A

Hybrid Simulations of Heterogeneous Biochemical Models in SBML

HUI JU K. CHIANG, Graduate Institute of Electronics Engineering, National Taiwan University
FRANÇOIS FAGES, Inria Paris-Rocquencourt, France
JIE HONG R. JIANG, Graduate Institute of Electronics Engineering, National Taiwan University
SYLVAIN SOLIMAN, Inria Paris-Rocquencourt, France

Models of biochemical systems presented as a set of formal reaction rules can be interpreted in different formalisms, most notably as either deterministic Ordinary Differential Equations, stochastic continuous-time Markov Chains, Petri nets or Boolean transition systems. While the formal composition of reaction systems can be syntactically defined as the (multiset) union of the reactions, the composition and simulation of models in different formalisms remains a largely open issue. In this article, we show that the combination of reaction rules and events, as already present in SBML, can be used in a non-standard way to define stochastic and boolean simulators and give meaning to the hybrid composition and simulation of heterogeneous models of biochemical processes. In particular, we show how two SBML reaction models can be composed into one hybrid continuous-stochastic SBML model through a high-level interface for composing reaction models and specifying their interpretation. Furthermore, we describe dynamic strategies for automatically partitioning reactions with stochastic or continuous interpretations according to dynamic criteria. The performances are then compared to static partitioning. The proposed approach is illustrated and evaluated on several examples, including the reconstructions of the hybrid model of the mammalian cell cycle regulation of Singhania et al. as the composition of a Boolean model of cell cycle phase transitions with a continuous model of cyclin activation, the hybrid stochastic-continuous models of bacteriophage T7 infection of Alfonsi et al., and the bacteriophage λ model of Goutsias, showing the gain in both accuracy and simulation time of the dynamic partitioning strategy.

Categories and Subject Descriptors: I.6.5 [**Model Development**]: Modeling methodologies

General Terms: Algorithms, Performance

Additional Key Words and Phrases: hybrid simulation, stochastic simulation, computational systems biology, synthetic biology

ACM Reference Format:
Hui-Ju K. Chiang, François Fages, Jie-Hong R. Jiang, and Sylvain Soliman, 2014. Hybrid Simulations of Heterogeneous Biochemical Models in SBML. *ACM Trans. Model. Comput. Simul.* V, N, Article A (January YYYY), 22 pages.
DOI: http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

Systems biology aims at elucidating the high-level functions of the cell from their biochemical basis at the molecular level [Ideker et al. 2001]. A lot of work has been done for collecting genomic and post-genomic data, making them available in databases [Ashburner et al. 2000; Kanehisa and Goto 2000], and organizing the knowl-

This work has been supported by the French OSEO Biointelligence and ANR BioTempo projects, and by the European Eranet Sysbio C5Sys project.
Author's addresses: H.-J. Chiang and J.-H. Jiang, Graduate Institute of Electronics Engineering, National Taiwan University; F. Fages and S. Soliman, Inria Paris-Rocquencourt, France.

DOI: http://dx.doi.org/10.1145/0000000.0000000

edge on pathways and interaction networks into models of cell metabolism, signaling, cell cycle, apoptosis, etc. now published in model repositories (e.g. http://biomodels.net/). In particular, the Systems Biology Markup Language (SBML) [Hucka et al. 2003] provides a common exchange format for biochemical *reaction systems* and is nowadays supported by a majority of modeling tools.

According to the knowledge available on the system and to the nature of the queries that will be asked to the model, e.g. qualitative or quantitative predictions, these rule-based reaction systems can be interpreted (and simulated) under different semantics as either:

— ordinary differential equations (continuous semantics),
— continuous-time Markov chains (stochastic semantics),
— Petri nets (discrete semantics),
— Boolean transition systems (Boolean semantics), and many variants.

These different interpretations can be related by either approximation [Gillespie 1977; Gillespie 2001; Gillespie 2009] or abstraction [Fages and Soliman 2008a] relationships. Many modeling tools support several of them but provide no support for the combination of heterogeneous models. However, in the perspective of applying engineering methods to the analysis and control of biological systems, the issue of building complex models by composition of elementary models is a central one. While reaction systems can be formally composed by the multiset union of reaction rules, and interpreted by one common semantics, there is also a need to compose models with different semantics, as will be shown here by some examples from the literature. What we call a *hybrid model* is a model obtained by composition of models with heterogeneous semantics (continuous, stochastic, Boolean, etc.), and *hybrid simulation* is the topic of simulating such hybrid models.

In [Pahle 2009], the author observes that "A very promising direction is the development of hybrid methods because they directly deal with the important problem of stiffness, which is often present in biochemical models. [...] There exist already a few software tools, which allow for hybrid simulation, [...] and this number is expected to grow in the future." In this paper, we propose a general approach to progress in that direction by showing that the combination of reaction rules and events, as already present in SBML, can be used in a non-standard way to give meaning to the hybrid composition and simulation of heterogeneous reaction models. In particular, we show how hybrid continuous-Boolean models and hybrid continuous-stochastic models can be assembled and simulated, through the specification of a *high-level interface for composing heterogeneous models*, and producing as output a hybrid model in standard SBML format, which can thus be executed with any SBML-compatible simulator.

Our high-level interface, prototyped in the modeling environment BIOCHAM [Calzone et al. 2006; Fages and Soliman 2008b], takes two models with synchronization information as inputs, and produces one SBML model with reactions and events as output. For hybrid continuous-Boolean composition, it transforms a Boolean state transition model to events with extra triggers which express the links with the continuous variables and the parameters of the continuous reaction model. For hybrid continuous-stochastic composition, the interface described in this paper transforms stochastic reactions to a set of events, which implements Gillespie's direct method for stochastic simulation, and can be freely combined with the simulation of continuous reactions. Furthermore, our framework supports the specification of *dynamic strategies* which automatically choose between the stochastic and continuous interpretations of the reactions according to particle counts, reaction propensities, or more specific model-dependent criteria. We show that without the need to conduct time-consuming fully stochastic simulations beforehand to obtain the scale information of particle count

and propensity for all reactions, dynamic partitioning results in higher accuracy and shorter simulation time than static partitioning – as static partitioning cannot adapt to the possibly substantial scale variations over time, which can render the initial partition inadequate.

This approach is illustrated and evaluated with several examples including the reconstructions of the hybrid model of the mammalian cell cycle regulation of Singhania et al. [Singhania et al. 2011] as the composition of a Boolean model of cell cycle phase transitions and a continuous model of cyclin activation, plus a hybrid Boolean-continuous-stochastic version of this model with dynamic partitioning strategy, and of the hybrid stochastic-continuous model of bacteriophage T7 infection of Alfonsi et al. [Alfonsi et al. 2005], and of bacteriophage λ of Goutsias [Henzinger et al. 2010], showing the gain in both accuracy and simulation time of the dynamic strategy.

Since XML, and hence SBML, are not easy to read by humans, in this article we use mathematical notation and BIOCHAM code. The BIOCHAM and SBML files of the examples of this paper are available at: http://lifeware.inria.fr/supplementary_material/TOMACS/. The BIOCHAM files can be executed via the BIOCHAM web application http://lifeware.inria.fr/biocham/online without any installation.

Related Work

Hybrid simulation is a classical topic in physics, e.g. for numerically solving equations describing stochastic systems using ordinary differential equations whenever possible in place of stochastic equations, in order to speed-up simulations [Alfonsi et al. 2005; Salis and Kaznessis 2005]. It is also ubiquitous in computer science for programming and verifying hybrid systems which have both discrete and continuous dynamics [Alur et al. 2001; Henzinger et al. 1997]. Hybrid modeling is also used in Systems Biology for reducing the complexity of many modeling task, e.g. [Matsuno et al. 2000; Alur et al. 2001; Ghosh and Tomlin 2001; Bockmayr and Courtois 2002; Kiehl et al. 2004; Ahmad et al. 2006; Singhania et al. 2011; Berestovsky et al. 2013], for speeding up stochastic simulations [Salis et al. 2006; Hellander and Lotstedt 2007; Henzinger et al. 2010; Golightly and Wilkinson 2011], and achieving whole cell simulation [Karr et al. 2012]. A review of the different approximate stochastic and hybrid methods used in Systems Biology can be found in [Pahle 2009].

Due to the structure of SBML, which mostly relies on explicit and global reactions and events, the composable modelling at the core of hybrid process algebra, e.g. [Galpin et al. 2008; Akman et al. 2010] is out of reach of the presented work. On the other hand, we show that SBML can express various form of hybrid systems. Indeed, a set of SBML events and continuous reactions can also be visualized as a hybrid automaton [Henzinger 1996] in which there is a state with a particular ODE for each combination of the trigger values, and there is a transition from one state to another state when at least one trigger changes value from false to true in the source state. Stochastic hybrid automata [Hahn et al. 2013] can be similarly simulated in SBML with a random number generator coded by events. Since our focus in this paper is on SBML, we are mainly focussed on simulations and on the reproduction of simulation results, as examplified for instance by the notion of "curated" model in the BIOMODELS project at http://biomodels.net/. The use of existing verification tools for hybrid systems is thus beyond the scope of this paper.

Another line of work also exists on the extension of Boolean models with continuous time delays. René Thomas's discrete modeling of gene regulatory networks (GRN) [Thomas 1973] is a well known approach to study the logical dynamics of a set of interacting genes. It deals with a graph of positive and negative influences between genes and logical functions that determine the possible trajectories in the state space. Those parameters are a priori unknown, but they may generally be deduced from a

large set of biologically observed behaviors in various conditions. Besides, it neglects the time delays for a gene to pass from one level of expression to another one. In [Ahmad et al. 2006], it is shown that one can account for time delays depending on the expression levels of genes in a GRN, while preserving powerful enough computer-aided reasoning capabilities. The characteristic of this approach is that, among possible execution trajectories in the model, one can automatically find out both viability cycles and absorption in capture basins. Model-checking techniques developed for hybrid systems are used for this purpose [Ahmad et al. 2008]. The authors describe a Hybrid model for the mucus production in the bacterium *Pseudomonas aeruginosa* and show that they are able to discriminate between various possible dynamical behavior [Ahmad et al. 2006; Ahmad et al. 2008]. Such a model can be presented and compiled in a set of reaction rules with events as described in this article.

Time constraints provide another means to refine Boolean or discrete models which are often too coarse to be useful. In [Maler and Batt 2008], the authors present a new technique for over-approximating (in the sense of timed trace inclusion) continuous dynamical systems by timed automata for the purpose of efficiently checking timed (as well as untimed) properties. The essence of this technique is the partition of the state space into cubes and the allocation of a clock for each dimension. This is in contrast with other approaches which use only one clock. This idea is a specific case of rectangular hybrid automata. This makes it possible to get better approximations of the behavior. The timed automata produced by these techniques can be similarly composed in our tool for simulation.

Organization of the article

In the next Section, we review reaction rules and events in SBML and discuss their semantics. Then in Section 3, we focus on hybrid continuous-stochastic models. We show how Gillespie's stochastic simulation algorithm can be implemented in SBML with events, and how this can be used to compose continuous and stochastic reaction models in SBML, using a preprocessor which produces as output an SBML model with events for emulating a hybrid simulator. Then in Section 4, we focus on dynamic partitioning criteria for automating the choice between the continuous and the stochastic interpretations of the reactions. We describe our implementation of the dynamic partioning strategy and evaluate its performance on simple examples from the literature, both in terms of computation time and accuracy of the hybrid simulation. Section 5 shows how our SBML preprocessing approach can also be used to compose continuous reaction models with Boolean models. This is illustrated with the hybrid cell cycle model of Singhania et al. [Singhania et al. 2011], and generalized to a hybrid Boolean-continous-stochastic model using dynamic partitioning. Finally, we conclude on the merits of our non-intrusive approach for hybrid simulation based on a non-standard use of SBML events.

2. SBML REACTION RULES AND EVENTS

2.1. Reaction Rules and Kinetics

In SBML [Hucka et al. 2003], a reaction rule is composed of a reaction rate, a left-hand side and a right-hand side of molecular species given with stoichiometric coefficients. In this article, an SBML reaction i is written in mathematical notation:

$$\sum_j l_{ij} \times S_j \xrightarrow{v_i} \sum r_{ij} \times S_j$$

or in the code in BIOCHAM syntax:

$$v_i \texttt{ for } \sum_j l_{ij} * S_j => \sum r_{ij} * S_j$$

where the S_j are the species of the model, l_{ij} and r_{ij} are stoichiometric coefficients, and v_i is any mathematical function (given in a subset of MathML notation in SBML, and in BIOCHAM with the abbreviation MA for mass action law kinetics) of the species concentrations and parameters of the system, which defines the rate of reaction i. A reaction model is a finite set of reactions.

According to the data available on the system and to the nature of the queries that will be asked to the model, e.g. qualitative or quantitative predictions, a reaction model can be interpreted under different semantics: continuous, stochastic, discrete or boolean.

The stochastic semantics, detailed in Section 3.1 associates to a reaction model the Continuous-Time Markov Chain (CTMC) where the states are defined by molecule numbers, and the transitions by the reactions, with their reaction rates as propensities giving the transition probabilities after normalization. This CTMC realizes the solution of the Chemical Master Equation [Gillespie 1977].

The continuous semantics associates to a reaction model the following Ordinary Differential Equation (ODE) system:

$$\frac{d[S_j]}{dt} = \sum_i (r_{ij} - l_{ij}) \times v_i$$

which describes the time evolution of molecular species concentrations according to the reaction rates. The continuous semantics approximates the mean behavior of the CTMC for large numbers of molecules. The continuous semantics usually leads to numerical integration, whereas the stochastic semantics is either used for exact or approximate simulation, or for stochastic model checking (see for instance [Kwiatkowska et al. 2008]).

The discrete semantics of a reaction model is a Petri net [Gilbert et al. 2007] which forgets about the reaction rates v_i but keeps the stoichiometric information. The Petri net semantics can be seen as an abstraction of the stochastic semantics.

The Boolean semantics forgets about precise stoichiometry and keeps only information about whether or not a species is active. It can be defined as an abstraction of the previous discrete semantics, provided the combinatorics of all possible consumptions is kept [Fages and Soliman 2008a]. The Boolean semantics of large networks is usually used for proving reachability properties using symbolic model checkers [Chabrier and Fages 2003] instead of doing simulations.

2.2. Events

SBML models can also be described with *events*. An SBML event is basically twofold: it is built by a *trigger*, determining when it fires, and by an *action*, i.e., its influence on the current state (parameters, concentrations), in the form of a list of assignments. In this article, we will write an event in BIOCHAM syntax as follows:

$$event(trigger,\ [s_1, \ldots, s_n],\ [f_1, \ldots, f_n])$$

where the s_i indicate the variables that are modified by the event, and the f_i are mathematical functions of the state that give the new value to s_i.

There are many possible semantics for events but the basic idea is that an event fires when its trigger changes from *false* to *true*. This induces however several issues:

— what happens at the start of the simulation?
— how to find the precise time when a trigger becomes true?
— what happens if several events are enabled simultaneously?

The first point is easy to settle, deciding whether events that are true at time 0 should fire or not is an arbitrary choice which does not impact the expressive power of the formalism, since the initial state can be adapted to both conventions. The simplest choice is to avoid the firing of events at the initial point of the simulation, they will only get triggered when going from false to true, and to reflect events that were meant to be triggered at the start by modifying accordingly the initial state.

The second point has been solved in practical tools for a long time: since numerical integration of ODEs goes by steps, one detects changes in triggers only in the interval of a simulation step. If some triggers become true, one can thus go back in time until one finds—with a given precision—the first time point where the first trigger becomes true. Note however that if arbitrarily complex conditions appear in the events, a numerical integrator unaware of the events can hide inside a single step that a trigger went from false to true and back to false again. Therefore, a cautious implementation is necessary, and fixed step size integration methods may be recommended to use in presence of events, instead of more efficient adaptive step size methods.

The third point is again a question with multiple possible answers. Generally, the set of events that are enabled simultaneously at a given time will all be fired, whatever the actions of the events are, but what if several events modify the same variable? It is possible to assume a *synchronous* semantics, where the simultaneous events execute their actions in parallel, but then one must forbid events with conflicting actions, i.e., events that would modify in different ways the same variable at the same time point. A more common choice is an *asynchronous* semantics, that will fire all the events enabled at a given time one after the other. Conflicts in actions are then solved by the ordering of events, which can be either random, i.e. non-deterministic, or given by the modeller, e.g. by the order of writing (BIOCHAM choice) or by priorities (SBML choice). However, if some actions invalidate the trigger of other enabled events, these events should be disabled in a purely asynchronous semantics.

The SBML Level 3 choice[1] is to keep a very flexible semantics, with semi-asynchronous events, which can use either the values at the time they were enabled, or the *current values* at the time they are actually executed, after the execution of the simultaneous events with higher priority, and which can specify the *permanence* of an event in order to define if it should be fired even if its trigger has been invalidated by previous events firing at the same time.

In BIOCHAM, there are no priorities, and the events that are enabled simultaneously are executed in the order of their writing using current values. An event with n assignments of f_i to s_i is therefore equivalent to the sequence of n events with the same trigger for each assignment f_i to s_i. The semantics of events implemented in BIOCHAM can thus be defined in SBML Level 3 using the current value and permanence options and priorities corresponding to the order of writing.

It is worth noting that in SBML Level 3, thanks to the Distributions package, random variables can be represented. This would be useful in the following sections to implement stochastic semantics with SBML events. However, since the SBML Test Suite Database and the page of the Distributions package show that there is apparently no software currently able to cope both with prioritized events and random variables, we have implemented a simple linear pseudo-random number generator (PRNG) using

[1]The Versions and Releases of the SBML Level 3 Core specification and officially-supported Level 3 package specifications are available at: http://sbml.org/Documents/Specifications/SBML_Level_3.

SBML events. The files generated by our hybridization interfaces can therefore be run with any SBML Level 3 core compatible simulator.

3. HYBRID CONTINUOUS-STOCHASTIC MODELS

Chemical reactions, originated from random collisions of particles, are discrete and stochastic in nature. Although there is no way to predict the exact state of a chemical system at a specific time point, its *statistical* behavior can be effectively calculated from known probabilistic properties, as done by Gillespie's stochastic simulation algorithm (SSA) [Gillespie 1976], to be detailed in Section 3.1. The SSA simulation can be especially slow if one or more of the reactions have fast reaction rates (or high event occurrences) because the next reaction time will be very short due to the high probability of firing (one of the) fast reactions.

Despite the fact that all reactions are innately stochastic, those with large reactant counts and high reaction rates can be accurately approximated in terms of deterministic behavior expressed by ODEs. By incorporating both continuous and stochastic semantics into one simulator, an optimal balance between simulation runtime and accuracy can be achieved. This potentially lifts the scalability of simulating large biological systems. In Section 3.2, we provide an event-based view on the SSA, that serves as basis to a hybrid continuous-stochastic simulator built upon an ODE simulator with events.

3.1. Gillespie's Stochastic Simulation Algorithm

A reaction model with kinetic expressions can be interpreted under the stochastic semantics as a continuous-time Markov chain (CTMC). A CTMC can be simulated with a stochastic simulation algorithm (SSA), for example, *Gillespie's direct method* [Gillespie 1976]. Rather than solving all possible trajectories' probabilities as in the case of Master equations, the algorithm generates statistically correct trajectories.

Gillespie's direct method first calculates *when* the next reaction will occur, then decides *which* reaction should occur with the help of a random number generator. The probability that a certain reaction i will be the next one is determined by the propensities α of the reactions: $\alpha_i = (\#\text{combinations of reactants}) \cdot k_i$ where k_i is the rate coefficient of reaction i. The algorithm repeats the following steps.

(1) Calculate *how long from now* ($\triangle t$) the next reaction will occur as a Poisson event.

$$\triangle t = \frac{-1}{\sum_j \alpha_j} \cdot \log(ran_1),$$

where ran_1 is a random number within range $[0, 1]$ and the α_j are propensities at the current state.

(2) Choose which reaction will occur according to the probability distribution of reactions. This is done by generating a random number ran_2 within range $[0, 1]$, and letting the reaction i be chosen for

$$\frac{\sum_{k=1}^{i-1} \alpha_k}{\sum_j \alpha_j} < ran_2 \leqslant \frac{\sum_{k=1}^{i} \alpha_k}{\sum_j \alpha_j}.$$

(3) Update the numbers of molecules to reflect the execution of reaction i, and set current time to $t = t + \triangle t$.

3.2. Event-based Implementation of the Stochastic Simulation Algorithm

By considering every firing of a chemical reaction as one firing of an event, the *event* semantics of Section 2 enables a direct embedding of stochastic reactions into an intrinsically continuous framework without additional implementation of a separate stochas-

tic simulation algorithm. Under this framework, *time* is the only unifying variable to keep track of current state at each instant. This event-based approach permits the simple integration of ODE and stochastic simulation as will be elaborated in Section 3.3.

Notice that, in the SSA of Section 3.1, *when* the next reaction will occur is independent of *which* reaction will occur, and also that only one reaction is chosen each time. These facts make it possible for *the complete set of stochastic reaction rules* to be interpreted correctly as *a single event*. Essentially the simulation can be accomplished by compiling the *when* and *which* questions Gillespie's direct method asks into an event. Specifically the event is triggered by the calculated next reaction time (`tau`); the event obtains a new random variable (`ran`) and then conditionally updates the particle counts depending on which reaction is chosen to occur next. To accommodate all stochastic rules in one event, each update entry is composed of conditional expressions over the propensities and the random number that decides which reaction occurs.

Example 3.1. Given the stochastic reaction rules $A + 2B \xrightarrow{k_1} C$ and $C \xrightarrow{k_2} 2A$ from [Gillespie 1976], we derive their propensities by

$$\texttt{alpha1} = k_1 \times (nA) \times \frac{(nB) \times (nB - 1)}{2}$$

$$\texttt{alpha2} = k_2 \times (nC)$$

where "nX" denotes the particle count of species X. Then the next reaction time from the current time point can be decided by

$$\texttt{e} = \frac{-1}{\texttt{alpha\_sum}} \cdot \log(ran_1)$$

for ran_1 a random number within $[0, 1]$ and where $\texttt{alpha\_sum} = \texttt{alpha1} + \texttt{alpha2}$. The first reaction is chosen for the next occurring reaction if $0 < (\texttt{alpha\_sum} \times ran_2) \leqslant \texttt{alpha1}$, which leads to the consumption of one A and two B's and producing one C.

This is achieved by the following event:

```
event(Time>tau, [ran, tau, ran, nA, nB, nC],
                [rand, Time + e, rand,
                 if alpha_sum*ran =< alpha1 then nA-1 else nA+2,
                 if alpha_sum*ran =< alpha1 then nB-2 else nB,
                 if alpha_sum*ran =< alpha1 then nC+1 else nC-1]).
macro(rand, 1664525 * ran + 1013904223 / 4294967296 -
            floor(1664525 * ran + 1013904223 / 4294967296)).
```

Where `rand` is a macro implementing a PRNG as explained in Section 2.2 and `e` is defined as shown above. Note that the update of the particle counts of the first reaction is reflected in the three `then` entries, and that of the second reaction is reflected in the three `else` entries. Note also that a single parameter `ran` is used twice to store first ran_1 then ran_2, this is in order to use our simple linear PRNG with a single seed.

This encoding relies on the left to right ordering of the different events associated to a single trigger (see Section 2.2). This ordering is imposed to two kinds of parameters – the random number, and a reaction's propensity function – such that possible errors are avoided. Because the two kinds of parameters depend on the *current* number of molecules, they are listed in front of molecular species. So their values are not changed before the *completion* of reaction firing, that is, all species' counts have been updated according to the chosen reaction.

3.3. Preprocessor for Composing Continuous and Stochastic Models

The purpose of our *preprocessor* for composing heterogeneous biochemical models automatically is to provide a simple interface for specifying hybrid simulations without digging into algorithmic details. The only work for the user is to decide the semantic model for each of the reactions under simulation. The models are then *processed* into a composed hybrid model suitable for simulation or analysis.

In classical work on hybrid simulation [Alfonsi et al. 2005; Kiehl et al. 2004], chemical reactions are divided into two groups according to their *propensities* and *reactants' concentrations*: one consisting of reactions to be simulated stochastically using SSAs, and the other consisting of reactions to be simulated deterministically using ODEs. The former is referred to as the *stochastic reactions* and the latter *continuous reactions*. While continuous reactions simply advance continuously according to their governing ODEs with the pass of time, stochastic reactions fire discretely in time with frequency determined by their propensities. When the *reactant concentrations/particle counts* and the *propensity* of a reaction are sufficiently large, ODE simulation can be faithfully applied without introducing unacceptably large errors in particle counts when using the total counts of corresponding species as references. (i.e. keeping the ratio $\frac{|nX_{\texttt{expected}} - nX_{\texttt{ODE}}|}{nX_{\texttt{ODE}}}$ acceptably small, where $nX_{\texttt{expected}}$ is the expected particle count of species X in fully stochastic simulations, and $nX_{\texttt{ODE}}$ is the particle count obtained when ODE simulation is allowed.) At the same time, frequent updates of particle counts within a small time interval are avoided, thus accelerating the simulation speed.

Hybrid species are referred to as those involved in both stochastic and continuous reactions. This kind of species requires special attention because they are influenced by two different mechanisms: *ODEs* that govern differential behavior by continuously changing related *concentrations*, and *events* that regulate stochastic behavior by modifying *particle counts* discretely whenever triggered. So a hybrid species is under two kinds of modification: one targets at the evolution of macroscopic concentrations and the other targets at the changes in microscopic particle counts.

In our implementation, a fresh new variable is introduced for each hybrid species to represent its total quantity (the summation of the numbers of particles from both continuous and stochastic models). That variable is set equal to the sum of a continuous variable multiplied by the corresponding volume and a small discrete number of particles. ODEs will act on the continuous part, whereas discrete events will impact the discrete one[2]. In all kinetic expressions (i.e. in rate equations and propensity functions), the hybrid species are expressed by the corresponding new variables representing the total amount. It is then a simple matter to put together the ODEs for the continuous part and the events corresponding to the encoding of the stochastic part as described in the previous section. It is worth noting however that while the total amount of each species is guaranteed to be nonnegative, the continuous part can sometimes become negative.

Example 3.2. The reaction model of bacteriophage T7 infection described in [Alfonsi et al. 2005] is an interesting example that can be hybridized by static partitioning of the reactions with continuous semantics for protein synthesis and with stochastic semantics for gene activation. The partition in [Alfonsi et al. 2005] consists in taking the fifth and sixth reactions with the continuous semantics and the other reactions with stochastic semantics, as follows:

[2]This specific implementation is related to the constraint that, contrary to the SBML specification, BIOCHAM continuous variables cannot be modified by events.

```
% Continuous reaction rules
MA(c5) for tem => tem + struc.
MA(c6) for struc => _.

parameter(c5, 1000).
parameter(c6, 1.99).
```

```
% Stochastic reaction rules
MA(c1) for gen => tem.
MA(c2) for tem => _.
MA(c3) for tem => tem + gen.
MA(c4) for gen + struc => virus.

parameter(c1, 0.025).
parameter(c2, 0.25).
parameter(c3, 1).
parameter(c4, 0.0000075).
```

In this example, `tem` and `struc` are hybrid species representing respectively the template viral nucleic acids and the viral structural proteins, while `gen` and `virus` are purely stochastic and represent the genomic viral nucleic acids and the final virus. The full input files with parameters and output file after preprocessing in both BIOCHAM and SBML are available at http://lifeware.inria.fr/supplementary_material/TOMACS/Alfonsi/. All experiments in this paper are conducted in BIOCHAM on a 2.9GHz Intel Core i7 platform with 16GB 1600MHz DDR3 memory.

Table I summarizes the result from 1000 simulations over a time horizon of 100 days. The experimental results shows that the hybrid simulation improves by three orders of magnitude the simulation time. The accuracy of the hybrid simulation technique will be demonstrated in more detail in Example 4.3.

method	#fired events	CPU time (sec)
stochastic	276556	218.7
hybrid	832	0.75
ratio	0.003	0.003

Table I: A comparison between purely stochastic and hybrid simulation implemented using chemical reactions and events. Columns *#fired events* and *CPU time* respectively hold the number of events triggered and runtime in seconds. All values are the average of 1000 simulations over a time horizon of 100 days. The last row shows the ratio of hybrid to stochastic statistics.

4. DYNAMIC STRATEGIES FOR HYBRID CONTINUOUS-STOCHASTIC SIMULATIONS

The above discussion assumes a static partition of a set of reactions into two subsets interpreted under continuous and stochastic semantics. Once the partition is established based on the system's initial conditions and partition criteria, it stays *fixed* throughout a simulation process. However, such a partition strategy may be inadequate for two reasons: firstly, a good static partition may not be known *a priori* given only initial conditions, secondly, a good static partition may not even exist. Essentially a fixed semantic interpretation of a reaction can lead to inaccurate and/or inefficient simulation when the reaction's reactants' counts and/or its propensity fluctuate substantially over time, thus violating the legitimacy of abstraction with continuous semantics and/or being unnecessarily trapped in the too frequent firing of reaction events. It is therefore desirable to adjust the reaction partition *dynamically* along the progress of a simulation.

4.1. Dynamic Partitioning Criteria

Particle count and *propensity value* [Alfonsi et al. 2005] are predominant factors of choice between the stochastic and continuous interpretation of reactions. Examples

of other higher-level factors derived from the two include: critical relative fluctuation [Bentele and Eils 2005] that describes a reaction's influence on a species' count relative to each one's total count; particle count of substrate involved, and ratio of a reaction's propensity to the sum of all reactions' propensities [Puchaka and Kierzek 2004]. In [Wagner et al. 2006] however, the partitioning criteria themselves, composed of particle count and propensity value, do not possess explicit meanings, rather they are derived to guarantee that the error of each approximation is smaller than the user-specified value.

We adopt a partition strategy that takes both particle counts and propensities into account: A reaction can be interpreted as differential only if its propensity value exceeds some target threshold *and* its related species' particle counts all exceed a certain threshold. In the sequel, we will refer to the two threshold values as *propensity threshold* and *particle count threshold*, respectively. To preserve flexibility on the user's side to decide the trade-off between accuracy and efficiency, both non-negative thresholds can be tuned by users according to the need. Increasing the value(s) leads to more accurate and less efficient simulations, while lowering the value(s) leads to more efficient but less accurate simulations. Note that a threshold's value can be set to zero if the accuracy degradation caused by its corresponding property is assumed to be non-substantial.

Consider the SBML reactions of Section 2.1. For reaction i with $\sum_j l_{ij} \times S_j \rightarrow \sum_j r_{ij} \times S_j$, under the time step size Δ of the ODE simulator used, by default we let

$$\begin{aligned} \text{propensity threshold} &= (n_1 \times \Delta)^{-1}, \text{ and} \\ \text{particle count threshold} &= n_2 \times max_{i,j}|r_{ij} - l_{ij}|, \end{aligned}$$

where n_1 and n_2 are two parameters of non-negative real values. To determine the value of n_1, note that the expected time period from present to a reaction's next firing equals the reciprocal of its propensity value. To avoid simulation being trapped by the frequent firing of a fast (with respect to Δ) reaction, we can interpret a reaction as continuous only if its expected time period from present to its next firing is shorter than the reciprocal of the propensity threshold. Thereby we may potentially skip unnecessarily many event updates. The smaller the value of n_1 is, the lesser the efficiency is gained from continuous semantics. On the other hand, to determine n_2, note that $max_{i,j}|r_{ij} - l_{ij}|$ is the largest possible change in particle count by one reaction firing among all reactions. For continuous semantics to be legitimate, particle counts should be large enough. Furthermore, for continuous semantics to be a good approximation, the change in the particle count of each species by one reaction occurrence should be relatively small compared to the species' total count. The larger the value of n_2 is, the more stringent the condition is for a reaction to be interpreted as continuous.

4.2. Implementation

There are two directions of semantic switching in dynamic partitioning: (1) from continuous to stochastic, and (2) from stochastic to continuous. During simulation, instead of monitoring the switching criteria all the time, the reactions are checked against the criteria within the event that realizes stochastic reactions, and only at the time of reaction firing. At the start of a simulation, all reactions are classified as stochastic by default. When a reaction event is triggered, apart from updating the particle counts according to the reaction selected, all reactions are checked against the user-specified criteria whether they are eligible for continuous semantics. The eligibility is usually based on the requirement of being theoretically sound (for accuracy concern) and can favorably include being practically beneficial (to improve efficiency). Switching from continuous to stochastic occurs when a reaction no longer satisfies the criteria; a reac-

tion is switched from stochastic to continuous if its current condition can satisfy the criteria.

For the first switching direction, postponement in switching can result in accuracy degradation. Indeed, one continuous reaction requires switching to stochastic only when the small error assumption for continuous interpretation is no longer satisfied. With our event formulation, the delay is *at most* the time period between now and the next reaction time of current set of stochastic reactions, provided that there is at least one stochastic reaction. When there is no stochastic reaction, the sum of propensities will be zero and thus resulting in infinite waiting time till the next reaction. To avoid this infinite waiting problem, the absence of stochastic reactions can be detected to enforce progress in simulation (this is achieved by the last macro, i.e., function definition, as shown in the BIOCHAM code of Example 4.1).

For the second switching direction, postponement in switching does not lead to loss of accuracy, although early switching can improve simulation efficiency. Since switching in both directions are realized in the same event, the upper bound of the delay is the same as that of the first switching direction. To make the most out of the unavoidable trade-off between accuracy and efficiency, once the partitioning strategy and its corresponding criteria are set, the goal becomes one of always maximizing the set of continuous reactions without violating the criteria.

4.3. Simple Example

The following example shows the general implementation of dynamic partitioning by SBML events for Example 3.1. Once the partitioning strategy is chosen, the corresponding part of partitioning criteria that determine when a reaction can switch from stochastic to continuous are incorporated into the event as conditions, in ways demonstrated in the example.

Example 4.1. Consider again the system of two reactions: $A + 2B \xrightarrow{k_1} C$ and $C \xrightarrow{k_2} 2A$. The main structure of BIOCHAM code used to fulfill simulation with dynamic partitioning is as follows:

```
% Continuous semantics
MA(k1_diff) for A + 2*B => C.
MA(k2_diff) for C => 2*A.

% Event for stochastic semantics and dynamic partitioning
event(Time>tau,
      [ran, tau, ran, k1_diff, k1_stoch, k2_diff, k2_stoch, nA, nB, nC],
      [rand, Time + e, rand,
         if (condition for reaction 1 to be continuous is satisfied)
             then k1 else 0,
         if k1_diff=0 then k1 else 0,
         if (condition for reaction 2 to be continuous is satisfied)
             then k2 else 0,
         if k2_diff=0 then k2 else 0,
         if alpha_sum*ran =< alpha1 then nA-1 else nA+2,
         if alpha_sum*ran =< alpha1 then nB-2 else nB,
         if alpha_sum*ran =< alpha1 then nC+1 else nC-1]).

% Hybrid species
macro(A_total, [A]*volume + nA).
macro(B_total, [B]*volume + nB).
macro(C_total, [C]*volume + nC).
macro(alpha1, k1*A_total*B_total*(B_total-1)/2).
macro(alpha2, k2*C_total).
```

```
macro(alpha_sum, alpha1 + alpha2).
macro(e, if alpha_sum=0
        then (-1/propensity threshold)
        else (-1/alpha_sum)*log(ran)).
```

Under dynamic partitioning, all species are treated as hybrid continuous-stochastic species; each reaction can become either continuous or stochastic, but not both, at any time point. Specifically, each rate constant k_i is duplicated into k_i*_di* and k_i*_stoch*, to simplify the process of semantic switching to value alteration between 0 and the real value of rate constant. A reaction r is *stochastic* if and only if k_r*_di* is set to 0 and k_r*_stoch* is set to the r's rate constant value, while it is *continuous* if and only if k_r*_stoch* is set to zero and k_r*_di* is set to the value of its natural rate constant.

The last macro decides the next reaction time of the set of stochastic reactions, which is also the next time point for checking and adjusting the partition until consistent with the strategy imposed. The *else* part is the same as that in the static partitioning, which implements Gillespie's Direct Method as described previously in Section 3.1. The *then* part serves to avoid, when all reactions become continuous, the problem of infinite waiting time before the next reaction. Note that the value is also the upper bound of semantic switching delay, which is set here to be the average firing period of the fastest possible stochastic reaction under current strategy. This is to make sure that the average particle count error resulted from delayed switching to stochastic semantics will not exceed the species' stoichiometric number in the reaction.

Note that this encoding allows to trace which reactions of the SBML model were chosen to be stochastic (resp. continuous) at which point in time, simply by observing the value of k_i*_stoch* (resp. k_i*_di*), which is non-null when the reaction is stochastically (resp. continuously) evaluated.

4.4. Performance Evaluation

The effectiveness of dynamic over static partitioning by our proposed framework is evaluated in Examples 4.2 and 4.3 below. Additionally, implementations of different partitioning strategies and a comparison among them is presented in Example 4.3.

Example 4.2. We study Goutsias model [Henzinger et al. 2010] to demonstrate the effectiveness of dynamic partitioning. The model describes the transcription regulation of a repressor protein M in bacteriophage λ. It involves 6 different species and 10 reactions listed as follows:

$RNA \xrightarrow{c_1} RNA + M$
$M \xrightarrow{c_2} \emptyset$
$DNA.D \xrightarrow{c_3} RNA + DNA.D$
$RNA \xrightarrow{c_4} \emptyset$
$DNA + D \underset{c_6}{\overset{c_5}{\rightleftharpoons}} DNA.D$
$DNA.D + D \underset{c_8}{\overset{c_7}{\rightleftharpoons}} DNA.2D$
$M + M \underset{c_{10}}{\overset{c_9}{\rightleftharpoons}} D$

Assume the particle counts and parameters are initialized as follows:
$\#RNA_{t=0} = \#DNA.D_{t=0} = \#DNA.2D_{t=0} = 0$
$\#M_{t=0} = \#D_{t=0} = 10$
$\#DNA_{t=0} = 2$

c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}
0.043	7×10^{-4}	71.5	3.9×10^{-6}	0.02	0.48	4×10^{-4}	9×10^{-12}	0.08	0.5

For static partition used in this example, reactions $M + M \underset{c_{10}}{\overset{c_9}{\rightleftharpoons}} D$ are interpreted under differential semantics while all other reactions are stochastic. The partition is based on the fact that molecules M and D have the greatest initial counts, and both have initial propensities no less than 5 while all other reactions' initial propensities are much smaller than 1. As for dynamic partition used, the propensity threshold and the particle count threshold are set to 5 (with $n_1 = 20$) and 20 (with $n_2 = 10$), respectively; a reaction is interpreted as continuous only if its propensity value exceeds the propensity threshold *and* its related species' particle counts all exceed the particle count threshold. This criterion aims to take both population and propensity into account for the following reasons: firstly, in this model, discreteness from extremely low particle counts is the main cause of violation to the continuous semantics' assumption. Secondly, the rate constants of the system span orders of magnitudes, even among reactions with shared reactants. So it can be highly probable that the large difference in reaction rates can introduce inefficiency during simulation.

With both static and dynamic strategies partitioning reactions into continuous and stochastic based on the same considerations, i.e. particle counts *and* propensities, the only major difference between the two strategies is the allowed time point for information gathering and making corresponding semantic alterations. For static strategy, reactions are partitioned once and for all based on initial particle counts and propensities. Dynamic strategy, on the other hand, updates the partition according to current system state whenever an event is triggered. Figure 1 shows the average results from 1000 simulations. Note that even as static partition strategy has taken initial conditions into account, the difference between static partition strategy and the expected result (obtained by averaging over 1000 fully stochastic simulations) is already much larger than that of dynamic partition strategy after 5 time units.

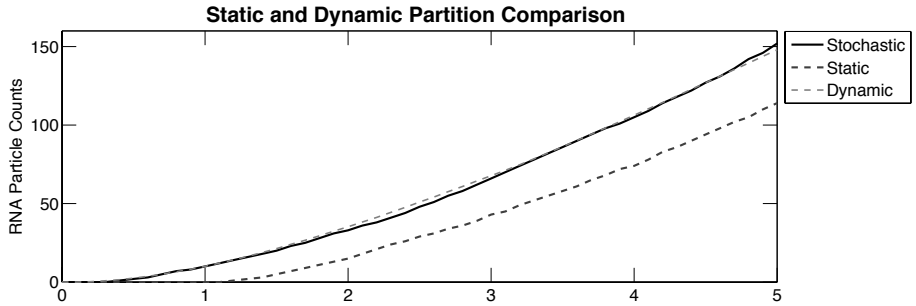

Fig. 1: Comparison of static and dynamic partition strategy with stochastic simulation result. Each curve represent the average of 1000 simulation runs of corresponding strategy, with simulation horizon = 5 time units.

Apart from the accuracy improvement shown in Figure 1, a substantial reduction on the firing of events and thus CPU time is achieved by the dynamic partition, as is shown in the last two rows of Table II. Notice that the reduction on event count is more substantial than the reduction on run time because of the extra checking needed in the dynamic partition to decide potential switchings at each event firing.

Figure 2 explains these results by showing the behavior of the dynamic partioning strategy in this example. On the long time horizon, the dynamic strategy interprets reactions $\{1, 9, 10\}$, i.e., the production and reversible dimerization of protein M, as continuous and the other ones as stochastic. However, on the first 7 units of time,

method	#fired events	CPU time (sec)
purely stochastic	141 036.91	96.07
static partition	9931.68	9.67
dynamic partition	126.42	1.61
ratio over stochastic	0.000 896	0.0168
ratio over static partition	0.0127	0.166

Table II: Average number of events fired and average runtime from 100 simulations with simulation horizon set to 100 time units, comparing over three simulation methods. The last two rows are the ratios of dynamic partition strategy's statistics to that of purely stochastic and static partition strategy's, respectively.

the dynamic strategy applies a completely different choice, with stochastic interpretation for those reactions and reaction 3, the RNA production, continuous. Then, for a transient time of around 20 units, reactions $\{1, 3, 9, 10\}$ are mainly continuous with a decreasing frequency for reaction 3.

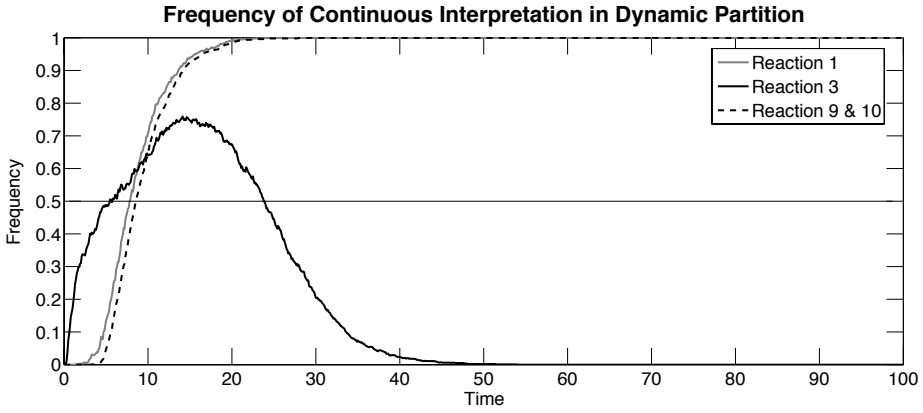

Fig. 2: The frequency of each reaction being interpreted as continuous under dynamic partition strategy, over time horizon of 100 units, and calculated over 1000 simulations. Reactions not listed are never interpreted as deterministic during the simulation horizon.

Example 4.3. Let us consider again the model of intracellular growth of bacteriophage T7 of Example 3.2 with the static partitioning strategy of [Alfonsi et al. 2005], noted $\{1, 2, 3, 4\}$ since the first four reactions are always stochastic and the last two ones always continuous, and with a different static partition $\{1, 3\}$ in which only the first and third reactions are stochastic, the others being continuous. For dynamic partition, the propensity threshold and the particle count threshold are set to be 10 (with $n_1 = 10$) and 5 (with $n_2 = 5$), respectively.

Figure 3 depicts the relative frequencies of the numbers of *tem* molecules after 50, 100, 150, 200 days, obtained with that static partition, with the dynamic partitioning strategy, and with SSA. Each bar represents the relative frequency of *tem* molecule count falling in that region after certain amount of time. As can be clearly seen in the graph, bars of static partition deviate from those of purely stochastic simulation, while bars of dynamic partition are closer to the purely stochastic ones.

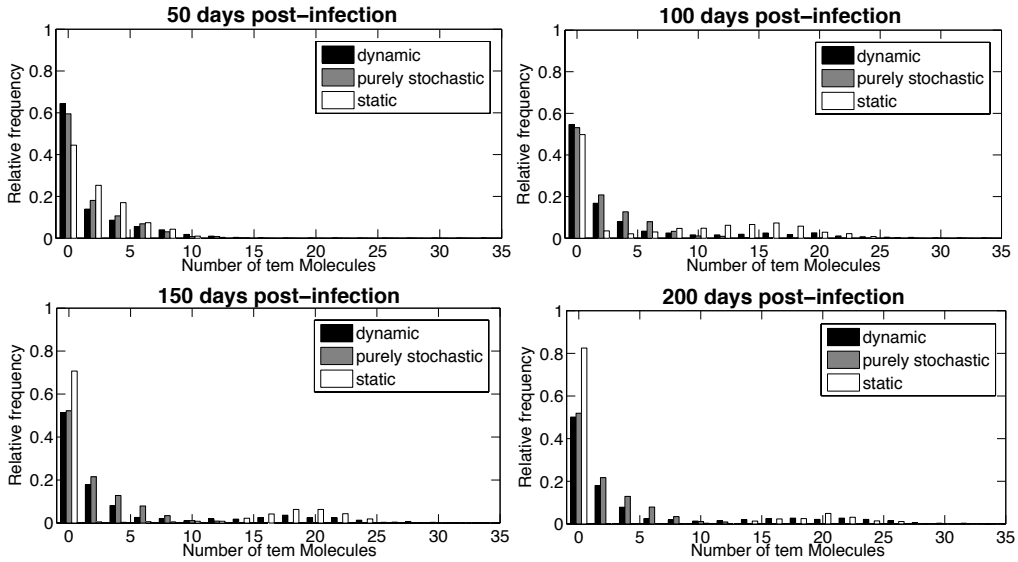

Fig. 3: Comparison of post-infection distributions of *tem* particle counts obtained at time= 50, 100, 150, 200 days, by stochastic, static hybrid and dynamic hybrid simulations (based on 1000 simulation runs of each strategy.)

These observations can be made quantitative using statistical distances. Let us use the *two sample Kolmogorov-Smirnov test* as distance measure (KS distance) to compare the relative frequencies. Table III shows the KS distance between the distributions obtained by SSA and the static partitioning $\{1, 2, 3, 4\}$ of [Alfonsi et al. 2005], the static partitioning $\{1, 3\}$ and the dynamic partitioning respectively.

Post-infection Time (days)	50	100	150	200
KS distance SSA - static hybrid $\{1, 2, 3, 4\}$	0.0525	0.7145	0.8035	0.836
KS distance SSA - static hybrid $\{1, 3\}$	0.3815	0.9225	0.624	0.6055
KS distance SSA - dynamic hybrid	0.0515	0.116	0.1485	0.161

Table III: Post-infection distributions of *tem* molecules from simulations using different hybrid strategies compared to the reference fully stochastic model. Each row contains the outcome of applying two-sample Kolmogorov-Smirnov test on the distributions obtained from 1000 simulations using specified hybrid strategy and the reference fully stochastic model. The smaller the value, the more similar the two distributions involved. Distributions at four sampling points are used for comparison through the evolution of time, as listed in four corresponding columns.

By taking the particle count distribution of purely stochastic simulation as the reference, this example shows that the dynamic strategy always beats the static partition strategies and improves the accuracy of the simulation to a small distance from SSA along all time points.

5. HYBRID BOOLEAN MODELS

In this section we demonstrate how Boolean models can also be composed with continuous and even hybrid continuous-stochastic models in SBML.

5.1. Preprocessor for Composing Continuous and Boolean Models

In this section, we consider the composition of continuous reaction models with Boolean transition systems. One typical use of this form of composition is for modeling the interactions between gene expression and metabolism on different time scales. Gene networks can be modeled by simple Boolean regulatory networks representing the on/off states of the genes and the possible transitions from one state to another, while metabolic networks are naturally modeled by chemical reactions with continuous semantics. Hybrid models of gene expression and metabolism can thus be naturally built as hybrid continuous-Boolean models, and analyzed and simulated as such.

A continuous-Boolean composition necessitates specifying:

— the link between the discrete/continuous variables and the Boolean variables, e.g. by fixing particle count or concentration threshold values,
— the relationship between the discrete logical time of the Boolean model and the continuous real time of the continuous reaction model, e.g. by adding delays on Boolean transitions,
— the integrity constraints between both dynamics.

There is currently no general method for these tasks. Our high-level interface takes as input

(1) a reaction model that accommodates both stochastic and continuous semantics,
(2) a Boolean transition system,
(3) an interface specifying for each Boolean transition, the triggers and actions on the reaction model variables,

and produces as output a system of reactions and events which synchronize the execution of both input models.

5.2. Hybrid Composition of Continuous-Boolean Cell Cycle Models

In [Singhania et al. 2011], Singhania et al. have proposed a simple hybrid model of the mammalian cell cycle regulation. This cell cycle model of low dimension has been evaluated in terms of flow cytometry measurements of cyclin proteins in asynchronous populations of human cell lines. The few kinetic constants in the model are easier to estimate from the experimental data than the numerous kinetic constants of a single large ODE model.

In this model, cyclin abundances are tracked by piecewise linear continuous equations for cyclin synthesis and degradation. Cyclin synthesis is regulated by transcription factors whose activities are represented by discrete variables (0 or 1) and likewise for the activities of the ubiquitin-ligating enzyme complexes that govern cyclin degradation. The discrete variables change according to a predetermined sequence, with the times between transitions determined by the amount of cyclin presented as well as exponentially distributed random variables.

This model can be reconstructed using our interface as the hybrid composition of a purely continuous reaction model of cyclin activation and degradation, with a Boolean model of cell cycle phase transitions. We provide here the real examples and thus the ASCII syntax for the BIOCHAM constructs described in Section 2.1. Beside the syntax introduced before, the `present` command specifies the initial concentration, and the `macro` command defines a function that makes the reaction rates dependent on the value of boolean variables, as specified in the original article.

The inputs are:

(1) *the continuous reaction model of cyclin activation*, which provides an always progressing continuous behavior:

```
% Initial Conditions
present(CycA, 1).
present(CycB, 1).
present(CycE, 1).

% Reaction Rules
k_sa for _ => CycA.
MA(k_da) for CycA => _.

k_sb for _ => CycB.
MA(k_db) for CycB => _.

k_se for _ => CycE.
MA(k_de) for CycE => _.

macro(k_sa, 5+6*B_tfe+20*B_tfb).
macro(k_sb, 2.5+6*B_tfb).
macro(k_se, 0.02+2*B_tfe).
macro(k_da, 0.2+1.2*B_cdc20a+1.2*B_cdh1).
macro(k_db, 0.2+1.2*B_cdc20b+0.3*B_cdh1).
macro(k_de, 0.02+0.5*B_scf).
```

(2) *the Boolean transition system of the cell cycle progression*, which is given in [Singhania et al. 2011] as the following limit cycle of state transitions. The `add_boolean_state` command defines a numbered state, and associates the boolean variables true in that state; the `add_boolean_transition` command defines a named transition between two states. Here is an excerpt of the file:

```
% States and corresponding active boolean species
add_boolean_state(1, [B_cdh1]).
add_boolean_state(2, [B_tfe, B_cdh1]).
add_boolean_state(3, [B_tfe]).

...

set_initial_boolean_state(1).

% Transitions between states
add_boolean_transition(T12, 1, 2).
add_boolean_transition(T23, 2, 3).

...
```

(3) *the synchronization between both models*, specified as a set of triggers and actions (similar to the ones in events described in Section 2.2) associated to the Boolean transitions via the `add_boolean_transition` command. In this hybrid model, the time for the Boolean transitions are given by random variables. This is represented by a parameter `tau` and a macro `next_event` as can be seen in the following excerpt:

```
parameter(tau, 0).
macro(next_event, Time - lambda * log(ran)).
event(Time = 0, [ran, tau], [rand, next_event]).

parameter(theta_e, 80).
parameter(theta_a, 12.5).
parameter(theta_1_b, 21.25).
parameter(theta_2_b, 3).
```

```
add_interface(T12, Time > tau, [ran, lambda, tau], [rand, 0,
    next_event]).
add_interface(T23, Time > tau and [CycE] * masst >= theta_e,
   [ran, lambda, tau], [rand, 0.01, next_event]).

...
```

The result of the composition is an SBML model formed of the continuous reaction model augmented with a list of events. The events implement the Boolean transition cycle from state 1 to 9 and back to 1, and their synchronization with the continuous reaction model. In this form, the hybrid model can be simulated using any simulator of SBML models. The simulation over a time horizon of *100 hours* takes 150 ms. The simulation result is shown in the upper plot in Figure 4.

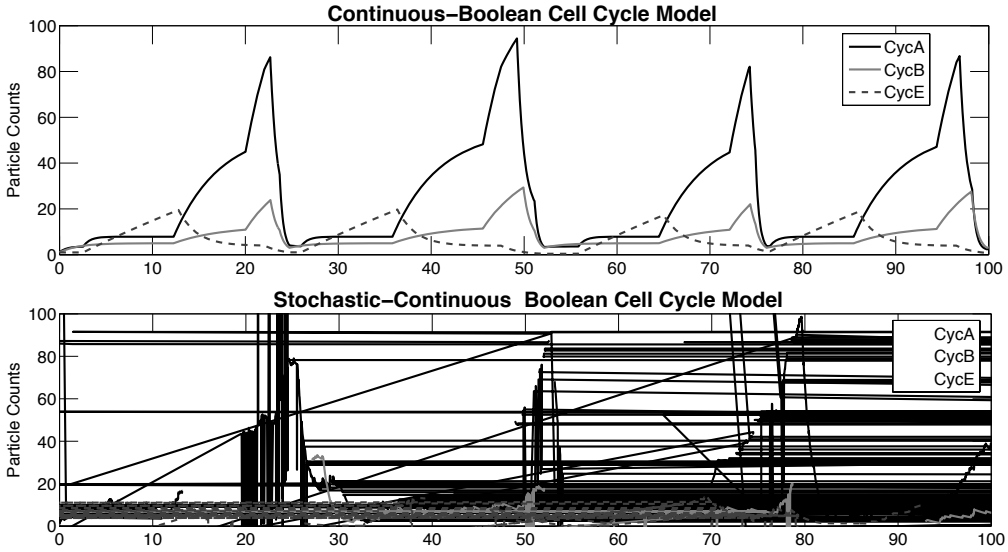

Fig. 4: Evolution of the cyclin's particle counts in Singhania et al. model of the cell cycle. Simulation horizon = 100 hours. (Upper) original continuous-Boolean model (with stochastic delays), with average runtime = 0.15 seconds; (Lower) hybrid stochastic-continuous-Boolean model, with average runtime = 8.42 seconds.

5.3. Hybrid Simulation of Stochastic-Continuous-Boolean Cell Cycle Model

A continuous-Boolean model can be easily generalized to a more realistic stochastic-continuous-Boolean one by extending the purely continuous reaction model to a stochastic-continuous one, using event-based methods as shown in Section 3.2 and Section 3.3.

In the lower plot in Figure 4, we demonstrate the simulation result of a stochastic-continuous-Boolean cell cycle model. This model extends the purely-continuous reaction model of cyclin activation proposed by Singhania et al., making it more realistic by allowing stochastic semantics for reactions. In the model simulated here, cyclin synthesis reactions are stochastic and cyclin degradation reactions are continuous.

6. CONCLUSION

The combination of events with kinetic reactions, as already present in the SBML standard format, provides enough expressive power for combining discrete and continuous dynamics. Although introduced in SBML for handling a limited number of events in biochemical models, such as the division of the mass variable by two at each cell division in cell cycle models, we have shown that SBML events can be used in a non-standard way on a large scale to define stochastic, boolean and hybrid simulators.

We have presented a high-level interface for composing hybrid models, compiling them into SBML reactions plus events, and running hybrid simulations. We have first shown how Gillespie's direct method for stochastic simulation can be implemented with SBML events. This makes it possible to compose a stochastic model with a continuous reaction model to create a hybrid continuous-stochastic reaction model in standard SBML format, and perform hybrid simulations with any simulation tool supporting SBML Level 3 core.

The benefits of this approach and the gain of performance over purely stochastic simulations, have been illustrated on the model of bacteriophage T7 of [Alfonsi et al. 2005]. Furthermore, we have shown how dynamic strategies based on particle count and reaction propensities can be easily implemented in this framework, to dynamically partition the continuous and stochastic reactions. The gain in accuracy of the dynamic partitioning strategy over static partition strategies, has been shown on the bacteriophage T7 example, and on another example of transcription regulation in bacteriophage λ.

Then, we have shown that hybrid Boolean-continuous models can be composed by specifying the input models, the conditions on the continuous variables, and the time delays of the Boolean transitions. This has been illustrated by a reconstruction of the hybrid mammalian cell cycle model of [Singhania et al. 2011], and by a hybrid stochastic-continuous-Boolean version of it obtained by just specifying in our interface that the synthesis reactions are stochastic.

ACKNOWLEDGMENTS

This work has been supported by the French OSEO Biointelligence and ANR BioTempo projects, and by the European Eranet Sysbio C5Sys project. We acknowledge fruitful discussions with our partners in these projects and with Robin Philip for his early work on this topic during an internship with us at Inria.

REFERENCES

Jamil Ahmad, Gilles Bernot, Jean-Paul Comet, Didier Lime, and Olivier Roux. 2006. Hybrid Modelling and Dynamical Analysis of Gene Regulatory Networks with Delays. *ComplexUs* 3 (2006), 231–251.

Jamil Ahmad, Olivier Roux, Gilles Bernot, Jean-Paul Comet, and Adrien Richard. 2008. Analysing formal models of genetic regulatory networks with delays. *International Journal of Bioinformatics Research and Applications* 4, 3 (2008), 240–262.

Ozgur E. Akman, Maria Luisa Guerriero, Laurence Loewe, and Carl Troein. 2010. Complementary approaches to understanding the plant circadian clock. In *Proceedings Third Workshop From Biology To Concurrency and back, FBTC 2010, Paphos, Cyprus, 27th March 2010. (EPTCS)*, Emanuela Merelli and Paola Quaglia (Eds.), Vol. 19. Open Publishing Association, 1–19. DOI: http://dx.doi.org/10.4204/EPTCS.19.1

Aurélien Alfonsi, Eric Cancès, Gabriel Turinici, Barbara di Ventura, and Wilhelm Huisinga. 2005. Adaptive simulation of hybrid stochastic and deterministic models for biochemical systems. *ESAIM: Proc.* 14 (Sept. 2005), 1–13. DOI: http://dx.doi.org/10.1051/proc:2005001

Rajeev Alur, Calin Belta, Franjo Ivanicic, Vijay Kumar, Max Mintz, George J. Pappas, Harvey Rubin, and Jonathan Schug. 2001. Hybrid modeling and simulation of biomolecular networks. In *Proceedings of the 4th International Workshop on Hybrid Systems: Computation and Control, HSCC'01 (Lecture Notes in Computer Science)*, Vol. 2034. Springer-Verlag, Rome, Italy, 19–32.

Michael Ashburner, Catherine A. Ball, Judith A. Blake, David Botstein, Heather Butler, J. Michael Cherry, Allan P. Davis, Kara Dolinski, Selina S. Dwight, Janan T. Eppig, Midori A. Harris, David P. Hill, Laurie Issel-Tarver, Andrew Kasarskis, Suzanna Lewis, John C. Matese, Joel E. Richardson, Martin Ringwald, Gerald M. Rubin, and Gavin Sherlock. 2000. Gene Ontology: tool for the unification of biology. *Nature Genetics* 25 (2000), 25–29.

Martin Bentele and Roland Eils. 2005. General Stochastic Hybrid Method for the Simulation of Chemical Reaction Processes in Cells. In *CMSB'05: Proceedings of the third international conference on Computational Methods in Systems Biology (Lecture Notes in Computer Science)*, Vol. 3082. Springer-Verlag, Paris, France, 248–251.

Natalie Berestovsky, Wanding Zhou, Deepak Nagrath, and Luay Nakhleh. 2013. Modeling Integrated Cellular Machinery Using Hybrid Petri-Boolean Networks. *PLoS Computational Biology* 9, 11 (Nov. 2013), 1003306. DOI : http://dx.doi.org/10.1371/journal.pcbi.1003306

Alexander Bockmayr and Arnaud Courtois. 2002. Using hybrid concurrent constraint programming to model dynamic biological systems. In *Proceedings of ICLP'02, International Conference on Logic Programming (Lecture Notes in Computer Science)*, Vol. 2401. Springer-Verlag, Copenhagen, 85–99.

Laurence Calzone, François Fages, and Sylvain Soliman. 2006. BIOCHAM: An Environment for Modeling Biological Systems and Formalizing Experimental Knowledge. *Bioinformatics* 22, 14 (2006), 1805–1807. DOI : http://dx.doi.org/10.1093/bioinformatics/btl172

Nathalie Chabrier and François Fages. 2003. Symbolic model checking of biochemical networks. In *CMSB'03: Proceedings of the first workshop on Computational Methods in Systems Biology (Lecture Notes in Computer Science)*, Corrado Priami (Ed.), Vol. 2602. Springer-Verlag, Rovereto, Italy, 149–162. http://contraintes.inria.fr/ fages/Papers/CF02cmsb.pdf

François Fages and Sylvain Soliman. 2008a. Abstract Interpretation and Types for Systems Biology. *Theoretical Computer Science* 403, 1 (2008), 52–70. DOI : http://dx.doi.org/10.1016/j.tcs.2008.04.024

François Fages and Sylvain Soliman. 2008b. Formal Cell Biology in BIOCHAM. In *8th Int. School on Formal Methods for the Design of Computer, Communication and Software Systems: Computational Systems Biology SFM'08 (Lecture Notes in Computer Science)*, M. Bernardo, P. Degano, and G. Zavattaro (Eds.), Vol. 5016. Springer-Verlag, Bertinoro, Italy, 54–80. DOI : http://dx.doi.org/10.1007/978-3-540-68894-5_3

Vashti Galpin, Jane Hillston, and Luca Bortolussi. 2008. {HYPE} Applied to the Modelling of Hybrid Biological Systems. *Electronic Notes in Theoretical Computer Science* 218 (2008), 33–51. DOI : http://dx.doi.org/10.1016/j.entcs.2008.10.004 Proceedings of the 24th Conference on the Mathematical Foundations of Programming Semantics (MFPS XXIV).

Ronojoy Ghosh and Claire Tomlin. 2001. Lateral inhibition through Delta-Notch signaling: A piecewise affine hybrid model. In *Proceedings of the 4th International Workshop on Hybrid Systems: Computation and Control, HSCC'01 (Lecture Notes in Computer Science)*, Springer-Verlag (Ed.), Vol. 2034. Rome, Italy, 232–246.

David Gilbert, Monika Heiner, and Sebastian Lehrack. 2007. A Unifying Framework for Modelling and Analysing Biochemical Pathways Using Petri Nets. In *CMSB'07: Proceedings of the fifth international conference on Computational Methods in Systems Biology (Lecture Notes in Computer Science)*, Muffy Calder and Stephen Gilmore (Eds.), Vol. 4695. Springer-Verlag, Edinburgh, Scotland. DOI : http://dx.doi.org/10.1007/978-3-540-75140-3_14

Daniel T. Gillespie. 1976. General Method for Numerically Simulating Stochastic Time Evolution of Coupled Chemical-Reactions. *J. Comput. Phys.* 22 (1976), 403–434.

Daniel T. Gillespie. 1977. Exact Stochastic Simulation of Coupled Chemical Reactions. *Journal of Physical Chemistry* 81, 25 (1977), 2340–2361.

Daniel T. Gillespie. 2001. Approximate accelerated stochastic simulation of chemically reacting systems. *Journal of Chemical Physics* 115, 4 (July 2001), 1716–1733.

Daniel T. Gillespie. 2009. Deterministic Limit of Stochastic Chemical Kinetics. *The Journal of Physical Chemistry B* 113, 6 (2009), 1640–1644. DOI : http://dx.doi.org/10.1021/jp806431b

Andrew Golightly and Darren J. Wilkinson. 2011. Bayesian parameter inference for stochastic biochemical network models using particle Markov chain Monte Carlo. *Interface Focus* (2011). DOI : http://dx.doi.org/10.1098/rsfs.2011.0047

Ernst Moritz Hahn, Arnd Hartmanns, Holger Hermanns, and Joost-Pieter Katoen. 2013. A compositional modelling and analysis framework for stochastic hybrid systems. *Formal Methods in System Design* 43, 2 (2013), 191–232. DOI : http://dx.doi.org/10.1007/s10703-012-0167-z

Andreas Hellander and Per Lotstedt. 2007. Hybrid method for the chemical master equation. *J. Comput. Phys.* 227, 1 (2007), 100–122. DOI : http://dx.doi.org/10.1016/j.jcp.2007.07.020

Thomas A. Henzinger. 1996. The Theory of Hybrid Automata. In *Proceedings of the 11th Annual Symposium on Logic in Computer Science (LICS)*. IEEE Computer Society Press, 278–292. An extended version appeared in Verification of Digital and Hybrid Systems.

Thomas A. Henzinger, Pei-Hsin Ho, and Howard Wong-Toi. 1997. HYTECH: A Model Checker for Hybrid Systems. In *CAV'97: Proceedings of the 9th International Conference on Computer Aided Verification*. Springer-Verlag, 460–463.

Thomas A. Henzinger, Linar Mikeev, Maria Mateescu, and Verena Wolf. 2010. Hybrid numerical solution of the chemical master equation. In *Proceedings of the 8th International Conference on Computational Methods in Systems Biology (CMSB '10)*. ACM, New York, NY, USA, 55–65. DOI: http://dx.doi.org/10.1145/1839764.1839772

Michael Hucka and others. 2003. The Systems Biology Markup Language (SBML): A Medium for Representation and Exchange of Biochemical Network Models. *Bioinformatics* 19, 4 (2003), 524–531. http://sbml.org/

Trey Ideker, Timothy Galitski, and Leroy Hood. 2001. A New Approach to Decoding Life: Systems Biology. *Annual Review of Genomics and Human Genetics* 2 (2001), 343–372.

Minoru Kanehisa and Susumu Goto. 2000. KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Research* 28, 1 (2000), 27–30.

Jonathan R. Karr, Jayodita C. Sanghvi, Derek N. Macklin, Miriam V. Gutschow, Jared M. Jacobs, Benjamin Bolival Jr, Nacyra Assad-Garcia, John I. Glass, , and Markus W. Covert. 2012. A Whole-Cell Computational Model Predicts Phenotype from Genotype. *Cell* 150, 2 (2012), 389,401.

Thomas R. Kiehl, Robert M. Mattheyses, , and Melvin K. Simmons. 2004. Hybrid simulation of cellular behavior. *Bioinformatics* 20, 3 (2004), 316–322. DOI: http://dx.doi.org/10.1093/bioinformatics/btg409

M. Kwiatkowska, G. Norman, and D. Parker. 2008. Using probabilistic model checking in systems biology. *SIGMETRICS Performance Evaluation Review* 35, 4 (2008), 14–21.

O. Maler and G. Batt. 2008. Approximating continuous systems by timed automata. In *First International Workshop on Formal Methods in Systems Biology, FMSB'08 (Lecture Notes in Bioinformatics)*, J. Fisher (Ed.), Vol. 5054. Springer-Verlag, 77–89.

Hiroshi Matsuno, Atsushi Doi, Masao Nagasaki, and Satoru Miyano. 2000. Hybrid Petri Net Representation of Gene Regulatory Network. In *Proceedings of the 5th Pacific Symposium on Biocomputing*. Stanford, Hawaii, USA, 338–349.

Jürgen Pahle. 2009. Biochemical simulations: stochastic, approximate stochastic and hybrid approaches. *Briefings in Bioinformatics* 10, 1 (2009), 53–64. DOI: http://dx.doi.org/10.1093/bib/bbn050

Jacek Puchaka and Andrzej M. Kierzek. 2004. Bridging the Gap between Stochastic and Deterministic Regimes in the Kinetic Simulations of the Biochemical Reaction Networks. *Biophysical Journal* 86, 3 (2004), 1357–1372. DOI: http://dx.doi.org/10.1016/S0006-3495(04)74207-1

Howard Salis and Yiannis N. Kaznessis. 2005. Accurate hybrid stochastic simulation of a system of coupled chemical or biochemical reactions. *The Journal of Chemical Physics* 122, 5 (2005), 54103. DOI: http://dx.doi.org/10.1063/1.1835951

Howard Salis, Vassilios Sotiropoulos, and Yiannis N. Kaznessis. 2006. Multiscale Hy3S: Hybrid stochastic simulation for supercomputers. *BMC Bioinformatics* 7, 1 (2006), 93. DOI: http://dx.doi.org/10.1186/1471-2105-7-93

Rajat Singhania, R. Michael Sramkoski, James W. Jacobberger, and John J. Tyson. 2011. A Hybrid Model of Mammalian Cell Cycle Regulation. *PLOS Computational Biology* 7, 2 (Feb. 2011), e1001077.

René Thomas. 1973. Boolean formalisation of genetic control circuits. *Journal of Theoretical Biology* 42 (1973), 565–583.

Holger Wagner, Mark Mller, and Klaus Prank. 2006. COAST: Controllable approximative stochastic reaction algorithm. *The Journal of Chemical Physics* 125, 17 (2006), 174104. DOI: http://dx.doi.org/10.1063/1.2361284