CBNSimulator: a simulator tool for understanding the behaviour of complex biomolecular networks using discrete time simulation

Ali Ayadi, François de Bertrand de Beuvron, Cecilia Zanni-Merk, Saoussen Krichen

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 112 (2017) 514–523

Procedia Computer Science

www.elsevier.com/locate/procedia

21th International Conference on Knowledge Based and Intelligent Information and Engineering Systems

CBNSimulator: a simulator tool for understanding the behaviour of complex biomolecular networks using discrete time simulation

Ali Ayadi[a,c,*], François de Bertrand de Beuvron[a], Cecilia Zanni-Merk[b], Saoussen Krichen[c]

[a] *ICUBE/SDC Team (UMR CNRS 7357)-Pole API BP 10413, Illkirch 67412, France*

[b] *LITIS Laboratory, Fédération CNRS Norm@STIC FR 3638, INSA de Rouen Normandie, Avenue de l'Université, 76801 Saint-Etienne-du Rouvray, France*

[c] *LARODEC Laboratory, Institut Supérieur de Gestion de Tunis, University of Tunis, Rue de la liberté, 2000 Bardo, Tunisia*

Abstract

Because of the lack of satisfactory solutions to explain biological systems, biologists usually focus on modelling and simulation tools to understand the behaviour of these complex organisms. Indeed, computational modelling and simulation of cells plays a pivotal role in systems biology. In this paper, we tackle the problem of studying the behaviour of human cells by reproducing the behaviour of complex biomolecular networks. To this end, we present in this paper an approach for simulating complex biomolecular networks inspired by the discrete-event simulation model (DEVS), a formalism developed for supporting the modelling of complex systems. In this paper, we propose a simulation tool, named "CBNSimulator", based on a logical model of the biomolecular network and taking advantage of the performance of a discrete-time simulation model for understanding the evolution and the behaviour of complex biomolecular networks as a discrete sequence of events in time. The proposed tool has been applied to the case study of a ribosomal protein regulation network, named "the bacteriophage T4 gene 32", and results given by this simulation tool are in agreement with the expert's judgement. Moreover, the graphical user interface of CBNSimulator allows biologists to easily reproduce, analyse and understand behaviour of complex biomolecular networks through discrete simulation.

Peer-review under responsibility of KES International

Keywords: Complex biomolecular network , modelling and simulation, DEVS, discrete-time algorithm, CBNSimulator.

1. Introduction

Systems Biology is aimed at analyzing the behaviour and interrelationships of biological systems and is characterized by combining experimentation, theory, and computation [1,2]. Indeed, due to the lack of sufficient solutions to explain the biological systems, biologists usually focus on modelling and simulation tools to understand the behaviour of these complex systems. Biological systems are considered by Ren et al. in [3] as "Complex systems characterized by nonlinearity, i.e., emergence, which also means 'the whole is not equal to the sum of its parts'. The whole behavior of the biological system is an emergent behavior of many local components' interactions." It means, that the entire

* Corresponding author. Tel.: +33 6 56 76 34 46.
E-mail address: ali.ayadi@unistra.fr

1877-0509 © 2017 The Authors. Published by Elsevier B.V.
Peer-review under responsibility of KES International
10.1016/j.procs.2017.08.157

behaviour of a biological system, such as a cell, emerges from interactions among several molecular components such as, DNA, RNA, proteins and metabolites. In fact, each component has its simple behavioural rules, whereas a set of components can produce an emergent complex behaviour. This summarises the challenges faced by modelling and simulation tools to explain and understand the behaviour of complex organisms.

This issue has already been addressed in Wu et al.'s research[4], where they introduce and define the transittability of complex biomolecular networks. Indeed, the transittability expresses the idea of steering the complex biomolecular network from an unexpected state to a desired state[4]. Our work belongs to this context with the novelty of developing a platform to simulate the state changes of the complex biomolecular networks with the hope of understanding their transittability. Precisely, this platform will allow the computation of an optimal set of external stimuli to be applied during a predetermined time interval to steer the biomolecular network from its current state to a desired state[5].

Moreover, the complexity of biomolecular networks is firstly due to its large number of coupled components, but also to the diversity of these components and to their intricate interactions[6]. Thus, biologists require tools that will allow them to gain insights into the behaviour of complex biomolecular networks by simulating the different states of their components over time.

The contribution of this paper is to propose a simulator tool able to reproduce the behaviour of complex biomolecular networks and their components over time. This simulator is based on the combination of a logical-based modelling of complex biomolecular networks (which is presented in our previous work[7]) and a discrete time simulation algorithm inspired by the discrete event modelling and simulation environment (DEVS)[8].

The remaining of this paper is organized as follows. In Section 2 we describe some related work that tackle the same problem but with different approaches. In Section 3 we provide some background knowledge of the study by presenting the complex biomolecular networks and introducing the notions of modelling and simulation in systems biology. The DEVS formalism and the DEVSjava environment are also described. We then present our proposed approach in Section 4. In section 5, we give a case study and make experiments to evaluate the performance of our simulator tool. Finally, we provide the concluding remarks and discuss about future work in Section 6.

2. Related work

There have been a some approaches dedicated to the simulation of biological networks[9]. In this section, we highlight several kinds of simulation tools have been proposed in the recent years[10].

Simulation tools based on stochastic simulation algorithm are widely used. Within the literature there have been many proposed works using this method such as Spiral[11], COPASI[12], Jdesigner[13], CellDesigner[14], CellIllustrator[15], works of Frazier et al.[16], and Voliotis et al. . Many others specialize in simulating Gene Regulatory networks, i.e.TinkerCell[17], GenoCAD[18], etc. Szekely and Burrage[19] presented a critical review of stochastic simulation methods in systems biology describing their advantages and disadvantages. These stochastic methods propose the use of a mathematical model that is composed of equations for simulating the behaviour of biomolecular systems. The numerical solutions to these equations is sometimes complicated to solve considering the high number of molecular components and their heterogeneity. Moreover, these stochastic methods do not follow individuals over time, instead, they track only the total populations. They also consider that the interactions among molecular components are homogeneous and assume that the entire system is just the sum of its components, which is not necessarily true. Other simulation tools based on boolean networks such as the one of Mizera et al.[20] were developed for the identification of disease-associated genes. These simulation tools model the molecular components by boolean variables that represent active and inactive states. At each time step, the state of each gene is determined by a logic rule which is a function of the state of its regulators and the state of all genes forms a global state that changes synchronously[21]. This type of approach is suited to simulate small network and particularly Gene Regulatory Networks and signalling pathways. However, it becomes impractical to simulate large biomolecular network (for n nodes, we have 2^n possible states).

It is to be remarked that no method can be said to be better than the rest, only more suitable for a certain problem. Each of these tools has its own uses, and is best suited for solving problems of certain scale and complexity. Most of these methods consider the biomolecular network as a simple network, usually taking account only one level and by only focusing on modelling isolated parts of this network. However, the interconnection among different network levels reflects the importance of a general approach that focuses on the multiscale properties of a biomolecular network to replace these traditional reductionist methods. The difference between our works and these approaches interest

is the level at which we view the biomolecular network and the information in which we are interested. Rather than focusing on traditional reductionist methods, we think that the behaviour of the complex biomolecular network emerges from the network-level interaction and requires an integrative simulation tool. Thus, we propose the first version of a simulator tool – the CBNSimulator – that is specifically designed to reproduce the states of the different biomolecular network components.

3. Background

3.1. DEVS formalism

Discrete event modelling and simulation environment (DEVS) defines a syntax based on the formalization of systems. This formalization itself take its origin in discrete mathematics. In DEVS, the representation of time is essential. Indeed, in a model with discrete events, it is the occurrences of events that determine the progress of time. We present the basic concepts of the devs formalism using the DEVS's terms as defined by Zeigler et al. in [8].

3.1.1. DEVS atomic model

A *DEVS atomic model* describes a simple system and is defined as: $M = \langle X, Y, S, \delta_{int}, \delta_{ext}, \delta_{con}, \lambda, ta \rangle$ where: $X = \{(p,v)|p \in IPorts, v \in V_X\}$ the input events set of values, with V_X the set of possible values on the input ports and $IPorts$ the set of names of the input ports through which external events are received; $Y = \{(p,v)|p \in OPorts, v \in V_Y\}$ the ouput events set of values, with V_Y the set of possible values on the output ports and $OPorts$ the set of names of the output ports through which external events are sent; S the set of system state values; $\delta_{int} : S \to S$ the *internal transition* function, which specifies to which next state the system will transit after the time given by the time advance function has elapsed; $\delta_{ext} : Q \times X^b \to S$ the *external transition* function, which specifies how the system changes state when an input is received. The next state is computed on the basis of the present state, the input port and value of the external event, and the time that has elapsed in the current state, with $Q = \{(s,e)|s \in S, 0 \leq e \leq ta(s)\}$ is the *total-state set*, and e is *time elapsed* since last transition. X^b denotes the collection of bags over X (sets in which some elements may occur more than once); $\delta_{con} : Q \times X^b \to S$ the *confluent transition* function, which is applied when an input is received at the same time that an internal transition is to occur – the default definition simply applies the internal transition function before applying the external transition function to the resulting sate; $\lambda : S \to Y^b$ the *output* function, which generates an external output just before an internal transition takes place; $ta : S \to R^+_{0,\infty}$ the *time advance* function which represent the liftetime of a state s ($ta(s)$ might be a real number, 0 and ∞) and controls the timing of internal transitions.

3.1.2. DEVS coupled model

A *DEVS coupled model* is the composition of several sub-models which can be atomic and/or coupled. These sub-models are connected together via their input and output ports. A DEVS coupled model is formally defined as: $N = \langle X, Y, D, \{M_d|d \in D\}, EIC, EOC, IC \rangle$ where: X is the set of input ports for the reception of external events; Y is the set of output ports for the emission of external events; D is an index of the set of components of a coupled model, M_d is the DEVS model for each $d \in D$ (atomic or coupled); EIC is *the external input coupling* representing the set of input links that specifies the connections among external and component inputs. It connects the inputs of the coupled model to one or more of the inputs of the components that it contains; EOC is *the external output coupling* representing the set of output links that describes the connections among component and external outputs. It connects the outputs of one or more of the contained components to the output of the coupled model; IC is *the internal coupling* representing the set of internal links that ensures the connections among components themselves. It connects the output ports of the components to the input ports of the components in the coupled models.

4. Simulation model

This section starts by a brief recall of the logical formalism of complex biomolecular networks by describing its structural, functional and behavioural modelling (Figure 1). Then it introduces how this logical modelling is associated with a discrete time simulation algorithm based on the DEVS formalism.

4.1. Complex Biomolecular Networks

The cell is a complex system consisting of thousands of diverse molecular entities (genes, proteins and metabolites) which interact with each other physically, functionally and logically creating a biomolecular network[4,22]. The complexity of the biomolecular network appears by its decomposition into three levels: the genome level models the genetic material of an organism, the proteome level describes the entire set of proteins and the metabolism level contains the complete set of small-molecule chemicals[23]. Depending on the type of its cellular components and their interactions, we can distinguish the three basic types of networks: the Gene Regulatory networks (GRNs), the Protein-Protein-Interaction networks (PPINs) and the Metabolic networks (MNs), that were logically and semantically formalized in our previous works[5,7]. In order to propose a simulation tool that reproduce the evolution of a complex biomolecular network over time, we extended the logical modelling of complex biomolecular networks presented in our previous works[7,24] by associating it with a discrete time simulation algorithm inspired from the DEVS modelling.

4.2. Fundamentals of the logical modelling

Figure 1 formally recall the logical formalism of complex biomolecular networks by describing its three pillars: the structural, functional and behavioural modeling. These concepts are directly applied to the biological example of the autoregulation of the bacteriophage T4 gene 32 detailed in Section 5.

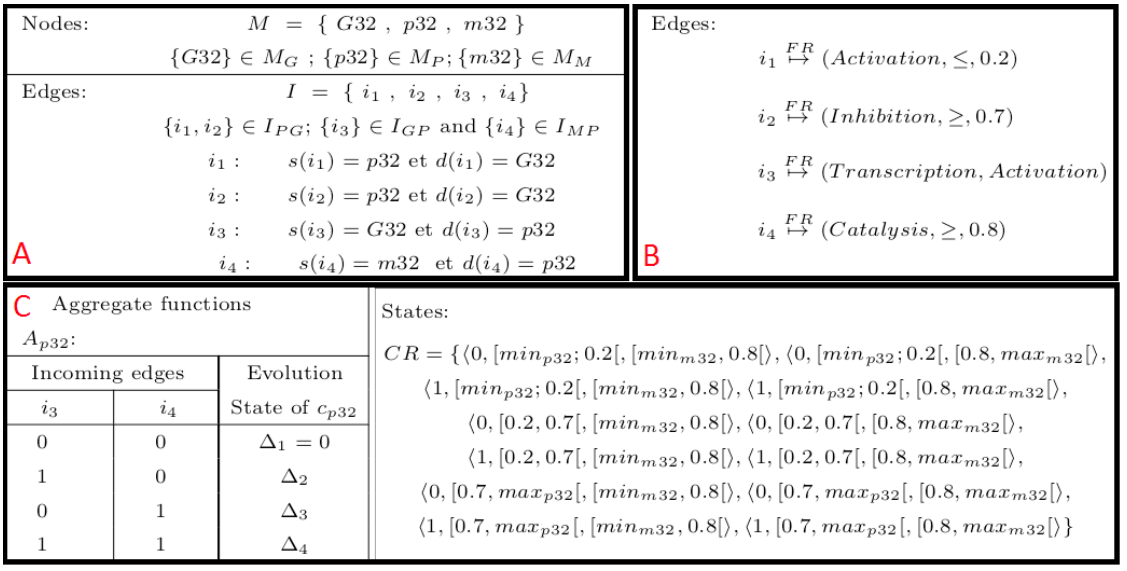

Fig. 1. An illustration of the three main concepts of the logical modelling. **A** The structure; **B** The function and **C** The behaviour.

4.2.1. Structural modelling

The *structure* of the biomolecular network SR is a graph denoted by $SR = (M, I)$. Where: M denotes all the molecules composing the network and represents the nodes of the graph defined by a finite set of nodes $M = \{m_1, m_2, \ldots, m_n\}$. We distinguish a tripartite partition of M: M_G the set of genes, M_P the set of proteins and M_M the set of metabolites. And I denotes the set of interactions between the network's molecules. It describes the edges of the graph SR defined by a finite set of edges $I = \{i_1, i_2, \ldots, i_m\}$. An edge $i = (m_s, m_d)$, (where $m_s, m_d \in M$) which starts from m_s (origin) and comes to m_d (destination) is also noted $m_s \rightarrow m_d$. Thus, for an edge $i \in I$, we note that $s(i)$ the starting node and $d(i)$ the destination node. The partition of the graph nodes induces a partition into a range of different types of interactions. We distinguish three interactions between molecular components of the same type (intraomic interactions), four interactions (among the 6 possibilities) between the nodes belonging to different net-

works (interomic interactions), and two interactions I_{GM} and I_{MG} are not taken into account because there is no direct interaction between the genes and metabolites and vice versa.

4.2.2. Functional modelling

The *function* of the biomolecular network, denoted by FR, associates to each one of the graph's edges $i_{m_s,m_d} \in I$ the type of its interaction and the condition that activates it. It depends on the type of the starting node m_s:

$$FR : \left| i_{m_s,m_d} \overset{FR}{\mapsto} \begin{cases} (TypeInteraction, Activation) & \text{if } m_s \in M_G. \\ (TypeInteraction, \leqslant OR \geqslant, Threshold) & \text{if } m_s \in M_P \cup M_M. \end{cases} \right.$$

If the starting node $m_s \in M_G$, the function FR associates to each arc $i_{m_s,m_d} \in I$, a couple consisting of a label $TypeInteraction$ representing the nature of its interaction that indicates whether the interaction is triggered on the action or on the deactivation of the gene. If the starting node $m_s \in M_G \cup M_M$, the function FR associates to each arc $i_{m_s,m_d} \in I$, a triplet consisting of a label $TypeInteraction$ representing the type of its interaction, a comparison operator ($\leqslant$ or $\geqslant$) that will be used to compare the value of the concentration of the starting node m_s to the threshold associated with this arc and finally the *Threshold* which defines the condition for activating the interaction i_{m_s,m_d} depending on the concentration of the molecule represented by the starting node m_s.

4.2.3. Behavioural modelling

First point. The *state of the network* at a given time if defined by a function $en(m,t)$ which associates to each node m its state at the moment t. $en : \left| (m,t) \overset{en}{\mapsto} \begin{cases} Activation \in \{True, False\} & \text{if } m \in M_G. \\ [c_m(t)] \in \mathbb{R} & \text{if } m \in M_P \cup M_M. \end{cases} \right.$

Where: $c_m(t)$: the value of the concentration of the molecule denoted by the node m at a given time t, and $Activation$ a label representing the states of gene.

Second point. We define $ER(t)$ the state of the biomolecular network at an instant t with a set representing the states of all components in the network at any given time t: $ER(t) = \langle en(m_1,t), en(m_2,t), ..., en(m_n,t) \rangle$.

Third point. The state of a node at time $t+1$ depends on its state at time t, as well as the possible influence of each one of its incoming edges. This influence obviously depends on the state of the starting node of the arc in question.

For each node m, we define an *aggregate function* A_m (relating to the node m) which calculates the evolution of the node status between two successive instants of the simulation. This *aggregate function* A_m depends on the current state of the node m, the state of its predecessor nodes $Pred(m)$ and the characteristics of its incoming edges $ie(m)$.

$$en(m,t+1) = A_m(en(m,t),\ ie(m),\ en(n,t)\ ; n \in Pred(m))$$

Fourth point. To simulate the different transition states of a biomolecular network, we give a state $ER(0)$ at time t_0 and a time step size ΔT. Then the successive states $ER(t+1)$ are calculated from the current state $ER(t)$ according to the interactions and the aggregate functions defined by the network, and the external stimuli. At a given time $t+1$, for each $m \in M$ we have:

- If there are no external stimuli in time t for the node m then: $en(m,t+1) = A_m(en(m,t),\ ie(m),\ en(n,t))$ where: $n \in Pred(m)$
- Else If $m \in M_G$: $en(m,t+1) = \Delta_c$
- Else ($m \in M_P \cup M_M$): $en(m,t+1) = A_m(en(m,t),\ ie(m),\ en(n,t)\ ) + \Delta_c$ where: $n \in Pred(m)$

Fifth point. The *behaviour* of the biomolecular network $CR_{[t_0,t_n]}$ is given by the sequence of its successive states during the simulation time, as: $CR_{[t_0,t_n]} = [ER(0), ER(1), ..., ER(n)]$ Indeed, the behaviour of the network extends between two distinct instants t_0 and t_n forming the simulation interval $[t_0,t_n]$.

4.3. Mapping the logical based modelling with the DEVS formalism

Before proposing a discrete time algorithm following the DEVS approach[8], it is necessary to link the different parts of the logical modelling with their corresponding notions of the DEVS formalism. As discused in Section

4.2, the logical-based modelling is based on three basic modelling pillars: (1) **The structural modelling** *SR*, to describe the architecture of the biomolecular network. (2) **The functional modelling** FR, to describe what can carry out each component of the biomolecular network, specifying the conditions for these activities. (3) And **the behavioural modelling** $CR_{[t_0,t_n]}$, to describe how the biomolecular network and its individual components evolve during the simulation period $[t_0,t_n]$. Therefore, the biomolecular network *BN* is defined as: $BN = (SR, FR, CR_{[t_0,t_n]})$.

We follow this tripartite classification to make the mapping between the logical modelling and the DEVS formalism. The structure of the biomolecular network *SR* (Section 4.2.1) is composed by the set of nodes M which corresponds to the set of DEVS components D, and the set of interactions I corresponds to the set of the internal links among DEVS components IC. The function of the biomolecular network, represented by the function FR, corresponds to the internal transition function δ_{int}. In the logical modelling, the behaviour of complex biomolecular network is represented with multiple parameters. The first one is both of the function $en(m,t)$ that define the set of each component and the function $ER(t)$ that defines the state of the network at time t. These functions correspond to the set of DEVS components S in the DEVS formalism. The aggregate function A_m corresponds to a set of functions in DEVS formalism which consists of the internal transition function δ_{int}, the external transition function δ_{ext}, the confluent transition function δ_{con} and the time advance function ta. The external stimuli represented by S corresponds in DEVS modelling to the set X of input events. Finally, the behaviour of the network $CR_{[t_0,t_n]}$ corresponds in DEVS modelling to the set Y of output events which represents the evolution of each DEVS components during the period of simulation.

4.4. Discrete time simulation algorithm

In this section, we describe the basic model of the proposed discrete time simulation algorithm, with specific reference to the logical-based modelling developed in our previous works[7,24] (presented in Section 4.2) and following the approach in Zeigler et al.[8] (detailed in Section 3.1). Algorithm 1 provides a high level description of the general simulator algorithm of a complex biomolecular network. The main steps of this algorithm are: (1) The initialization time and the state of all molecular components; (2) The execution of the aggregate function to calculate the evolution of the node in the next iteration; (3) Evaluate the node state; (4) Launch the specific reaction if the node state reached a threshold; and (5) Update the novel value of the node.

Algorithm 1 Pseudocode of the general simulator algorithm functioning

```
1: Initialization(t, ER(t))                                   ▷ Initialisation of time and network's state.
2: for All time step t from begining to end_of_simulation do
3:     for Each component m_i ∈ M do
4:         Execution of the aggregate function A_{m_i}        ▷ Launch the aggregate function manages the evolution of the node m_i
5:         (Value, Threshold) = TestState(en(m_i,t), FR(m_i))  ▷ Evaluate the node state
6:         if Value = true then                               ▷ If the state of a node state achieves a threshold
7:             LaunchReaction(FR(en(m_i,t))) = True           ▷ Launch the reaction defined by the function FR
8:             Update the node's state an(m_i,t))             ▷ Update the novel state of the node
9:         end if
10:    end for
11: end for
```

At the beginning of the simulation, the simulator initialises the network's state and time. Algorithm 2 shows the pseudo-code of this *Initialization* function. This function requires the specification of two parameters: the initial time t which its value is often set to $t = 0$ and the network's state $ER(t)$ which is set to $ER(0) = \langle en(m_1,0), en(m_2,0), ..., en(m_n,0)\rangle$.

After each iteration, the simulator evaluates the state of each node. This step is done by the *TestState* function 3. This function requires the specification of two parameters: the state of the node m_i at this time t which is provided by the function $en(m_i,t)$ (see Section 4.2.3) and its function $FR(m_i)$. This function compares the value of the state with the set of Thresholds defined by its function FR. If the value of $en(m_i,t)$ reached a threshold, it returns a boolean value equal to true and the reached threshold (as it is described in Section 4.2.2, else it returns the boolean value false.

Algorithm 2 Pseudocode of the Initialisation function

function INITIALIZATION($t, ER(t)$)
  $t \leftarrow 0$ ▷ Initialize time
  for Each component $m_i \in M$ **do** ▷ Initialise the state of the network's component
    $en(m_i, t_0) \leftarrow InitialState$
  end for
  $ER \leftarrow ER(0)$ ▷ Initialise network state
  return $t_0, ER(0)$
end function

Algorithm 3 Pseudocode of the TestState function

function TESTSTATE($en(m_i, t), FR(m_i)$)
  for Each $Threshold \in ae(m_i)$ **do** ▷ Compare the state of the molecule m_i with all thresholds
    if $en(m_i, t)$ reached $Threshold$ **then**
      return (true, Threshold); ▷ If a threshold is reached it returns true and the reached threshold
    else
      return (false, —);
    end if
  end for
end function

Once the comparison has been done and according to the result returned by the TestState function, the simulator runs the *LaunchReaction* procedure defined by Algorithm 4. This procedure requires the specification of two parameters: the $Threshold$ returned by the *TestState* function and the function $FR(m_i)$. According to these two parameters, the procedure launch the specific reaction corresponding to the type of the interaction defined by the function FR (Section 4.2.2). This label $TypeInteraction$ represents the type of the interaction defined by the edge i_{m_i,m_d} (with m_i the actual node and m_d the target node). These types of interactions belongs to the set of concepts of the Interaction Ontology (which is detailed in our previous works [7]). After applying the specific reaction, the novel value of is updated. This process will continue for all the nodes M and until the end of the simulation. Finally, the simulator returns the sequence of the successive states during the simulation time $CR_{[t_0, t_{end\_of\_simaulation}]} = [ER(0), ER(1), ..., ER(end\_of\_simaulation)]$ defining the behaviour of the biomolecular network. These results are presented graphically.

Algorithm 4 Pseudocode of the LaunchReaction procedure

procedure LAUNCHREACTION($Threshold, FR(m_i)$)
  switch $Threshold$ **do**
    case $TypeInteraction_1$
      `assert`(Launch the reaction corresponding to the this type of interaction)
    case $TypeInteraction_2$
      `assert`(Launch the reaction corresponding to the this type of interaction)
    case $TypeInteraction\_..$
      `assert`(Launch the reaction corresponding to the this type of interaction)
    case $TypeInteraction_n$
      `assert`(Launch the reaction corresponding to the this type of interaction)
end procedure

5. Biological example: Autoregulation of the bacteriophage T4 gene 32

5.1. Description

We test the performance of the proposed CBNSimulator by using a real example of biomolecular network, the bacteriophage T4 gene 32[25]. This biomolecular network consists of three nodes a **gene G32** coding for a **protein p32** and a **metabolite m32** which can catalyse the protein $p32$. In this network, the concentration of $p32$ is regulated by itself and normally should remain between 0.2 10^{-6} Mol and 0.7 10^{-6} Mol. When the concentration of $p32$ exceeds the threshold $S_{p32} = 0.7\ 10^{-6}\ Mol$, we talk about an **Inhibition** in which the protein $p32$ inhibits the translation of its gene $G32$ making it deactivated. However, when the concentration of $p32$ decreases and becomes lower than the threshold $S_{p32} = 0.2\ 10^{-6}\ Mol$, we talk about an **Activation** in which the protein $p32$ activates the translation of its gene $G32$ making it activated. When the gene $G32$ is activated by the protein $p32$, we talk about a **Translation** in which we have a production of $p32$ by increasing the value of its concentration. When the concentration of $m32$ exceeds the threshold $S_{m32} = 0.8\ 10^{-6}\ Mol$, the metabolite $m32$ catalyses the $p32$ by decreasing the value of its concentration, here we treat a **Catalysis**.

5.2. Simulation results

Before simulation of the given example starts, it is necessary to define its structure in a text file that follows the structure SR of a biomolecular network defined in Section 4.2.1. This text file represents the data input of CBNSimulator. CBNSimulator reads the network data from the defined text file. Figure 2 shows the text file containing the necessary elements describing the structure of the bacteriophage T4 gene 32 network.

bacteriophage_T4_G32-definition.txt

```
Molecule Network ::=(<Comment>)*
                    <NetworkName>
                    (<Node> | <Link> | <Comment>)*
<Node> ::= ("Molecule" | "Gene" | "Protein" | "Metabolite") <NodeId> <Value> <Comment>
<Link> ::= "Link" <LinkId> <NodeId> <NodeId> <Value>

<NetworkName> ::="NetwrorkName" (String | space |Number)*
<NodeId> ::=(String | Number)*
<LinkId> ::=(String | Number)*
<Value> ::= Number
<Comment> ::= "#" String


Defintion of the autoregulation T4 gene32:

#This is a comment for my molecular definition
NetworkName  BioMolecular Network

# Definition of molecules
Gene G32 1        # gene G1 is activated
Protein p32 2.3     # protein with its concentration
Metabolite m32 7.1 # protein with its concentration


# Add some link between our molecules
Link G32p32 G32 p32 1 # add link (nammed G32p32) between the gene G32 and the protein p32 with a value 1.
Link G32p32 p32 G32 0.2 # add link (nammed G32p32) between the gene G32 and the protein p32 with a value 0.2 (threshold).
Link G32p32 p32 G32 0.7 # add link (nammed G32p32) between the gene G32 and the protein p32 with a value 0.7 (threshold).
Link G32p32 m32 p32 0.8 # add link (nammed G32p32) between the gene G32 and the protein p32 with a value 0.8 (threshold).
```

Problems Javadoc Declaration Console Progress Call Hierarchy

Fig. 2. Definition of the necessary elements describing the structure of the bacteriophage T4 gene 32 network.

To simulate the behaviour of the given network, we implement the algorithm presented in the previous section and define the different interactions. For example, to define the Activation interaction by a statement such as: "If the value of the concentration of the protein $p32$ rises above a certain threshold, then the transcription of the gene $G32$ is switched on.", we implement its formula. Thus, this statement will look like:

$$\text{If} \quad \text{g is activated by ?p} \quad \text{Then} \quad transcription(g) = !p\ .\tau_\varepsilon$$

Interactions among molecular components can interact with one other, triggering other reactions. These Interactions are ensured by inputs and outputs. So, for each interaction, we affect its trigger which represents the input (denoted by $?x$) and its product called output (noted by $!x$). When the gene $G32$ is activated (by interacting with its input the protein $?p32$), a Translation interaction is activated creating a protein production ($!p\ .\ \tau_\varepsilon$) which means that

the output of this interaction is a production of the protein $p32$ ($!p32$) with an increase of its concentration associated to specific rate of production. Further to this Activation reaction and once the gene is activated, a Transcription reaction will automatically occur by creating increased concentration of the protein $p32$, which will additively increase its production to $\Delta_c\%$ (the change in concentration caused by the production interaction). It is the same with the Inhibition interaction. We define it by the following statement: "If the value of the concentration of the protein $p32$ rises above a certain threshold, then the transcription of the gene $G32$ is switched on.", we implement its formula. Thus, this statement will look like:

$$\text{If} \quad \text{g is deactivated by ?p} \quad \text{Then} \quad transcription(g) = !p \,.\tau_\varepsilon \,.\, 0$$

When the gene $G32$ is deactivated (by interacting with its input the protein $?p32$), the Translation interaction is also deactivated, as well as stopping the protein production ($!p \,.\, \tau_\varepsilon \,.\, 0$) which means that the production reaction is inert and never performs any actions. This allows to maintain a stable level of the protein p or its degradation. Further to this Inhibition reaction and once the gene is deactivated, the Translation reaction is automatically stopped enabling the degradation of the concentration of the protein $p32$.

In cooperation with our biological collaborators (the Integrative Bioinformatics and Genomics LBGI team – ICube Laboratory), we define the values of the thresholds for each interaction for example: 0.2 for the activation reaction and 0.7 for the inhibition reaction. We also estimate the value of a set of parameters needed to simulate the interactions with the discrete time algorithm simulation proposed in Section 4.4. Among them the production rate (which describes the rate of production of the target protein per unit time when the Activation reaction occurs), the degradation rate (which describes the rate of attenuation of the target protein per unit time when the Inhibition reaction occurs).

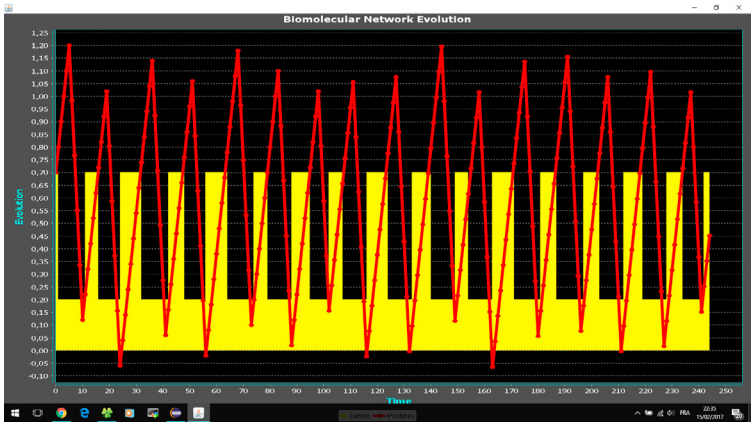

Fig. 3. The simulator's graphical interface. Evolution of the component's behaviour during the simulation period: the red curve represents the different values of the protein $p32$ during the period of simulation and the yellow surface represents the different states of the gene $G32$.

The simulation results of the given example were synthetically represented in graphical form. During the simulation, the concentration of the protein $p32$ is represented but the red curve and the state of the gene $G32$ by the yellow surface that appears when the gene is activated. We run the simulation with different starting states and observe its results. We note that CBNSimulator can successfully reproduce the behaviour of the bacteriophage T4 gene 32 network. In fact, expert biologists agree with the simulator results. Moreover, with the current graphical interface, the user can easily analyse and observe the different states of the network components and consequently deduce the behaviour of the biomolecular network. In addition, these results correspond with results which were obtained earlier with a qualitative reasoning method presented in our previous work[26] and a semantic approach based on a system based on rules SWRL[24].

6. Conclusions and future work

In this paper we integrate a discrete time simulation algorithm inspired by the DEVS formalism, into the logical modelling of biomolecular networks. This approach was implemented through a simulation tool named the CBNSim-

ulator, which aims at providing biologists with a flexible tool for simulating biomolecular networks by reproducing their behaviour and the state of their components over time, and consequently permitting them to analyse and understand simulated cell phenomena. The main goal of CBNSimulator is to analyse and interpret complex biomolecular networks and their behaviour before moving on to in vivo experiments in real cells. This approach has been verified on the bacteriophage T4 gene 32 biomolecular network use case. The simulation results obtained with the use of the CBNSimulator were formally treated and validated by expert biologists. Indeed, these results correspond to their domain knowledge.

Future studies should apply this simulator to large-scale complex biomolecular networks. Moreover, an important issue towards a generic simulation tool for studying the transittability of complex biomolecular networks would be integrate the simulation with optimisation approaches to compute an optimal set of external stimuli to be applied during a predetermined time interval to steer the biomolecular network from its current state to a desired state.

References

1. Brodland, G. W.: How computational models can help unlock biological systems. In Seminars in cell & developmental biology.*Academic Press*. 2015;62-73.
2. Kitano, H.: Systems biology: a brief overview. *Science*. 2002; **295(5560)**:1662-1664.
3. Ren, L. H., Ding, Y. S., Shen, Y. Z., & Zhang, X. F.: Multi-agent-based bio-network for systems biology: protein–protein interaction network as an example.*Amino acids*. 2008; **35(3)**:565-572.
4. Wu, F. X., Wu, L., Wang, J., Liu, J., & Chen, L.: Transittability of complex networks and its applications to regulatory biomolecular networks. *Scientific reports*. 2014; **4**.
5. Ayadi A., de Bertrand de Beuvron F., Zanni-Merk C., & Thompson J.: Formalisation des réseaux biomoléculaires complexes. *EGC 2016 - 16èmes Journées Francophones "Extraction et Gestion des Connaissances". Revue des Nouvelles Technologies de l'Information*. Reims; 2016.
6. Sauro, H. M., Harel, D., Kwiatkowska, M., Shaffer, C. A., Uhrmacher, A. M., Hucka, M., ... & Tyson, J. J.: Challenges for modeling and simulation methods in systems biology.*In Simulation Conference, 2006. WSC06. Proceedings of the Winter*.2006;1720-1730.
7. Ayadi, A., Zanni-Merk, C., de Bertrand de Beuvron, F. and Krichen, S.: Logical and Semantic Modeling of Complex Biomolecular Networks. *Procedia Computer Science*. 2016; Volume **396**:475-484.
8. Zeigler, Bernard P., Herbert Praehofer, & Tag Gon Kim. Theory of modeling and simulation: integrating discrete event and continuous complex dynamic systems.*Academic press*. 2000.
9. Laubenbacher, R.: Modeling and simulation of biological networks.*American Mathematical Soc*. 2007; **64**.
10. Wu, H., Von Kamp, A., Leoncikas, V., Mori, W., Sahin, N., Gevorgyan, A., ...& Stewart, G. R.: MUFINS: multi-formalism interaction network simulator.*npj Systems Biology and Applications*. 2016;**2**:16032.
11. Voliotis, M., Thomas, P., Grima, R., & Bowsher, C. G.: Stochastic simulation of biomolecular networks in dynamic environments.*PLoS Comput Biol*. 2016;**12(6)**:e1004923.
12. Mendes, P., Hoops, S., Sahle, S., Gauges, R., Dada, J., & Kummer, U.: Computational modeling of biochemical networks using COPASI.*Systems Biology*. 2009; 17-59.
13. Bergmann, F. T., & Sauro, H. M.: SBW-a modular framework for systems biology.*In Simulation Conference, 2006. WSC 06*. Proceedings of the Winter. 2006; 1637-1645.
14. Funahashi, A., Matsuoka, Y., Jouraku, A., Morohashi, M., Kikuchi, N., & Kitano, H.: CellDesigner 3.5: a versatile modeling tool for biochemical networks.*Proceedings of the IEEE*. 2008;**96(8)**:1254-1265.
15. Nagasaki, M., Saito, A., Jeong, E., Li, C., Kojima, K., Ikeda, E., & Miyano, S.: Cell illustrator 4.0: a computational platform for systems biology.*In silico biology*.2010; **10(1, 2)**:5-26.
16. Frazier, J. M., Chushak, Y., & Foy, B.: Stochastic simulation and analysis of biomolecular reaction networks.*BMC systems biology*. 2009;**3(1)**:64.
17. Chandran, D., Bergmann, F. T., & Sauro, H. M.: Computer-aided design of biological circuits using TinkerCell.*Bioengineered bugs*. 2010; **1(4)**:276-283.
18. Czar, M. J., Cai, Y., & Peccoud, J.: Writing DNA with GenoCAD.*Nucleic acids research*. 2009;**37(suppl 2)**:40-47.
19. Szekely, T., & Burrage, K.: Stochastic simulation in systems biology.*Computational and structural biotechnology journal*. 2014;**12(20)**:14-25.
20. Mizera, A., Pang, J., & Yuan, Q.: Fast simulation of probabilistic Boolean networks.*In International Conference on Computational Methods in Systems Biology. Springer International Publishing*. 2016; 216-231.
21. Machado, D., Costa, R. S., Rocha, M., Ferreira, E. C., Tidor, B., & Rocha, I. : Modeling formalisms in systems biology.*AMB express*. 2011;**1(1)** :45.
22. Karp, Gerald.: Biologie cellulaire et moléculaire: Concepts and experiments. *De Boeck Supérieur*. 2010.
23. Wu, F., Chen, L., Wang, J., & Alhajj, R.: Biomolecular Networks and Human Diseases. *BioMed research international*. 2014; **2014**:363717.
24. Ayadi A., Zanni-Merk C., & de Beuvron de Bertrand F.: Understanding the Behaviour of Complex BiomolecularNetworks by Combining Logical and Semantic Modeling. *In Proceedings of the 9th International SWAT4LS Conference Semantic Web applications and tools for life sciences*. Amsterdam. 2016.
25. B. Lewin, C. Sanlaville: Gènes VI,*De Boeck Supérieur*, 1998.
26. Ayadi, A., Zanni-Merk, C. and de Bertrand de Beuvron, F.: Qualitative Reasoning for Understanding the Behaviour of Complex Biomolecular Networks. *Proceedings of the 8th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2016)*. 2016; Volume **2**:144-149.