Pacific Symposium on Biocomputing 14:239-250 (2009)

FROM ODES TO LANGUAGE-BASED, EXECUTABLE MODELS OF BIOLOGICAL SYSTEMS

A. PALMISANO, I. MURA AND C. PRIAMI
CoSBi, Trento, Italy
E-mail: {palmisano, mura, priami}@cosbi.eu

Modeling in biology is mainly grounded in mathematics, and specifically on ordinary differential equations (ODE). The programming language approach is a complementary and emergent tool to analyze the dynamics of biological networks. Here we focus on BlenX showing how it is possible to easily re-use ODE models within this framework. A budding yeast cell cycle example demonstrates the advantages of using a stochastic approach. Finally, some hints are provided on how the automatically translated model can take advantage of the full power of BlenX to analyze the control mechanisms of the cell cycle machinery.

1. Introduction

Abstract models of biological systems are becoming an indispensable conceptual and computational tool for biologists. In order to be useful, a model has to allow automatic analysis, and permit the integration of new knowledge without changing too much the already built model.

Chemical kinetics has traditionally been analyzed using a mathematical formalism in which continuous variables evolve deterministically, even if molecular populations in a biological systems are integer variables that evolve stochastically [1]. There is not yet an agreement upon the characterization of the best “usable form” in modeling biology, because some aspects are better handled with the deterministic approach, while for others the stochastic one is more suitable. For example, in the literature, beside deterministic ordinary differential equations (ODE for short), we find stochastic cell cycle models built with stochastic ODE Langevine type [2], with the Gillespie method [3] and with stochasticity on transitions [4]. Moreover, as a recent paper by Nurse on Nature [5] points out, the life science field requires the development of new and more appropriate languages to describe biological systems.

After the work of Regev et al. [6], a promising trend in this direction is to use programming languages to generate executable models of biological systems. This strategy diverges from classical mathematical modeling because it is executable and not just simply solvable [7]. Execution means that we can describe the flow of control between species and reactions, i.e. not only the time, but also the causality relation among the events that constitute the history of the dynamics of the model. This interpretation is very similar to programming the behaviour of a system rather than describing only its outcome with respect to time. As a consequence, a number of process calculi have been adapted or newly developed for building biological models and performing stochastic simulations (i.e. stochastic π-calculus[8,9], BioAmbients[10], Brane Calculi[11], CCS-R[12], k-calculus[13], Bio-PEPA[14]).

The main contribution of this paper is a semi-automatic method to translate existing deterministic models written with ODEs into programs written in the stochastic modeling language BlenX [15,16]. The simple translation of general ODE terms into a stochastic BlenX model is possible because of the expressive power of the language that allows the definition of general rate functions for the transitions.

The paper is organized as follows. In the next section we concisely recall the two languages used for modeling biological systems, i.e. the mathematical (ODEs) and the computational (BlenX) one. Sect. 3 illustrates a semi-automatic procedure to translate the former into the latter and Sect. 4 applies this procedure on a biological relevant example, the cell cycle machinery of *Saccharomyces cerevisiae.* Validation results and improvements of this model are then discussed in Sect. 4.3, conclusions and future work directions close the paper in Sect. 5.

2. Modeling of Biological Systems

2.1. *The mathematical language*

The usual way in which life scientists build models starts from the organization of the known interactions of the relevant molecules into a wiring diagram [17]. There are no unambiguous rules or general conventions for building these diagrams and so the information contained in those diagrams is quite difficult to be translated automatically in different modeling languages. However, some information contained in those maps is fundamental to disambiguate between different reactions that are associated to the same mathematical model.

Once the model is represented diagrammatically, the biochemical relationships are converted into ODEs. A rate equation is written for each biochemical entity whose concentration changes over time. Those rate equations are written following different kinetic laws that take into account the known properties of the chemicals involved, possibly abstracting some unknown mechanisms.

Before doing the translation of the ODE model in BlenX, we want to make a remark, already discussed by several authors [18,14,19], about the usage of the Gillespie SSA algorithm [20] when high-level mathematical representation of non-elementary reactions are incorporated in the form of rate-dependent functions. The rate dependent functions are defining the reaction propensities of the stochastic model. Following the approach of [21] we assume that the fundamental hypothesis of Gillespie, i.e. each reaction time is a random variable following a negative exponential distribution with rate equal to the value of the propensity function, is verified for the biological system we are modeling. If this hypothesis is valid, a stochastic characterization of the reaction times as negatively distributed random variables is an accurate modeling choice, as proved by Gillespie [20]. In some studies, even considering Michaelis-Menten reactions [18], the applicability of the fundamental hypothesis was mathematically proved, and in some others dealing with the circadian rhythm this same hypothesis was experimentally verified in silico [22]. In our case, because of the presence of many non-elementary reactions, a careful validation similar to the one in [21] has been performed: here just few examples will be shown for the sake of conciseness and because the main focus of the paper is the translation procedure and not the validation process that has to be done on each single class of non-elementary reactions.

2.2. *The* BlenX *Language*

A detailed description of the BlenX language is out of the scope of this paper; here we just summarize the sub-part of the BlenX language needed for understanding the code of the presented examples. We refer the reader to [15,16] for a detailed description of the language and its modeling approach.

The basic metaphor we keep is that a *biological entity*, i.e. a component that is able to interact with other components to accomplish some biological functions, is represented by a *box* in BlenX. This box has an interface (its set of *binders*) and an internal structure that drives its behaviour (see Fig. 1).

For example, in a box modeling a protein, binders may represent *sensing* and *effecting domains*. Sensing domains are the places where the protein receives signals, effecting domains are the places that a protein uses for

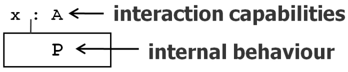

Figure 1. Boxes as abstractions of biological entities.

propagating signals, and the *internal structure* codifies the actions that transform an input signal into, for example, the activation/inactivation of a specific domain. The exchanging of signals can happen between boxes whose binders have a certain degree of affinity, which codes the strength of their interaction.

The basic primitives of the language that will be used in Sect. 3 to translate ODEs into BlenX are summarized in graphical form in Fig. 2.

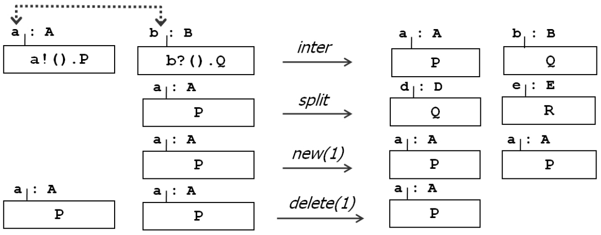

Figure 2. Intuitive behaviour of some BlenX primitives. Each row represents one of the primitives used in our translation. The first primitive codifies the interaction between two boxes, through the exchange of an input/output signal (input is in the form of b?() and the output is in the form a!()). The last three rows are the graphical representation of events of the form: "when(*conditions*) *verb*", where the action *verb* is triggered when *conditions* are satisfied. The *verb* can be one of split, new or delete, that model respectively the substitution, creation, and deletion of boxes in the system (graphically depicted above). Conditions, in the models presented here, are in the form of "*entity_name : : function*", whose meaning is that the action after the condition is triggered, at rate *function*, on the entity *entity_name*.

Rate functions are associated to action and events, and those rates can be declared using real numbers that will be used as base rate for the elementary mass action law, or arbitrary functions (e.g. Hill response) that are useful when a box represents an aggregated process or when the precise mechanism of interaction between entities is not known.

3. From ODEs to BlenX models

The rationale is to use the same level of abstraction adopted in the deterministic model so that we can define an easy mapping of ODEs into BlenX.

This translation requires using few primitives of the language. Later in the paper, we propose some hints on how the user can tune the model produced by this mapping, using more powerful BlenX capabilities: this is a fundamental step toward the generation of a more effective translation of the functioning of the logic circuits that underpin biological phenomena [5].

We first modify the differential equations to get the variation of the number of molecules instead of concentrations, using a scalar constant α defined as $\alpha = (N_A 10^{-6} V)^{-1}$, where N_A is the Avogadro's number and V is the volume of the modelled system. Applying $Conc_S = \alpha \cdot MolNum_S$ conversion for each species S, we have an ODE system with the original kinetics constant, but written in terms of number of molecules. This allows us to use directly those terms as stochastic rate functions, without the need of giving to each single kinetic constant its stochastic numerical counterpart.

The second algebraic modification of the ODE system that we need is to add the species that are appearing "implicitly" in the system. If a species is present in a constant total amount but it can switch between an active and an inactive state, the time evolution of just one of the two states is usually explicitly considered, because the other (the "implicit") one can be derived from the first. In our initial simplified usage of BlenX we have to specify all the states in which a species can be, and so we need a definition also for the states not explicitly written in the initial ODEs. Hereafter, when we talk about species, we include both the original *explicit* and the added *implicit* ones.

The general mathematical description of a system of N reactant species $S_1, \ldots, S_N$ expressed in terms of number of molecules $M_1, \ldots, M_N$ and involved in $R_1, \ldots, R_R$ reactions has the following form:

$$\frac{dM_i}{dt} = \sum_{j=1}^{R_i} f_j^i(M_1, \ldots, M_N) \qquad (1)$$

where $i = 1, \ldots, N$, $R_i \leq R$, and f_j^i is a rate function that can contain constants rate coefficients and/or other discrete/continuous variables dependent from time and/or species in the model (see the example of $CycB$ variable introduced in Sect. 4.1). So the complete definition of the system above has to be coupled with the set $Var_1, \ldots, Var_T$ of mathematical expressions for the variation of each discrete/continuous variable.

The translation from an ODE system written with the above conventions to BlenX is summarized by the following pseudocode:

```
Input:    the set (1) of ODEs;
Output:   the definition of the BlenX model and the rate functions.
```

Method:

```
(1) expand all the possible factors of the ODEs, so that all the equations are
    written as summations of positive/negative terms;
(2) for(i in [1..$N$])
      for(j in [1..$R_i$])
        define function(f_i_j) = ABS(mathematical_expression($f_j^i$))
(3) for (i in [1..$T$])
        define variable(var_i) = mathematical_expression($Var_i$)
(4) declare all the constants contained in the previous definitions
(5) for(i in [1..$N$])        declare empty-box(i)
(6) for(i in [1..$N$])
     for(j in [1..$R_i$])
      switch(kind of $f_j^i$ reaction):
       case 'synthesis':                add when (i : : f_i_j) new(1);
       case 'degradation':              add when (i : : f_i_j) delete(1);
       case 'changing from i to k':     add when (i : : f_i_j) split(Nil, k);
       case 'changing from k to i':     add when (k : : f_i_j) split(Nil, i);
(7) remove all copies but one of the replicated events that have been created
    for reactions appearing in multiple ODEs
(8) set the initial state of the model.
```

Note that the choice on step (6) has to be done manually, looking at the wiring diagram of the system: this is the only step that cannot be done automatically, because of the *inverse problem* discussed in the conclusions of the paper.

In the next section, following [21], the whole process is applied to a budding yeast cell cycle model [23] in order to show that the translation is straightforward and that the stochastic simulations are, from one side, consistent with the results obtained by the solution of the ODE system, but from the other side, the stochastic model matches some characteristics that cannot be found with the deterministic one.

4. Cell Cycle Machinery of Saccharomyces cerevisiae

4.1. *Deterministic Model of Budding Yeast Cell Cycle*

The budding yeast is a well-studied and understood example of how the cell cycle can be controlled by the activity of only a few cyclin-dependent protein kinases. We will focus hereafter on the biochemical machinery that

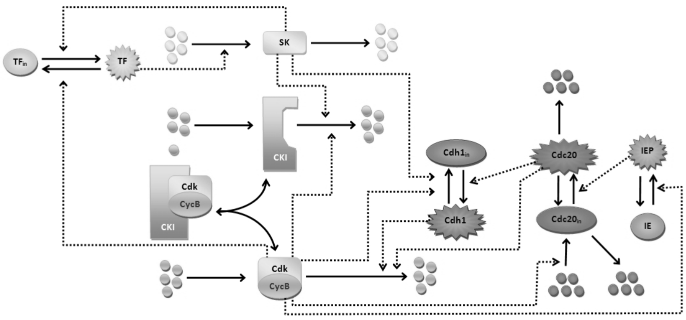

Figure 3. Graphical representation of cell cycle engine. Solid lines link reactants to products, dashed lines represent the mediation effect that some species have on reactions.

controls Cdks activity that has been modeled with ODEs by Novak and Tyson (see [23], pag. 270), and that is depicted in Fig. 3.

For the sake of conciseness, we do not report here the 8 equations composing the model in [23], but only the ODEs of two species that contain all the basic features needed for the translation in BlenX of the model.

$$\begin{aligned}
\frac{d|CDC20\_IN|}{dt} &= \underbrace{k5p/\alpha}_{synthesis} + \underbrace{\frac{(k5s/\alpha)*(\alpha*m*CycB)^n}{(J5)^n+(\alpha*m*CycB)^n}}_{(induced)\ synthesis} - \underbrace{k6*|CDC20\_IN|}_{degradation} + \\
&\quad + \underbrace{\frac{k8*|CDC20\_A|}{J8+\alpha*|CDC20\_A|}}_{inactivation} - \underbrace{\frac{k7*\alpha*|IEP|*|CDC20\_IN|}{J7+\alpha*|CDC20\_IN|}}_{activation} \qquad (2)\\
\frac{d|CDC20\_A|}{dt} &= \frac{k7*\alpha|IEP||CDC20\_IN|}{J7+\alpha|CDC20\_IN|} - \frac{k8|CDC20\_A|}{J8+\alpha|CDC20\_A|} - k6|CDC20\_A| \qquad (3)\\
\frac{dm}{dt} &= \mu*m*(1-m/mstar) \qquad (4)
\end{aligned}$$

Figure 4. Equations for the activation/inactivation of the Cdc20 protein and the rate law for the growing of the mass. $|S|$ is the number of molecules of species S and the different ks (and Js) are the deterministic kinetic parameters. The part of the model above is also referring to a discrete variable ($CycB$) that represents the activity of the dimer CyclinB/Cdk. $CycB$ is calculated with an algebraic expression which can be found in [24] with the complete set of equations and variables.

4.2. *Stochastic Model of Budding Yeast Cell Cycle*

We now apply the translation steps described in Sect. 3 to each term of the above ODEs generating the BlenX code of the model. The model can be given as input to BetaWB[a] framework to perform stochastic simulations.

The $CDC20\_IN$ species contains a positive term for the rate of its **synthesis** ($k5p/\alpha$) and a negative term for the rate of its **degradation** ($k6 * |CDC20\_IN|$). The result of their codification is in Fig. 5(a).

(a)

```
let k5p : const = 0.005; let k6 : const = 0.1;       //step 4
let alpha : const = 0.00302023;                       //step 4
let d_dtCDC20_IN_1 : function = k5p/alpha;            //step 2
let d_dtCDC20_IN_5 : function = k6*|CDC20_IN|;        //step 2
```

(b)

```
let CDC20_IN : bproc = #(a,CDC20_IN)[ nil ];          //step 5
when(CDC20_IN : : d_dtCDC20_IN_1 ) new(1);            //step 6
when(CDC20_IN : : d_dtCDC20_IN_5 ) delete(1);         //step 6
```

(c)

```
let d_dtCDC20_IN_3 : function =                       //step 2
 (k8*|CDC20_A|)/(J8+alpha*|CDC20_A|);
let d_dtCDC20_IN_4 : function =                       //step 2
 (k7*alpha*|IEP|*|CDC20_IN|)/(J7+alpha*|CDC20_IN|);
```

(d)

```
let CDC20_A : bproc = #(a,CDC20_A)[ nil ];              //step 5
when(CDC20_IN::d_dtCDC20_IN_4) split(Nil,CDC20_A);      //step 6
when(CDC20_A::d_dtCDC20_IN_4) split(Nil,CDC20_IN);      //step 6
```

Figure 5. BlenX code showing the result of the steps of the translation method on the specific equations in Fig. 4: synthesis/degradation (a) and activation/inactivation (c) rate functions, synthesis/degradation (b) and activation/inactivation mechanism (d).

The next step is the encoding of the structure of the model. We define an empty box for the species $CDC20\_IN$ and then we add a new event that represents its synthesis and a delete event that represents its degradation. The rates of the events are the functions defined in Fig. 5(a) and structure of the model is in Fig. 5(b). The $CDC20\_IN$ species contains a positive Michaelis-Menten term that is representing the **inactivation** of the $CDC20\_A$ and a negative Michaelis-Menten term that is representing the **activation** of the $CDC20\_IN$. Coding this rate terms similarly to the previous case, we obtain the code in Fig. 5(c). In the model we just have to add the box for the CDC20_A species because we have already defined the box for the inactive species. Then we add two events, one encoding the inactivation of the active Cdc20 and one encoding the activation of the

[a] BetaWB is available at http://www.cosbi.eu/

inactive Cdc20. Their rate are the ones in Fig. 5(c) and the code for this part of the model is in Fig. 5(d). All the terms of the complete ODE system can be seen as one of the previous cases, so even if they are driven by complex kinetics (as the Hill function in the synthesis of $CDC20\_IN$) they can be easily coded by a new/delete/split event.

With the rules introduced so far we encode almost the whole set of ODEs from [23]. The only one left is ODE (4), which is not representing a chemical species but the mass of the cell. It can be seen as a variable that has to be updated, following the function in equation (4), with discrete time steps. With BlenX is possible to define those kind of variables simply copying their ODE with the other rate functions, declaring it as a **var** rather than a **function**. Also cell division is an event that is not explicitly coded in the ODE system, and it halves the mass value when the concentration of active Cdk/CycB falls below an assigned threshold (0.1, as in the [23] model) after having raised above another threshold (here 0.2). In BlenX we can simply add the event "**when**(:mCycB→0.2,mCycB←0.1:)**update**(m,mass_div);", which tells the simulator to trace the state of the variable $mCycB$ and whenever this variable overcomes the 0.2 threshold and then goes back under the 0.1 threshold, the event that updates the value of the mass with its halved value is executed. With this last rule we can complete the translation of the model from ODE to BlenX.

4.3. *Validation of the stochastic model*

Following [21], we compare the results provided by the BlenX model and the deterministic one for the wild type of budding yeast.

The model simulation was performed with BetaWB and the only difference among the runs of the model is the stochastic fluctuation (Fig. 6a-6b).

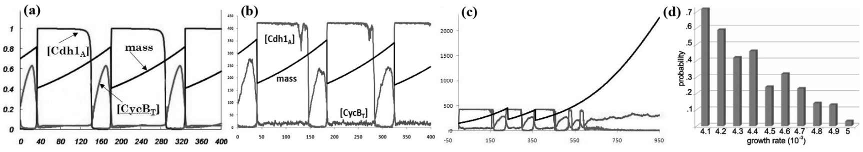

Figure 6. Deterministic (a) and stochastic (b) model results for the wild type. We plot the time courses of the cell mass, cyclin [CycB$_T$] and APC mediator [Cdh1A] concentration for a [0-400] time interval. (c) One stochastic result for the mutant with growth rate 0.0045. The cell was able to generate 2 offspring generations before losing the ability of surviving further. (d) Probability of having at least 10 generation of the mutant varying the growth rate.

Following the careful validation process of [21], we built some mutants starting from the above model: here for the sake of conciseness we show just one of them. All the results of the stochastic simulations are in agreement with the deterministic model of those mutants. In order to show the insights that can be obtained with the stochastic model, we here show results for the mutant obtained with the deletion of the destruction box of cyclin Clb2 (which reduces its degradation rates), and with the deletion of cyclin Clb5 (which reduces the overall CycB level). This model can be obtained changing just some specific parameters of the original model.

Experimental results show that this mutant is viable, but only with a decreased growth rate [25]. Therefore we reduce the rate constant of growth speed from the default value 0.005 min^{-1} to 0.0041 min^{-1}. Both the deterministic and the stochastic model results correctly show its viability. It is interesting to observe that the deterministic model is able to fit the lethality of the mutation with growth rate 0.005 and its viability with growth rate 0.0041, but cannot predict the intermediate situations: it is instead reasonable to expect a continuous transitions as the growth rate varies in the interval [0.0041-0.005], with some mutant cells that have a limited survivability for values of the growth rate inside the interval and a death probability increasing approaching the lethal situation of 0.005.

Small colonies of mutants cells with growth rate in this interval have been experimentally observed [25], so we are expecting that if a mutant is able to complete a sufficient number of cells cycles before dying, a colony may develop, even if its overall growth would be slow. We conducted an in-silico experiment to evaluate the probability that a single mutant cell would be able to generate at least 10 offspring generations before dying, varying the growth rate within the interval [0.0041-0.005] (we ran 100 simulations for each value). The results (Fig.6c-6d)) clearly show that colonies of the mutant may exist for values of the growth rate higher than the threshold which sets the upper limit for the viability of the mutant in the ODE model. We can conclude that stochastic simulations are important to check the partial viability of mutants at the border of life and death.

4.4. *Improving the structure of the model*

In this section we provide some hints on how the model structure can be improved by using the full power of BlenX. The underlying mathematical model is exactly the same as before, but using the peculiarities of this new language, as pointed out in [5], can be a fundamental step toward the generation of a more effective model that can unravel the precise mechanisms of interactions that governs biological systems.

The first modification is to use the `die` action for the degradation mechanism of a box, so that the death of a box is driven by its own internal behaviour and not by a global event (e.g. the independent degradation of CycBT at rate k2p). The second modification is using the communication between boxes through binders for modeling simple mass action kinetics. For example the degradation of the CycB/Cdk dimer caused by its interaction with Cdh1 (Fig. 3) can be modeled explicitly as in Fig. 4.4 and by assigning an affinity rate `k2s_stoch` of communication between CYCBT_TYPE and CDH1_TYPE binders, where `k2s_stoch` is the deterministic rate k2s converted to stochastic.

```
let CYCBT: bproc = #(x,CYCBT_TYPE)
[ die(rate(k2p)) + x?().die.nil ];
let CDH1 : bproc = #(y,CDH1_TYPE)
[rep y!().nil];
```

$$\frac{d|CYCBT|}{dt} = -k2p|CYCBT| + \\ -k2s|CDH1||CYCBT| + \ldots$$

Figure 7. Code for a mass action dynamic (left) with the subpart of the deterministic model it is referring to (right).

Using this kind of mechanism on the whole set of ODEs we create a model that is more prone to be modified. As new knowledge is acquired on a specific protein, this can be added to the internal behavior of that specific box in parallel or choice with the existing coded behavior. The only drawback of this approach is that we need to have a detailed description of the interactions between the components and usually this is not the case (e.g. in the cell cycle model described above, the Hill response has been used in order to abstract a cooperative mechanism whose details are unknown).

5. Conclusions and Future Work

We presented a translation of existing ODE models into BlenX computational model. The procedure cannot be completely automatized, because it has to face the *inverse problem* of reaction kinetics [26]: more than one structurally different network of chemicals can produce the same set of ODEs. To disambiguate between those networks we need to have the knowledge of the interactions between all the species, but usually this knowledge is hidden in wiring diagrams that are not following strict or general conventions.

Our translation creates a model that can be stochastically simulated in order to answer questions that cannot be directly tackled by deterministic models. The simple mapping is possible because of the expressive power of BlenX that allows the definition of general rate functions for reactions. However the usage of the SSA algorithm with simple and complex kinetic laws leaves to the user the responsibility to validate simulation results. The example proposed here uses those different levels of abstraction and it has been validated against experimental results. It is our future goal to further analyze the mechanisms that are behind those complex kinetics in order to make more detailed considerations on the effect of the stochastic noise.

Acknowledgments

We thank A. Csikasz-Nagy and P. Lecca for the discussions on mathematical modeling, and L. Dematté and A. Romanel for bright conversations on computational approach.

References

1. D. Gillespie, *Annual Review of Physical Chemistry* **58**, 35 (2007).
2. R. Steuer, *Journal of Theoretical Biology* **228**, 293 (2004).
3. M. Sabouri-Ghomi, A. Ciliberto, S. Kar, B. Novak and J. Tyson, *J. Theor. Biol.* **250**, 209 (2007).
4. P. Lecca and C. Priami, in *Proc. of BioConcur 2003*, (ENTCS, 2007).
5. P. Nurse, *Nature* **454**.
6. A. Regev and E. Shapiro, *Nature* **419**, 353 (2002).
7. J. Fisher and T. Henzinger, *Nature Biotechnology* **25**, 1239 (2007).
8. C. Priami, *The Computer Journal* **38**, 578 (1995).
9. C. Priami, A. Regev, E. Shapiro and W. Silverman, *Inform. Process. Lett.* **80**, 25 (2001).
10. A. Regev, E. Panina, W. Silverman, L. Cardelli and E. Shapiro, *Theoretical Computer Science* **325**, 141 (2004).
11. L. Cardelli, in *Proc. of CMSB 2004*, (LNCS, 2005).
12. V. Danos and J. Krivine, in *Proc. of CONCUR 2004*, (LNCS, 2004).
13. V. Danos and C. Laneve, *Theoretical Computer Science* **325**, 69 (2004).
14. F. Ciocchetta and J. Hillston, in *Proc. of FBTC 2007*, (ENTCS, 2007).
15. L. Dematté, C. Priami and A. Romanel, *Briefings in Bioinformatics* (2008).
16. L. Dematté, C. Priami and A. Romanel, in *Proc. of SFM 2008*, (LNCS, 2008).
17. J. Sible and J. Tyson, *Methods* **41**, 238 (2007).
18. A. Arkin and C. Rao, *Journal of Chemical Physics* **118**, 4999 (2003).
19. I. Mura, Tech. Rep. TR-12-2008, CoSBi (2008).
20. D. Gillespie, *Journal of Physical Chemistry* **81**, 2340 (1977).
21. I. Mura and A. Csikasz-Nagy, *Journal of Theoretical Biology* (2008).
22. D. Gonze, J. Halloy and A. Goldbeter, *J. of Biol. Phys.* **28** (2002).
23. B. Novak and J. Tyson, *Cell Cycle Controls*, in *Computational Cell Biology*, (Springer, 2003), pp. 261–284.
24. A. Palmisano and C. Priami, Tech. Rep. TR-10-2008, CoSBi (2008).
25. F. Cross, *Developmental Cell* **4**, 741 (2003).
26. P. Érdi and J. Tóth, *Mathematical Models of Chemical Reactions* 1989.