Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science 229 (2009) 75–92
Electronic Notes in Theoretical Computer Science
www.elsevier.com/locate/entcs

The Importance of Being (A Little Bit) Discrete

Luca Bortolussi[1]

Department of Mathematics and Computer Science, University of Trieste, Italia
Center for Biomolecular Medicine, Area Science Park, Trieste

Alberto Policriti [2]

Department of Mathematics and Computer Science, University of Udine, Italia
Institute for Applied Genomics, Udine

Abstract

We compare the hybrid, stochastic, and differential semantics for stochastic Concurrent Constraint Programming, focussing on the exhibited behavior of models and their robustness. By investigating in detail two case studies, a circadian clock model and the Repressilator, we comment on the effect of the introduction of a limited amount of discreteness in the description of biological systems with hybrid automata. Experimental evidence suggests that discreteness increases robustness of the models.

Keywords: Biological Modeling, Hybrid Systems, Robustness, Discreteness, Stochastic Noise.

1 Introduction

Mathematical modeling of biological networks [16] is dominated by two formalisms: *ordinary differential equations* (ODE) and stochastic processes, mainly *Continuous Time Markov Chains* (CTMC [22]). Differences are evident: ODEs describe the systems as *continuous* and *deterministic*, while CTMC are *discrete* and *stochastic*. An approach standing somewhere in between consists in modeling biological systems using Hybrid Automata (HA [14,1]), a formalism having a mixed discrete/continuous dynamics, or their stochastic counterpart, *Stochastic Hybrid Automata* (SHA [9]). As one could expect, there is no *a-priori* correct formalism to use, but rather this choice depends on the system under examination and on the properties one is interested to check. Whatever is the choice, however, is important to keep in mind

[1] Email: luca@dmi.units.it
[2] Email: policriti@dimi.uniud.it

doi:10.1016/j.entcs.2009.02.006

that even though CTMC models have firmer physical bases [12], their analysis is very expensive from a computational point of view.

A further crucial issue in Systems Biology is the *language* used to describe systems, which should have features like *compositionality* and *model reusability*. Many *Stochastic Process Algebras* (SPA) have been applied in systems biology [10,18]. Their semantics is defined classically in terms of CTMC and, more recently, also in terms of ODEs [15].

In this paper we focus on a particular SPA, namely stochastic Concurrent Constraint Programming (sCCP [5]), which has already been applied successfully to biological modeling [8]. In addition to the standard CTMC-based semantics, sCCP has also an ODE based semantics [6], as well as a further one based on Hybrid Automata [7]. These three semantics associate different models to the same system, with a varying degree of discreteness and continuity. This feature can be exploited to study the *interplay* between discrete and continuous dynamics and its drawbacks on modeling the behavior of the system. As a matter of fact, the relationship between discreteness and continuity raises several interesting philosophical questions: Is discreteness an essential property of biological systems? If so, in which cases must it be maintained as part of the description and when, instead, can it be safely continuously approximated? Does maintaining a level of discreteness increase robustness? And, finally, what is the role of stochasticity? We believe a better understanding of these questions can lead to an improvement and a speed-up of the analysis of models.

The purpose of this paper is to start the investigation of these issues exploiting the above mentioned three different semantics of sCCP. In particular, we will focus on two properties of models: one is the exhibited *dynamical behavior* and the other is the *robustness* of the system. More specifically, we will concentrate on systems that are expected to oscillate in alternating phases. Hence, we will investigate if oscillations are (qualitatively) preserved when kinetic parameters governing the dynamics are perturbed.

In the following we will discuss two case studies. The first is the well known Repressilator [11], a synthetic regulatory network supposed to exhibit an oscillatory behavior. We will compare the stochastic, differential, and hybrid sCCP models, performing a robustness analysis of the latter. The second system is a simplified model of the circadian clock [21]. In particular, we will be concerned with the effect on the stability of oscillations caused by the introduction of a small degree of discreteness and stochasticity.

The paper is organized as follows: Section 2 provides some background on sCCP and hybrid automata. Sections 3 and 4 discuss the Repressilator and the circadian clock, respectively. Conclusions and future perspectives are the content of Section 5.

2 Basics

In this section we review some concepts that are needed in the following. First we present sCCP and its mapping to ODEs, then we introduce hybrid automata, and

$$Def = \varepsilon \mid Def.Def \mid p{:}\text{-}M \qquad N = p \mid p \parallel N$$
$$\pi = [g(X) \to u(X,X')]_\lambda \qquad M = \pi.p \mid M + M$$

Table 1
Syntax of (a restricted version of) sCCP. Agents $p \in Def$ are defined as simple summations prefixed by basic actions π. Each of such actions is a guarded update of the form $g(X) \to u(X,X')$, where $g(X)$ is the guard and $u(X,X')$ is the update predicate, a conjunction of basic constraints $X' = f(\mathbf{X})$, replacing the current value of X with the value of its primed version X'. The stochastic duration of π is given by the function λ.

finally we describe the hybrid semantics of sCCP.

2.1 Stochastic Concurrent Constraint Programming

Stochastic Concurrent Constraint Programming (sCCP [5]) is a process algebra extending CCP [20] in which *agents* interact by exchanging information in the form of *constraints* through a *shared store.* We schematically introduce now a simplified version of sCCP. The interested reader is referred to [5,8] for further details.
An sCCP program is a tuple $\mathcal{N} = (Def, N, \mathbf{X}, init(\mathbf{X}))$. Def, defined according to Table 1, consists of a collection of agents restricted to be *sequential* (i.e. not containing any occurrence of the parallel operator $\parallel$). N, instead, is the initial *network* of the program, a parallel composition of agents of Def (cf. again Table 1). The store consists only in a finite set of *global variables* $\mathbf{X} = \{X_1, \ldots, X_n\}$, usually taking integer values.[3] Finally, $init(\mathbf{X})$ is a predicate on X of the form $\mathbf{X} = \mathbf{x_0}$, assigning an *initial value* to each store variable.
The basic actions π each agent can execute are *guarded updates* of store variables. Such actions have a stochastic duration, given by an exponentially distributed random variable with rate determined by a function $\lambda : X \to \mathbb{R}^+$, depending on the state of the store. This results in a semantics of the language [5] given in terms of a Continuous Time Markov Chain (CTMC [22]).

In the following we will make use of a graphical description of sCCP sequential agents, defined in [6]. Consider the labeled multi-graph whose vertices correspond bi-univocally to the different agents in Def, and whose edges are labeled by stochastic actions π, connecting p to p' if and only if $p = \pi.p' + M$. The portion of this graph reachable from the vertex corresponding to agent p is called the *Reduced Transition System* (RTS) of p.

In [8] we showed how to use sCCP to model biological systems. In particular, we exploited the functional form of rates to encode different chemical kinetics. As an example, in Table 2 we show the model of a simple genetic regulatory network, consisting of one gene self-repressing its own expression. We model the gene as

[3] Here we consider only *stream variables*, i.e variables that can change value over time. Formally, they can be represented as a growing list with an unbounded tail.

$$\text{gene}_{on}(X) :\!-\ [* \to X' = X+1]_{k_p}.\text{gene}_{on}(X) + [X \geq 1 \to *]_{k_b X}.\text{gene}_{off}(X)$$
$$\text{gene}_{off}(X) :\!-\ [* \to *]_{k_u}.\text{gene}_{on}(X)$$
$$\text{degrade}(X) :\!-\ [X \geq 0 \to X' = X-1]_{k_d X}.\text{degrade}(X)$$

Table 2
Model of a self-repressing genetic network. The model of the gene is essentially a *neg-gate*, cf. [3] and next section. In sCCP code, $*$ is a shorthand for true, k_p is the production rate of X, k_b is the binding rate of X to the gene, k_u is the corresponding unbinding rate, and k_d is the degradation rate of π.

a two-state agent (gene$_{on}$ and gene$_{off}$), which can produce a transcript (and the protein) when is active and can be switched off at a rate proportional to the amount of its repressor. We also included an agent to account for degradation of the protein. The RTS of these agents are shown in Figure 1(a).

A *fluid-flow approximation* [15] of the *entire* sCCP program can be defined by treating variables as continuous and describing their time-evolution by means of ODEs as in [6]. In fact, tarting from a sCCP program $\mathcal{N}$, consider the set Def of all defined sequential agents and *associate a (fresh) continuous variable to each of them*. Such variables, together with all variables $\mathbf{X}$ of the store, will be governed by differential equations. Differential equations can be introduced defining an *interaction matrix* I of the sCCP-network. This matrix captures the effect of each action of a sequential agent (i.e. of each edge in its RTS) on system's variables: it has as many rows as system's variables and as many columns as the edges in the RTS of all components, each entry $I[X,e]$ storing the net variation on the variable X caused by the update of edge e. In order to specify the (system of) ODEs, we simply need to store in a vector $\mathbf{r}$ the (functional) *rates* of each transition (times the variables associated to the exit state), following the same order used in the interaction matrix, and compute the product $I \cdot \mathbf{r}$. For the example considered above, there are three variables: X, G_1 (associated to the agent gene$_{on}$), and G_0 (associated to the agent gene$_{off}$).

$$I = \begin{array}{c} X \\ G_1 \\ G_0 \end{array} \begin{pmatrix} 1 & 0 & 0 & -1 \\ 0 & -1 & +1 & 0 \\ 0 & +1 & -1 & 0 \end{pmatrix} \quad \mathbf{r} = \begin{pmatrix} k_p G_1 \\ k_b X G_1 \\ k_u G_0 \\ k_d X \end{pmatrix} \quad ode : \begin{cases} \dot{X} = k_p G_1 - k_d X \\ \dot{G_1} = k_u G_0 - k_b X G_1 \\ \dot{G_0} = k_b X G_1 - k_u G_0 \end{cases}$$

2.2 *Hybrid Automata*

Hybrid automata are dynamical systems presenting both discrete and continuous evolution. Essentially, they are defined using a set of variables evolving continuously in time, subject to instantaneous changes induced by the happening of discrete *control* events. When discrete events happen, the automaton enters its next *mode*, where the laws governing the flow of continuous variables may change. The *traces* of the system are the time traces of the continuous variables. Hybrid automata are generally non-deterministic, hence there can be different traces starting from the same initial value. Therefore, the *simulation* of a hybrid automaton consists in the generation of (a set of) admissible traces. The reader is referred to [14] for an introductory survey. Formally, a hybrid automaton is a tuple $H = (V, E, \mathbf{X}, flow, init, inv, jump, reset)$, where:

- $\mathbf{X} = \{X_1, \ldots, X_n\}$ is a finite set of real-valued variables (the time derivative of X_j is denoted by $\dot{X}_j$, while the value of X_j after a change of *mode* is indicated by X'_j).
- $G = (V, E)$ is a finite labeled graph, called *control graph*. Vertices $v \in V$ are the *(control) modes*, while edges $e \in E$ are called *(control) switches* and model the happening of a discrete event.
- Each vertex $v \in V$ is associated with a set of ordinary differential equations[4] $\dot{\mathbf{X}} = flow(v)$ (referred to as the *flow conditions*). Moreover, $init(v)$ and $inv(v)$ are two formulae on $\mathbf{X}$ specifying the *admissible initial conditions* and some *invariant conditions* that must be true during the continuous evolution of variables in v (forcing a change of mode to happen when violated).
- Edges $e \in E$ of the control graph are labeled by $jump(e)$, a formula on $\mathbf{X}$ stating for what values of variables each transition is active (the so called *activation region*), and by $reset(e)$, a formula on $\mathbf{X} \cup \mathbf{X}'$ specifying the change of the variables' values after the transition has taken place.

2.3 Hybrid Semantics for sCCP

The technique presented at the end of Section 2.1 constructs a system of differential equations governing the entire network. In order to properly describe the dynamics of the system, we were forced to introduce new variables, associated to all the states of sequential agents. A different possibility is that of defining different sets of ODEs, depending on the state of each agent. This leads directly to hybrid automata. More specifically, the translation of a sCCP program $\mathcal{N} = (Def, N, \mathbf{X}, init)$ to a hybrid automaton proceeds in two phases: first, each sequential component p_i of the initial network N is converted into a hybrid automaton, then these hybrid automata are "glued" together using a suitable *product of automata* construction. In the following we sketch the definition of this hybrid semantics. More details can be found in [7].

The first part of the construction, namely the definition of hybrid automata associated to sequential components of the network, is more or less direct: the control graph coincides with the RTS of the component, after removing all looping edges. Flows, instead, are obtained by localizing the general technique for fluid-flow approximation to a single state of a sequential component, constructing interaction matrix and rate vector, using only edges belonging to the RTS looping in that state, and the variables $\mathbf{X}$ of the constraint store. The delicate point in this phase is the definition of activation conditions on edges, with special care in correctly capturing the timing of the associated sCCP transitions. Activation conditions are defined introducing one variable Y_e for each edge e, whose purpose is to control time

[4] Other form of flow's specification are possible (differential inclusions, first order formulae, etc.) but sets of differential equations are sufficient for our purposes here.

varying rates $\lambda = \lambda(t)$. [5] The crucial observation is the fact that every transition, when isolated from the context, constitute a *non-homogeneous Poisson process* [19]. Thus, we can define the *cumulative rate function*

$$\Lambda(t) = \int_{t_0}^{t} \lambda(s)ds,$$

which is a monotone function of t and use the fact, following from the theory of non-homogeneous Poisson processes, that the number of firings at time t behaves like a Poisson variable with rate equal to $\Lambda(t)$. Hence, *the average number of firings of the transition at time t equals* $\Lambda(t)$. Therefore, we may activate the transition whenever $\Lambda(t) \geq 1$, corresponding to the happening of at least one firing on average. This condition is expressed in the hybrid automaton as $Y_e \geq 1$, with the associated transition variable Y_e evolving according to

$$\dot{Y}_e = \frac{d\Lambda(t)}{dt} = \lambda(X). \tag{1}$$

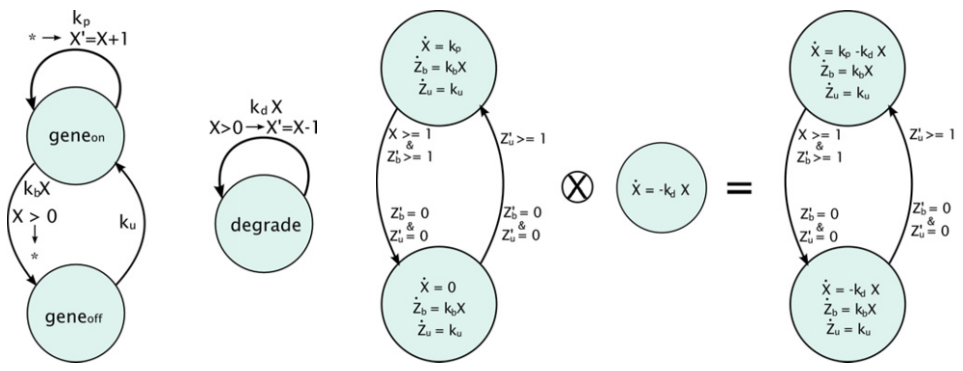
The above point is the kernel of the construction: in order to properly define activation conditions reflecting stochastic behavior in a given interval of time, it is sufficient to control cumulative rate functions. In addition, guards and resets associated to edges of the RTS are added also to edges of the HA.

To complete the construction, hybrid automata associated to sequential components have to be combined together to form the hybrid automaton $H(\mathcal{N})$ associated to the network. The key point is that the same variable of the store must be allowed to be modified by several agents concurrently. Hence, the product automaton must superimpose fluxes, adding the right-hand side of the differential equations of each component for all shared variables. An almost classical product of two hybrid automata can be carried out (see [14]), with the only difference of a special treatment of fluxes for variables shared among the factors. Therefore, modes of the automaton $H(\mathcal{N})$ are the combination of modes of the sequential components. The variables of $H(\mathcal{N})$, instead, are the store variables $\mathbf{X}$ of the system, shared by all components, for which fluxes are added, and the variables involved in the activation conditions of transitions, which are local within each component. The hybrid automaton for the simple genetic network of Section 2.1 is shown in Figure 1.

2.3.1 Non-determinism and Stochasticity

The discrete transitions of the hybrid automaton $H(\mathcal{N})$ associated to an sCCP network $\mathcal{N}$ are, in the simplest case, *urgent*, meaning that they are executed whenever their guard becomes true. As a consequence, $H(\mathcal{N})$ is deterministic: it has a unique temporal trace, determined by the initial conditions (assuming that no two edges can fire at the same time). This rather strong requirement can be relaxed in two ways. First of all, some form of non-determinism can be introduced in the activation conditions substituting $Y_e \geq 1$ with $Y_e \in [\mu_1, \mu_2]$, for an interval $[\mu_1, \mu_2]$

[5] Rates depend on store variables, which vary continuously over time. Hence, also rates are time-varying functions.



(a) Reduced Transition System (b) Hybrid Automaton

Fig. 1. (bf1(a)) Reduced Transition System for the agents of Table 2. (bf1(b)) Hybrid Automata obtained from the agents of Table 2. Variables Z_b and Z_u are associated with the edge from gene$_{on}$ to gene$_{off}$ and from gene$_{off}$ to gene$_{on}$ of the RTS.

containing 1. This approach is discussed in more detail in [7]. Another possibility, instead, is to keep discrete transitions stochastic, associating to each edge of $H(\mathcal{N})$ a non-homogeneous exponential distribution, with cumulative distribution function $F(T \leq t) = 1 - e^{\Lambda(t)}$. This distribution can be simulated, using standard Monte Carlo techniques [22], by drawing an uniform random variable U and solving for t the equation $F(T \leq t) = U$, leading to the condition $Y_e \geq -\log(U)$. The mathematical framework to study such stochastic systems is that of stochastic hybrid automata (SHA [9]), although our treatment in this paper will be more informal.

3 Repressilator

The Repressilator [11] is a synthetic genetic regulatory network composed by three genes expressing three proteins, **tetR**, **λcI**, and **LacI**. These proteins are transcription factors, acting as repressors of their genes in a cyclic fashion, i.e. **tetR** represses **λcI**, **λcI** represses **LacI**, and **LacI** represses **tetR**. The Repressilator, which is expected to oscillate, has been extensively studied in literature, especially in the context of general modeling approaches to genetic networks [4].
In [11], the authors study a mathematical model based on ODE with (cooperative) Hill dynamics for gene production. Their model has either a stable limit cycle or a stable stationary point, depending on the value of parameters.
In [4], instead, authors provide a stochastic model, described in π-calculus using simple building block units, the gene gates. The model is extremely simplified, yet it exhibits a clear oscillatory pattern. Specifically, in the model the protein production is described as a single step action, while repression is modeled by collision of a single repressor on the promoter region, instead of representing explicitly the binding/unbinding mechanism. Clearly, this model abstracts away many biological details, hence it can provide only a qualitative picture of the system dynamics. Taking its biological significance for granted, we will encode it in sCCP and study the relationships among the three different semantics at our disposal. The real bio-

$$\mathrm{Neg}(X,R) :- \quad [* \to X' = X+1]_{k_p}.\mathrm{Neg}(X,R) + [R \geq 1 \to *]_{k_b R}.[* \to *]_{k_u}.\mathrm{Neg}(X,R)$$
$$\mathrm{Degrade}(X) :- [X > 0 \to X' = X-1]_{k_d X}.\mathrm{Degrade}(X)$$
$$\mathrm{Neg}(A,C) \parallel \mathrm{Neg}(B,A) \parallel \mathrm{Neg}(C,B) \parallel \mathrm{Degrade}(A) \parallel \mathrm{Degrade}(B) \parallel \mathrm{Degrade}(C)$$

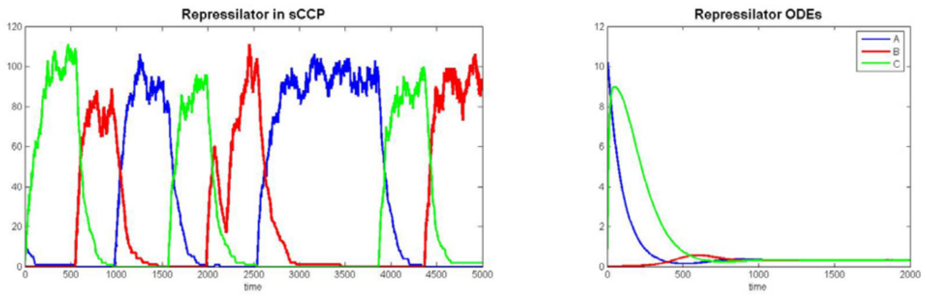
Table 3
sCCP code for the Repressilator. We are using three template processes (with template variables X for the protein and R for the repressor) that are instantiated with the three global stream variables of the system, namely A, B, C. In the code, k_p is the production rate of each gene, k_b and k_u are the binding and unbinding rates of the repressor, and k_d is the degradation rate of each protein.

logical systems remains in the background: we know that it should oscillate, hence we will deem good any model of Repressilator showing distinct oscillations. As a matter of fact, in order to extract also quantitative information, more detailed models should be considered, explicitly representing a cooperative mechanisms of binding/unbinding [4,11]. An sCCP formalization of this model of Repressilator is straightforward [8] and is given in Table 3. The hybrid automaton associated to one *neg-gate*, i.e. one gene, is very similar to the one of Figure 1 (the gene agent of the example of Section 2.1 is indeed a self-repressing neg-gate).

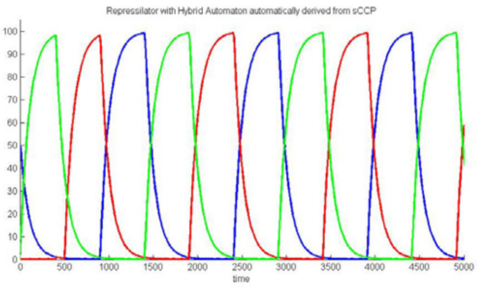
This hybrid model of Repressilator has been first introduced in [7], where it is used to experimentally justify the definition of the hybrid semantics for sCCP. Actually, looking at Figure 2, we can see that while the stochastic and the hybrid model oscillate, the ODEs have a radically different behavior, converging to a stable state. As a matter of fact, the key difference between the stochastic and hybrid models on one side and the ODEs on the other is that the dynamics of gene activations and deactivations is discrete for the former and continuous for the latter. This discreteness is thus essential for the oscillations to exist at the level of detail of this model.

In this section we will analyze the hybrid model of Repressilator, with particular attention to robustness of the oscillatory behavior with respect to parameter changes. We will also compare the hybrid and the stochastic systems, trying to see if stochasticity increases robustness of the model, or if discreteness of genes is enough to produce oscillations. We perform this analysis following the suggestions of [4], where a thorough analysis of the stochastic model has been presented. First of all, what really matters are the ratios between parameters, hence we can fix one of them, letting only the other three vary. We choose to fix the production rate k_p (cf caption of Table 3) to the nominal value of 1.0.

Analyzing the robustness for hybrid systems is a challenging task, as the techniques available for ODEs are not applicable due to the discreteness of the dynamics. In order to get an idea of what happens for Repressilator, we did an extensive experimental study, simulating the hybrid automaton for different combinations of parameters and using this information to draw a coarse grained map of the parameter space. We use urgent transitions, guaranteeing that the dynamics of the automaton is deterministic, so that a single (accurate) numerical simulation suffices to understand dynamical properties. The results of this analysis are shown in Figure 3, where we use color maps to describe the presence/absence of oscillations and their amplitude relative to the maximum value attainable by proteins under the combination of parameters considered (equal to k_p/k_d). As we can see, the



(a) (b)



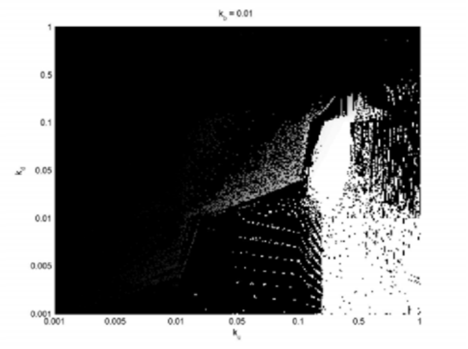
(c)

Fig. 2. 2(a) Stochastic time trace for the Repressilator system of Table 3. Parameters are $k_p = 1$, $k_d = 0.01$, $k_b = 1$, $k_u = 0.01$. 2(b) Solution of the differential equations associated to the sCCP program of Table 3. Parameters are the same as in stochastic simulation. 2(c) Trace of the hybrid automaton model of Repressilator, using the same parameters as before.

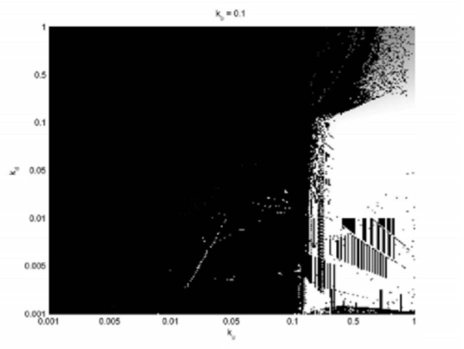
oscillations are present for a wide area of the parameters space. Furthermore, we can observe how the increase of the binding rate k_b stabilizes the behavior of the system, expanding the region of oscillations. Moreover, oscillations are present more frequently if $k_u > k_d$. These results are in line with the behavior of the stochastic system, cf. [4].

In order to compare the hybrid and the stochastic models, we consider parameter combinations near the boundary of the oscillatory region, comparing the two systems. What happens is depicted in Figure 4: the hybrid system, when it oscillates, has small and brief oscillations. The corresponding stochastic system, instead, exhibits a very noisy oscillatory pattern, due to the small amount of molecules into play. Actually, the low amplitude of oscillations should warn us of a potentially disruptive effect of noise. However, the fact that the hybrid model oscillates and that stochastic noise tends to disrupt this behavior tells us that the oscillations are indeed induced by the discreteness of gene dynamics rather than by an effect of stochasticity.

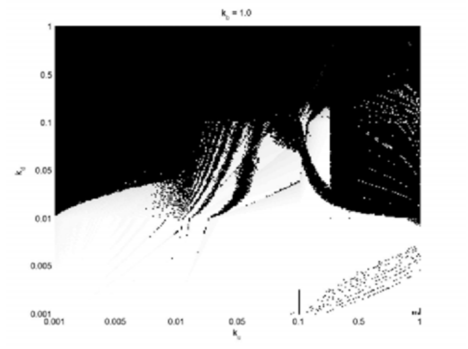
The hybrid model of Repressilator we just discussed is symmetric, as all three genes share the same parameters. If the deterministic model starts from identical initial conditions (i.e. $A_0 = B_0 = C_0$), then the automaton has no way to break this starting symmetry, hence it should not exhibit oscillations with alternating peaks. This is indeed the case, as Figure 5(a) shows. Nevertheless, a little amount



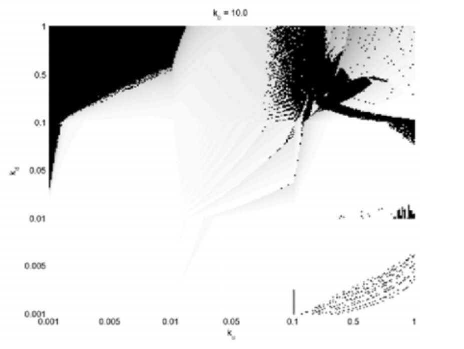
(a) $k_b = 0.01$



(b) $k_b = 0.1$



(c) $k_b = 1$



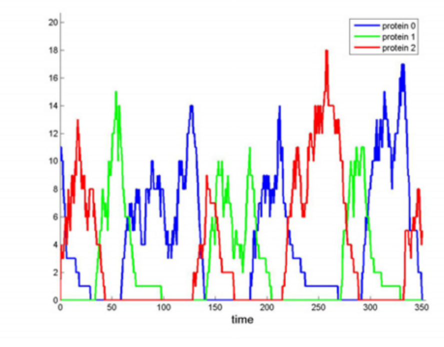
(d) $k_b = 10$

Fig. 3. **3(a)–3(d)**Presence or absence of oscillations and their magnitude (relative to the maximum amplitude k_p/k_d) in the k_u-k_d plane (unbinding rate, degradation rate, resp.), for various values of k_b (binding rate). k_u and k_d range from 0.001 to 1. The color map assigns light colors to bigger oscillations, thus white points correspond to full oscillations, while black point denote absence of oscillations.

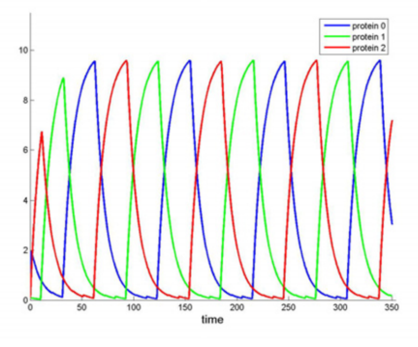
of non-determinism can be introduced according to Section 2.3.1, and its effect is to break the initial symmetry, leading the automaton away from the (unstable) solution of Figure 5(a) into the usual (stable) oscillatory pattern (cf. Figure 5(b)). In simulating the non-deterministic automaton, we considered an uniform measure on the set of admissible traces, generating one at random.

The Repressilator can be generalized by considering n instead of 3 genes, still repressing cyclically [17]. It can be shown, for an ODE model using cooperative Hill's equations, that oscillations are possible if and only if n is odd. The same effect can be observed also for the generalized hybrid Repressilator (results not shown).

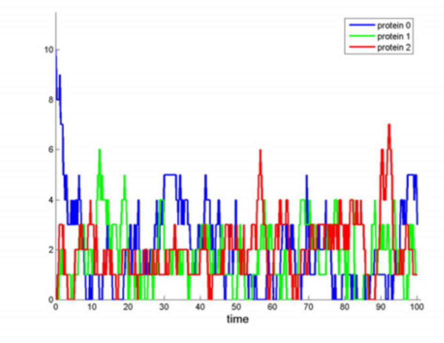
The experimental evidence presented shows that discreteness can have an important role in stabilizing the dynamics of a system, hence it cannot always be ignored safely in the modeling activity.



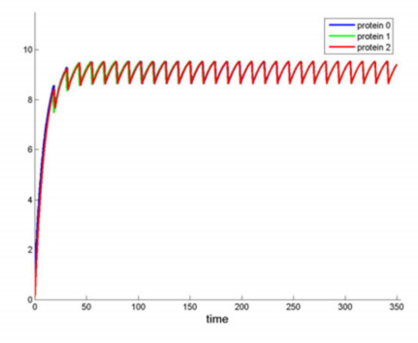
(a) CTMC, oscillating



(b) HA, oscillating

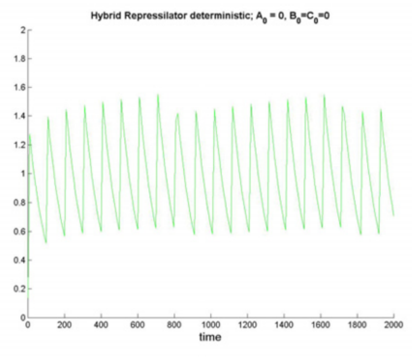


(c) CTMC, not oscillating

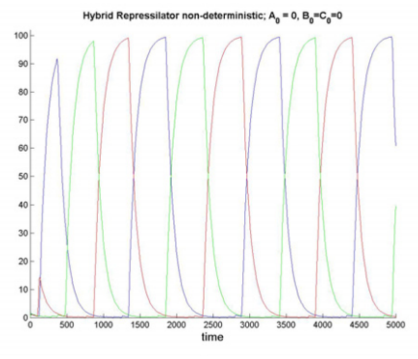


(d) HA, not oscillating

Fig. 4. **4(a)** Simulation of the stochastic system for $k_b = 10.0$, $k_d = k_u = 0.1$. Oscillations are still present, though extremely noisy. **4(b)** Simulation of the hybrid system for $k_b = 10.0$, $k_d = k_u = 0.1$. Oscillations are still present, although with a small absolute amplitude (approximately equal to 10). **4(c)** Simulation of the stochastic system for $k_b = 0.01$, $k_d = 0.5$, and $k_u = 1.0$. The behavior is extremely noisy. **4(d)** Simulation of the hybrid system for $k_b = 0.01$, $k_d = 0.5$, and $k_u = 1.0$. The system does not present an oscillatory behavior, but fluctuates around a steady state.



(a) Deterministic HA



(b) Non-deterministic HA

Fig. 5. **5(a)** Simulation of the deterministic hybrid Repressilator, with parameters as in Figure 2 and initial conditions $A_0 = B_0 = C_0 = 0$. **5(b)** Simulation of the non-deterministic hybrid Repressilator with the same parameters as above and with activation conditions $Y_e \in [0.9, 1.1]$.

4 Circadian Clock

We investigate now the mass-action model of the circadian clock presented in [21], where the authors show that both the ODE-based model and the stochastic model exhibit regular oscillations (for suitable parameters' values). However, the stochastic model is more robust as if internal noise was exploited by Nature to increase stability of function (i.e., for a clock, oscillations).

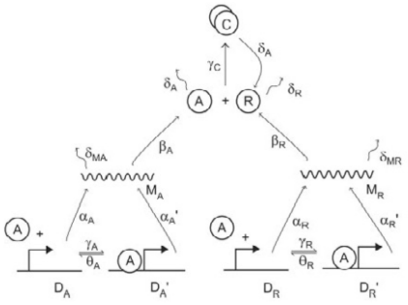
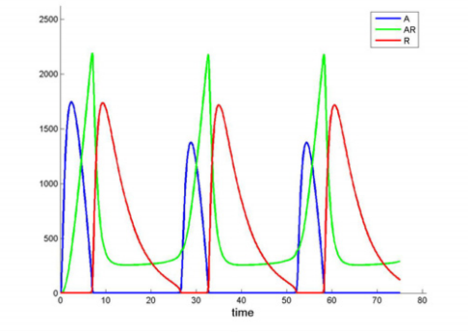


Fig. 6. Biochemical network for the circadian rhythm regulatory system. The figure is taken from [21], like numerical values of rates. Rates are set as follows: $\alpha_A = 50$, $\alpha'_A = 500$, $\alpha_R = 0.01$, $\alpha'_R = 50$, $\beta_A = 50$, $\beta_R = 5$, $\delta_{MA} = 10$, $\delta_{MR} = 0.5$, $\delta_A = 1$, $\delta_R = 0.2$, $\gamma_A = 1$, $\gamma_R = 1$, $\gamma_C = 2$, $\theta_A = 50$, $\theta_R = 100$.

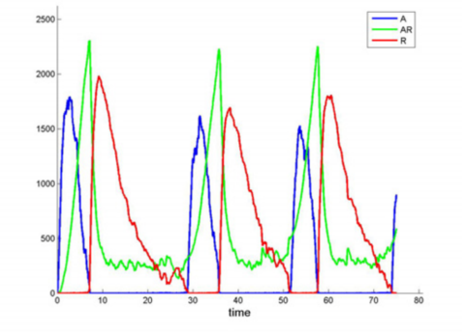
The circadian system is schematically depicted in Figure 6. It is a simple abstraction of the machinery involved in the regulation of the circadian rhythm of living beings, a typical mechanism for responding to environmental stimuli, in this case the periodic change between light and dark. Basically, this system behaves like a clock, expressing proteins A and R periodically with a stable period. We stress that the *stability* of the period is an essential requirement for a circadian clock model to be realistic.
The system consists of two genes, one expressing an activator protein A, the other producing a repressor protein R. The transcription and the translation phases are modeled explicitly. Protein A is an enhancer for both genes, meaning that it regulates positively their expression. Repressor R, instead, can capture protein A, forming the complex AR and making A inactive. Proteins A and R are degraded at a specific rate (see caption of Figure 6 for more details about the numerical values), but R can be degraded only if it is not in complexed form, while A can be degraded in any form. Notice that regulation activity of A is modeled by an explicit binding to the gene, which remains stimulated until A unbinds.

We will focus on two main questions: What is the interplay between discreteness and stochasticity in the increase of robustness? What are the key interactions that should be kept discrete/stochastic? In order to tackle these questions, we first encode the model in sCCP, turning then to an experimental study of the relationships among the three semantics at our disposal: stochastic, ODE-based, and hybrid.

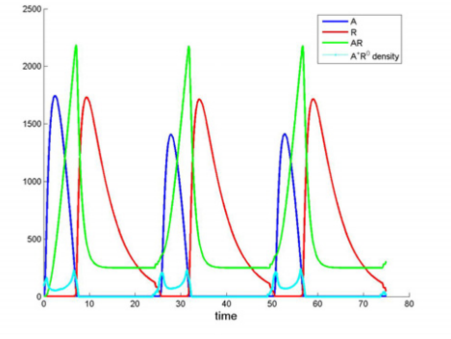
The code of the sCCP program modeling the circadian clock [8] is straightforward: each reaction is associated to a single looping agent (similar to degradation agents of Section 2.1 and Table 3), while genes are modeled by two-state agents, describing explicitly the binding and unbinding of the enhancer. The hybrid automaton obtained from the sCCP model has 4 states (it is the product of 2 two-state automata and few one-state automata), corresponding to the possible combinations of gene states (enhanced or normal).



(a) ODE



(b) CTMC



(c) HA

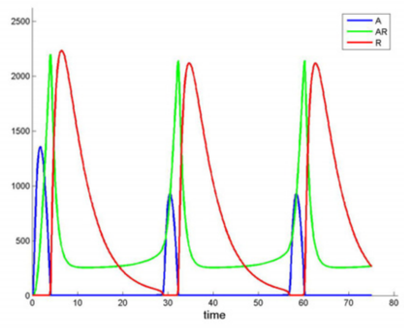
Fig. 7. **7(a)** Simulation of ODE-based model of the circadian clock, using parameters of Figure 6. **7(b)** Simulation of the CTMC-based model of the circadian clock. **7(c)** Simulation of the HA-based model of the circadian clock. The small cyan curve indicates a *temporal density* for the combination of gene states in which A is in the enhanced state and R is in the normal state. Each point represents the fraction of time in which genes are in this specific combination, for a 10 seconds long time-window, centered in the considered instant. The curve is scaled by a factor of 1000.

In Figure 7 we show a comparison of stochastic, ODE, and hybrid models, for the set of parameters given in [21] (cf. also caption of Figure 6): The behavior is essentially the same for all three systems.

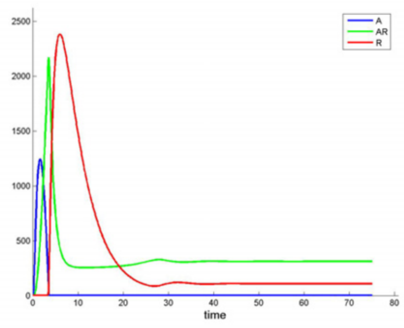
What are the interactions responsible for oscillations? Looking at the plots of Figure 6, we can see that a peak of A is followed by a peak of AR and then by a peak of R. Actually, an increase of the amount of A stimulates the transcription of both genes A and R. However, the production of A is faster than the production of R both in the normal and in the excited states. Therefore, for some time, R is not able to saturate A by complexation, hence A increases. This effect is contrasted by the much slower degradation of the mRNA of R and of the protein R itself, with respect to the mRNA of A and the protein A. This makes the concentration of R

grow so that A becomes present only in the complexed form, thus inhibiting the enhancing of gene expression. This, in turn, makes R gradually decay, due to its slow basic production rate. The cycle then starts again.

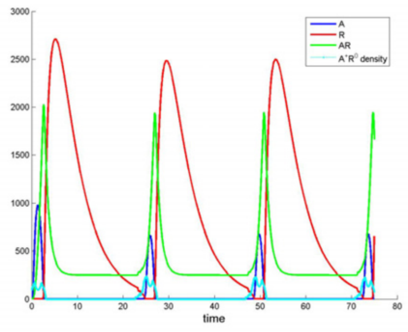
The previous discussion gives an idea of the complexity of this process, which is the result of the interaction of many factors. However, one parameter that should be important for the process is the translation rate of R. Increasing this parameter, in fact, should decrease both duration and height of the peak of expression of A, eventually making the oscillations disappear. This intuition is indeed confirmed in Figures 8(a) and 8(b), where solutions of the ODEs are presented for β_R equal to 12 and 15, respectively. As we can see, the size of A's peak decreases until it disappears.



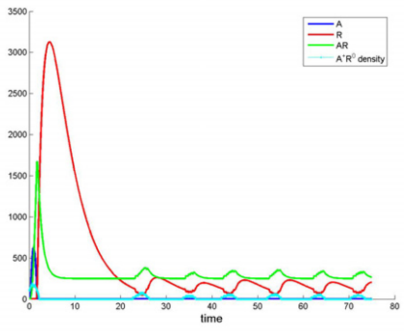
(a) ODE, $\beta_R = 12$



(b) ODE, $\beta_R = 15$



(c) HA, $\beta_R = 25$

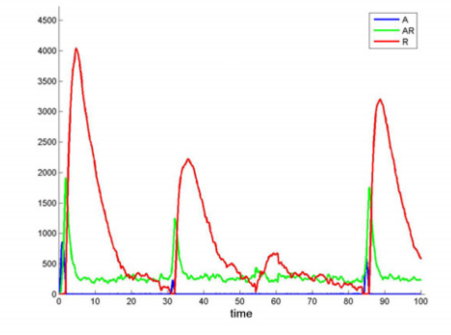


(d) HA, $\beta_R = 50$

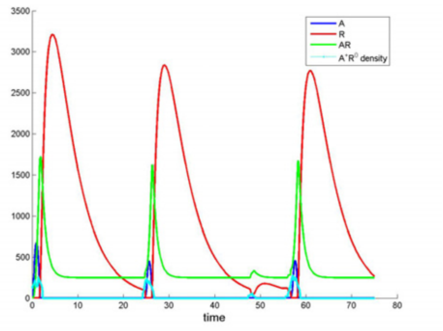
Fig. 8. **8(a)** Simulation of the ODE-based model of the circadian clock with $\beta_R = 12$ and all other parameters as in the caption of Figure 6. **8(b)** Simulation of the ODE-based model of the circadian clock with $\beta_R = 15$. **8(c)** Simulation of the HA-based model of the circadian clock with $\beta_R = 25$ and all other parameters as in the caption of Figure 6. **8(d)** Simulation of the HA-based model of the circadian clock with $\beta_R = 50$.

Now, this is the effect of β_R on the system of differential equations, where the state of the gene is represented by a variable taking values in $[0, 1]$. We may wonder whether a discrete description of the gene dynamics will increase or decrease robustness. Indeed, the oscillatory region is extended, as confirmed by Figures 8(c) and 9(c), the first showing a plot of the hybrid model for $\beta_R = 25$, the second comparing the stability between different formalisms.

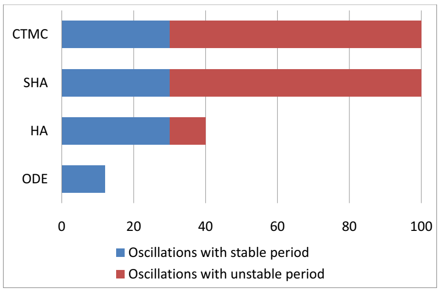
However, the authors in [21] claim that it is internal noise that increases the stability of oscillations. As a matter of fact, the introduction of stochasticity increases the interval of β_R in which the system oscillates, as can be seen from Figures 9(a) (showing a stochastic simulation for $\beta_R = 50$) and 9(c). The question is now to understand how much stochasticity is necessary: do we need to describe the whole system as stochastic, or just a few transitions will suffice? In order to answer, we considered the stochastic hybrid automaton model introduced in Section 2.3.1. The only stochastic transitions of this automaton are the binding and unbinding of A to the promoter region of the two genes. Its simulation for $\beta_R = 50$, presenting an oscillatory pattern, is shown in Figure 9(b).



(a) CTMC, $\beta_R = 50$



(b) SHA, $\beta_R = 50$



(c) Stability w.r.t. β_R

Fig. 9. **9(a)** Simulation of the CTMC-based model of the circadian clock with $\beta_R = 50$ and all other parameters as in the caption of Figure 6. **9(b)** Simulation of the SHA-based model of the circadian clock with $\beta_R = 50$. **9(c)** Stability diagram for β_R. The blue bar represents oscillations with stable period. The red bar, instead, indicates the region of the parameter space with oscillations with variable period. This region, for SHA and CTMC, extends well beyond 100. It is noteworthy that the region with stable oscillations essentially coincides for HA, SHA, and CTMC.

The experimental tests presented suggest that discreteness is an important factor in increasing the robustness of the model. Actually, only a limited amount of discreteness seems sufficient in this respect, namely the description of genes as two-state automata.

As a matter of fact, also stochasticity plays an important role: Treating the binding and unbinding of the enhancer as discrete and stochastic basically guarantees the same behavior as the CTMC model. But how does stochasticity influence the oscillations? Remember that oscillation starts with a peak of the free enhancer A. This event corresponds to a situation in which gene A is enhanced but gene R is in its normal state. Specifically, a peak in the amount of A is triggered whenever this combination of gene states occurs for a sufficiently long time, as confirmed by the small cyan curves in Figures 7(c), 8(c) representing a sort of "temporal density" (cf. caption of Figure 7(c)). In the deterministic case, the increase of β_R gradually reduces this density below a critical threshold, under which oscillations disappear. Fluctuations of the stochastic system, on the other hand, can still bring the density over the threshold, and trigger an oscillation loop. Notice that, the bigger is β_R, the harder is for fluctuations to make the density overcome this threshold, hence oscillations have no more a stable period.
Actually, as the diagram of Figure 9(c) shows, the stochastic models have the *same robustness* as the deterministic HA if we consider only oscillations with a *stable* period. This is another hint on the importance of being (a little bit) discrete.

5 Conclusions

In this paper we analyzed a model of Repressilator and a model of the circadian clock, using as a key instrument the fact that we have at our disposal one description language, sCCP, equipped with three different semantics. Thus, we described these systems in sCCP and studied the hybrid automaton associated to them, comparing its behavior with the one exhibited by the ODE and the CTMC models. Specifically, we have seen that the hybrid Repressilator oscillates for a broad range of parameters, similarly to the stochastic system. Hence, it is a robust model. We have also seen how the introduction of a controlled amount of non-determinism can overcome some rigidities connected with the deterministic evolution of the hybrid system.
The circadian clock is also stabilized by the introduction of discreteness. In this case, however, stochasticity has a further stabilizing role. This effect is depending on events that, even if inhibited by the deterministic evolution, can be still triggered in the stochastic system due to its intrinsic variability.

We can then conclude that, at least in the two cases considered, robustness is increased by the discrete treatment of gene repression or enhancement. All in all, a small amount of discreteness can guarantee a more precise description (i.e. qualitatively similar to the CTMC-based model), without increasing too much the computational cost of simulation with respect to numerical integration of ODE.

From a philosophical point of view, the examples discussed suggest that, in cases in which the stochastic systems are more robust than their ODE-bases counterpart, the essential ingredient for robustness is *the discreteness of dynamics*, rather than the stochasticity. Testing extensively this hypothesis requires a more refined hybrid semantics for sCCP. In fact, the hybrid automata associated to sCCP programs have a rigid degree of discreteness, determined by the logical structure of agents.

Nevertheless, the mapping from sCCP to HA, however, can be generalized allowing a variable level of discreteness, letting the user specify what transitions are to be kept discrete and what transitions are to be approximated continuously. This will associate to an sCCP program an entire *lattice of hybrid automata*, differing in the degree of discreteness. An analysis of such lattice can identify the key discrete transitions, thus refining the study of robustness presented here. In this direction we are investigating an approach based on temporal logic. Essentially, we would like to understand how satisfiability of certain temporal formulae (expressing dynamical properties of interest) varies in the lattice, connecting patterns of satisfiability with structural properties of the lattice. As a matter of fact, also stochasticity can be introduced parametrically, associating to an sCCP program a lattice of SHA.

Our approach has also connections with hybrid simulation algorithms for multi-scale systems, like [13]. In fact, it is not difficult to describe such algorithms as simulations of a certain (stochastic) HA associated to the original system. Hence, they may be described within the generalization of our formal framework in terms of the lattice of (S)HA. This point of view can possibly provide new insights on these methods.

Another direction that we wish to further investigate is the description of genetic networks by *hybrid gene gates*, i.e. by the hybrid automata associated to gene gates. An interesting feature of such systems is that the continuous dynamics of proteins is decoupled: proteins influence each other expression only through discrete transitions.

Acknowledgement

We thank FIRB LIBi and PRIN “BISCA” projects for financial support.

References

[1] R. Alur, C. Belta, F. Ivancic, V. Kumar, M. Mintz, G. Pappas, H. Rubin, and J. Schug. Hybrid modeling and simulation of biomolecular networks. In *Proceedings of Fourth International Workshop on Hybrid Systems: Computation and Control*, volume LNCS 2034, pages 19–32, 2001.

[2] M. Antoniotti, A. Policriti, N. Ugel, and B. Mishra. Model building and model checking for biochemical processes. *Cell Biochemistry and Biophysics*, 38(3):271–286, 2003.

[3] R. Blossey, L. Cardelli, and A. Phillips. A compositional approach to the stochastic dynamics of gene networks. *T. Comp. Sys. Biology*, pages 99–122, 2006.

[4] R. Blossey, L. Cardelli, and A. Phillips. Compositionality, stochasticity and cooperativity in dynamic models of gene regulation. *HFPS Journal, in print*, 2007.

[5] L. Bortolussi. Stochastic concurrent constraint programming. In *Proceedings of 4th International Workshop on Quantitative Aspects of Programming Languages (QAPL 2006)*, volume 164 of *ENTCS*, pages 65–80, 2006.

[6] L. Bortolussi and A. Policriti. Stochastic concurrent constraint programming and differential equations. In *Proceedings of Fifth Workshop on Quantitative Aspects of Programming Languages, QAPL 2007*, volume ENTCS 16713, 2007.

[7] L. Bortolussi and A. Policriti. Hybrid approximation of stochastic concurrent constraint programming. In *To be presented at IFAC 2008. Available upon request*, 2008.

[8] L. Bortolussi and A. Policriti. Modeling biological systems in concurrent constraint programming. *Constraints*, 13(1), 2008.

[9] M.L. Bujorianu and J. Lygeros. Theoretical foundations of stochastic hybrid systems. In *Proceedings XVI Int. Symp. on Math. Theory of Networks and Systems (MTNS 2004)*, 2004.

[10] M. Calder, S. Gilmore, and J. Hillston. Modelling the influence of rkip on the erk signalling pathway using the stochastic process algebra PEPA. *Transactions on Computational Systems Biology*, 4230:1–23, 2006.

[11] M.B. Elowitz and S. Leibler. A synthetic oscillatory network of transcriptional regulators. *Nature*, 403:335–338, 2000.

[12] D.T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *J. of Physical Chemistry*, 81(25), 1977.

[13] E.L. Haseltine and J.B. Rawlings. On the origins of approximations for stochastic chemical kinetics. *J. Chem. Phys.*, 123, 2005.

[14] T. A. Henzinger. The theory of hybrid automata. In *LICS '96: Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science*, 1996.

[15] J. Hillston. Fluid flow approximation of PEPA models. In *Proceedings of the Second International Conference on the Quantitative Evaluation of Systems (QEST'05)*, 2005.

[16] H. Kitano. Computational systems biology. *Nature*, 420:206–210, 2002.

[17] S. Müller, J. Hofbauer, L. Endler, C. Flamm, S. Widder, and P. Schuster. A generalized model of the repressilator. *J. Math. Biol.*, 53:905–937, 2006.

[18] C. Priami, A. Regev, E. Y. Shapiro, and W. Silverman. Application of a stochastic name-passing calculus to representation and simulation of molecular processes. *Inf. Process. Lett.*, 80(1):25–31, 2001.

[19] S. M. Ross. *Stochastic Processes.* Wiley, New York, 1996.

[20] V. A. Saraswat. *Concurrent Constraint Programming.* MIT press, 1993.

[21] J. M. G. Vilar, H. Yuan Kueh, N. Barkai, and S. Leibler. Mechanisms of noise resistance in genetic oscillators. *PNAS*, 99(9):5991, 2002.

[22] D. J. Wilkinson. *Stochastic Modelling for Systems Biology.* Chapman & Hall, 2006.