BRIEFINGS IN BIOINFORMATICS. VOL 9. NO 5. 437–449
Advance Access publication May 7, 2008
doi:10.1093/bib/bbn023

The Beta Workbench: a computational tool to study the dynamics of biological systems

Lorenzo Dematté, Corrado Priami and Alessandro Romanel

Submitted: 30th November 2007; Received (in revised form): 4th April 2008

Abstract

We introduce the Beta Workbench (BWB), a scalable tool built on top of the newly defined BlenX language to model, simulate and analyse biological systems. We show the features and the incremental modelling process supported by the BWB on a running example based on the mitogen-activated kinase pathway. Finally, we provide a comparison with related approaches and some hints for future extensions.

Keywords: *systems biology; modelling and simulation; computational biology*

INTRODUCTION

Systems Biology [1] investigates the interactions and relationships among the components of biological systems to understand how they globally work. Several approaches have been developed and used to model and study complex interaction mechanisms in biological systems mainly based on mathematical modelling, which generally takes the form of a system of ordinary differential equations (ODEs) and for which ODE solvers with various interfaces are available. We also mention the approach based on chemical equations for which several methods for qualitative and quantitative analysis are provided like stochastic and deterministic time course simulation, steady state analysis and sensitivity analysis [2–5].

We describe the systems' behaviour by relying on computational modelling. A computational model differs from a mathematical one, because it is executable and not just simply solvable [6]. Execution means that we can predict/describe the flow of control between species and reactions (e.g. not only the time, but also the causality relation among the events that constitute the history of the dynamics of the model). In other words, our interpretation of computational modelling is similar to programming, the step by step behaviour of a system, rather than describing only the outcome of the system or scripting some code to solve mathematical formulations of problems.

Various computational approaches have been proposed and equipped with supporting software tools. Boolean and probabilistic Boolean networks textually describe models on which simulation of the network dynamics and computation of network statistics can be performed [7, 8]. Petri nets and their stochastic variants are used to visually describe models and to perform stochastic and deterministic time course simulation and pathway analysis [9–12]. The reactive animation framework is based on a reactive system that drives the display of an animation by combining classical tools for statecharts with tools for animation [13]. Logical formalisms have also been defined to perform probabilistic model checking analysis, i.e. to test

Corresponding author. Alessandro Romanel, Dipartimento di Ingegneria e Scienza dell'Informazione, The Microsoft Research, University of Trento, Centre for Computational and Systems Biology, Università di Trento, Povo, TN, Italy. Tel: +390-461-882-804; Fax: +390-461-882-814; E-mail: romanel@cosbi.eu

Lorenzo Dematté is a PhD student at the ICT International Doctorate School at the University of Trento and he is also currently carrying out research activities at the Microsoft Research, University of Trento Centre for Computational and Systems Biology.

Corrado Priami received his PhD in Computer Science from the University of Pisa, Italy. He is currently President & CEO of the Microsoft Research, University of Trento Centre for Computational and Systems Biology and full professor of Computer Science at the University of Trento.

Alessandro Romanel is a PhD student at the ICT International Doctorate School at the University of Trento and he is also currently carrying out research activities at the Microsoft Research, University of Trento Centre for Computational and Systems Biology.

whether a given logical formula is satisfied by a given model described in one of different possible formalisms (e.g. Continuous or Discrete Time Markov Chains) [14–16]. Process calculi are formal language-based descriptions that have been equipped with stochastic engines to execute models [17–19].

We consider here programming oriented languages inspired by process calculi that we believe more akin to an executable philosophy of computational modelling. Furthermore, programs written in a programming language are compact and executable textual representations so that they are well-suited for storage and to overcome the difficult manageability of graphical formalisms in the presence of large models. Finally, process calculi (and their derived languages) have the ability of handling concurrency, execution causality, non-determinism, stochastic behaviour and cooperation/competition for resources that are usual features of computational approaches.

The basic metaphor we keep as a reference to represent biological systems is inspired by [20]. A biological entity is represented by an instance of a computer program. The parallel execution of the programs corresponding to the biological entities describes the interactions between the entities through messages passing over communication channels between programs. The existence of communication channels of the right type is a measure of the specificity or affinity of biological entities. The state change of the overall system due to local interactions describes the dynamics of the represented biological systems.

The Beta Workbench (BWB) defines and implements the BlenX programming language [21] based on the process calculus Beta-binders [22]. BWB overcomes some limitations of the other process-calculi based tools [17–19], because it has been designed for biology from the beginning, instead of relying on formalisms defined for computer science applications. In the next section and in the example we highlight specific issues and we discuss the differences that make us prefer BWB.

The article is organized as follows: in the next section we describe the characterizing features of BWB and we report a comparison with the main process calculi-based approaches as well as with a representative of the mathematical/chemical approaches. Section 3 shows the features highlighted in Section 2 in a case study. The last section briefly mentions some future extensions and ongoing work on BWB.

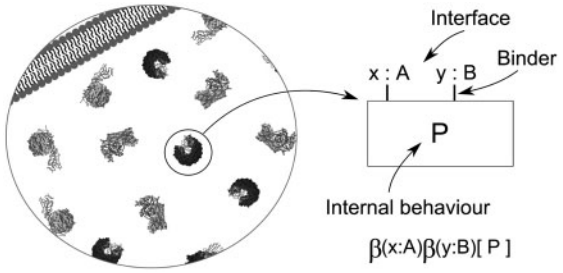


Figure 1: Boxes as abstractions of biological entities. At the bottom of the figure the textual representation of the box is given.

THE BETA WORKBENCH VERSUS OTHER COMPUTATIONAL AND MATHEMATICAL MODELLING APPROACHES

In this section, we present the main and innovative features of BWB. While discussing these features, we compare BWB with other process calculi-based approaches and with mathematical/chemical modelling to justify our choice of developing and using a new approach.

The BWB is freely available [23, 24] to model, analyse and simulate biological behaviour and it is inspired by Beta-binders [22, 25]. BWB has already been tested in modelling large biological systems like the full Nuclear Factor-kappa B (NF-κB) pathway [26] or the study of the evolutionary genesis of the Mitogen-Activated Protein (MAP) kinase (MAPK) cascade [27]. BWB has also been used as a target environment to compile a narrative-like specification of biological systems very close to natural language and exemplified on the Gp130/Signal Transducers and Activator of Transcription (STAT) signalling pathway [28].

The main goal of BWB is to facilitate and engineer the model-building process of biological systems at different levels of abstractions. The model design provides a static description of the system by just listing the components of the initial state with their amount and specificity for interaction. BWB allows the user to or *boxes* to represent biological species both in graphical and textual form (Figure 1, where B is a box). Any of the two representations can be automatically translated into the other so that the user can select the favoured specification approach. We interpret *biological entities* as the components that interact to accomplish some biological function: for example, proteins, enzymes, organic or inorganic compounds as well as cells or tissues. *Boxes* have well-defined interaction sites, called *binders*, and an internal

structure (the code needed to handle the stimuli received along the interfaces and implement the responses), such as that of biological entities. The binders represent, for example, protein domains or cell receptors and the internal program of a box codifies the response to an external stimulus.

We now discuss the features of BWB that distinguish it from other process calculi tools [17–19] and from mathematical/chemical modelling:

(i) Typed and dynamically varying interfaces of biological components: the status of an interface (*binder*) can be available/active, hidden (for instance when a 3D structure changes and it hides a domain of a protein from further interactions) or complexed (when the binder is bound to the binder of another entity). Furthermore any interface is equipped with a type that describes its properties. Primitives are available to dynamically hide or unhide binders, to create new binders and to change the type of a binder to vary its properties. This feature is not available in other process calculi-based tools.

(ii) Sensitivity-based interaction: quantities associated to binders stochastically determine the possibility of interaction between two entities. We release the key–lock interaction mechanism based on the notion of exact complement of communication channel names (in fact this is an inheritance from computer science modelling where two programs can interact only if they know the exact address of the interacting partners), and implemented in all the other process calculi tools. This allows us to avoid any global policy on the usage of names in order to make components interact, i.e. we do not need centralized authorities that decide how to name entities or interfaces. We will see in the next section that this feature is essential in order to move towards incremental and distributed model building as well as composition of models.

(iii) One-to-one correspondence between biological components and boxes specified in the model: a biological component that can be in n different states is just a box in our approach, differently from mathematical/chemical modelling in which n variables are needed to represent the n different states.

(iv) Description of complexes and dynamic generation of complexes: BWB allows the user to input complexes (set of components bound together through specific interfaces) from the beginning or it can generate complexes during the simulation relying on the specification of the components and on the complexation affinities of their interfaces. This feature highly reduces the number of components to be specified in the initial state because the products of interactions are generated by the execution and they need not to be described initially. The high parallelism of the execution will then show all the possible scenarios. This feature is an improvement over other process calculi-based approaches and also relaxes the assumption of mathematical/chemical modelling that imposes the specifications of all the species, complexes and their states (variables) in the initial state (set of equations).

(v) Spatial information: types of binders can be used to model compartments or locations as well. A special interface $\{\text{loc}: T\}$ can be associated with any box and T can vary over the relevant compartments and locations of interest for the system at hand. The primitives available to change the types of binders can then simulate translocation or movement. Conditional checks on the special binder loc may or may not enable interactions over the other interfaces of the same bio-process. For example, we could allow an interaction only if the two partner entities are in the same compartment. We used this feature in modelling the pathway Gp130/STAT in [28]. To the best of our knowledge no other computational modelling approach that supports spatial information has an associated software implementation. Some theoretical extensions of process calculi have been instead defined to cope with spatial modelling [29, 30].

(vi) Hybrid parameter specification: the specification of the quantitative parameters that drive the simulation and the dynamic behaviour of the system can be done also through named general functions computed on the global state of the system at a given time. Besides global functions, BWB also allows the user to define state variables that can be inspected and plotted or used as triggering conditions at any point of time during the execution of the model. This is a unique feature of BWB with respect to other process calculi-based tools. Actually, quantities specified through named functions make our approach more general than pure Gillespie-based approaches because rates are not constant and

associated to interaction channel statically, but can be dynamically computed during the simulation as functions of (possibly global) variables like, e.g. concentrations of other species or mathematical functions of concentrations at that point of time. This feature allows the user to recover the ODE-like modelling principles once all the reactions are specified as global conditions that trigger specific events (see below). In short, the hybrid approach of BWB provides the user with a higher level of flexibility than similar proposals and it can be used to address a larger selection of problems (see next section).

(vii) Events: global conditions can be expressed by the amount of components or complexes in the system at a given time, by the simulation time or simulation steps. Conditions trigger the enabling of particular actions called events that are then stochastically selected including them in the set of standard interaction-enabled actions. Events are used to remove or inject entities from/into the system, to join two entities into a single one, or to split an entity into two entities. This feature is essential to program perturbation of the systems triggered by particular conditions emerging during simulation and to observe how the overall behaviour is affected. An example could be the knock-out of a gene at a given time. Another possible use of events is to program sensitivity analysis of the system into the model driven by dynamic conditions emerging during simulation. Events are essential if we want to develop an in-silico lab. No other process calculi-based tool supports events in the manner described above.

(viii) De-coupling the qualitative description of the model from the quantities needed to drive execution: this novel feature with respect to computational as well as mathematical modelling approaches makes BWB suitable to exploit model composition. In fact, we only need to add a few values in the affinity definition of the interfaces of components coming from different models to let them interact (see next section for an example of easy composition of MAPK and extracellular signal-regulated kinase (ERK) pathways). Note that global conditions on concentration or structure of species are not affected by adding new species. If we want conditions on the new boxes as well, we can simply add new events to the list.

(ix) Markov chain generation: the automatic generation of all the states reachable from the initial configuration of a system (Figure 3) allows us to produce the Continuous Time Markov Chain (CTMC) describing the stochastic evolution of the biological system. When the system is not finite state, in order to deal with the problem of finiteness we put some constraints on the CTMC generation (e.g. limitation on the concentration of species and limitation on the global number of states of the CTMC). This feature allows the user to rely on classical numerical analysis techniques, to check the properties of the system using logical characterizations and tools like [14], or even to use mathematical/chemical modelling tools because we recover the whole set of reactions that govern the dynamics of the system.

The usage of the features listed above is shown in an example in the next section.

The main obstacle for the use of computational modelling in biology is the need to be able to manage formal computer science theories. We work on this issue by hiding as many formal details from the user as possible through providing modelling templates. In particular, the graphical specification formalisms supported by BWB, although not suitable for large systems, help in understanding the modelling principles.

The model is then compiled into an executable internal representation optimized to run a variant of the Gillespie's algorithm [31] similar to the Next Reaction Method [32]. The execution of a model produces the time course concentrations of the components and a reaction graph that allows hierarchical structuring of events and interactive exploration of their causal relationships. This last feature is essential when examining a system in order to infer new hypotheses that could lead to new experiments as advocated by the iterative cycle of systems biology. The set of output visualization facilities and their full interconnection is extremely valuable in inferring properties of the modelled system.

We end this section by comparing the BWB with respect to other process calculi-based tools (BioSPI [17] version 2.2.3 and SPiM [18] version 0.05) and a representative of the mathematical modelling approaches (Dizzy [3] version 1.11.4). We make the comparison for increasing sizes of the model to check how the performance of the approaches compares with respect to the complexity of the

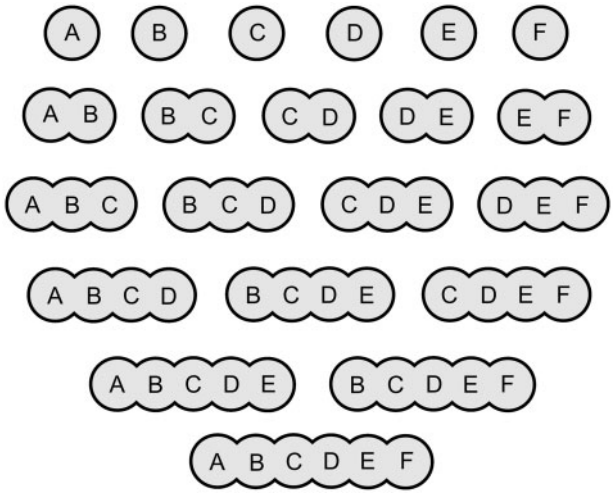


Figure 2: Multiple complexation scenario. Species A, B, C, D, E and F can bind together through several intermediate products to form the complex ABCDEF. All the reactions are reversible.

Table 1: Simulation time results for the multiple complexation example

	Dizzy (s)	BWB (s) (complexes)	BWB (s) (events)	SpiM (s)	BioSPI (s)
1	<1	<1	<1	1	5
10	<1	2.9	<1	4.2	39
100	3.6	44.7	10.1	31.3	556
1000	36.4	509.2	106.8	309.5	*

We fix the time limit simulation parameter to 1000 s and we run simulations changing the initial population of entities. Cells marked with * show that the corresponding tool has not finished the simulation within 30 min.

models. We consider both the time needed to execute the sample models and their size in terms of the elements to be specified in the initial state. The comparison has been performed on a Dual-Core AMD Opteron^{TM} 64 bit Processor 1218 2.60 GHz, 2.00 GB of RAM on a Windows XP operating system. The examples we selected are models which permit us to fully exploit the peculiarities of BWB. The first example is related to multiple complexation. Figure 2 shows species A, B, C, D, E and F that can bind together through several intermediate products to form the complex ABCDEF.

This example allows us to show how the usage of complexes can help in reducing the model size. The complete specification with chemical reactions requires the definition of 21 species (6 initial species and 15 complexes) and 70 reactions of complex formation and complex breaking. Similarly, the

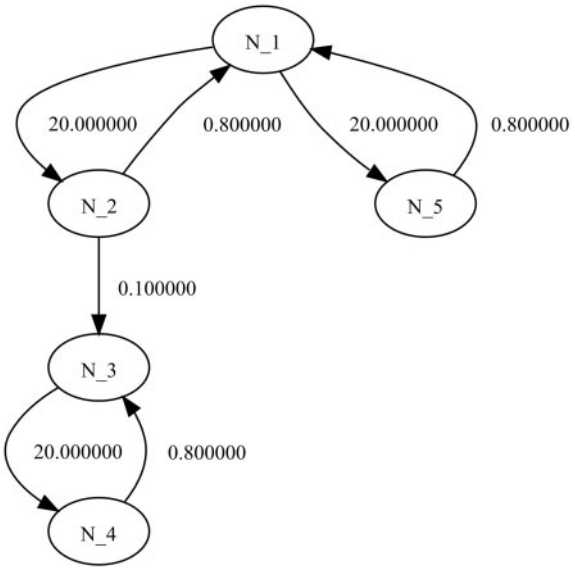


Figure 3: Continuous Time Markov chain of the enzymatic reaction with inhibitor. Node N.1 corresponds to the initial state with concentrations of one *E*, one *S* and one *I*. Node N.2 corresponds to the state in which complex *ES* has been formed, while node N.3 corresponds to the state in which the substrate *S* has been transformed in the product *P*. Nodes N.4 and N.5 correspond to the states in which complex *EI* is present. Numbers on edges represent overall stochastic rates associated to state changes; each number corresponds to the sum of the stochastic rates associated to reactions that causes the specific state change.

process algebra specifications for BioSPI and SPiM requires the definition of 21 processes and at most 70 reaction channels (depending on the rates value, the number of reaction channels can be reduced; we assume all the rates are equal to 1.0 and hence we reduce the number of channels to 50). We build two models in BWB: the first model is based on events and the second one is based on complexes. The model with events has the same order of complexity of the other specifications (21 boxes and 70 events), while the model based on complexes requires only the definition of 6 boxes (species A, B, . . . , F), and the specification of 5 binder affinities (any species can bind with the following one). The model size of the BWB specification with complexes grows linearly with the number of the initial species, while the size of all the other specifications has a quadratic growth. The time performances of the tools on the considered models (Table 1) show that the BWB simulator is faster than all the other process calculi-based tools if we consider the model with events, and slower than Dizzy. As expected, the BWB simulator is slower

than SPiM and Dizzy when considering the specification with complexes, because of the time needed to manage the on-the-fly generation of complexes. However, this comparison suggests us that our modelling approach with a combined use of complexes and events is a good compromise between model size, scalability and efficiency in simulation.

The second example that we consider is an enzymatic reaction $E + S \underset{d}{\overset{a}{\rightleftarrows}} ES \xrightarrow{k} E + P$ with inhibitor $E + I \underset{f}{\overset{e}{\rightleftarrows}} EI$ (Figure 3).

To specify the intermediate complexes *ES* and *EI* in BioSPI and SPiM, we can use special and tricky features of process calculi usually by called private names. This allows us to reduce the size of the model, at the price of the simulation time courses of the concentrations of intermediate complexes not being able to be plotted. On the contrary, the primitives for complexation and decomplexation available in BWB allow us to obtain a compact specification which also runs faster than the ones obtained with private names for BioSPI and SPiM. Furthermore, complexes are first class objects in our formalism, so they can be recognized and traced by the system during the simulation in an efficient way. Finally (last row of Table 2), BWB can handle even larger models than SPiM and BioSPI.

The considered tools implement the simulation loop with different methods, all derived from the Gillespie Stochastic Selection Algorithm [31]. One of the most efficient methods is the *Next Reaction Method* by Gibson and Bruck [32] (In computer science notation, the complexity of the method is $O(\log r)$, where r is the number of reactions in the system); we implemented a variant of this method, where the *dependency graph* data structure is replaced by a *hash map* data structure that can be updated and queried in amortized constant time. As previously mentioned, another difference of our approach is the ability to count species and complexes on-the-fly with a linear cost in the number of species; this allows us to reduce considerably the number of reactions in the system and hence to improve the implementation efficiency with respect to other process-calculi approaches, where reaction channels are created between every entity in the system. On the other hand, this on-the-fly check adds some complexity with respect to the original *Next Reaction Method* (In computer science notation, the complexity of our method is $O(k + \log r)$, where r is the number of reactions and k the number of species in the current state of the system; k is always smaller than r).

For further analysis of the main differences with non-process calculi-based computational approaches we refer the reader to [33], and for a comparison of process calculi-based solutions we refer the reader to [34].

Table 2: Simulation time results for the enzymatic reaction with inhibitor example

	Dizzy (s)	BWB (s) (complexes)	SpiM (s) (private names)	BioSPI (s)
10	<1	<1	1.2	3
100	<1	1.1	5.4	33
1000	<1	11.4	198.8	329
10 000	4.4	114.65	*	*

We fix the time limit simulation parameter to 500 s and we run simulations changing the initial population of entities. Cells marked with * show that the corresponding tool has not finished the simulation within 30 min.

AN EXAMPLE

We introduce the model-building process, simulation and analysis carried out through our BWB using a simple but complete model of the mitogen-activated protein kinase pathway, a highly conserved series of three protein kinases implicated in diverse biological processes (Figure 4).

A general model for the core MAPK cascade is described in [35]: it is built using 10 chemical equations of the form $E + S \underset{d}{\overset{a}{\rightleftarrows}} ES \xrightarrow{k} E + P$, for a total of 30 single reactions (Table 3).

KKK denotes Mitogen-Activated Protein (MAP) Kinase Kinase Kinase (MAPKKK), KK denotes Mitogen-Activated Protein Kinase Kinase (MAPKK) and K denotes MAPK. The signal Start transforms KKK to KKKP, which in turn transforms KK to KKP to KKPP, which in turn transforms K to KP to KPP. When an input Start is added, the output of KPP increases rapidly. The transformations in the reverse direction are the result of the signal End, the KKPase and the KPase phosphatases. In particular, by removing the signal Start, the output level of KPP reverts back to zero.

We built the same model presented in Table 1 and in Figure 4 with BWB by creating seven boxes:

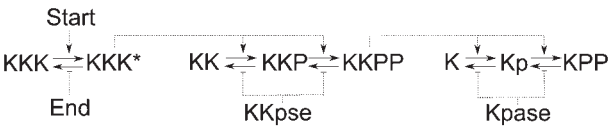


Figure 4: The MAPK biochemical pathway.

one for each kinases, two signal molecules (Start, End) and two phosphatases. Each box must have one or more binders to interact with other boxes. In our example, signal molecules and phosphatases will have a single *binder*, representing the capability of binding in the former case and the enzyme active site in the latter. The three kinases each have two binders: one represents the phosphorylation site, the other one the kinase domain.

Compatibility and strength of interaction between binders are declared using real numbers (affinities) that represent rates for specific actions based on mass action laws, or arbitrary functions. Functions are especially useful when a box represents a higher aggregation entity (e.g. a cell). In this case the input–output relation can be more non-linear, such as the sigmoidal dose responses for signalling molecules.

In the core MAPK cascade, kinases react according to mass action type rates. In this case, we specify three affinities for every pair of binders, i.e. for complex (*bind*), decomplex (*unbind*) and communication (*inter*) actions. When bind and unbind affinities are specified, complex and decomplex operations are enabled to attach and detach the corresponding boxes through the interaction sites. The biological counterpart is the binding of a ligand to a receptor, or of an enzyme to a substrate. For instance, affinities are set so that Start can bind to KKK, and when bound it can perform a reaction with a certain rate. We use hidden binders to model inactive domains, de-phosphorilated proteins, methylated receptors and so on. Initially, KKK is inactive, so we set its *p* binder—of type KKKkase—as hidden. The model for the core MAPK cascade is shown in Figure 5.

To check the compositionality of the BWB, we build two well-known models of the EGF (epidermal growth factor) -activated ERK cascade, and we compose each of them with the core

Table 3: The 10 chemical equations of the MAPK model in [35], were it is used to derive a system of 25 mathematical equations (18 ODEs plus 7 conservation equations)

(1) $KKK + Start \underset{d1}{\overset{a1}{\rightleftarrows}} KKK \cdot Start \xrightarrow{k1} KKK^* + Start$

(2) $KKK^* + End \underset{d2}{\overset{a2}{\rightleftarrows}} KKK \cdot End \xrightarrow{k2} KKK + End$

(3) $KKK^* + KK \underset{d3}{\overset{a3}{\rightleftarrows}} KKK^* \cdot KK \xrightarrow{k3} KKK^* + KKP$

(4) $KKP + KKPse \underset{d4}{\overset{a4}{\rightleftarrows}} KKP \cdot KKPse \xrightarrow{k4} KK + KKPse$

(5) $KKK^* + KKP \underset{d5}{\overset{a5}{\rightleftarrows}} KKK^* \cdot KKP \xrightarrow{k5} KKK^* + KKPP$

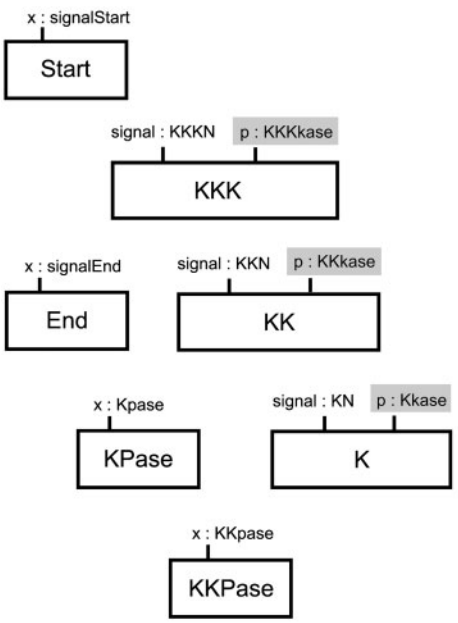
(6) $KKPP + KKPse \underset{d6}{\overset{a6}{\rightleftarrows}} KKPP \cdot KKPse \xrightarrow{k6} KKP + KKPse$

(7) $KKPP + K \underset{d7}{\overset{a7}{\rightleftarrows}} KKPP \cdot K \xrightarrow{k7} KKPP + KP$

(8) $KP + KPase \underset{d8}{\overset{a8}{\rightleftarrows}} KP \cdot KPase \xrightarrow{k6} K + KPase$

(9) $KKPP + KP \underset{d9}{\overset{a9}{\rightleftarrows}} KKPP \cdot KP \xrightarrow{k9} KKPP + KPP$

(10) $KPP + KPase \underset{d10}{\overset{a10}{\rightleftarrows}} KPP \cdot KPase \xrightarrow{k10} KP + KPase$




```
[steps = 15000, delta = 1.0]

<<BASERATE: inf>>

let Start: bproc = #(x:1,signalStart)
   [ !x<add>.nil ];

let End: bproc = #(x:1,signalEnd)
   [ !x<remove>.nil ];

//state2: back to KP
let pKPP : pproc =
signal{}.ch(signal,UN).hide(p).ch(signal,KP).
state1<e>.nil;
//state one
let pKP : pproc =
   !state1{}.
      (signal{what}.what<e>.nil |
         (add{}.ch(signal,UN).ch(signal,KPP).
          unhide(p).pKPP +
remove{}.ch(signal,UN).ch(signal,KN).
x<e>.nil));
//base
let pK : pproc =
   (signal{}.ch(signal,UN).ch(signal,KP).
    state1<e>.nil);

let K : bproc = #h(p:1,Kkase), #(signal:1,KN)
   [ !x{}.pK | pK | pKP | !p<add>.nil ];
```

Figure 5: Our model of the core MAPK pathway.

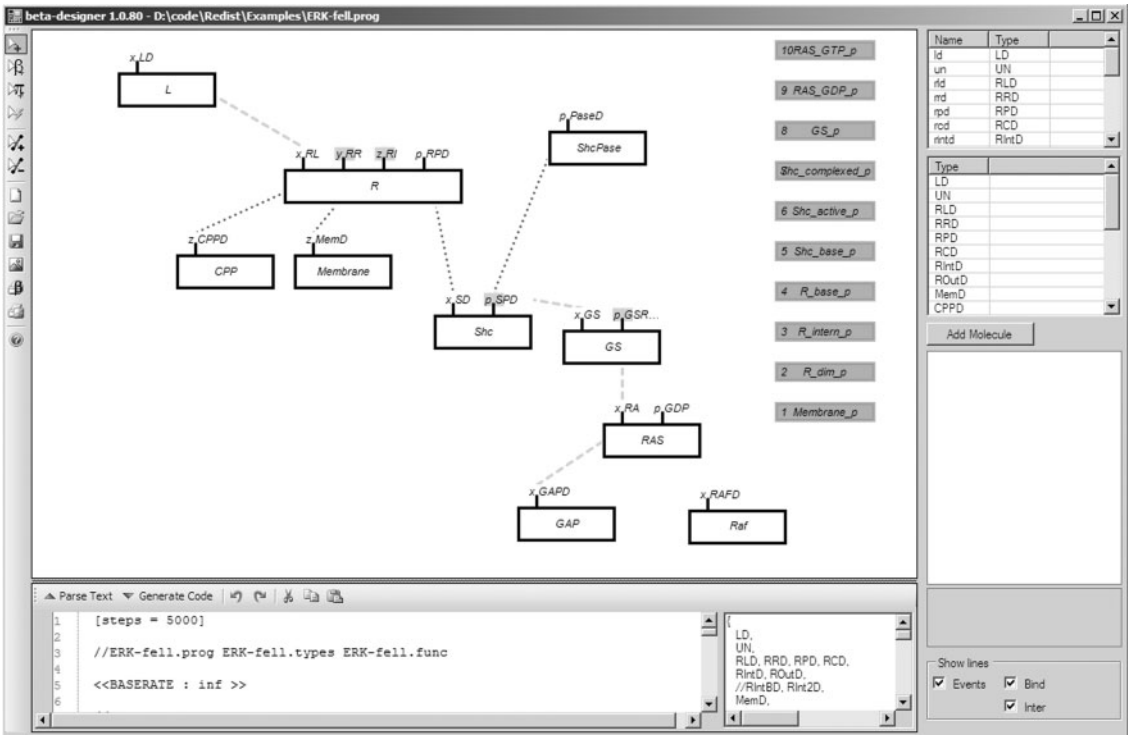


Figure 6: The EGF-activated ERK cascade model [37] implemented in our language.

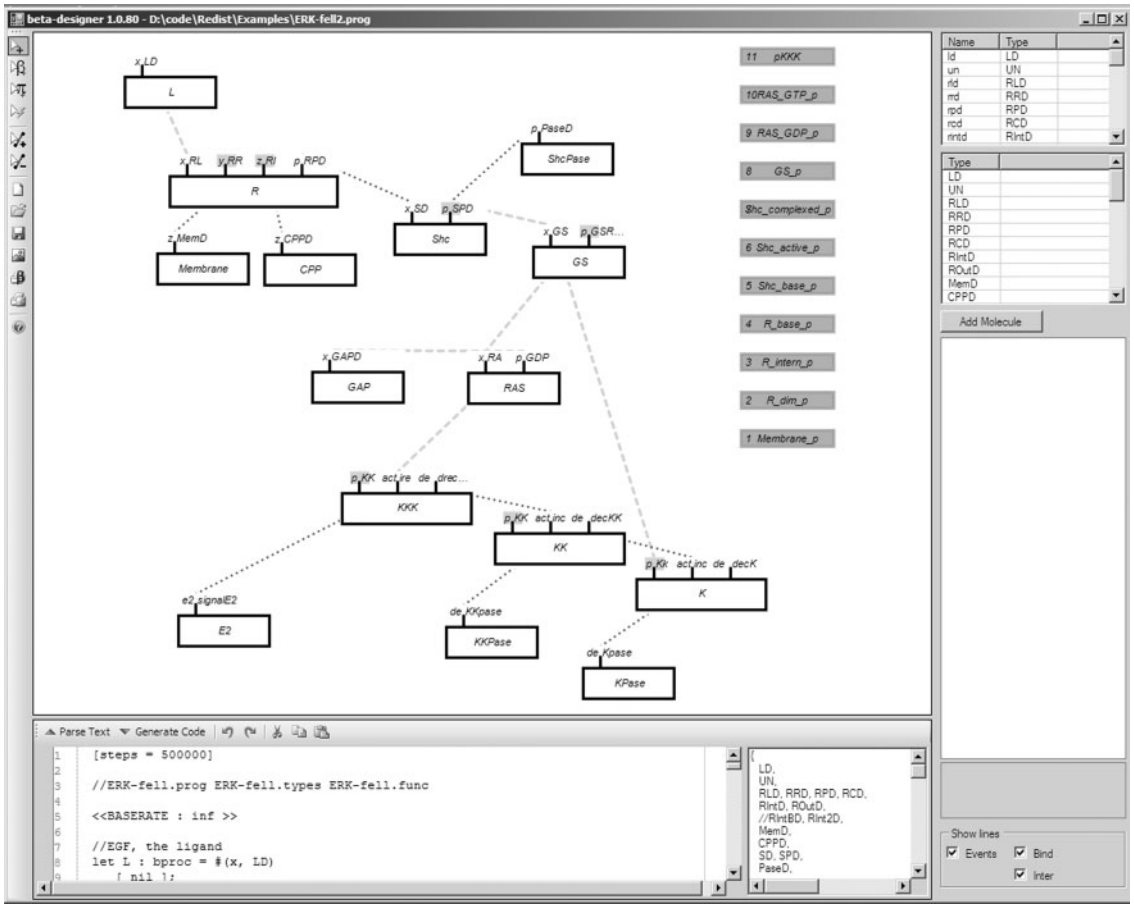


Figure 7: The EGF-activated ERK cascade model composed with core MAPK. Note the feedback regulation between K and GS.

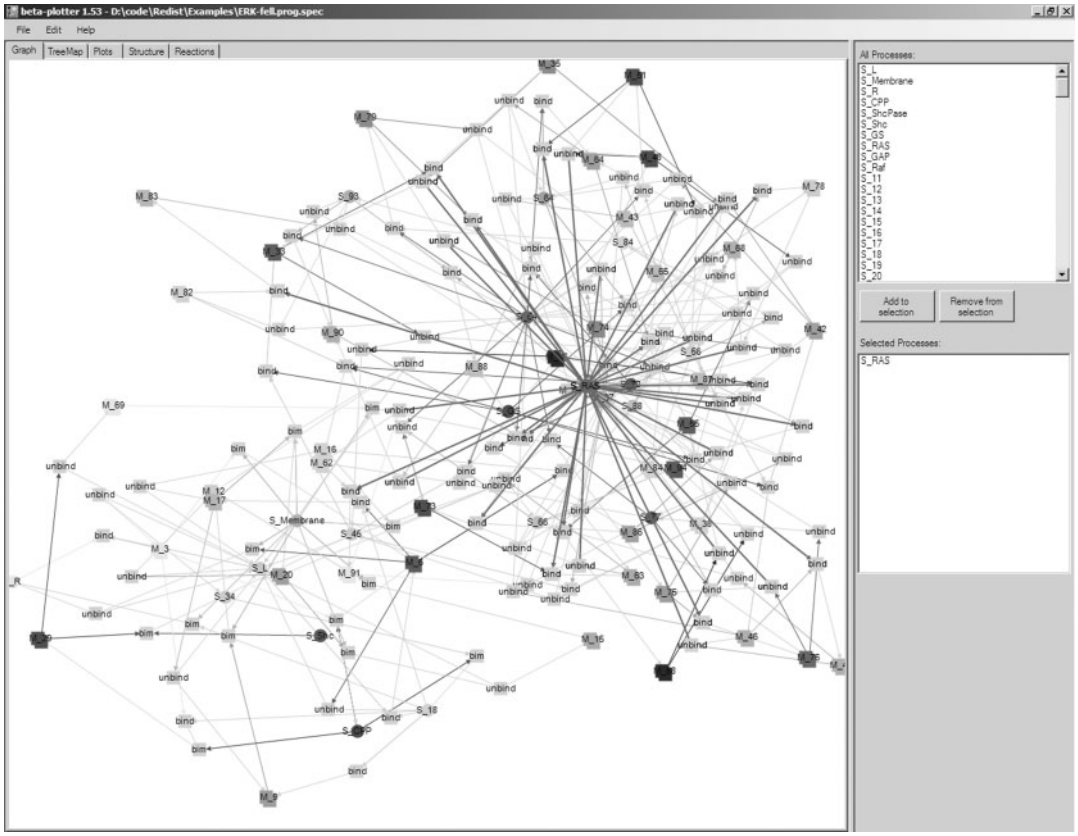


Figure 8: The output of the simulation of our MAPK model in the BWB Plotter. Nodes in the graph are the different states that biological entities modelled by boxes in Figure 7 traverse during the simulation.

MAPK model. The two models are based on [36] and [37] (Figure 6).

Plugging the core MAPK cascade into these two models is extremely simple: we only need to delete the Signal box and insert it in the detailed model, setting the affinities between the RASD binder of the Ras protein and KKK to the appropriate kinetic rate. This is achieved by adding one line of code to the affinity definition file. The Fell's model also requires a feedback regulation between K and Gs; the addition of another affinity between the KN and GSD domains is sufficient to accomplish this task (Figure 7). The possibility to plug a model into an existing one by composition, exchanging part of them as we did, can be very useful in building large scale models.

Note that a single bio-process is specified for every biological component. The execution of the model generates the different states of the same component (active forms, bound entities, receptor dimers, internalized complexes and so on). The full models are available at [23].

The language allows us to compress a part or a whole pathway with a *one-box* simplified model. In this case a box represents a higher aggregation entity and the input–output relation can be non-linear.

Arbitrary functions that specify affinities between binders allow us to effectively compress the MAPK pathway modelling and its sigmoid response as a Hill function [35]. In this compact representation, the starting signal is directly connected to a box through the following user-defined rate function:

```
let fHill: function = (pow (|Kact|, n)/
(pow(|Kact|, n) + pow (Omega, n)));
```

This *one-box* model can perfectly substitute the complete model, and can also be composed with a more detailed or higher level model.

BWB allows the user to examine the results of simulations by visualizing the entities, states and complexes generated, the variation in concentration of the various entities during simulation time, the reactions that took place and the causal relationship between entities and reactions. Figure 8 shows the graph of the possible reactions between components generated by the execution of the complete

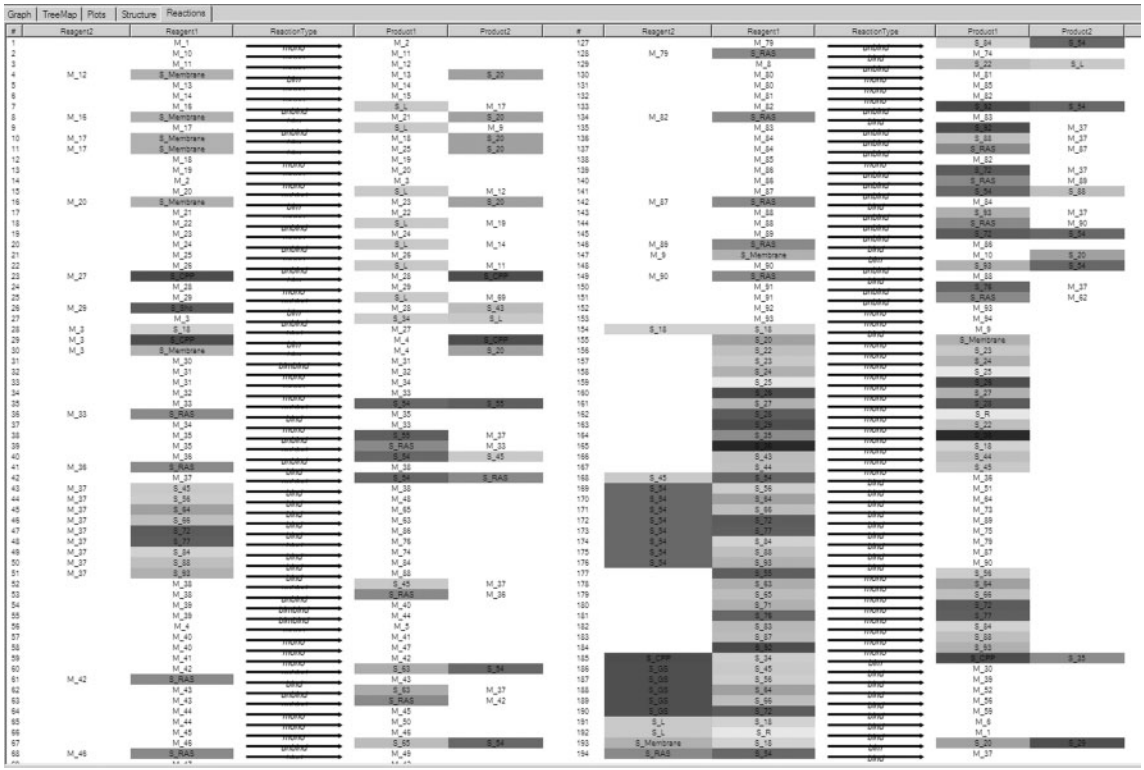


Figure 9: List of reactions of the model in Figure 7.

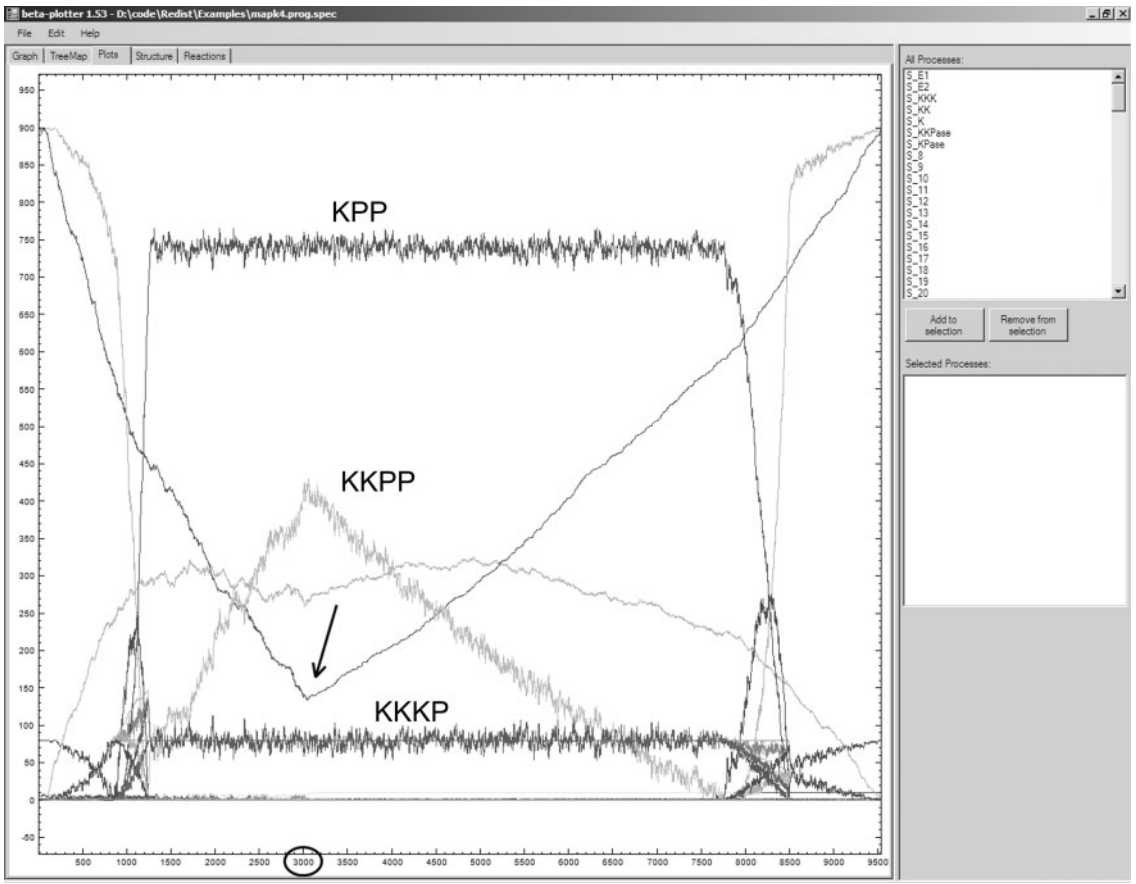


Figure 10: Variation in the concentration of entities over time of the complete model.

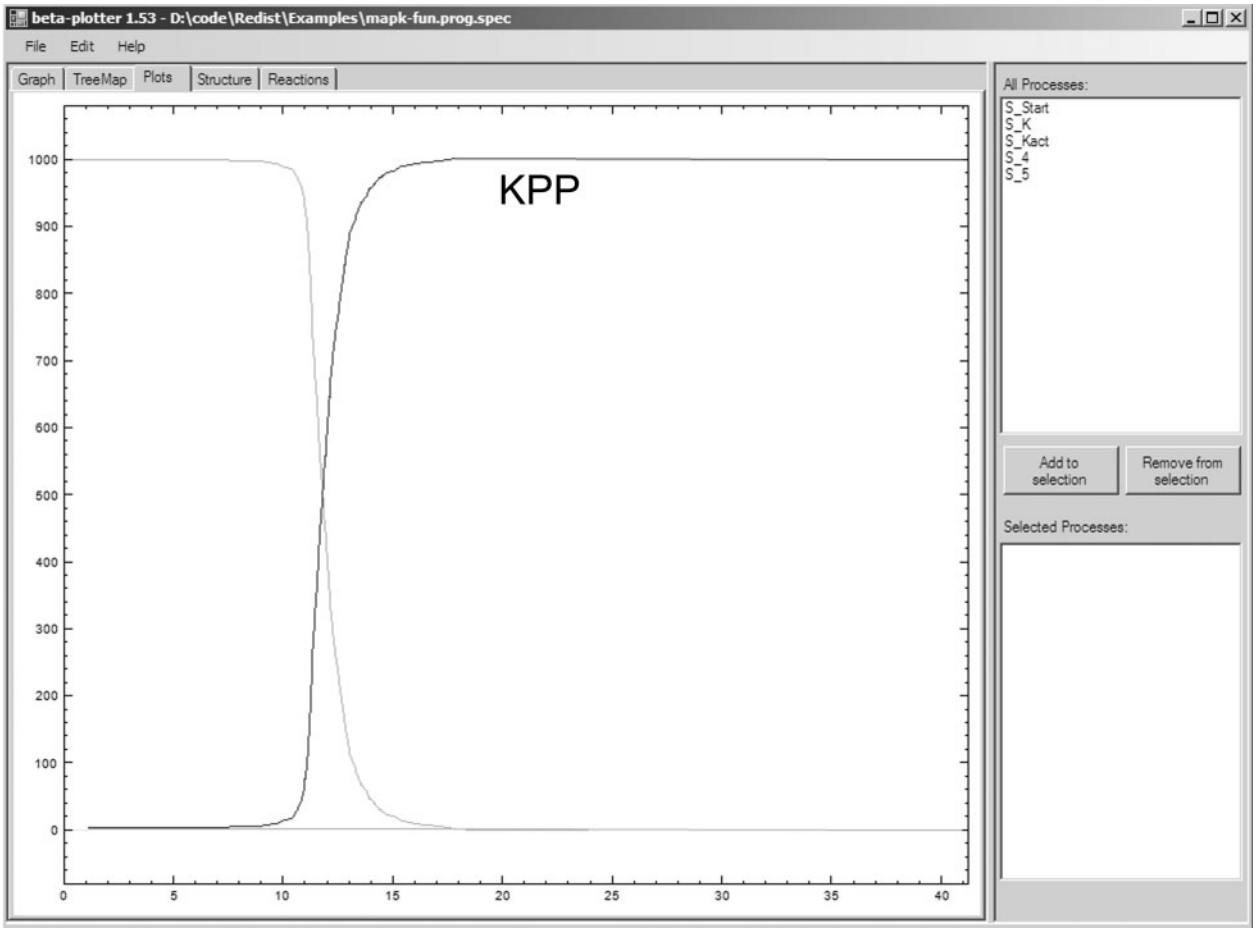


Figure 11: Variation in the concentration of entities over time of a one-box simplified model, where the sigmoidal response is modelled as an Hill function, as discussed in [35].

ERK-MAPK model in Figure 7. Note that a fairly simple and compact model generates a quite complex reaction graph that in other approaches need to be fully specified. Reactions can also be automatically generated and displayed by BWB in the classic way (Figure 9). Note that we are considering binary interactions as they happen at the lowest level, so we only need two reactants and two products at most. An extension to multi-reactions can be based on transactions mechanisms as described in [38]. Transactions would allow us to overcome one of the main limitations of process calculi in representing chemical reactions.

Figures 10 and 11 depict the variation of concentrations of the entities in the system as time passes. Note the inversion in the tendency of the various curves at step 3000, when the Start signal is removed from the system with an event. The behaviour of the system is in agreement with the conventional ODE model and wet-lab experiments presented in [35–37].

CONCLUSIONS

We presented the BWB, a tool made to build models of biological systems, to simulate their dynamic behaviour and to inspect and query the outcome of the simulations. We discussed the main and innovative features of the approach compared to other process calculi tools and mathematical/chemical modelling approaches. We also reported scalability and performance considerations. An example based on the composition of the MAPK and ERK model built separately shows how large scale models could be implemented easily.

Future work aims at making the writing of models simpler, through a collection of templates that encode common biological behaviours, such as enzymatic behaviour, internal structure of kinases and phosphatases and so on. In this way the user can build a complete and complex model through a simple point-and-click, while letting more advanced users write and optimize code for specific behaviour. We are also enhancing visualization and analysis on

the generated reaction graph. The tool is actively developed, maintained and expanded: we are creating different simulation algorithms to speed up simulation and management of the dynamic generation of complexes, including deterministic approaches, and algorithms for rate inference from experimental data and for 3D diffusion mechanisms. Finally, we are implementing export/import feature to connect our tool to public databases and standardization efforts like SBML in order to re-use and share model designs and analyses.

We hope that the development of a general tool to carry out *in silico* experiments can enhance the usage of formal computational models in the life science community: a step that we find essential in order for the systems biology community to grow.

Key Points

- We introduce a scalable tool to model, simulate and analyse biological systems. We compare the proposed approach with other process calculi-based tools and with the mathematical/chemical approach. We consider modelling issues, analysis capabilities and performance scaling according to the size of the system in hand.
- The peculiar features of the proposed tool are illustrated through biological examples. Modularity and compositionality are discussed by showing how to merge different pathways in an easy way. In particular we stress that the tool was designed for biology from the beginning, rather than checking whether computer science approaches apply to biology.
- The new approach is different from mathematical modelling because it relies on executable specifications of systems. The execution provides the time course history of components and hence the models are not just simply solvable as systems of equations. Furthermore, the representation of biological systems is done though the definition of abstract entities that turns out to be in a one-to-one correspondence with concrete biological components independently of the number of states that they can pass through. In fact, the intermediate product of reactions is generated automatically by the execution of the model without the need of specifying an object or a variable for them in the initial state.
- The modelling approach based on programming language theory allows us to implement on top of our models a set of analyses that extends simulations such as causality between interactions, spatial information of interactions, etc.

References

1. Kitano H. *Foundations of Systems Biology*. Cambridge, MA, USA: MIT Press, 2001.
2. Adalsteinsson D, McMillen D, Elston T. Biochemical network stochastic simulator (BioNetS): software for stochastic modeling of biochemical networks. *BMC Bioinformatics* 2004;**5**:24.
3. Ramsey S, Orrell D, Bolouri H. Dizzy: stochastic simulation of large-scale genetic regulatory networks. *J Bioinform Comput Biol* 2005;**3**:415–36.
4. Hoops S, Sahle S, Gauges R, *et al*. COPASI – a Complex pathway simulator. *Bioinformatics* 2006; **22**:3067–74.
5. Hucka M, Sauro H, Finney A, et al. The ERATO systems biology workbench: an integrated environment for multi-scale and multitheoretic simulations in systems biology. In: Hiroaki Kitano (ed). *Foundations of Systems Biology*. Cambridge, MA: MIT Press, 2001.
6. Fisher J, Henzinger TA. Executable cell biology. *Nat Biotechnol* 2007;**25**:1239–49.
7. Kauffman SA. Metabolic stability and epigenesis in randomly constructed genetic nets. *J Theor Biol* 1969; **22**:437–67.
8. Shmulevich I, Dougherty E, Kim S, *et al*. Probabilistic boolean networks: a rule-based uncertainty model for gene regulatory networks. *Bioinformatics* 2002;**18**:261–74.
9. Nagasaki M, Doi A, Matsuno H, *et al*. A versatile petri net based architecture for modeling and simulation of complex biological processes. *Genome Inform* 2004;**15**:180–97.
10. Dill D, Knapp M, Gage P, *et al*. The pathalyzer: a tool for analysis of signal transduction pathways. In: *Proceedings of Systems Biology and Regulatory Genomics 2005, LNCS* 2005; **4023**:11–22.
11. Alla H, David R. Continuous and hybrid petri nets. *J Circuits Sys Comp* 1998;**8**:159–88.
12. Marsan M, Balbo G, Conte G, *et al*. *Modelling with Generalized Stochastic Petri Nets*. Wiley Series in Parallel Computing. New York: John Wiley & Sons.
13. Efroni S, Harel D, Cohen IR. Reactive animation: realistic modeling of complex dynamic systems. *Computer* 2005; **38**:38–47.
14. Heath J, Kwiatkowska M, Norman G, *et al*. probabilistic model checking of complex biological pathways. In: *Proceedings of the International Workshop on Computational Methods in Systems Biology (CMSB 2006)*, LNCS 2006; **4210**:32–47.
15. Ciesinski F, Grossner F. On probabilistic computation tree logic. *Validation of Stochastic Systems*. Series: Lecture Notes in Computer Science, Vol. 2925. Heidelberg, Germany: Springer, 2004,147–188.
16. Aziz A, Sanwal K, Singhal V, *et al*. Model-checking continuous-time Markov chains. *ACM T Comput Log* 2000;**1**:162–70.
17. Priami C, Regev A, Silverman W, *et al*. Application of a stochastic name passing calculus to representation and simulation of molecular processes. *Inform. Process. Lett.* 2001;**80**:25–31.
18. Phillips A, Cardelli L. A correct abstract machine for the stochastic pi-calculus. In: *Proceedings of Concurrent Models in Molecular Biology (Bioconcur'04)*, affiliated with CONCUR'04. 2004.
19. Gilmore S, Hillston J. The PEPA Workbench: a tool to support a process algebra-based approach to performance modelling. In: *Proceedings of the Seventh International Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, *LNCS* 1994;**794**:353–68.
20. Regev A, Shapiro E. Cellular abstractions: cells as computation. *Nature* 2002;**419**:334–43.
21. Dematté L, Priami C, Romanel A. The BlenX Language: a tutorial. In: *Proceedings of SFM 2008, LNCS* 2008; **5016**:313–65.

22. Priami C, Quaglia P. Beta binders for biological interactions. In: *Proceedings of the second International Workshop on Computational Methods in Systems Biology (CMSB04), LNBI* 2005;**3082**:21–34.
23. Beta Workbench. http://www.cosbi.eu/Rpty_Soft_BetaWB.php (4 April 2008, date last accessed).
24. Romanel A, Dematté L, Priami C. The Beta Workbench. Technical Report TR-3-2007, The Microsoft Research - University of Trento Centre for Computational and Systems Biology, February 2007.
25. Priami C, Quaglia P. Operational patterns in beta-binders. *Transactions on Computational Systems Biology* 2005;**1**:50–65.
26. Ihekwaba AE, Larcher R, Mardare R, *et al*. Poster abstract: BetaWB: a language for modular representation of biological SYSTEMS. In: *Proceedings of The Eighth International Conference on Systems Biology*, 2007;61–62.
27. Dematté L, Priami C, Romanel A, *et al*. A formal and integrated framework to simulate evolution of biological pathways. In: *Proceedings of International Workshop on Computational Methods in Systems Biology (CMSB 2007), LNCS* 2007;**4695**:106–20.
28. Guerriero ML, Heath JK, Priami C. An automated translation from a narrative language for biological modelling into process algebra. In: *Proceedings of International Workshop on Computational Methods in Systems Biology (CMSB 2007), LNCS* 2007;**4695**:106–20.
29. Guerriero ML, Priami C, Romanel A. Modeling static biological compartments with beta-binders. In: *Proceedings of Algebraic Biology (AB2007), LNCS*. Berlin, Germany: Springer, 2007;**4545**:247–61.
30. John M, Ewald R, Uhrmacher AM. A spatial extension to the π Calculus. *Electron. Notes Theor. Comput. Sci* 2008;**194**:133–48.
31. Gillespie DT. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *J Comput Phys* 1976;**22**:403–34.
32. Gibson MA, Bruck J. Efficient exact stochastic simulation of chemical systems with many species and many channels. *J Phys Chem A* 2000;**104**:1876–89.
33. Priami C, Quaglia P. Modeling the dynamics of biosystems. *Brief Bioinform* 2004;**5**:259–69.
34. Guerriero ML, Prandi D, Priami C, *et al*. Process calculi abstractions for biology. Technical Report TR-13-2006, The Microsoft Research - University of Trento Centre for Computational and Systems Biology, 2006.
35. Huang CY, Ferrell JE. Ultrasensitivity in the mitogen-activated protein kinase cascade. *Proc Natl Acad Sci USA* 1996;**93**:10078–83.
36. Kholodenko BN, Demin OV, Moehren G, *et al*. Quantification of short term signaling by the epidermal growth factor receptor. *J Biol Chem* 1999;**274**:30169–81.
37. Brightman FA, Fell DA. Differential feedback regulation of the MAPK cascade underlies the quantitative differences in EGF and NGF signalling in PC12 cells. *FEBS Lett* 2000; **482**:169–74.
38. Ciocchetta F, Priami C. Biological transactions for quantitative models. *Electron. Notes Theor. Comput. Sci* 2007;**171**:55–67.