Available online at www.sciencedirect.com
ScienceDirect
ELSEVIER
Electronic Notes in Theoretical Computer Science 253 (2009) 83–98
Electronic Notes in Theoretical Computer Science
www.elsevier.com/locate/entcs

Exploiting non-Markovian Bio-Processes

I. Mura[b] D. Prandi[b,1] C. Priami[a,b] A. Romanel[a,b]

[a] *Dipartimento di Ingegneria e Scienza dell'Informazione, Università di Trento*

[b] *The Microsoft Research - University of Trento Centre for Computational and Systems Biology*

Abstract

The Stochastic Simulation Algorithm (SSA) is a milestone in the realm of stochastic modeling of biological systems, as it inspires all the current algorithms for stochastic simulation. Essentially, the SSA shows that under certain hypothesis the time to the next occurrence of a biochemical reaction is a random variable following a negative exponential distribution. Unfortunately, the hypothesis underlying SSA are difficult to meet, and modelers have to face the impact of assuming exponentially distributed reactions besides the prescribed scope of applicability. An opportunity of investigation is offered by the use of generally distributed reaction times.
In this paper, we describe how general distributions are introduced into BlenX, a programming language designed for specifying biological models. We then experiment the new extension on few examples of increasing complexity and discuss how the quantitative behaviour of a model is affected by the choice of the reaction time distribution.

Keywords: Semantics, Systems Biology, Non-Markovian Processes, Quantitative analysis.

1 Motivations

Systems Biology [17] studies the relations and interactions among the components of biochemical systems in order to understand how they globally work. Many different approaches have been used and developed over the last years to model, simulate and analyze complex interactions mechanisms characterizing biochemical systems. Recently, there has been a significant interest in the stochastic modeling approach, mainly because there is enough experimental evidence that stochasticity could play an important role (see, e.g., [28,1]). Two of Gillespie's papers [12,13] enabled this research line by introducing the theoretical basis underlying the stochastic dynamics of biochemical systems together with an effective simulation algorithm, namely the Stochastic Simulation Algorithm (SSA). The essence of Gillespie's work is a precise characterization of the stochastic behaviour of chemical systems. In the stochastic realm the amount of a molecule i at time t is specified by a discrete random variable

[1] Correspondig author: prandi@cosbi.eu

doi:10.1016/j.entcs.2009.10.007

$X_i(t)$, and a system is represented by a vector of random variables $\boldsymbol{X}$. The evolution of $\boldsymbol{X}$ is then modeled by a joint probability distribution $\mathcal{P}(\boldsymbol{X}, t)$, i.e. the Chemical Master Equation (CME) [20], expressing the probability that at time t there are X_i molecules of the ith species. In general the CME is hard to solve either analytically or numerically. Therefore, Gillespie's papers introduce a set of sufficient conditions at the molecular level that, when satisfied, allow to assume the time to the next occurrence of a reaction to be a random variable following a negative exponential distribution. This enables the definition of SSA, a Monte Carlo simulation method, whose execution produces a correct trace in the state space defined by the CME. This is true under the proviso that the Gillespie's hypotheses are satisfied, i.e., systems are well-mixed, under thermal equilibrium, and only for specific types of chemical reactions, often termed *elementary*. In this context, an elementary reaction is one that does not abstract any intermediate species.

Although the SSA also works besides the prescribed scope of applicability, as it seems to be proved by the successful validation of many stochastic models of biochemical systems against experimental data, it is easy to realize the difficulties in describing a biochemical system in terms of elementary reactions. Quite often, there is an incomplete knowledge of the full set of reactions, and mesoscopic or macroscopic transformations are the only observable ones. Therefore, systems biologists are confronted with the necessity of including abstract reactions in their models. Besides being dictated by a lack of detailed knowledge, abstractions are a convenient means to limit the size of models. However, abstractions may (and usually do) introduce approximations in the behaviour of models, the impact of which should be understood or at least estimated. Consider for instance the process of gene transcription [2], which is commonly modeled by a single synthesis reaction. Let us ask whether the fundamental hypothesis would hold for such a reaction. If one considers the complexity of the transcription process, which encompasses the sequential assembly of a long nucleotide sequence based on the gene template scan, it is quite intuitive to understand that it is not an elementary reaction in itself. If we assume that every reaction step in the transcription process is elementary, satisfying the fundamental hypothesis of Gillespie, the time to the occurrence of a single transcription event will have a distribution that results from the composition of those of the elementary reactions. Apart for the operations of multiplication for a scalar and minimum, the class of negative exponential random variables is not closed with respect to composition. For instance, the sum and maximum of a set of negative exponential random variables is not a negative exponential random variable. Thus, a simple mathematical argument shows that Gillespie's fundamental hypothesis does not hold for gene transcription modeled by a single synthesis reaction. Indeed, modeling studies [27] indicate that the process of transcript production exhibits less variability than a simple Poisson process, the one assumed if the transcription times follow a negative exponential distribution. However, when the pausing that occurs in gene transcription is frequent enough, the transcript production process tends to become a Poisson process [27].

Given that it would not feasible to model every biological system at the elemen-

tary level, we need to devise adequate abstractions for their quantitative behaviour. An opportunity is offered by the inclusion of generally distributed reaction times into models. Reduced variability distributions (such as Erlang, Hypoexponential) and increased variability (Hyperexponential) provide an improved flexibility to modelers, allowing better matches with statistical characteristics of biological phenomena. This requires extending simulation engines to treat stochastic models beyond the class of SSA inspired algorithms.

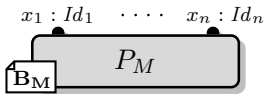
Here we rely on the flexibility of process calculi to reason about general distributions and biological systems. Some related work on generally-distributed stochastic process calculi can be found in [4,15].

The paper is organized as follows. Sect. 2 gives a brief tutorial on BlenX [10], a process calculi inspired programming language designed for modeling biological systems, and we comment on how general distributions have been managed. Then, Sect. 3 discusses the impact of assuming abstract reactions as exponentially distributed through a few biological examples of increasing complexity. Finally, Sect. 4 exploits the capability of models including generally distributed reaction times.

2 The BlenX Language

The BetaWB framework [18] is a computational tool that supports textual and visual programming with BlenX. The BetaWB can be seen as an *in-silico* laboratory, where (in-silico) experiments can be designed (i.e., a BlenX program is written), simulated and analyzed. The quantitative component of the experiments is guaranteed by the stochastic capability of BlenX, on the line of [9], where a continuous-time Markov process is taken as foundational quantitative model. This section provides an introduction to the visual version of BlenX and introduces the formal tools for managing non-Markovian processes.

BlenX is a programming language inspired by the Beta-binders [26] process calculus, and it is specifically designed for modeling entities that can change their behaviour in response to external stimuli. A general biological molecule M with n interaction sites is depicted as a *box* $\mathbf{B_M}$:



The program P_M is called *process* and allows describing the behaviour of $\mathbf{B_M}$. In particular, P_M activates proper replies to external signals caught by *interaction sites* $x_i : Id_i$. *Types* Id_i discriminate among allowed and disallowed interactions, mimicking interaction mechanisms based on compatibility [23]. The *name* x_i is used by the process P_M to modify or to interact through the associated type Id_i. Process P_M is written in a process calculi style and therefore it has few primitives inspired by both π-calculus [21] and molecular biology [2]. Instead of a long and boring description of the syntax and semantics of BlenX we prefer to put the language on the road by considering an example that will be used in the next section.

Enzymes [2] are by far the most important elements of any biological system,

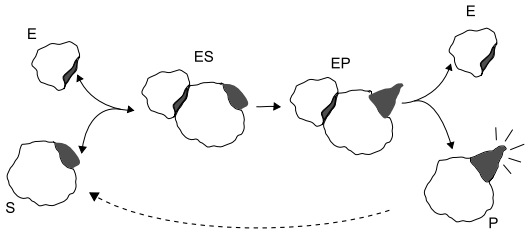
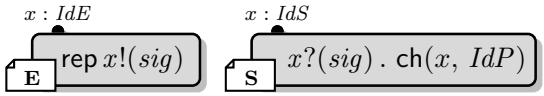


Fig. 1. Enzymatic activation.

because they catalyze biochemical reactions that make life possible. Basically, enzymes bind to one or more ligands, called *substrates*, and convert them into one or more chemically modified *products*. A simple [2] enzymatic reaction is considered in Fig 1. The enzyme E and the substrate S encounter to form the enzyme-substrate intermediate ES. This reaction can be reversed, but the formation of ES is favored when many substrates molecules are available. When S is bound, E sends a signal enabling the modification of substrate S into product P. Finally, the product P is released, and the enzyme E regenerated. Enzyme and substrate can be modeled by the parallel composition of two boxes **E** and **S**, written $\mathbf{E} \parallel \mathbf{S}$:

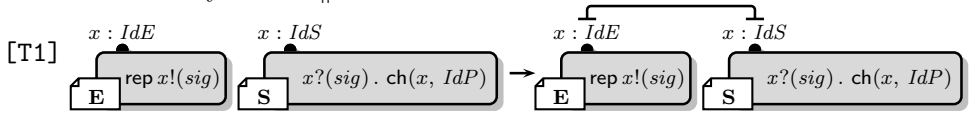


The primitive $x!(sig)$ sends a signal sig through interaction site $x : IdE$. Replication rep assures that the process sends a signal each time it is needed, i.e., each time substrate and enzyme interact. The primitive $x?(sig)$ waits a signal on the interaction site $x : IdS$ that enables the change of the type in IdP by means of $\mathsf{ch}(x, IdP)$.

In order to interact, enzyme and substrate have to bind forming the enzyme-substrate intermediate. BlenX allows to specify if two types can complex in a 5-tuple. For instance,

$$(IdE, IdS, 0.5, 0.1, 0.01) \tag{1}$$

means that type IdE and IdS, expressed by boxes **E** and **S** respectively, may bind with a rate of 0.5. The rate is the unique parameter of a negative exponential distribution that describes the stochastic behaviour of the action. The other two values in the tuple (1) are the rates for decomplexing and interacting and will be commented later in this section. The tuple (1) allows inferring the following evolution of the system $\mathbf{E} \parallel \mathbf{S}$:

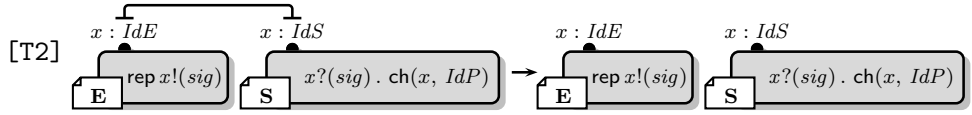


BlenX is equipped with syntax-driven rules, the so-called *operational semantics* [22], that allows the inference of the possible transitions of a BlenX program. For instance, transition [T1] above means that there is a proof that the system $\mathbf{E} \parallel \mathbf{S}$ can perform a transition, written $\rightarrow$, to a system where boxes **E** and **S** are bound through

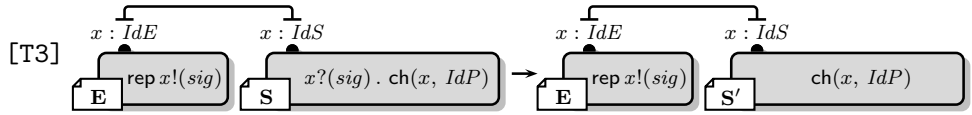
[2] do not consider dashed arrow at this stage

interaction sites $x : IdE$ and $x : IdS$, respectively.

Following Fig. 1, the reaction leading to the intermediate form enzyme-substrate can be reversed, namely boxes **E** and **S** can decomplex. This is represented by transition [T2] below, where the rate of this transition is specified by the forth parameters, 0.1 in this case, of tuple (1).

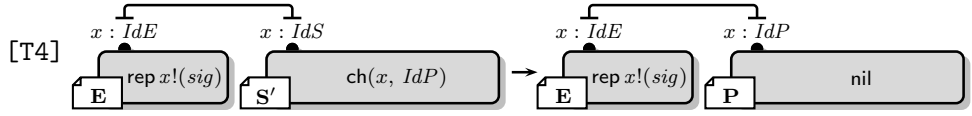


However, after transition [T1] the enzyme-substrate complex can undergo certain modifications that lead to the release of the product. This is modeled in BlenX as a communication on the private channel created by the binding of the interaction sites, followed by a change in the type of the interaction site $x : IdS$. In particular, the process $\mathsf{rep}\, x!(sig)$ can send a signal sig through the interaction site $x : IdE$; the process $x?(sig)\,.\,\mathsf{ch}(x, IdP)$ can receive a signal from the interaction site $x : IdS$ and then ("then" is represented by an infix dot) enable the change primitive. Since box **E** can output and box **S** can input, and they are connected in the right way, a communication is possible, leading to:



The rate of the communication in transition [T3] is specified in the fifth component of definition (1). Notice that the process inside box **E** is not changed by [T3] because the guarded replication operator $\mathsf{rep}\_$ allows to regenerate output $x!(sig)$ each time it is consumed.

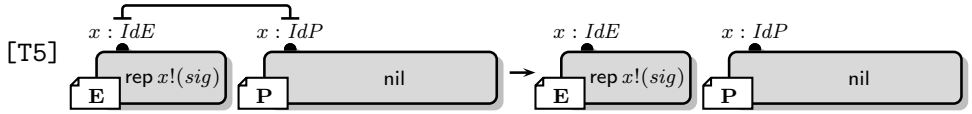
The process $\mathsf{ch}(x, IdP)$ into box $\mathbf{S}'$ changes the type of the interaction site $x : IdS$ from IdS to IdP, completing the transformation from substrate S to product P, described by transition [T4]:



The process nil is the empty process, but more complex behaviour for box **P** can be specified. Looking back to Fig. 1, the last step of enzyme activation is the release of the product and the regeneration of the enzyme. The unbind of boxes **E** and **P** is specified by adding a new tuple

$$(IdE, IdP, 0, inf, 0) \tag{2}$$

that prescribes that each time the types IdE and IdP are bound they have to unbound *immediately*, represented by the key-word *inf* (i.e. infinite rate). The zeros in the third and fifth position means that types IdE and IdP cannot bind and cannot interact if they are bound. The tuple (2) allows inferring transition [T5]:



We conclude this brief tutorial on the BlenX language introducing *events*. The primitives we describe above work with elementary reactions, but, as we already pointed, it is difficult to describe biological systems only in terms of elementary reactions. Events specify transitions that are not elementary reactions. Here we consider two classes of events, namely *join* and *split*. The join event `[E1]`

```
[E1]   when(B1,B2::rate_parameter_E1) join(B)
```

is enabled when a box `B1` and a box `B2` are available. The left part in brackets of the event is called *condition*, while *join(B)* is called *verb*. `[E1]` removes boxes `B1` and `B2` and adds a box `B`. The duration of the transition is specified by the `rate_parameter_E1` value, as usual, the unique parameter of a negative exponential distribution. The split event `[E2]`

```
[E2]   when(B::rate_parameter_E2) split(B1,B2)
```

reverses event `[E1]` by removing box `B` and adding both `B1` and `B2`.

Events add flexibility to BlenX enabling the description of biological systems with different levels of detail in the same model. Consider again the enzymatic reaction of Fig. 1. We provide a detailed description of the transformations that lead to the release of the product P. Now suppose there is an unknown pathway that transform P into substrate S. This pathway is represented by a dashed arrow in Fig. 1. Events allow integrating in the detailed model of the enzymatic reaction the partial knowledge about a feedback mechanism that transforms the product into the substrate. In particular, a split event is enough:

```
[E3]   when(P::0.0078) split(S,Nil)
```

Event `[E3]` uses a tricky version of event `[E2]`. In fact, the second output box Nil is the BlenX representation of an empty system and therefore event `[E3]` transforms box **B** into box **S** with rate 0.0078. Note that the BlenX language offers a richer set of primitives w.r.t. the ones presented here; for a detailed description of the full language we refer the reader to [10].

The BetaWB framework lets to stochastically simulate the evolution of a BlenX program. Essentially, the BetaWB offers a run-time support that associates each possible transition of a BlenX program with a stochastic rate, i.e., a real number being the unique parameter of a negative exponential distribution describing the time required by the transition. The BetaWB has also a built-in simulation algorithm inspired by SSA but designed for optimizing simulation of BlenX programs. However, as the reader can image, the use of exponential distributions and SSA inspired algorithms collides with the specification of models with different levels of detail. For instance, event `[E3]` above represents complex and unknown transformations. Currently `[E3]` is managed by the simulator engine as a reaction following negative exponential distribution, but it is hard to see `[E3]` as *elementary*. For this reason, we move BlenX and the associated BetaWB framework toward the support

of generally distributed reaction times.

Since BlenX is rooted into process calculi theory, we look at process calculi for including general distributions support. General distributions are not *memoryless*, that is the probability distribution of a transition is not independent from the transitions already happened. Therefore, general distributions introduce the problem of tracking the time consumed by each transition, since this influences the probability distributions of the transitions that are going to be executed. The problem was faced by the process calculi community following three main approaches, namely, *counters*, *ST semantics*, and *enhanced operational semantics*.

The semantics of TIPP [14] uses counters to track how many times an action has not been selected to happen, and adjusts probability distribution accordingly. The approach taken in [5] integrates clocks into models and expresses clocks start and clocks termination events through the use of an ST semantic; in this way dependency between transitions can be considered. Finally, in [25] the reach labels of the enhanced operational semantics allow to derive the firing distributions of enabled transitions. We adopt this last approach mainly because enhanced operational semantics can be also used besides the scope of general distributions to retrieve interesting biological information as causality and locality [8]. The details about the enhanced semantics of BlenX and how general distributions are derived can be found in [24]. Here we only sketch the general ideas rephrasing the example above.

Suppose to have a system with two instances of product box $\mathbf{P}$, say $\mathbf{P}^1$ and $\mathbf{P}^2$. Also consider $i \in \{1, 2\}$. Thus, there are two transitions $\texttt{[T6]}^i$ from $\mathbf{P}^i$ to $\mathbf{S}^i$. SSA generates two tentative times t_i and the faster transition, say $\texttt{[T6]}^1$, is chosen and executed. Memoryless property of exponential distributions assures that the time of transition $\texttt{[T6]}^2$ does not depend on the time t_1 consumed by $\texttt{[T6]}^1$. In a general setting memoryless does not hold and the time distribution of $\texttt{[T6]}^2$ is influenced by the time consumed by $\texttt{[T6]}^1$. Enhanced operational semantics allows defining *causality*: a transition `[Ti]` *causes* a transition `[Tj]` if `[Tj]` cannot be executed before `[Ti]`. The notion of causality supports *enabling memory discipline* [19], namely the stochastic distribution of the time consumed by a transition `[T]` must be influenced by all the transitions fired from the states where `[T]` was first enabled. Therefore, the runtime support of BlenX is extended to support causality and enabling memory discipline. In particular, inspired by [11], we implemented an ad-hoc extension of the BlenX stochastic simulator engine based on classical optimized discrete event simulator solutions. From a user point of view, the extended version of BlenX allows to specify also general distributions and not only unique rate parameters. For instance, event `[E3]` can be extended as:

```
[E3]   when(P::0.0078) split(S,Nil);
[E3_1] when(P::gamma(2,64.1)) split(S,Nil);
[E3_2] when(P::hyperexp((0.12879,0.003),(0.87121,0.01))) split(S,Nil);
```

events `[E3_1]` and `[E3_2]` have the same qualitative behaviour of `[E3]`, but different time distributions. In particular `[E3_1]` has associated a Gamma distribution with shape 2 and scale 64.1, while `[E3_2]` is distributed as an Hyperexponential, i.e., the sum of two exponentials with rates 0.003 and 0.01, and weight 0.12879 and 0.87121,

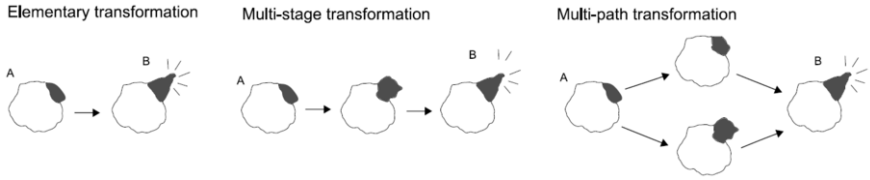


Fig. 2. Different protein transformation scenarios.

respectively. Playing with general distributions lets to test different hypothesis for, e.g., the pathway that leads to box **S** from box **P**. Next section builds on this example and gives some insight about the use of general distributions for modeling complex biological systems.

3 Experiments

In this section we introduce a few examples of biological systems that include abstract reactions. We present their modeling within the extended BlenX language and the results obtained via the extended stochastic simulator [3]. The purpose of this few case studies is twofold: they allow quantifying the impact of assumptions such as supposing all the reactions as elementary; and they provide the basis for further discussing the advantages offered by the inclusion of generally distributed reaction times into models.

3.1 Protein transformation

We start by considering an example of protein transformation. In the simplest case we can imagine to model it as an elementary reaction (see Fig. 2); the *effecting domain* (dark grey in Fig. 2) is modified through an elementary reaction following a negative exponential distribution. Now, suppose we are not sure whether the transformation is elementary; we can consider scenarios (see Fig. 2) in which the transformation follows a multi-stage process, for instance when multiple phosphorylations are necessary to make the site active, or a multi-path process, for instance when multiple enzymes can catalyze the reaction with different affinities.

In BlenX, the initial protein A can be simply modeled as a box **A** where the effecting domain is a binder with an identifier *Aid*:

For simplicity we set the internal process as nil but in general we can imagine a complex internal process. All the three kinds of transformations can be modeled using split events and since split event verbs refer to box definitions, then we also model the modified protein B as the box **B** above. The main difference w.r.t. to

[3] The extended simulator and the complete code of the examples can be obtained by emailing to betawb@cosbi.eu

the box **A** is that the binder modeling the effecting domain has another identifier. Notice that also in this case we put nil as internal process only for simplicity.

Now, suppose we model the elementary transformation with an event using a negative exponential distribution with rate $0.0078s^{-1}$, which means the expected time to the reaction is $128.2051s$:

```
when(A::0.0078) split(B,Nil);
```

Then, assuming we want to abstract from intermediate species, multi-stage and multi-path transformations can be modeled using respectively Erlang and Hyper-exponential distributions with parameters equal to the ones reported in Tab.1. In

Distribution	Parameters	Mean	Variance
Exponential	$\lambda = 0.0078$	128.2051	16436.554
Erlang	$k = 2$, $\lambda = 0.0156$	128.2051	8218.2774
Hyperexponential	$p_1 = 0.3$, $p_2 = 0.7$	128.2051	79755.80924
	$\lambda_1 = 0.0025$, $\lambda_2 = 0.085312$		

Table 1
The three distributions used to model the protein transformation.

BlenX, this transformations are modeled with split events that uses the corresponding general distributions:

```
when(A::gamma(2,64.1)) split(B,Nil);
when(A::hyperexp((0.12879,0.003),(0.87121,0.01))) split(B,Nil);
```

Note that we model the transformation with probability distributions that have the same expected value but different variances. The Erlang is expressed through the more general Gamma distribution. We run 10^3 stochastic simulations for each of the three models using the BlenXsimulator, assuming an initial population of A equal to 1000. In Fig. 3 we report the time courses of means and variances of B for the three transformations. These measures were estimated at a 95% confidence level, and the relative width of the confidence intervals (not shown for the sake of clarity) is in all cases within 10% of the estimate.

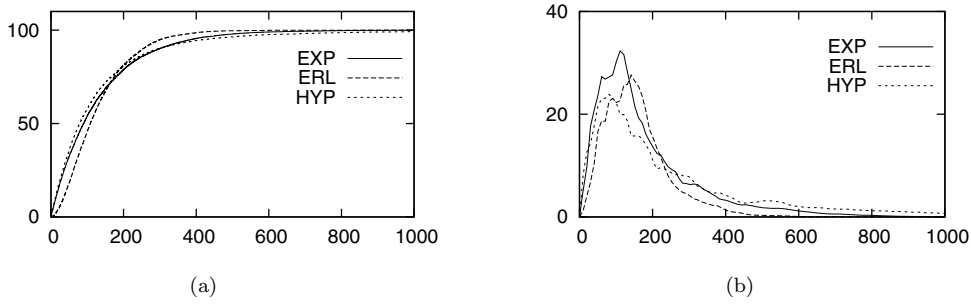


Fig. 3. (a) Time courses for the average number of molecules of B and (b) for its variance for the three transformations scenarios over 10^3 stochastic simulations.

From Fig. 3(a) we can observe that the abundance of species B is influenced by the type of distribution selected for the transformation. The major differences

among time courses are visible at the onset of the reaction, with the multi-stage (Erlang) transformation resulting in a slower accumulation of the reaction product and the multi-path (Hyperexponential) in a faster accumulation compared to the purely exponential one. A similar behaviour is shown by the variance time courses in Fig. 3(b), which also point out the effects of the different spread of the three types of distributions used in the models in the later stages of the reaction.

3.2 Enzymatic activation

Now, we use the previous model to show and discuss the effects of varying the type of stochastic transformation on the model of enzyme activation presented in Sect. 2. We concentrate on the reaction that transforms the product P into the substrate S (dotted arrow in Fig. 1). As before, we can assume this last transformation to be an elementary reaction or to be a multi-stage or a multi-path transformation. Proteins A and B of Sect. 3.1 corresponds to product P and substrate S, respectively. The rate of complexation, decomplexation, and interaction are specified by the following tuples:

$$(Sid, Eid, 0.5, 0.1, 0.01) \text{ and } (Pid, Eid, 0, inf, 0)$$

The reaction we are interested in is modeled using an event, which works on P and is different depending on the kind of transformation we want to consider:

```
when(P::0.0078) split(S,Nil);
when(P::hyperexp((0.12879,0.003),(0.87121,0.01))) split(S,Nil);
when(P::gamma(2,64.1)) split(S,Nil);
```

Note that for simplicity we use the same distributions as the previous example (see Tab.1).

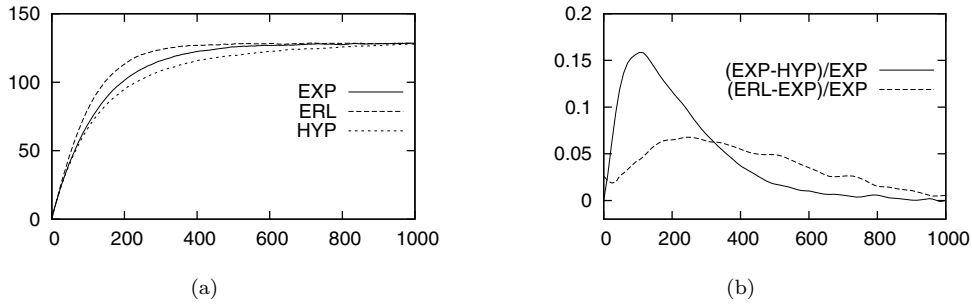


Fig. 4. Time courses of the average number of molecules of species P (a) for the three transformations scenarios over 10^3 stochastic simulations (b) and for their relative distances.

As before we run 10^3 stochastic simulations for each of the three models assuming an initial population of S equal to 1000 and the one of E equal to 100. In Fig. 4 we report the time course for the average value of molecules of P for the three transformations and the point-wise relative difference between the results obtained from the Erlang and the Exponential models of transformation, i.e., (ERL-EXP)/EXP, and between those of the Exponential and the Hyperexponential models, i.e. (EXP-HYP)/EXP. Whereas the three models reach in the long run the same steady-state

value for the average number of molecules of species P, significative differences can be observed in the transient behaviour in the three cases, as pointed out by Fig. 4(b). It is worthwhile noticing that differences in concentration of species as large as 10-15% may easily turn out in profound changes in the dynamics of a model, for instance when the molecules of P are regulating other processes in a non-linear fashion, as in the example below.

3.3 Gene regulation network

Here we concentrate on a model that describes a network with negative feedback through dimers in the control circuit for the λ repressor protein cI of phage λ in *E. coli* [6,7]. A variety of studies have been done on this specific λ repressor system (e.g. [3,16]). This ensures that all the concepts and ideas we discuss here are not purely abstract or theoretical, but can be interesting and important also w.r.t. modeling aspects of real biologically relevant systems.

In particular, we refer to one of the models presented in [6] from which we take all the quantitative parameters. In the BlenX model we implemented, each element composing the network is represented as a box; we have a box **D** representing the DNA string, a box **DR** representing the complex of the DNA with a DNA polymerase R, a box **M** representing the mRNA, the box **P** representing the protein P expressed, a box **P2** representing the protein P dimer and finally a box **Q** representing the complex of the dimer and the DNA, i.e., the DNA inhibition. All the chemical reactions reported in [6] can be translated simply in BlenX using events. We have the transcription, translation, and the decay of the mRNA and of the protein P which we describe using, respectively, events:

```
when(DR::0.0078) split(D,M);      //mRNA Transcription
when(M::0.043) split(M,P);        //Protein P Translation
when(M::0.0039) delete(1);        //mRNA Degradation
when(P::0.0007) delete(1);        //Protein P Degradation
```

where all the rates are expressed as s^{-1} (s = seconds). Here we use a *delete* event [10], which allows a deletion from the system of a certain number of specific boxes; the event condition specifies the box species and the distribution it follows, while the verb contains the number of boxes subject to deletion. Then we have all the complexation and decomplexation reactions. We have the formation and breaking of complex **DR**:

```
when(D::1.14) split(DR,Nil); //DNA-DNA Polymerase R complex formation
when(DR::0.3) split(D,Nil);  //DNA-DNA Polymerase R complex breaking
```

where we abstract from the presence of the DNA polymerase R, i.e., we assume population of DNA polymerase always constant. Then we have the events representing the formation and breaking of the protein P dimer:

```
when(P,P::0.025) join(P2);        //Protein P dimer P2 formation
when(P2::0.5) split(P,P);         //Protein P dimer P2 breaking
```

and for creating and breaking the complex between the dimer and the DNA:

```
when(D,P2::0.012) join(Q);    //Dimer P2 and DNA complex formation
when(Q::0.9) split(D,P2);     //DNA and P2 dimer complex breaking
```

Note that all the complexation rates are expressed as $s^{-1}(nM)^{-1}$ (nM = nanomoles), while the decomplexation rates are expressed as s^{-1}.

We focus our attention on the event representing the transcription. As already mentioned, although gene transcription is a complex process encompassing a multitude of reactions, it is usually modeled as an elementary reaction following a negative exponential distribution. Anyway, since detailed models of transcription [27] indicate that mRNA production can be quite different from a pure Poisson process, we want to quantify the affect of assuming the transcription to be a non elementary reaction. Also in this case we refer to two variants based on Erlang and Hyperexponential distributions with parameters equal to the one reported in Tab.1. We run 10^3 stochastic simulations for each of the three models. All simulations start in an initial state where the only entity is a single molecule of DNA, represented in the models by the presence of one box **D**.

In Fig. 5 we report the time course of the average number of molecules of the protein P for the three different transcription processes. In all the three cases, the amount of protein P initially increases and then finds an equilibrium that is determined by the negative feedback exerted on the DNA by its dimerized form. The three time courses shown in Fig. 5 point out that the transient behaviours as well as the equilibrium values are quite different for the three models. Thus, the choice of the distribution reveals to have an important effect on the predictions that can be obtained from a stochastic model. Again, it is important to stress that the three distributions considered for modeling the transcription process have the same expected value and that only the variance is different.

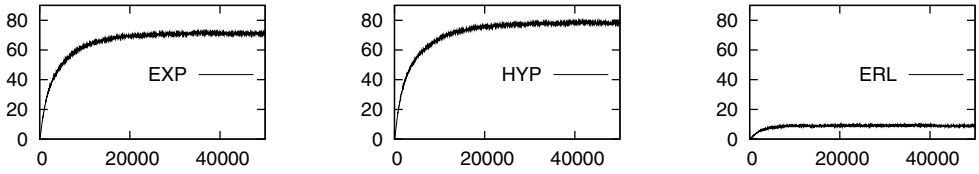


Fig. 5. Time courses of the average number of molecule of protein P for the three transcription processes over 10^3 stochastic simulations.

4 Exploiting non-Markovian Modeling Capabilities

The simulation results presented in the previous section for the three considered examples of biochemical systems suggest that the assumption of elementariness of a reaction should be carefully considered. Without any willingness to stigmatize current approaches to stochastic modeling, we believe it is however fair claiming that many models of biological systems are introducing approximations without making them explicit. Thus, the occurrence times of complex biological processes are commonly being modeled with simple negative exponential distributions without

discussing the impacts of this choice.

It is an easy guess to conjecture that complex non-linear systems such as the biochemical ones can be sensitive to variances of event occurrence times: we provided some evidence for it with three simple examples. Unfortunately, assuming a negative exponential distribution of every reaction occurrence time limits the possibility of defining accurate models of non-elementary transformations, as there is no lever to match statistical properties of phenomena beyond their expected occurrence times. An important implication of this limitation is that, to make models to fit wet-lab data, the rates of reactions may need to be changed to unrealistic values. Consider for instance the third example provided in the previous section. In the gene regulatory network model that uses general distributions, to fit an observed steady-state level of protein expression, one can tune the variance of a transcription distribution without changing its expected value. However, if modelers assume elementariness of every reaction, the only possibility left to match experimental data is to change their rate constants.

In this respect, the extension of BlenX with general distributions of reaction times can help in alleviating this limitation. When only mesoscopic or macroscopic biological phenomena are experimentally observable, general distribution provide the support for matching measured quantitative information and representing them into models in a more accurate way. As technological progress in experimental biology reduces the spurious uncertainties introduced by protocols and machinery, it becomes more and more possible to reliably determine the inherent variability of biological phenomena. This improved characterization of biological transformations provides a key input for selecting which distribution can adequately represent the specific aspects of the considered system.

Moreover, having the possibility of including generally distributed reaction times opens new possibilities for better managing the level of abstraction in models of biological systems. Abstraction is a powerful means to limit the size of models, allowing modelers to focus only on key parts of systems. It is important here to notice that stochastic models of biological systems easily stretch the capabilities of modeling and simulation tools, mostly because of the abundance of entities. Thousands of copies of proteins exist in the small volume of a cell nucleus, which inevitably turns out in millions of reaction events occurring in very short time-scales. Thus, even in those cases when all the molecular details of the biological transformations are known, it would be desirable to have the freedom of modeling them in an aggregated way. Managing the level of abstraction requires the ability to accurately represent the aggregated behaviour of subsystems, so that parts and motifs occurring in biochemical networks can be modeled as single reaction events accounting for a multiplicity of elementary transformations. Obviously, we expect the stochastic properties of such aggregated behaviours to span over different types of distributions, and modeling such properties calls for increased modeling capabilities such as the ones we coded into BlenX.

Besides the intrinsic importance and relevance of being able to model and simulate classes of non-Markovian biological processes, here we also want to explore

	exp 5	exp 10	exp 15	exp 20	**Erlang**
1000	0.112s	0.118s	0.182s	0.303s	**0.101s**
2000	0.178s	0.304s	0.407s	0.505s	**0.156s**
4000	0.221s	0.444s	0.706s	1.260s	**0.364s**
6000	0.323s	0.627s	1.100s	1.525s	**0.698s**
8000	0.413s	0.794s	1.398s	1.997s	**1.169s**
10000	0.489s	1.702s	1.741s	2.496s	**1.781s**

Table 2
The first four columns represent the simulation times for multi-stage transformations of different length for different initial populations of A. Since simulation times for the Erlang abstractions of different length are really similar, we consider only representative values.

how general distributions can be used, within certain limits and for specific classes of systems, to increase the computational efficiency of a stochastic simulation. A similar study can be found in [11]. We consider the first example and in particular the multi-stage protein transformation. We want to compare models in which the transformation is implemented using chains of elementary reactions with models in which it is implemented using an Erlang distribution. Notice that the abstraction is correct because an Erlang distribution with shape n and scale λ is the sum of n exponentially distributed random variables with parameters λ. We run stochastic simulations considering different populations of protein A and transformation chains of different lengths; we impose a limit time enough large to permit the consumption of all the initial protein A. In Tab. 2 we report the results we obtained. The table shows clearly that within certain limits the Erlang abstraction allows to speed up the simulation time. In particular, the longer the multi-stage transformation, the more we can speed up simulations for increasing initial populations of protein A. This fact can be particularly useful if we want to increase the efficiency of stochastic simulations in classes of models that presents chains of sequential elementary monomolecular reactions where all the intermediate species are inactive. If we know that the populations of the species subject to these transformations live in the right range, the Erlang abstraction allows us to speed up the simulation time.

An interesting modeling facet that surely deserves to be subject of future work is the exact representation of the competition among reactions. In fact, this is quite a delicate area of modeling, whose importance becomes evident when dealing with general distributions. Consider for example the transcription relation in Example 3.3. Obviously, the first stage of the transcription process is in competition with the polymerase decomplexation reaction. However, if we model the whole transcription process as a single elementary reaction, we cannot limit the scope of this competition to the first stage only. Thus, it would be like saying that the polymerase could decomplex from DNA and stop gene transcription at any time during the process, which is not realistic. Indeed, this extended competition is what is actually modeled for the Example 3.3 in the literature [7] as well as in the model variants presented in this paper that used generally distributed transcription times.

This shortcoming of abstract reaction modeling is encountered whatever is the distribution used to model the transcription time. However, when we explicitly consider the multi-stage nature of the transcription process, for instance in the form of an Erlang distribution, we have a degree of freedom in playing with the

preemption at the various stages of the process. Thus, we could specify that only the first stage of the Erlang can be preempted because of the competition with the polymerase unbinding reaction. It is clear that the opportunities offered by this refined modeling only become available when considering the details of the staged nature of abstract reactions, and thus only in a framework that allows going beyond the pure elementariness of biochemical reactions.

5 Conclusions

The SSA algorithm is the de-facto standard for simulating the dynamic evolution of systems of stochastic reactions. SSA assumes *elementary* reactions and therefore the time to the next reaction is a random variable following negative exponential distribution. However, it is difficult to describe complex biological systems only in terms of elementary reactions. Quite often there is an incomplete knowledge of the full set of reactions comprising the systems under inspection. Thus, researchers have to introduce *abstract* reactions in their models, without considering the impact of using SSA like algorithms with abstract reactions. This calls for an investigation of the dichotomy of the *qualitative* abstract reactions and the *quantitative* elementariness assumed in the simulation algorithms.

We started by extending the modeling language BlenX and the associated computational framework BetaWB with general distributions. BlenX allows including in the same model different levels of (qualitative) abstraction and so it is well suited to be tested in an environment with general distributions. Here we described the general ideas underlying BlenX and its extension in a tutorial fashion, pointing the interested reader to [10]. Then, we described three examples that present an increasing complexity. The examples resulted sensitive to variances of reaction occurrence times as non-linearity and complexity of the system increase. Moreover, the extension presented can help in providing better matching between wet-lab experiments and in-silico results.

We plan to investigate general distributions as a quantitative tool for managing the level of abstraction of biological models, resulting in a powerful means to limit the size of models. Moreover, the models presented calls for a deep study of exact representations of the competition among biochemical reactions.

References

[1] Ahmad, K. and S. Henikoff, *Modulation of a Transcription Factor Counteracts Heterochromatic Gene Silencing in Drosophila*, Cell **104** (2001), pp. 839–847.

[2] Alberts, B., A. Johnson, J. Lewis, M. Raff, K. Roberts and P. Walter, "Molecular biology of the cell (IV ed.)." Garland science, 2002.

[3] Arkin, A., J. Ross and H. McAdams, *Stochastic Kinetic Analysis of Developmental Pathway Bifurcation in Phage λ-Infected* Escherichia coli *Cells*, Genetics **149** (1998), p. 1633.

[4] Bohnenkamp, H. C. and B. R. Haverkort, *Stochastic event structures for the decomposition of stochastic process algebra models*, in: J. Hillston and M. Silva, editors, *PAPM'99, Proceedings of the 7th International Workshop on Process Algebra and Performance Modelling* (1999), pp. 25–39.

[5] Bravetti, M., M. Bernardo and R. Gorrieri, *Towards Performance Evaluation with General Distributions in Process Algebras*, in: *CONCUR*, 1998, pp. 405–422.

[6] Bundschuh, R., F. Hayot and C. Jayaprakash, *Fluctuations and slow variables in genetic networks*, Biophys J **84** (2003), p. 2003.

[7] Bundschuh, R., F. Hayot and C. Jayaprakash, *The role of dymerization in noise reduction of simple genetic networks*, Journal of Theoretical Biology **220** (2003), pp. 261–269.

[8] Curti, M., P. Degano, C. Priami and C. Baldari, *Modelling biochemical pathways through enhanced π-calculus*, TCS **325** (2004), p. 111.

[9] Degano, P., D. Prandi, C. Priami and P. Quaglia, *Beta-binders for biological quantitative experiments*, Electr. Notes Theor. Comput. Sci **164** (2006), pp. 101–117.

[10] Dematté, L., C. Priami and A. Romanel, *The BlenX Language: A Tutorial*, in: *SFM 2008*, LNCS (2008), pp. 313–365.

[11] Gibson, M. and J. Bruck, *Efficient Exact Stochastic Simulation of Chemical Systems with Many Species and Many Channels*, Journal of Physical Chemistry A **104** (2000), pp. 1876–1889.

[12] Gillespie, D., *A general method for numerically simulating the stochastic time evolution of coupled chemical species*, Journal of Computational Physic **22** (1976), pp. 403–434.

[13] Gillespie, D., *Exact stochastic simulation of coupled chemical reactions*, Journal of Physical Chemistry **81** (1977), pp. 2340–2361.

[14] Götz, N., U. Herzog and M. Rettelbach, *TIPP - Introduction and Application to Protocol Performance Analysis*, in: *Formale Methoden für verteilte Systeme, GI/ITG-Fachgespräch, Magdeburg, 10.-11. Juni 1992*, 1992, pp. 105–125.

[15] Harrison, P. G. and B. Strulo, *Process algebra for discrete event simulation*, Qualitative Methods in Parallel Systems (1995).

[16] Hasty, J., J. Pradines, M. Dolnik and J. J. Collins, *Noise-based switches and amplifiers for gene expression*, PNAS **97** (2000), p. 2075.

[17] Kitano, H., "Foundations of Systems Biology," MIT Press, 2002.

[18] L.Dematté, C.Priami and A.Romanel, *The Beta Workbench: a computational tool to study the dynamics of biological systems*, Brief. Bioinform. **9** (2008), pp. 437–449.

[19] Marsan, M., G. Balbo, A. Bobbio, G. Chiola, G. Conte and A. Cumani, *The Effect of Execution Policies on the Semantics and Analysis of Stochastic Petri Nets*, IEEE TOSE **15** (1989), pp. 832–846.

[20] McQuarrie, D., *Stochastic approach to chemical kinetics*, Journal of Applied Probability **4** (1967), p. 413.

[21] Milner, R., "Communicating and mobile systems: the π-calculus," Cambridge Universtity Press, 1999.

[22] Plotkin., G. D., *A Structural Approach to Operational Semantics.*, Technical Report DAIMI-FN-19, Computer Science Department, Aarhus University (1981).

[23] Prandi, D., C. Priami and P. Quaglia, *Communicating by compatibility*, JLAP **75** (2008), p. 167.

[24] Prandi, D., C. Priami and A. Romanel, *Simulation of non-Markovian Processes in BlenX*, Technical Report TR-11-2008, CoSBi (2008).

[25] Priami, C., *Language-based performance prediction for distributed and mobile systems*, Information and Computation **175** (2002).

[26] Priami, C. and P. Quaglia, *Beta binders for biological interactions*, in: *CMSB*, LNCS **3082** (2004), p. 20.

[27] Voliotis, M., N. Cohen, C. Molina-Paris and T. Liverpool, *Fluctuations, Pauses, and Backtracking in DNA Transcription*, Biophysical Journal **94** (2008), pp. 334–348.

[28] Wijgerde, M., F. Grosveld and P. Fraser, *Transcription complex stability and chromatin dynamics in vivo*, Nature **377** (1995), pp. 209–213.