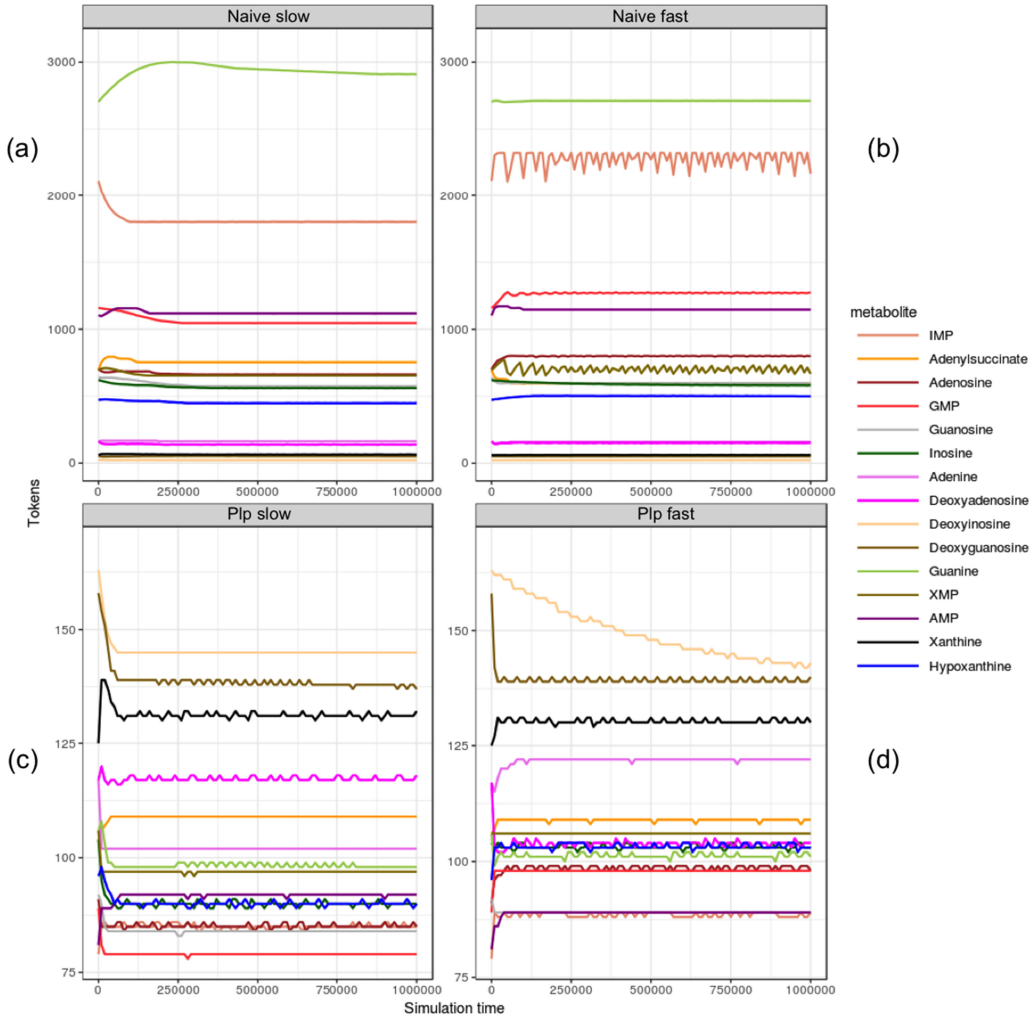

Fig. 8. Metabolite stability.

type of network considered. A system biologist should have an idea of how the metabolic pathway under study might behave over time and therefore make an estimate of such assertions for each metabolite. In sensitive networks, the more accurate the evaluations (i.e., the lower the values of range), the greater the effort for automatic extrapolation of parameters.

In our model, the inhibition mechanisms are represented through the inhibition arcs, which is an extension of the classical PN to represent the inhibition of a molecule when its concentration exceeds a certain threshold. We assumed that a metabolite can inhibit a reaction when it grows by 35% from its initial concentration.

The genetic algorithm used by the automatic input generator has been configured with a population size of 250 individuals, a mutation probability of 0.05, and a crossover probability of 0.1.

We defined the reference trend of the system as follows:

$$r_i(t) = m_i, \forall t > 0, i = 1, 2, \ldots, n,$$

Metabolite	Naive	PLP
	init (mM)	init (mM)
IMP	21,126	0,7991
AMP	11,0389	0,8174
XMP	unkown	unkown
GMP	11,5898	0,8981
Inosine	6,2093	1,0495
Hypoxanthine	4,7223	0,9668
Adenosine	7,0112	0,9121
Adenine	1,6299	1,172
Deoxyadenosine	unkown	unkown
dAMP	0,2681	1,3106
Guanosine	6,3851	0,9239
Guanine	27,0459	1,0649
Deoxyguanosine	0,5342	1,5833
dGMP	unkown	unkown
Deoxyinosine	0,2321	1,6381
Xanthine	0,6195	1,2584
Urate	0,519	1,1419
Adenylosuccinate	6,951	unkown

Fig. 9. Initial concentrations of metabolites for naive and PLP conditions.

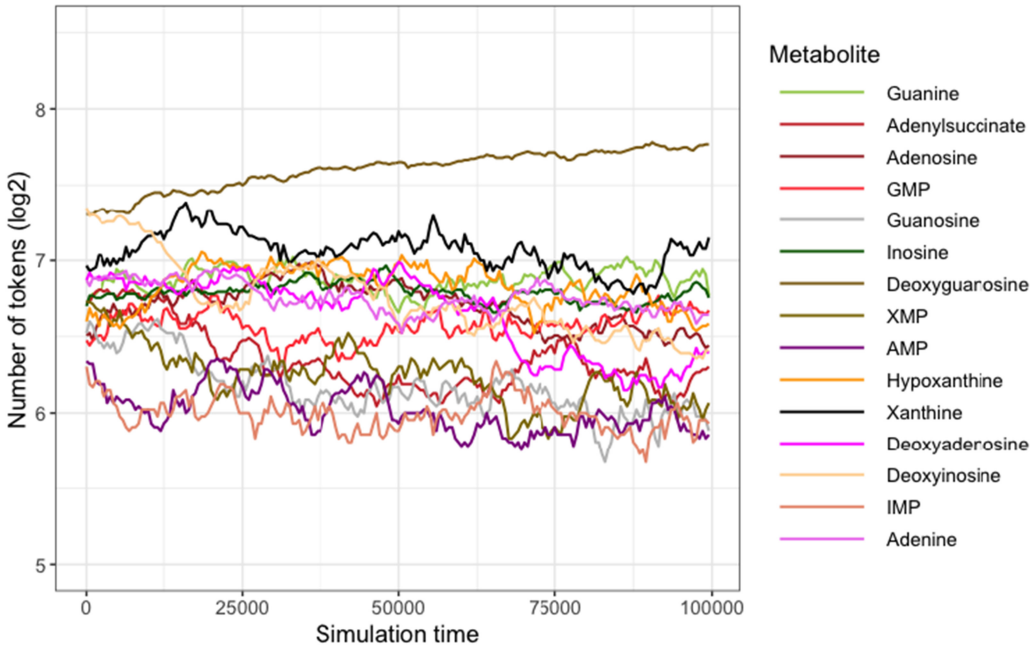

Fig. 10. Example of parameterization of the purine pathway in PLP-specific condition that leads the metabolites to stability within a simulation time of 10^5 clock cycles.

where m_i is the starting concentration of a metabolite and t is the simulation time. The state vector of the simulation trend is defined as $S = [c_1, c_2, \ldots, c_n]$, where c_i are the set of angular coefficients of the linear functions $s_i(t)$, linking the starting and the ending concentrations of the metabolites. The state vector of the reference trend was defined as $R = [0, 0, \ldots, 0]$. The fitness function was defined as the inverse of the Euclidean distance between the simulation and the reference trends. The selection method used to pick an individual from the population is *rank based*, meaning that the reproduction is always done by taking individuals with better fitness.

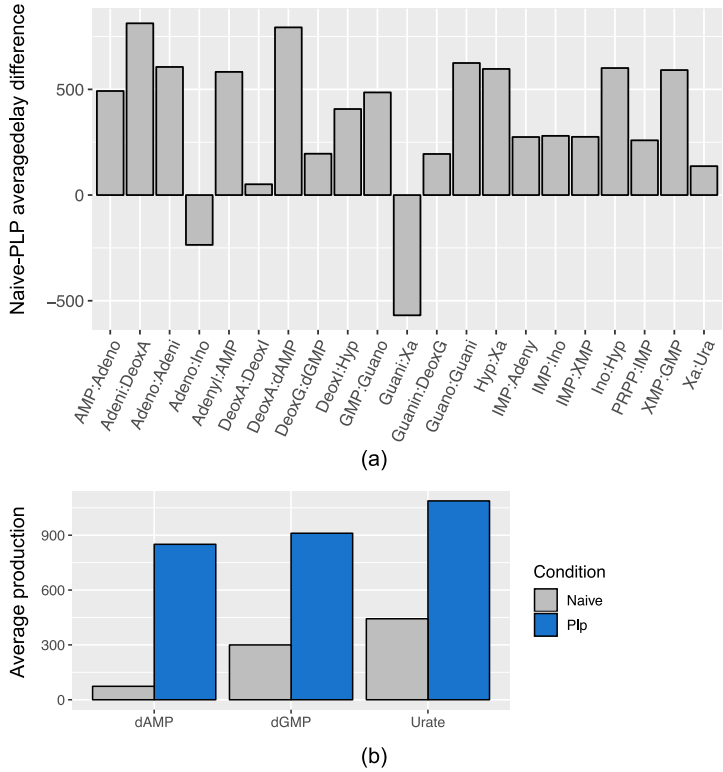

Fig. 11. Results of the analysis of 10 parameterizations of the purine pathway in condition of stability for naive and PLP-specific cells. (a) Average difference of the delays obtained parameterizing the pathway in naive and PLP-specific condition. (b) Difference in the final concentration of the metabolites dAMP, dGMP, and urate.

We obtained 10 parameter configurations of the purine pathway for each condition. Figure 8 shows the plot of the metabolite concentrations obtained with one of such configurations for each slow and fast flow speed, in both standard and proliferation conditions. For each parameter configuration, the network simulation required around 7 seconds to simulate 1 million time instants (simulation time in Figure 8). The complete parameterization phase required from 1 to 12 minutes for each network version. All the simulations were run on a machine equipped with an Intel(R) Xeon(R) CPU E5-2650 v4 clocked at 2,200 MHz and 16 GBs RAM, and the Ubuntu 16.04 operating system.

In general, the parameterization and simulation led to interesting differences in the regulation of the purine pathway, suggesting that most chemical reactions are highly favored in PLP-specific cells versus naive lymphocytes as shown in Figure 11(a) and in Figure 12, where gradients of colors are associated to reaction speeds from fast (antique rose) to slow (blue).

All metabolic reactions, with the exception of the reactions from Guanine to Xanthine (Guani:Xa) and from Adenosine to Inosine (Adeno:Ino), are sped up in the PLP-specific condition, having a lower average delay time generated by our framework (see Figure 11(a)). Further, the reaction from Deoxyadenosine to Deoxyinosine (DeoxA:DeoxI) had comparable delay times between the naive and PLP-specific condition. Overall, the observed speed-up in the PLP-specific condition resulted in a greater production of the fundamental elements of the pathway dAMP,

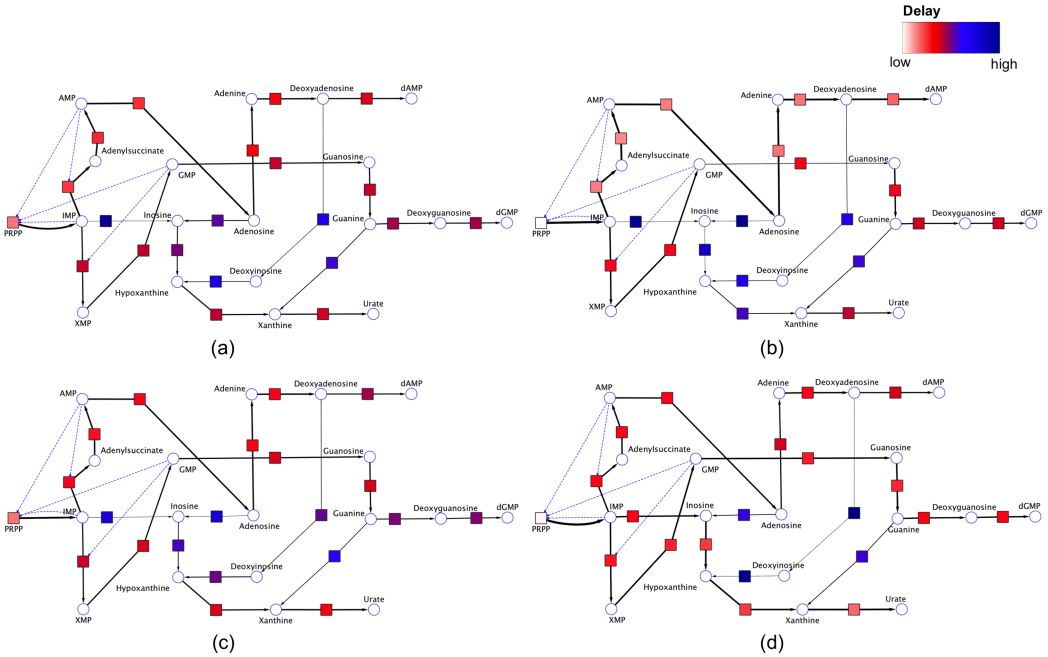

Fig. 12. Pathway activity based on PRPP→IMP flux: slow naive (a), fast naive (b), slow PLP-specific (c), and fast PLP-specific (d).

Table 1. Network Information and Simulation Results

Metabolic Network	Metabolites (#)	Reactions (#)	Met-Reaction Interactions (#)	Stochastic Sim. Steps (Avg #)	Sim. Time (Avg Sec.)
Purine network	19	21	42	1,752	0.067
Biomd619	10	13	26	1,996	0.044
Synthetic50	50	46	117	2,998	0.165
Biomd328	18	29	58	2,975	0.084

dGMP, and urate (Figure 11(b)). Notably, the increased urate, dGMP, and dAMP production in the PLP-specific network reflects our metabolomics data and a well-known metabolic feature of proliferating lymphocytes [20], validating the potentiality of the proposed framework in extrapolating parameters and simulating metabolic processes modeled through PN.

4.1 Analysis of Scalability

We finally tested the scalability of the framework in generating SystemC code for more complex PN models. We measured and compared the average time required to simulate networks of different characteristics.

Table 1 summarizes the evaluated benchmarks, which consist of two additional real metabolic networks, the acetaminophen (Paracetamol) pharmacokinetics in humans (Biomd619) and the atorvastatin metabolism in hepatic cells (Biomd328) [46], and a larger synthetic network to stress the system with a higher number of metabolites and reactions. The table reports the network characteristics in terms of metabolite and reaction numbers and information on their simulation

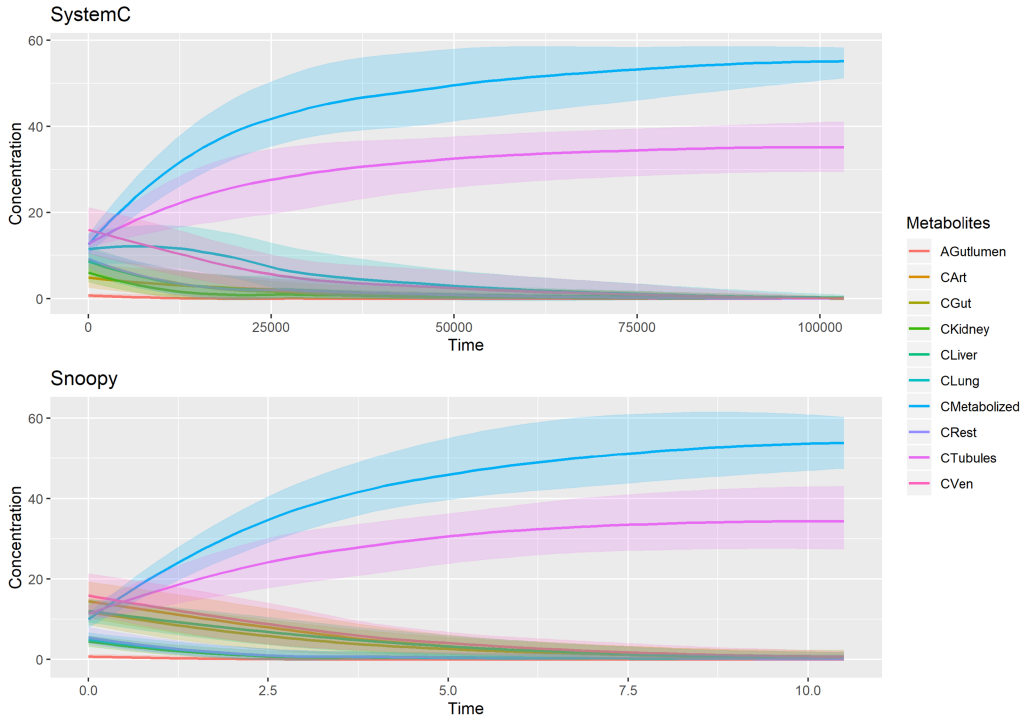

Fig. 13. Comparing simulation trends of the Biomd619 network with the SystemC model generated by the proposed framework (top) and those obtained with Snoopy [28] (bottom).

(stochastic simulation steps and simulation time). The average number of stochastic simulation steps is the number of steps during simulation that are necessary to lead the model into a dead state, meaning that all the initial tokens of the network are consumed by the reactions. One step corresponds to one fired reaction. The results show that the simulation time depends on two factors: the network complexity and the reaction rates. The first is strictly related to the number of reactions, while the second is related to the reaction delay and probability. The higher the probability of reactions to fire (i.e., the smaller the reaction delay), the larger the number of reactions to fire along the whole system simulation. In any case, in general, the total simulation time grows almost linearly with both network complexity and reaction rates.

We finally compared the simulation trends of all the analyzed networks with the SystemC models generated by the proposed framework (top) and those obtained with *Snoopy* [28] (bottom), which is a well-known and widespread software application used to visualize and simulate stochastic and deterministic PN models. We run 10 simulations for both softwares. Figure 13 shows, as an example, the results obtained with the Biomd19 network. The plots depict the minimum, maximum, and average values for each time instant. The results underline that, although the stochastic engine is behind the two softwares, the general trends in all the analyzed networks are matchable. It was not possible to accurately compare the two softwares in terms of simulation performance, as *Snoopy* requires a manual parameter setting and restart for each simulation phase. In any case, while the difference of simulation time for each network configuration appears negligible, *Snoopy* does not allow for automatic parameterization of the network, which is the main goal and contribution of this work.

5 CONCLUSION

This work presented a framework based on languages, techniques, and tools well established in the field of EDA for the simulation and parameterization of metabolic networks. Since these biological networks are generally represented by Stochastic Petri Nets, we proposed a method to modularly translate Petri Net models into SystemC code and a kernel for the stochastic simulation of such a model. We then applied assertion-based verification combined to an automatic parameter generator based on a genetic engine to extrapolate the configuration patterns that lead the system to satisfy user-defined properties. We applied the framework to study the purine metabolism pathway starting from metabolomics data obtained from naive lymphocytes and autoreactive T cells implicated in the induction of experimental autoimmune disorders. Thanks to the automatic parameterization of the model, we were able to reproduce the experimental results obtained in vitro and to simulate the system under different conditions. This was not possible by using the tools at the state of the art, which require the manual insertion of the reaction parameters to simulate the network evolution. From a biological point of view, the obtained simulation results suggest that the entire purine pathway is sped up in PLP-specific cells versus naive lymphocytes, according to our experimental data and literature. Finally, we applied the framework to translate and simulate real and synthetic metabolic networks of different sizes and characteristics to evaluate the framework scalability and to compare the results of the stochastic simulation from a qualitative point of view with the reference tool at the state of the art.

REFERENCES

[1] Luca Albergante, Jon Timmis, Lynette Beattie, and Paul M. Kaye. 2013. A Petri net model of granulomatous inflammation: Implications for IL-10 mediated control of Leishmania donovani infection. *PLoS Computational Biology* 9, 11 (2013), e1003334.

[2] Pavel Balazki, Klaus Lindauer, Jens Einloft, Jörg Ackermann, and Ina Koch. 2015. MONALISA for stochastic simulations of Petri net models of biochemical systems. *BMC Bioinformatics* 16, 1 (2015), 215.

[3] Albert-Laszlo Barabasi and Zoltan N Oltvai. 2004. Network biology: Understanding the cell's functional organization. *Nature Reviews Genetics* 5, 2 (2004), 101.

[4] B. Behinaein, K. Rudie, and W. Sangrar. 2018. Petri net siphon analysis and graph theoretic measures for identifying combination therapies in cancer. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 15, 1 (2018), 231–243.

[5] Nicola Bombieri, Rosario Distefano, Giovanni Scardoni, Franco Fummi, Carlo Laudanna, and Rosalba Giugno. 2014. Dynamic modeling and simulation of leukocyte integrin activation through an electronic design automation framework. In *International Conference on Computational Methods in Systems Biology*. Springer, 143–154.

[6] N. Bombieri, F. Fummi, V. Guarnieri, G. Pravadelli, F. Stefanni, T. Ghasempouri, M. Lora, G. Auditore, and M. N. Marcigaglia. 2015. Reusing RTL assertion checkers for verification of SystemC TLM models. *Journal of Electronic Testing: Theory and Applications (JETTA)* 31, 2 (2015), 167–180.

[7] N. Bombieri, F. Fummi, and G. Pravadelli. 2006. On the evaluation of transactor-based verification for reusing TLM assertions and testbenches at RTL. In *Proc. of ACM/IEEE DATE*, Vol. 1. 1–6.

[8] M. Boulé and Z. Zilic. 2008. *Generating Hardware Assertion Checkers: For Hardware Verification, Emulation, Post-fabrication Debugging and On-line Monitoring*. Springer.

[9] S. Caligola, T. Carlucci, F. Fummi, C. Laudanna, G. Constantin, N. Bombieri, and R. Giugno. 2019. Efficient simulation and parametrization of stochastic Petri nets in SystemC: A case study from systems biology. In *Proc. of the 2019 Forum on Specification and Design Languages (FDL'19)*.

[10] Yang Cao, Hong Li, and Linda Petzold. 2004. Efficient formulation of the stochastic simulation algorithm for chemically reacting systems. *Journal of Chemical Physics* 121, 9 (2004), 4059–4067.

[11] Claudine Chaouiya. 2007. Petri net modelling of biological networks. *Briefings in Bioinformatics* 8, 4 (2007), 210–219.

[12] Ming Chen and Ralf Hofestaedt. 2003. Quantitative Petri net model of gene regulated metabolic networks in the cell. *In Silico Biology* 3, 3 (2003), 347–365.

[13] K.-T. Cheng and A. S. Krishnakumar. 1996. Automatic generation of functional vectors using the extended finite state machine model. *ACM TODAES* 1, 1 (1996), 57–79.

[14] I.-Chun Chou, Harald Martens, and Eberhard O. Voit. 2006. Parameter estimation in biochemical systems models with alternating regression. *Theoretical Biology and Medical Modelling* 3, 1 (2006), 25.

[15] D. Coati, R. Distefano, N. Bombieri, F. Fummi, M. Mirenda, C. Laudanna, and R. Giugno. 2016. A SystemC-based platform for assertion-based verification and mutation analysis in systems biology. *17th IEEE Latin-American Test Symposium (LATS'16)*, 159–164.

[16] Claudionor-Nunes Coelho Jr. and Harry D. Foster. 2008. *Assertion-Based Verification*. Vol. 4. Springer.

[17] R. Distefano, F. Fummi, C. Laudanna, N. Bombieri, and R. Giugno. 2015. A systemc platform for signal transduction modelling and simulation in systems biology. *In Proc. of the ACM Great Lakes Symposium on VLSI (GLSVLSI'15)*, 233–236.

[18] R. Distefano, N. Goncharenko, F. Fummi, R. Giugno, G. D. Badery, and N. Bombieri. 2016. SyQUAL: A platform for qualitative modelling and simulation of biological systems. In *2016 IEEE International High Level Design Validation and Test Workshop (HLDVT'16)*, 155–161.

[19] Cristian Dittamo and Davide Cangelosi. 2009. Optimized parallel implementation of Gillespie's first reaction method on graphics processing units. In *International Conference on Computer Modeling and Simulation (ICCMS'09)*. IEEE, 156–161.

[20] T. Eleftheriadis, G. Pissas, A. Karioti, G. Antoniadi, S. Golfinopoulos, V. Liakopoulos, A. Mamara, M. Speletas, G. Koukoulis, and I. Stefanidis. 2013. Uric acid induces caspase-1 activation, IL-1β secretion and P2X7 receptor dependent proliferation in primary human lymphocytes. *Hippokratia* 17, 2 (2013), 141.

[21] Philippe Darondeau, Eric Badouel, and Luca Bernardinello. 2015. *Petri Net Synthesis*. Springer-Verlag, Berlin. DOI: https://doi.org/10.1007/978-3-662-47967-4

[22] Jasmin Fisher and Thomas A. Henzinger. 2007. Executable cell biology. *Nature Biotechnology* 25 (2007), 1239–1249.

[23] Hartmann Genrich, Robert Küffner, and Klaus Voss. 2001. Executable Petri net models for the analysis of metabolic pathways. *International Journal on Software Tools for Technology Transfer (STTT)* 3, 4 (2001), 394–404.

[24] Daniel T. Gillespie. 1976. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *Journal of Computational Physics* 22, 4 (1976), 403–434.

[25] Daniel T. Gillespie. 1977. Exact stochastic simulation of coupled chemical reactions. *Journal of Physical Chemistry* 81, 25 (1977), 2340–2361.

[26] Gautam Goel, I.-Chun Chou, and Eberhard O. Voit. 2008. System estimation from metabolic time-series data. *Bioinformatics* 24, 21 (2008), 2505–2511.

[27] Anja Hartmann and Falk Schreiber. 2015. Integrative analysis of metabolic models–from structure to dynamics. *Frontiers in Bioengineering and Biotechnology* 2 (2015), 91.

[28] Monika Heiner, Mostafa Herajy, Fei Liu, Christian Rohr, and Martin Schwarick. 2012. Snoopy–a unifying Petri net tool. In *Int. Conf. on Application and Theory of Petri Nets and Concurrency*. Springer, 398–407.

[29] Monika Heiner and Ina Koch. 2004. Petri net based model validation in systems biology. In *ICATPN*, Vol. 3099. Springer, 216–237.

[30] Alexander Hoffmann, Andre Levchenko, Martin L. Scott, and David Baltimore. 2002. The IκB-NF-κB signaling module: Temporal control and selective gene activation. *Science* 298, 5596 (2002), 1241–1245.

[31] M. Hucka, A. Finney, H. M. Sauro, H. Bolouri, J. C. Doyle, H. Kitano, A. P. Arkin, B. J. Bornstein, D. Bray, A. Cornish-Bowden, A. A. Cuellar, S. Dronov, E. D. Gilles, M. Ginkel, V. Gor, I. I. Goryanin, W. J. Hedley, T. C. Hodgman, J.-H. Hofmeyr, P. J. Hunter, N. S. Juty, J. L. Kasberger, A. Kremling, U. Kummer, N. Le Novre, L. M. Loew, D. Lucio, P. Mendes, E. Minch, E. D. Mjolsness, Y. Nakayama, M. R. Nelson, P. F. Nielsen, T. Sakurada, J. C. Schaff, B. E. Shapiro, T. S. Shimizu, H. D. Spence, J. Stelling, K. Takahashi, M. Tomita, J. Wagner, and J. Wang. 2003. The systems biology markup language (SBML): A medium for representation and exchange of biochemical network models. *Bioinformatics* 19, 4 (2003), 524–531. DOI: https://doi.org/10.1093/bioinformatics/btg015 cited By 1920.

[32] IEEE. 2017. Property Specification Language - PSL. https://standards.ieee.org/findstds/standard/1850-2010.html.

[33] Hawoong Jeong, Sean P. Mason, A.-L. Barabási, and Zoltan N. Oltvai. 2001. Lethality and centrality in protein networks. *Nature* 411, 6833 (2001), 41.

[34] Steffen Klamt, Julio Saez-Rodriguez, Jonathan A. Lindquist, Luca Simeoni, and Ernst D. Gilles. 2006. A methodology for the structural and functional analysis of signaling and regulatory networks. *BMC Bioinformatics* 7, 1 (2006), 56.

[35] I. Koch. 2015. Petri nets in systems biology. *Software and Systems Modeling* 14, 2 (2015), 703–710.

[36] L. Napione et al. 2009. On the use of stochastic Petri nets in the analysis of signal transduction pathways for angiogenesis process. In *International Conference on Computational Methods in Systems Biology*. Springer, 281–295.

[37] Hiroshi Matsuno, Yukiko Tanaka, Hitoshi Aoshima, Mika Matsui, Satoru Miyano, et al. 2003. Biopathways representation and simulation on hybrid functional Petri net. *In Silico Biology* 3, 3 (2003), 389–404.

[38] James M. McCollum, Gregory D. Peterson, Chris D. Cox, Michael L. Simpson, and Nagiza F. Samatova. 2006. The sorting direct method for stochastic simulation of biochemical systems with varying reaction execution behavior. *Computational Biology and Chemistry* 30, 1 (2006), 39–49.

[39] Melody K. Morris, Julio Saez-Rodriguez, Peter K. Sorger, and Douglas A. Lauffenburger. 2010. Logic-based models for the analysis of cell signaling networks. *Biochemistry* 49, 15 (2010), 3216–3224.

[40] Jacek Puchałka and Andrzej M. Kierzek. 2004. Bridging the gap between stochastic and deterministic regimes in the kinetic simulations of the biochemical reaction networks. *Biophysical Journal* 86, 3 (2004), 1357–1372.
[41] G. Russo, M. Pennisi, R. Boscarino, and F. Pappalardo. 2018. Continuous Petri nets and microRNA analysis in melanoma. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 15, 5 (2018), 1492–1499.
[42] Andrea Sackmann, Dorota Formanowicz, Piotr Formanowicz, Ina Koch, and Jacek Blazewicz. 2007. An analysis of the Petri net based model of the human body iron homeostasis process. *Computational Biology and Chemistry* 31, 1 (2007), 1–10.
[43] Karsten Schmidt. 2005. Controllability of open workflow nets. In *EMISA*. 236–249.
[44] Ralf Steuer and Bjorn H. Junker. 2009. Computational models of metabolism: Stability and regulation in metabolic networks. *Advances in Chemical Physics* 142 (2009), 105.
[45] Synopsys Inc.2003. Assertion-Based Verification. White paper, http://www.synopsys.com.
[46] Systems Biology - A portal suite for Systems Biology. 2019. Model Repositories. http://systems-biology.org/resources/model-repositories.
[47] Vo Hong Thanh, Corrado Priami, and Roberto Zunino. 2014. Efficient rejection-based simulation of biochemical reactions with stochastic noise and delays. *Journal of Chemical Physics* 141, 13 (2014), 10B602_1.
[48] Siren R. Veflingstad, Jonas Almeida, and Eberhard O. Voit. 2004. Priming nonlinear searches for pathway identification. *Theoretical Biology and Medical Modelling* 1, 1 (2004), 8.
[49] L. Watanabe, T. Nguyen, M. Zhang, Z. Zundel, Z. Zhang, C. Madsen, N. Roehner, and C. Myers. 2019. IBIOSIM 3: A tool for model-based genetic circuit design. *ACS Synthetic Biology* 8, 7 (2019), 1560–1563. DOI: https://doi.org/10.1021/acssynbio.8b00078 cited By 2.
[50] Karsten Wolf. 2019. *How Petri Net Theory Serves Petri Net Model Checking: A Survey*. Springer, Berlin, 36–63. DOI: https://doi.org/10.1007/978-3-662-60651-3_2
[51] Y. Abarbanel et al. 2000. FOCS–Automatic generation of simulation checkers from formal specifications. In *Computer Aided Verification*. Springer, 538–542.
[52] Choujun Zhan and Lam F. Yeung. 2011. Parameter estimation in systems biology models using spline approximation. *BMC Systems Biology* 5, 1 (2011), 14.

Received January 2020; revised August 2020; accepted September 2020