View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Catalogo dei prodotti della ricerca

Dynamic Modeling and Simulation of Leukocyte Integrin Activation Through an Electronic Design Automation Framework

Nicola Bombieri[1], Rosario Distefano[1], Giovanni Scardoni[2], Franco Fummi[1], Carlo Laudanna[2], and Rosalba Giugno[3]

[1]Dept. Computer Science, University of Verona, Italy
[2]Dept. Patology and Diagnostics, University of Verona, Italy
[3]Dept. Clinical and Molecular Biomedicine, University of Catania, Italy
{nicola.bombieri,rosario.distefano,giovanni.scardoni,franco.fummi,carlo.laudanna}@univr.it,giugno@dmi.unict.it

Abstract. Model development and analysis of biological systems is recognized as a key requirement for integrating in-vitro and in-vivo experimental data. In-silico simulations of a biochemical model allows one to test different experimental conditions, helping in the discovery of the dynamics that regulate the system. Several characteristics and issues of biological system modeling are common to the electronics system modeling, such as concurrency, reactivity, abstraction levels, as well as state space explosion during verification. This paper proposes a modeling and simulation framework for discrete event-based execution of biochemical systems based on SystemC. SystemC is the reference language in the electronic design automation (EDA) field for modeling and verifying complex systems at different abstraction levels. SystemC-based verification is the de-facto an alternative to model checking when such a formal verification technique cannot deal with the state space complexity of the model. The paper presents how the framework has been applied to model the intracellular signalling network controlling integrin activation mediating leukocyte recruitment from the blood into the tissues, by handling the solution space complexity through different levels of simulation accuracy.

Keywords: Biochemical networks, Dynamic modeling and simulation, SystemC

1 Introduction

Cells are the fundamental units of the living organisms. They interact with the environment and with other cells by processing and exchanging environmental informations. Each different input coming from the environment produces a set of chemical reactions, which are the *answer* of the cell to the input. Those reactions depend on some parameters, such as the concentration of the reactants and the chemical properties regulating the reaction speed, and generate linear reaction pathways in turn organized in concurrent non-linear complex networks [16].

Dynamic network modeling in systems biology aims at describing how such interactions among defined elements determine the time course of the state of the elements, and of the whole system, under different conditions. A validated dynamic model that correctly captures experimentally observed normal behavior allows researchers to track the changes in the system due to perturbations, to discover possible covariation between coupled variables, and to identify conditions in which the dynamics of variables are qualitatively similar [19].

Mathematical models, such as those based on differential equations [7], have definitely gained consensus in the network modeling community as they have the highest potential to accurately describe the system. Nevertheless, since they have the highest requirement for input information, they are difficult to obtain and analyse if the number of independent variables grows and if the relationships depend on quantitative events, such as concentration reaching a threshold value.

Computational models, such as Boolean networks [25], Petri nets [10], interactive state machines [24], and Process Calculi [23], offer an effective alternative if precise quantitative relationships are unknown, if they involve many different variables, or if they change over time [14]. A common way to explain a certain class of complex dynamical systems is to view them as highly *concurrent reactive systems*. Hand-in-hand with the central notion of reactivity go (i) the discrete event-based execution and simulation of dynamical systems, which requires a fundamental understanding of parallelism, interaction, and causality; (ii) the design of complex systems from building blocks, requiring means for composition and encapsulation; and (iii) the description of systems at different levels of granularity, requiring methods for abstraction and refinement [13].

All these issues related to concurrent reactive systems have been largely addressed in the past decades in the electronic design automation (EDA) field and a large body of methodologies and tools are at the state of the art. In this context, SystemC [4] has become the de-facto reference standard language for system-level modelling and simulation of Hardware/Software/Network electronic systems at different abstraction levels [8].

In this paper, we propose a framework for modeling and simulation of biochemical networks based on SystemC. The framework relies on a state machine-based computational model to model the behavior of each network element. The element models are implemented and connected to realize a system-level network in SystemC. Finally, the network is connected to a stimuli generator and monitor of results to run a discrete and deterministic network simulation. To handle the complexity of exploring the solution space, the proposed framework allows us to discretize the range of the variable values with different levels of accuracy. In addition, the framework allows us to reuse existing EDA techniques and tools to parallelize the SystemC simulation, both on GPUs [22] and on clusters [12].

The paper presents how the framework has been applied to model the signaling network controlling LFA-1 beta2 integrin activation mediating leukocyte recruitment from the blood into the tissues, a central event during the immune response. Such a case study has been chosen for the large number of independent variables, for the lack of quantitative information such as molecular concentrations, activation and inhibition delays and lifetimes, and for the relationships strongly depending on qualitative events. The dynamic simulation of the model has been conducted with the aim of exploring the occurrence of emergent properties in signaling events controlling leukocyte recruitment, such as oscillating behaviors and, more in general, to help in better understanding the overall dynamics of leukocyte recruitment.

The paper is organized as follows. Section 2 summarizes the most important concepts and constructs of SystemC for modeling protein networks. Section 3 presents the leukocyte integrin activation case study. Section 4 presents the proposed framework, while Section 5 reports the obtained experimental results. Section 6 is devoted to concluding remarks.

2 Background on SystemC

SystemC [4] is a set of C++ classes and macros that provide an *event-driven* simulation interface in C++. These facilities enable a designer to simulate *concurrent processes*, each described using plain C++ syntax. SystemC processes can communicate in a simulated real-time environment, using signals of all the datatypes provided by C++, some additional ones provided by the SystemC library, as well as user defined.

SystemC has been applied to system-level modeling, architectural exploration, performance modeling, software development, functional verification, and high-level synthesis of digital circuits since 2000. Nowadays, SystemC is the defacto reference standard in the EDA community. SystemC is defined and promoted by the Open SystemC Initiative (OSCI) - Accellera Systems Initiative, and has been approved by the IEEE Standards Association as IEEE 1666-2005. The SystemC Language Reference Manual (LRM) [5] provides the definitive statement of the semantics of SystemC. OSCI also provides an open-source proof-of-concept simulator, which can be downloaded from the SystemC website [4]. Several optimized simulators are also available in the commerce [1, 3, 2].

SystemC offers a greater range of expression, similar to object-oriented design partitioning and template classes. Although strictly a C++ class library, SystemC is sometimes viewed as being a language in its own right. Source code can be compiled with the SystemC library (which includes a simulation kernel) to give an executable. SystemC allows designers to model systems at different abstraction levels (i.e., with different levels of details) by providing modeling features such as structural hierarchy and connectivity, communication abstraction, dynamic processes, timed event notifications, transaction-level modeling [9].

The most important language features, which have been used for modeling and simulating the signaling network presented in this paper are the following:

- *Modules.* They are the basic building blocks of a SystemC design hierarchy. A SystemC model usually consists of several modules that communicate via ports. As explained in the following sections, each network element (i.e., protein and cofactor) has been modelled as a module, and all the elements have been hierarchically organized into a module representing the whole network.

- *Ports.* They allow communication from inside a module to the outside (usually to other modules) via signals.
- *Signals.* They are the communication elements of SystemC models. They have been used to model the activation/inhibition activity between elements.
- *Processes.* They are the main computation elements and they are concurrent. Each protein behaviour has been modelled through a process, which reacts to any activation or inhibition by an upstream protein and, in turn, activates or inhibits a downstream protein.
- *Events.* They allow for synchronization between processes. Events are the key objects in SystemC models to provide event-driven simulation.

3 The case study

In order to better explain how the proposed framework can be applied for modelling and simulation of signaling networks, we first present the case study, which will be used as a model system in the subsequent sections.

As a model system, we analysed the signaling mechanism controlling beta2 integrin LFA-1 affinity regulation by chemokines, a crucial event mandatory to the fulfilment of the leukocyte recruitment process from the blood into the tissues. This process is critical to immune system function and is modeled as a concurrent ensemble of leukocyte behaviors under flow, including tethering, rolling, firm adhesion, crawling, and transmigration [18]. A central step is the integrin-mediated arrest, comprising a series of adhesive events including increase of integrin affinity, valency and binding stabilization altogether controlling cell avidity. In this context, modulation of integrin affinity is widely recognized as the prominent event in rapid leukocyte arrest induced by chemokines [6, 11, 15, 17]. Regulation of integrin activation depends on a plethora of signaling proteins [6]. At least 67 signaling molecules modulate integrin activity by chemokines [21, 20]. In this context, we have previously described an integrated group of signaling proteins including RhoA, Rac1 and CDC42 small GTPases, along with the two effectors PLD1 and PIP5K1C, modulating conformer-selective LFA-1 affinity triggering and homing to secondary lymphoid organs by chemokines of human primary lymphocytes [6]. To date, signaling by rho- and rap-small GTPases are the best-studied mechanisms of integrin activation by chemokines.

Furthermore, and more recently, we have demonstrated that, in human primary T lymphocytes, chemokines control integrin affinity triggering and in vivo homing by means of tyrosine kinases of the JAK family acting as upstream transducer linking chemokine receptors to the activation of the rho and rap module of integrin [20]. Overall, an integrated macro module comprising JAKs, rho and rap small GTPases and a variety of upstream regulators and downstream effectors finely control integrin triggering and mediated lymphocyte recruitment by chemokines. Beside arrest under flow, integrin activation is also critical to support leukocyte crawling and transmigration (diapedesis) along with directional

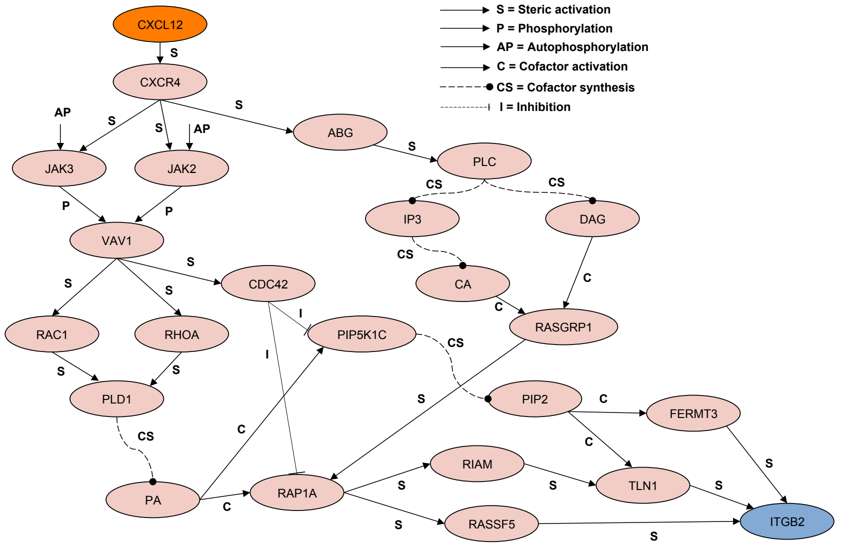

Fig. 1. The protein-protein interaction network of the leukocite integrin activation

movement toward a gradient of chemotactic factors that is chemotaxis. Figure 1 depicts the protein network and each different interaction between proteins or cofactors of the case study.

Notably, cell motility needs an on-off kinetic of integrin activation, allowing cycling between adhesion and de-adhesion event thus ensuring cell movement. Thus, control of the duration of cell adhesion is critical to control cell migration. This on-off, oscillatory, kinetics of integrin triggering likely depends on on-off kinetics of the signaling transduction machinery triggered by chemokines and controlling integrin-mediated cell adhesion. This suggests an equal relevance for both activators as well as inhibitors on integrin triggering. Although negative regulators of cell adhesion have been described, a comprehensive dynamic model of signaling events controlling on-off cycling of integrin activation is still lacking. Such a modeling is an important approach to explore the occurrence of emergent properties in signaling events controlling leukocyte recruitment, such as oscillating behaviors characterized by frequency and amplitude of agonist triggering. In turn, identification of these properties could help to better understand the overall dynamics of leukocyte recruitment.

4 The SystemC Framework for Modelling and Simulation of the Protein Network

The framework relies on three main steps. First, the behavior of each network element (i.e., protein and cofactor) is modeled through the Finite State Machine (FSM) formal model. The element models are then implemented in SystemC

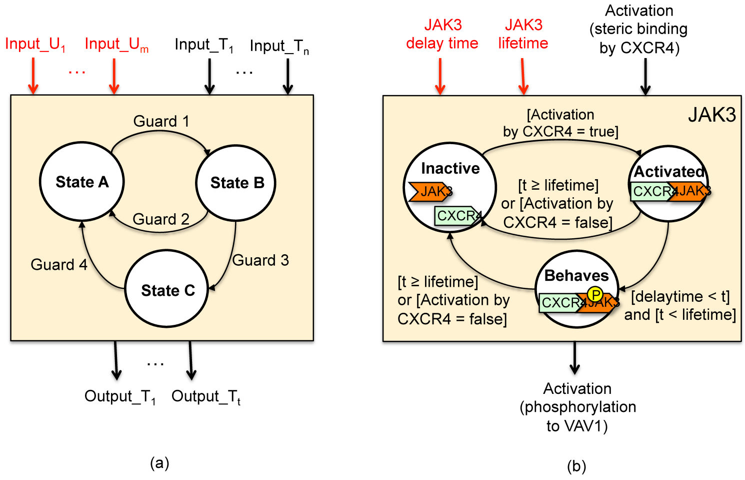
Input_U$_1$
…
Input_U$_m$
Input_T$_1$
…
Input_T$_n$
Guard 1
State A
State B
Guard 2
Guard 4
Guard 3
State C
…
Output_T$_1$
Output_T$_t$
(a)
JAK3 delay time
JAK3 lifetime
Activation (steric binding by CXCR4)
JAK3
[Activation by CXCR4 = true]
Inactive
JAK3
CXCR4
[t ≥ lifetime] or [Activation by CXCR4 = false]
Activated
CXCR4
JAK3
[t ≥ lifetime] or [Activation by CXCR4 = false]
Behaves
P
CXCR4
JAK3
[delaytime < t] and [t < lifetime]
Activation (phosphorylation to VAV1)
(b)

Fig. 2. The protein behavior representation through Finite State Machines. The protein template (a), and the JAK3 example

modules and connected through SystemC *signals* to realize a system-level network. Finally, the network is connected to a stimuli generator and monitor of results to run a reactive, event-driven network simulation.

4.1 Modelling proteins through Finite State Machines

The finite state machine model allows us to formally model each protein behavior and, similarly, each cofactor behaviour, in terms of states (e.g., inactive, activated/inhibited, activating/inhibiting, etc.), transitions between states, and guard conditions (i.e., boolean conditions).

Figure 2(a) depicts the proposed FSM template, while Figure 2(b) shows a modelling example of the JAK3 protein of the case study in Figure 1. Each protein changes state (i.e., a transition occurs) when the guard condition is evaluated to be true. The condition may be set on a particular reaction *event* (e.g., activation via phosphorylation, steric, auto-phosphorylation, cofactor or inhibition via phosphatase) generated by any upstream protein or on any environment status. As an example, JAK3 moves from the inactive state to the activated state (which represents the steric binding with CXCR4) as soon as CXCR4 activates JAK3. Once activated, JAK3 seeks for the phosphorylation of its own protein target (VAV1), which occurs after a *delay time* (i.e., the time spent to encounter a molecule of VAV1, to pick up an atom of phosphorus from an ATP molecule, and to add it to VAV1). *t* represents the time elapsed, which is constantly updated during simulation, while *lifetime* represents the maximum lifetime from the activation instant in which the protein carries out its biological function. JAK3 continues to phosphorylate new VAV1 molecules (*Behaves* state) as long as it is bounded with CXCR4 and the lifetime has not expired.

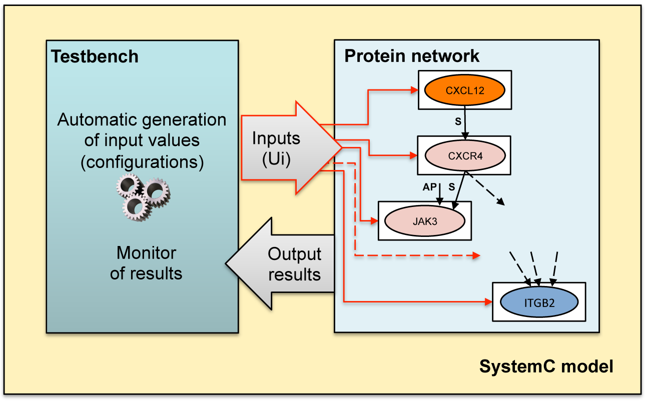

Fig. 3. The SystemC framework

The template distinguishes two sets of input data that can affect the model behavior and one set of generated output:

- *Topological inputs (Input_Ti)*: They are inputs whose values are calculated at simulation time and depend on the topological interaction of the modelled protein with upstream proteins. Some examples are the activation via phosphorylation, steric, cofactor, or inhibition.
- *Unknown inputs (Input_Ui)*: They are inputs whose values depends on the environment characteristics and status, which are unknown at modeling time. Some examples are the *delay time* (i.e., time spent by the protein to encounter a protein target), the molecular concentrations of the downstream proteins (which affect the delay time), the protein lifetime, etc. For each unknown input, the framework generates different values with the aim of observing, via simulation, how such values affect the system dynamics.
- *Topological outputs (Output_Ti)*: They are outputs whose values are calculated at simulation time and depend on the role of the protein towards downstream proteins (e.g., the ouput of the JAK3 module is set to true when JAK3 encounters and activates VAV1 via phosphorylation) .

4.2 Implementation of the protein models through SystemC

Each protein is implemented through a SystemC *module*, with both the topological and unknown inputs and outputs as SystemC *ports* (see Section 2). The protein behavior represented by FSM in Figure 2 is implemented through a SystemC *process*, which is sensitive to any *event* on the input signals. An activation/inhibition from an upstream protein is represented by an input (boolean) signal set to *true*. Being event-driven, the process *wakes up* and updates both the internal state and the output signals whenever a new event on inputs occurs.

The network model consists of every protein modules connected via SystemC signals (see right-most side Figure 3, which, for the sake of clarity, reports a part of the network).

The protein network is connected to a *testbench*, which generates the values for the unknown inputs of the protein network. The set of all input values represents a *configuration.* The testbench generates a configuration and runs (i.e., executes) a dynamic simulation of the network behavior for such a set of input values for a given simulation time. Then, the testbench generates a new different configuration for a new simulation. The run ends when all the possible configurations have been simulated.

The testbench also implements a monitor of results, which controls whether any condition or behavior of the network occurs, in order to identify which configurations have led to such a behavior. In the proposed case study, the monitored condition consists of the on-off, oscillatory, kinetics of integrin triggering represented, in the model, by the oscillatory state of ITGB2 between inactive and activate affinity state. Particularly, the monitoring activity of the testbench aims at identifying which configurations, in terms of protein lifetime, activation delays, and protein concentrations lead to a given number of oscillations, with a given oscillation period.

4.3 Simulation of the system-level network

The main problem in exploring the dynamics of protein networks is the complexity of the solution space. The solution space, that is, the number of configurations to simulate, grows exponentially over the number of unknown inputs. In addition, several inputs are continuous magnitudes (e.g., delay and lifetime), which would lead to an intractable problem if not properly discretized.

To handle such a complexity, the proposed framework allows us to discretize the range of the input values with different levels of accuracy. As an example, the lifetime of CDC42 in Figure 1 is an unknown input, whose value has to be generated by the testbench. Different values have been simulated, starting form a minimum to a maximum value, by steps of a given time period. The finer the step, the more accurate the space solution exploration, and, on the other hand, the higher the configuration number and the consequent overall simulation time. EDA techniques and tools at the state of the art can be applied to parallelize the SystemC simulation, both on GPUs [22] and on clusters [12] in order to improve the accuracy over the simulation time ratio.

In general, given a number of network elements, n, the total number of input configurations to be generated by the testbench is the following:

$$\prod_{i=1}^{n}(\frac{MConcentration_i}{MCStep_i})(Targets_i)(\frac{MaxDelayT_i-MinDelayT_i}{DelayStep_i})(\frac{MaxLifeT_i-MinLifeT_i}{LifetimeStep_i})$$

where *MConcentration* represents the molecular concentration of the protein (or cofactor), *Targets* represents the number of the downstream targets, *Max* and *MinDelayT* represent the observed range of delay time, *Max* and *MinLifeT* represent the range of the lifetime, while *MCStep*, *DelayStep*, and *LifetimeStep* represent the chosen periods of discretization.

The overall simulation time is linear over the number of configurations. It is possible to associate, before simulation, the required time for simulating the network dynamics with a chosen space exploration accuracy. In addition, parameters *MaxDelayT*, *MinDelayT*, *MaxLifeT*, *MinLifeT*, *MCStep*, *DelayStep*, and *LifetimeStep* can be tuned for each single element of the network. This allows us to explore, with different levels of detail, the behavior and the influence of each protein in the overall network dynamics.

The modular structure of the framework allows us to adopt different simulation models (e.g., stochastic simulations), by modifying the testbench module. The development of a testbench for stochastic simulations with the aim of relaxing the constraints on the input values is part of our current and future work.

5 Experimental Results

The case study presented in Section 3 has been implemented in SystemC with the aim of exploring pro-adhesive signaling events and to better understand the overall dynamics of leukocyte recruitment.

The main goal of the model simulation was identifying the system properties that lead to oscillating behaviors, which are characterized by frequency and amplitude of integrin triggering. In particular, the testbench has been implemented to monitor which configurations of input values lead to oscillations of ITGB2 with a period of 30-40 ms (15-20 ms in active state, 15-20 ms inactive state), which represents the average stopping time of a cell when it interacts with the blood vessel epithelium. Notably, although accurate experimental measurement of on-off dynamics of integrin triggering is, at the present, unavailable, the extremely rapid kinetics of leukocyte arrest under-flow conditions, occurring in the experimentally-determined range of few milliseconds clearly suggest that it is reasonable to consider this rapid time-frame as a correct reference time to simulate on-off dynamics of integrin triggering. Furthermore, since directional leukocyte motility (chemotaixs) appears to maintain constant speed, at least in the context of a chemotactic gradient, it is reasonable to expect the emergence of regular oscillatory dynamics of signaling mechanisms controlling integrin triggering.

In order to reduce the explosion of the exploration space, we assumed the following characteristics of the system, which are summarized in Table 1. Each protein and cofactor (listed in Table 1 with (P) and (C), respectively) have been simulated with three different molecular concentrations (1, half, and maximum molecular number). The delay time of each element has been fixed as a function of the molecular concentration of the target element, with minimum value equal to 2 ms.

The lifetime of each single protein (cofactor) has been explored by discretizing the time intervals, which have been fixed for each element as shown in the table.

	Unknown inputs				**Topological signals**	
	MConcentration (# molecules)	**downstream targets (#)**	**delay time (ms)**	**lifetime (ms)**	**inputs**	**outputs**
CXCL12 (P)	[1,400]	[1,1]	-	[250,250]	-	sig_CXCR4
CXCR4 (P)	[1,325]	[1,3]	[2,3]	[250,250]	sig_CXCL12	sig_JAK3 sig_JAK2 sig_ABG
JAK3 (P)	[1,300]	[1,1]	[2,5]	[250,250]	sig_CXCR4	pho_VAV1
JAK2 (P)	[1,175]	[1,1]	[2,5]	[42,42]	sig_CXCR4	pho_VAV1
ABG (P)	[1,200]	[1,1]	[2,5]	[31,37]	sig_CXCR4	sig_PLC
VAV1 (P)	[1,168]	[1,3]	[2,2]	[45,51]	pho_JAK3 pho_JAK2	sig_RAC1 sig_RHOA sig_CDC42
RAC1 (P)	[1,235]	[1,1]	[2,6]	[34,40]	sig_VAV1	sig_PLD1
RHOA (P)	[1,146]	[1,1]	[2,6]	[29,35]	sig_VAV1	sig_PLD1
CDC42 (P)	[1,256]	[1,2]	[2,2]	[35,41]	sig_VAV1	sig_PIP5K1C sig_RAP1A
PLC (P)	[1,210]	[1,2]	[2,4]	[33,33]	sig_ABG	syn_IP3 syn_DAG
IP3 (C)	[1,115]	[1,1]	[2,5]	[51,57]	syn_PLC	syn_CA
CA (C)	[1,140]	[1,1]	[2,5]	[44,50]	syn_IP3	sig_RASGRP1
DAG (C)	[1,123]	[1,1]	[2,5]	[56,62]	syn_PLC	sig_RASGRP1
RASGRP1 (P)	[1,127]	[1,1]	[2,4]	[32,38]	sig_CA sig_DAG	sig_RAP1A
PLD1 (P)	[1,67]	[1,1]	[2,4]	[28,28]	sig_RAC1 sig_RHOA	sig_PA
PIP5K1C (P)	[1,234]	[1,1]	[2,4]	[27,33]	sig_CDC42 sig_PA	sys_PIP2
PA (C)	[1,322]	[1,2]	[2,2]	[63,69]	sys_PLD1	sig_RAP1A sig_PIP5K1C
RAP1A (P)	[1,364]	[1,2]	[2,2]	[34,40]	sig_PA sig_RASGRP1 sig_CDC42	sig_RASSF5 sig_RIAM
PIP2 (C)	[1,243]	[1,2]	[2,3]	[55,61]	sys_PIP5K1C	sig_FERMT3 sig_TLN1
RIAM (P)	[1,435]	[1,1]	[2,4]	[39,39]	sig_RAP1A	sig_TLN1
RASSF5 (P)	[1,134]	[1,1]	[2,5]	[32,38]	sig_RAP1A	sig_ITGB2
FERMT3 (P)	[1,123]	[1,1]	[2,5]	[31,31]	sig_PIP2	sig_ITGB2
TLN1 (P)	[1,364]	[1,1]	[2,5]	[36,36]	sig_PIP2	sig_ITGB2
ITGB2 (P)	[1,125]	-	-	[43,49]	sig_FERMT3 sig_TLN1 sig_RASSF5	-

Table 1. The protein network characteristics

To better explore the behavior of the most interesting proteins of the network (e.g., CDC42, RAP1A, and PIP5K1C that can lead to oscillations upon inhibition), the lifetime ranges explored in simulation for such elements have been extended. JAK3 and JAK2 have a fixed lifetime (40 ms and 42 ms, respectively) since it has been accepted that, at present, there is not a known phosphatase process that can influence their behavior.

Each protein or cofactor can activate (inhibit) one target at a time. Activation (inhibition) of different targets are explored through different configurations. As an example, VAV1 activates either RAC1 or RHOA or CDC42 (see Figure 1) in a configuration run. Activation of all the targets is guaranteed and covered in different configuration runs.

For each configuration, the network dynamics has been simulated and monitored for a total time of 250 ms. For each configuration run, CXCL12 is always active. CXCL12 and ITGB2 have not delay time. In total, we run around seven billion configurations on a cluster of 16 dual-core CPUs for a total of 278 hours run time. As a result, we filtered the configurations that lead to periodic oscillations (0.06% of the total) from the configurations that lead to aperiodic oscillations or no oscillation (41.7% and 58.33%, respectively). Among the periodic oscillations, the majority of configurations (57.75%) lead to three oscillations in the overall simulated time (250 ms), 21% oscillations lead to two oscillations, while 11.14% and 9.45% lead to five and four oscillations, respectively. Such configurations represent different settings of the *unknown inputs* (see Section 4.1) that lead the model behavior close enough to what experimentally observed. These results encourage us for further model refinements and deeper investigations of the case study.

6 Concluding remarks

The paper presented a SystemC-based framework for modeling and simulation of the signaling network controlling LFA-1 beta2 integrin activation mediating leukocyte recruitment from the blood into the tissues. The framework relies on the FSM model to formally model the behavior of each network element and on the SystemC EDA language, which allows us to implement the network elements as concurrent and reactive processes. The framework also consists of a testbench, which generates *configurations* of values for each unknown parameters (e.g., molecular concentrations, activation delays, etc.). The framework simulates the system for each configuration to identify the system properties that lead to any experimentally observed behavior, such as the periodic oscillations of ITGB2 in the leukocyte integrin activation case study. The proposed approach allows us to handle the solution space complexity through different levels of simulation accuracy and to apply EDA techniques and tools at the state of the art to parallelize the SystemC simulation, both on GPUs and on clusters, to improve the accuracy over the simulation time ratio.

References

1. Cadence Palladium - System Design and Verification, `http://www.cadence.com/products/sd/Pages/default.aspx`
2. Mentor Graphics SystemVisio, `http://www.mentor.com/products/sm/`
3. Synopsys System Studio, `http://www.synopsys.com/Systems/Pages/default.aspx`
4. SystemC - Accellera Systems Initiative, `http://www.systemc.org`
5. IEEE 1666 Standard: SystemC Language Reference Manual. http://ieeexplore.ieee.org (2011)
6. Bolomini-Vittori, M., Montresor, A., Giagulli, C., Staunton, D., Rossi, B., Martinello, M., Constantin, G., Laudanna, C.: Regulation of conformer-specific activation of the integrin lfa-1 by a chemokine-triggered rho signaling module. Nat Immunol. 10, 185–194 (2009)
7. Butcher, J.C.: Numerical Methods for Ordinary Differential Equations. Wiley, Chichester, UK (2003)

8. Cai, L., Gajski, D.: Transaction level modeling: An overview. In: ACM/IEEE CODES+ISSS. pp. 19–24 (2003)
9. Cai, L., Gajski, D.: Transaction level modeling: An overview. In: Proceedings of the 1st IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis. pp. 19–24. CODES+ISSS (2003)
10. Chaouiya, C.: Petri net modelling of biological networks. Briefings in Bioinformatics 8(4), 210–9 (2007)
11. Constantin, G., Majeed, M., Giagulli, C., Piccio, L., Kim, J., Butcher, E., Laudanna, C.: Chemokines trigger immediate beta2 integrin affinity and mobility changes: differential regulation and roles in lymphocyte arrest under flow. Immunity 13, 759–769 (2000)
12. Ezudheen, P., Chandran, P., Chandra, J., Simon, B.P., Ravi, D.: Parallelizing SystemC kernel for fast hardware simulation on SMP machines. In: Proc. of ACM/IEEE PADS. pp. 80–87 (2009)
13. Fisher, J., Harel, D., Henzinger, T.A.: Biology as reactivity. Commun. ACM 54(10), 72–82 (Oct 2011)
14. Fisher, J., Henzinger, T.A.: Executable cell biology. Nature Biotechnology 25, 1239 – 1249 (2007)
15. Giagulli, C., Ottoboni, L., Caveggion, E., Rossi, B., Lowell, C., Constantin, G., Laudanna, C., Berton, G.: The src family kinases hck and fgr are dispensable for inside-out, chemoattractant-induced signaling regulating beta 2 integrin affinity and valency in neutrophils, but are required for beta 2 integrin-mediated outside-in signaling involved in sustained adhesion. J. Immunol. 177, 604–611 (2006)
16. Gilbert, D., Fuss, H., Gu, X., Orton, R., Robinson, S., Vyshemirsky, V., Kurth, M.J., Downes, C.S., Dubitzky, W.: Computational methodologies for modelling, analysis and simulation of signalling networks. Briefings in Bioinformatics 7(4), 339–353 (2006)
17. Kim, M., Carman, C., Yang, W., Salas, A., Springer, T.: The primacy of affinity over clustering in regulation of adhesiveness of the integrin $\alpha l\beta 2$. J Cell Biol. 167, 1241–1253 (2004)
18. Ley, K., Laudanna, C., Cybulsky, M., Nourshargh, S.: Getting to the site of inflammation: the leukocyte adhesion cascade updated. Nat. Rev. Immunol. 7, 678–689 (2007)
19. Melham, T.: Modelling, abstraction, and computation in systems biology: A view from computer science. Progress in Biophysics and Molecular Biology, vol. 111, pp. 129 – 136 (Apr 2013)
20. Montresor, A., Bolomini-Vittori, M., Toffali, L., Rossi, B., Constantin, G., Laudanna, C.: Jak tyrosine kinases promote hierarchical activation of rho and rap modules of integrin activation. J Cell Biol. 203(6), 1003–1019 (2013)
21. Montresor, A., Toffali, L., Constantin, G., Laudanna, C.: Chemokines and the signaling modules regulating integrin affinity. Front Immunol. 3:127 (2012)
22. Nanjundappa, M., Patel, H.D., Jose, B.A., Shukla, S.K.: Scgpsim: a fast systemc simulator on gpus. In: Proceedings of the 2010 Asia and South Pacific Design Automation Conference. pp. 149–154. ASPDAC '10 (2010)
23. Priami, C.: Stochastic pi-calculus. The Computer Journal 38(7), 578–589 (1995)
24. Sadot, A., Fisher, J., Barak, D., Admanit, Y., Stern, M.J., Hubbard, E.J., Harel, D.: Toward verified biological models. IEEE/ACM Transactions on Computational Biology and Bioinformatics 5(2), 223–34 (2008)
25. Srihari, S., Raman, V., Leong, H.W., Ragan, M.A.: Evolution and controllability of cancer networks: A boolean perspective. IEEE/ACM Transactions on Computational Biology and Bioinformatics 11(1), 83–94 (2013)