Modelling biological pathway dynamics with Timed Automata

Stefano Schivo*, Jetse Scholma*, Brend Wanders, Ricardo A. Urquidi Camacho,
Paul E. van der Vet, Marcel Karperien, Rom Langerak, Jaco van de Pol and Janine N. Post

***Abstract*—Living cells are constantly subjected to a plethora of environmental stimuli that require integration into an appropriate cellular response. This integration takes place through signal transduction events that form tightly interconnected networks. The understanding of these networks requires to capture their dynamics through computational support and models. ANIMO (Analysis of Networks with Interactive MOdelling) is a tool that enables construction and exploration of executable models of biological networks, helping to derive hypotheses and to plan wet-lab experiments. The tool is based on the formalism of Timed Automata, which can be analysed via the UPPAAL model checker. Thanks to Timed Automata, we can provide a formal semantics for the domain-specific language used to represent signalling networks. This enforces precision and uniformity in the definition of signalling pathways, contributing to the integration of isolated signalling events into complex network models. We propose an approach to discretization of reaction kinetics that allows us to efficiently use UPPAAL as the computational engine to explore the dynamic behaviour of the network of interest. A user-friendly interface hides the use of Timed Automata from the user, while keeping the expressive power intact. Abstraction to single-parameter kinetics speeds up construction of models that remain faithful enough to provide meaningful insight. The resulting dynamic behaviour of the network components is displayed graphically, allowing for an intuitive and interactive modelling experience.**

***Index Terms*—timed automata; signalling pathway; modelling; dynamic behaviour**

I. Introduction

Systems biology aims at understanding biological systems from a holistic rather than a reductionist perspective. As the complexity of dynamic molecular interaction networks exceeds the capacity of the human mind, computational support is needed to describe and understand such networks. Large volumes of data are particularly useful when used as reference for functional models that formalize the interactions between the components of a network. Such models can assist in exploration and transfer of knowledge. Functional disease models can be combined with patient-specific expression profiles to stratify patients into groups and maximize treatment efficacy.

Executable biology is a relatively new area in systems biology and takes the approach of *in silico* mimicking biological processes of interacting network components. This paradigm offers a close match with biological intuition [1]. ODEs (ordinary differential equations) are often used to model the dynamics of biochemical processes, and the availability of supporting tools [2], [3], [4] makes their power accessible to end users. Process calculi are also classified as executable approaches, and tools based on such formalisms have been successfully used in modelling complex biological events [5], [6]. However, most of the existing modelling tools require theoretical foundations or training in mathematics, and this may hamper the widespread creation of *in silico* models within the biological community. For this reason, we developed a modelling tool centred around user-friendliness, ANIMO [7], the technical foundations of which are described in detail in this paper. The formalism of Timed Automata [8] provides an excellent starting point for construction of executable models of biological networks and enables the use of the dedicated model checking tool UPPAAL [9]. The use of Timed Automata to model molecular interactions has been explored before [10], [11], [12], and is further elaborated by us and for the first time applied in a mature modelling tool. ANIMO is implemented as a plug-in to the network visualization tool Cytoscape [13], which contributes to the user friendliness and widens ANIMO's applicability within the biological community. The modelling process is entirely performed via the user interface, and user input is automatically translated into an underlying Timed Automata model.

This paper extends on the BIBE conference paper presented in 2012 [14]. In the present work, we describe in detail how Timed Automata guards and invariants are calculated from kinetic reaction equations and how unwanted side effects of rounding to integer time units are dealt with. Furthermore, we show how the use of UPPAAL's statistical model checker [15] improves simulation performance. Finally, a newly implemented feature is presented that assists in understanding the effect of modifications to the model, by using a heat-map that shows the dynamic per-node differences between two versions of the model. To exemplify this new functionality, we provide an illustrative case study that shows the added value for performing *in silico* experiments.

After a brief introduction on the basic aspects of biological signalling networks and Timed Automata in Section II, we will explain in Section III how our modelling approach works, showing an example application in Section IV. After discussing related work in Section V, Section VI concludes the

*These authors contributed equally to this work.

S. Schivo, R. Langerak and J. van de Pol are with the Formal Methods and Tools group, Faculty of EEMCS, University of Twente, Enschede, The Netherlands.

J. Scholma, R. A. Urquidi Camacho, M. Karperien and J. N. Post are with the Developmental BioEngineering group, MIRA Institute for Biomedical Technology and Technical Medicine, Enschede, The Netherlands.

B. Wanders and P. E. van der Vet are with the Human Media Interaction group, Faculty of EEMCS, University of Twente, Enschede, The Netherlands.

We acknowledge support from Netherlands Organisation for Scientific Research (NWO) CASIMIR to J.S.

paper and gives some perspectives for future developments.

II. Preliminaries

A. Signalling pathways in biology

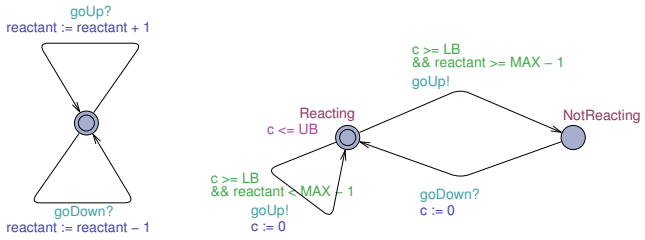
Intra-cellular signalling pathways represent the cellular events occurring when receptors are activated by specific ligands. A typical interaction occurring in a signalling pathway involves an upstream protein inducing a post-translational modification (e.g. phosphorylation) to a downstream substrate. Changing the state of a protein often results in a change in its activity. A cascade of activating reactions relays incoming signals to their downstream targets. Graphical representations of a signalling pathway typically depict proteins as nodes and reactions as edges, with $\rightarrow$ and $\dashv$ representing activation and inhibition, respectively.

Many nodes are shared by different signalling pathways and this crosstalk ties pathways into strongly interconnected networks that also contain feedback loops. However, the function of these networks does not only derive from their topology, but also from their dynamic behaviour, which cannot be deduced from traditional static representations. This is an incentive towards a method for enriching these representations with a formal description of the associated dynamics. Computational models of such complex, dynamic networks would enable *in silico* experiments, generation of hypotheses and rational design of wet-lab experiments.

B. Timed Automata

Timed Automata (TA) [8] are finite-state automata to which real-valued clocks and communication channels have been added. In particular, clocks are used to define conditions enabling transitions between locations of an automaton, or to limit the permanence in locations. These conditions are called *guards* and *invariants*, respectively. Performing a transition may require two automata to interact via *synchronization*, where each participant performs one of two complementary actions (termed *input* and *output*) on a shared communication channel. Such channels can also be defined as *broadcast*, allowing multi-part communication with one sender and many receivers. In order to obtain answers to interesting questions about the behaviour of a model, model checking [16] can be applied to a TA model, using a software tool such as UPPAAL [9].

As an example of TA, a basic model of a generic signalling network is given in Figure 1. The model represents the active fraction (further called *activity level*, or simply *activity* when not ambiguous) of a population of molecules as an integer variable (reactant, Fig. 1a), the value of which is increased (Fig. 1b) or decreased by reactions. In the example, the value of reactant is limited to the interval $[0, \mathsf{MAX}]$, and reaching a bound may cause disabling of a reaction (location NotReacting in Fig. 1b). More precise models of signalling networks also take into account the activity of upstream components when determining the availability of an activating reaction. This will be explained in the next section.



(a) Reactant (b) Activation

Fig. 1. TA templates to model a signalling pathway as represented by UPPAAL's user interface. Each automaton starts evolving from the location marked with two concentric circles. (a) This automaton updates the reactant variable whenever a reaction occurs, changing its activity level (reactant := reactant+1 increases the activity). (b) Activating reaction, which occurs when its internal clock c is inside the interval [LB, UB] (guard c >= LB and invariant c <= UB), making the target reactant's activity level increase. The current value of reactant ($>= \mathsf{MAX}-1$ or $< \mathsf{MAX}-1$) determines whether the automaton takes the upper transition or the transition at the lower left. The former ends in location NotReacting and disables the reaction because the reactant has reached its maximum activity. The latter ends in location Reacting and the reaction remains active. An inhibiting reaction can be defined in a similar way, with the effect of decreasing the reactant's activity. Shared broadcast channels goUp and goDown are used for communication between reaction and reactant automata.

III. Modelling signalling pathways

A. Abstraction and discretization

In order to provide experimental biologists with an intuitive way to formalize prior knowledge and experimental results, a key aspect is to choose a suitable abstraction level.

The first abstraction made in ANIMO is the representation of proteins in the network in terms of their activity. Distinct biological processes can lead to activation or inhibition of proteins. In ANIMO, these processes are all abstracted to reactions that change the activity of downstream reactants.

Second, we abstract from detailed elementary reactions. This is particularly useful in biology, where the construction of very detailed models is often hampered by a lack of knowledge about the exact details of the molecular mechanisms involved. Detailed representations of biological systems require detailed knowledge of reaction mechanisms as well as accompanying kinetic parameters for each individual step in the overall reaction. By choosing a higher abstraction level, elementary reaction steps can be aggregated into a single step in the model, reducing the number of parameters in the model. As such, this abstraction not only reduces the time to construct a model, it also facilitates the modelling of larger networks for which not all details of elementary reactions are known. Of course, a higher abstraction level also comes at the price of losing some descriptiveness, but we will demonstrate in Section IV that abstracted models can capture experimental data in a meaningful way.

A third level of abstraction is the discretization of activities into integer variables with a user-defined granularity, ranging from Boolean (2 levels) to almost continuous (100 levels). This discretization gives the user the flexibility to adjust the model abstraction to the level of prior knowledge, to the quality and nature of experimental data and to the biological questions at hand. It also allows control over the trade-off between level of detail and computational performance for large models.

B. Reaction kinetics

Reaction rates depend on the current activity levels of the involved reactants. For instance, a higher activity of upstream enzyme and/or a higher abundance of downstream substrate increase the reaction rate.

Different degrees of complexity are taken into account when defining reaction kinetics. Given a reaction in which A (enzyme) activates B (substrate), we define R as the time needed to increase the activity level of B by 1. R is computed using a kinetic function f that depends on the current activity levels of both A and B: $R = f(a, b)$

$$f(a,b) = \begin{cases} \frac{1}{r(a,b)} \times \textit{levelsScale} \times \textit{timeScale} & \text{if } r(a,b) \neq 0 \\ \infty & \text{otherwise} \end{cases}$$

The reaction rate r is computed from a single kinetic constant k and the activity levels of the reactants (a, b). Three kinetic scenarios are implemented :

Scenario 1: $r(a) = k \times a$: the reaction rate depends on the activity level of the upstream enzyme only.

Scenario 2: $r(a, b) = k \times a \times b$: the reaction rate depends on the activity levels of both the upstream enzyme and the downstream substrate. b represents the inactive fraction of substrate if the reaction is an activation, whereas it refers to the active fraction when the effect is inhibitory.

Scenario 3: $r(c, d) = k \times c \times d$: the reaction rate depends on the activity of two upstream components chosen by the user. This scenario performs a function analogous to an AND-gate in Boolean models.

levelsScale is a scale factor that allows to keep reaction kinetics independent from the granularity of its reactants. The value of *levelsScale* depends on the kinetic scenario used to compute r. With nA the granularity of reactant A (and similarly for other reactants), *levelsScale* is defined as:

Scenario 1: $\textit{levelsScale} = \frac{nA}{nB}$

Scenario 2: $\textit{levelsScale} = \frac{nA \times nB}{nB} = nA$

Scenario 3: $\textit{levelsScale} = \frac{nC \times nD}{nB}$, with nB the granularity of the reactant influenced by the reaction.

The numerators are used to normalize all input activity values to the $[0, 1]$ interval, which keeps the reaction kinetics independent from the input reactants' granularity. The denominator works similarly on the downstream reactant. The importance of this can be intuitively understood with a simple example. Consider an activating reaction $A \rightarrow B$, with A completely active and B completely inactive. If B has 10 granularity levels and no other reactions influence A or B, the reaction will take T seconds to completely activate B from 0 to 10 levels. When the granularity of B would be increased to 30 levels, the reaction should still take T seconds to completely activate B, bringing it from 0 to 30 levels. This implies that each reaction step that increases the activity of B by one level takes one third of the time it took in the situation with 10 activity levels for B. This scaling is accounted for by using nB as denominator in *levelsScale*.

timeScale is based on the ratio between real-life seconds and TA time units. This enables performing simulation runs using real-life units of time. Moreover, as an UPPAAL model must express all time bounds as integers, *timeScale* is automatically optimized to prevent rounding problems when R would be rounded to 0 for values < 0.5. Increasing the value for *timeScale* ensures that the smallest value for R becomes larger than 0. It is important to note that *timeScale* cannot simply be assigned an arbitrarily high value, because the resulting time constraints could overflow the maximum constant allowed in UPPAAL (currently, $2^{30} - 2$). ANIMO automatically chooses the optimum *timeScale* value by making a preliminary computation of all borderline values, i.e. the minimum and maximum time constraint for each reaction. The starting value for *timeScale* is computed as

$$\textit{timeScale} = \frac{1}{\textit{secondsPerStep}}$$

where *secondsPerStep* is the user-chosen ratio between real-life seconds and TA time units, with 1 as its default value. The resulting value is then optimized (raised or lowered) to ensure that all time bounds are inside the allowed interval $[0, 2^{30} - 2]$.

Finally, to take into account a natural variability that occurs in biological reactions in which reactants are present in low copy numbers, an uncertainty parameter can be set for each reaction. This uncertainty is translated to time intervals for model transitions, as is shown in the model in Figure 1 (cf. LB and UB time bounds). With an uncertainty value of 5%, the upper and lower bounds of R are computed as:

$$\begin{aligned} R(a,b)_{\text{lower bound}} &= f(a,b) \times 0.95 \\ R(a,b)_{\text{upper bound}} &= f(a,b) \times 1.05 \end{aligned}$$

for all values of a and b.

As we represent activity levels a, b via integer variables, there is a finite number of combinations for the values of a and b. A two-dimensional table is pre-computed, containing all possible values for R. As an example of this computation, Table I shows the lower and upper bounds table for a reaction $A \rightarrow B$, using scenario 2.

TABLE I

LOWER AND UPPER BOUNDS FOR A REACTION $A \rightarrow B$ WITH KINETIC SCENARIO 2, $k = 0.004$ (CORRESPONDING TO THE *medium* PRESET) AND 10 SECONDS PER UPPAAL TIME STEP. GRANULARITY OF A IS 5 (I.E. 5 ACTIVATION STEPS FROM 0 TO MAXIMUM ACTIVITY), GRANULARITY OF B IS 3. IN BOTH CASES, 0 MEANS COMPLETELY INACTIVE. WITH THESE SETTINGS, *levelsScale* $= 5.0$ AND *timeScale* $= 0.1$. UNCERTAINTY IS SET AT 5%. RED NUMBERS ARE LOWER BOUNDS, CYAN NUMBERS ARE UPPER BOUNDS.

B \ A	0	1	2	3	4	5
0	∞	[40, 44]	[20, 22]	[13, 15]	[10, 11]	[8, 9]
1	∞	[59, 66]	[30, 33]	[20, 22]	[15, 16]	[12, 13]
2	∞	[119, 131]	[59, 66]	[40, 44]	[30, 33]	[24, 26]
3	∞	∞	∞	∞	∞	∞

The choice of the scenario and the value for the corresponding parameter k are the only inputs requested when defining the kinetics of a reaction. To further simplify the modelling process, we implemented the possibility to use qualitative values for k, by providing a pre-defined set of reaction rates, labelled *very slow*, *slow*, *medium*, *fast*, *very fast*. These options encourage construction of a model in which relative reaction rates define the network dynamics. In subsequent steps, the

preliminary model can be fit to experimental data by more precisely setting the values for k.

C. Timed Automata model

An ANIMO network model contains an instance of the Reaction TA template (Fig. 2) for each reaction present in the model. An integer variable is defined for each of the n reactants of the network to represent the reactant's current activity level. This variable is initialized as specified by the user. Finally, a series of channels called reacting$_i$, with $i \in \{1, 2, \ldots, n\}$, is defined to allow reaction processes to communicate when the activity level of the i-th reactant has been updated.

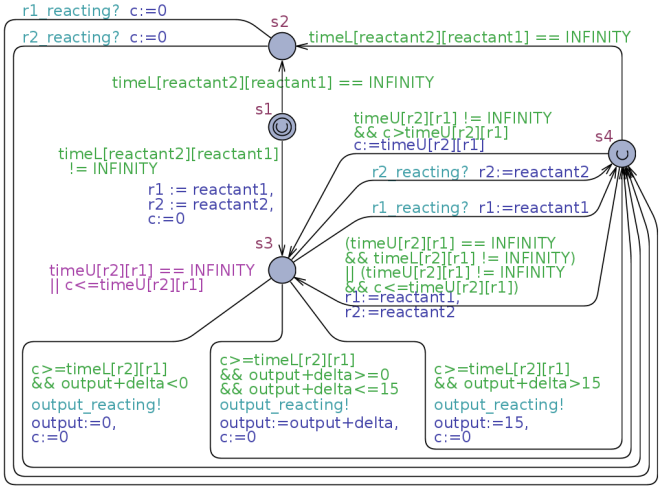


Fig. 2. The TA template for the reactions in the model. The two matrices timeL and timeU define the lower and upper time bounds for each possible combination of the activity levels of the reactants on which the reaction rate depends. Channels r1_reacting, r2_reacting and output_reacting are used for communicating modifications to the value of the reactants involved. Each of these three channels is a reference to a global shared channel reacting$_i$, where i is the index of the corresponding reactant. The output reactant is assumed here to have 15 levels of granularity, and the guards ensure that $0 \leq \text{output} + \text{delta} \leq 15$.

The TA template in Figure 2 depends on two input reactants, to which reactant1 and reactant2 are references, and influences one reactant (output). This template is used for kinetic scenarios 2 and 3, whereas a slightly simpler template is used for scenario 1 that relies on a single input. The basic concept underlying the Reaction TA is to perform a continuous cycle, where at each iteration the activity level of the target reactant (variable output) is updated upon completion of the reaction. The variable delta represents the increment caused in output when the reaction occurs: thus, delta contains $+1$ if the reaction is an activation and -1 for inhibitory reactions. The locations in the Reaction TA template in Figure 2 have been labelled s1, s2, s3, s4, where s1 is the starting location.

s1 is used to reset the internal clock c and start counting (transition to location s3) or to enter a "dormant" location if the reaction cannot occur (transition to location s2). This is the case when the lower bound of the reaction duration is declared as INFINITY (i.e. the reaction rate is 0, see the definition of $f(a, b)$ in Sect. III-B). For instance, if the reaction activates its target and the kinetics are based on scenario 2, the reaction cannot occur if either no inactive substrate or no active enzyme are available (cf. Tab. I). The U symbol inside locations s1 and s4 marks them as *urgent*: while there is at least one automaton in an urgent location, time cannot progress. In this way, all necessary updates are made before the reactions can continue.

The waiting location is identified by the label s3: the automaton can exit from this location when the activity level of an input reactant has been changed by another reaction (transitions from s3 to s4, receiving a communication on channel r1_reacting or r2_reacting), or when the current value of the internal clock c is inside the interval $[R[\text{r2}][\text{r1}]_{\text{lower bound}}, R[\text{r2}][\text{r1}]_{\text{upper bound}}]$, i.e. when the reaction can occur. The bounds for R under the current conditions are found in the tables timeL[ ][ ] (corresponding to $R[\,][\,]_{\text{lower bound}}$) and timeU[ ][ ] ($R[\,][\,]_{\text{upper bound}}$), which are indexed by the current activity levels of the two input reactants r1 and r2.

If the reaction cannot occur (e.g. because all substrate is already active), the automaton stays in location s2 until an update takes place, which can possibly change the current situation (transitions from s2 to s4).

Finally, location s4 is used to check that the clock settings are consistent with the current time bounds.

For an example run, consider two reactions $R_1 = A \rightarrow B$ and $R_2 = C \dashv B$, both based on scenario 2, with starting activity levels $A = 10/10$, $B = 0/10$, $C = 10/10$. The two automata for R_1 and R_2 will start from location s1 and move immediately to s3 and s2 respectively. As both reactions depend on the activity level of B (see the definition of scenario 2 in Sect. III-B) and B is completely inactive, R_1 can proceed at full speed, while R_2 cannot occur. After some time (depending on the parameter k of R_1), transition s3 $\rightarrow$ s4 will be taken by the automaton for R_1, increasing the activity level of the output reactant B by 1 (output = output + delta in the template). At the same time, a synchronization on channel reacting$_B$ (corresponding to output_reacting for R_1 and to r2_reacting for R_2) will allow the automaton for R_2 to reach location s4. reacting$_B$ also corresponds to r2_reacting in the automaton for R_1, but that automaton is already performing output_reacting!, and no more than one transition can be taken at a time. As R_1 can still occur with $A = 10/10$ and $B = 1/10$, the proper s4 $\rightarrow$ s3 transition is taken next. For the same reason, a transition s4 $\rightarrow$ s3 is taken in the automaton for R_2, making both reactions active. From this point, the evolution of the system will proceed depending on the kinetic parameters defined for the reactions, and the activity of B will vary depending on which of the two reactions will occur faster, i.e. more frequently.

D. Analysis with ANIMO

We use the statistical model checking engine [15] of the UPPAAL model checker to obtain a simulation run of the model. In order to obtain the required data, ANIMO generates a query of the form `simulate` $1\ [<= 36000]\{r_0, r_1, \ldots, r_n\}$, which can be read as "perform one simulation run until 36000 time units and produce a trace where the values of variables $r_0, r_1, \ldots, r_n$ (the reactant activities) are shown". This approach is considerably faster (especially in large models) than

the one we presented previously [14], because only strictly needed data is produced by UPPAAL, greatly reducing the time needed for parsing the result. Note that the number of simulations is normally fixed to one, so no real statistical model checking is done when asking for one simulation. In order to see the result of multiple simulation runs of the same model, the user needs to select the option *Compute X runs* on the ANIMO interface, choosing how many simulations have to be performed. Parsed traces can be graphically explored in the ANIMO user interface, where they are displayed as time-series graphs of the selected reactants. In case of multiple simulation runs, the user can choose whether to obtain a plot of the averages (with standard deviation bars) or an overlay plot of all the computed series. Experimental time series data can be added to the graph, enabling a direct comparison with model predictions. By moving a slider at the bottom of the time series graph, the original input model is interactively enhanced by colour codings, showing the activity levels of all reactants for each time instant.

IV. Case Study

As an example application of our approach, we present a model that describes signalling events downstream of two growth factors that regulate cell development and function in PC12 cells: epidermal growth factor (EGF) and nerve growth factor (NGF). PC-12 cells are a model cell line to study neuronal differentiation. We modelled part of the signalling network and compared the behaviour of our model with experimental data from Santos et al. [17].

It has been observed that the activation of extracellular regulated kinase (ERK) shows significantly different dynamics upon treatment with EGF as compared to treatment with NGF. A transient, peak-shaped activation is observed when PC-12 cells are treated with EGF, whereas treatment with NGF results in sustained activation of ERK, even after removal of the input signals via growth factor-neutralizing antibodies. These activation profiles are tied to different cellular outcomes, with transient activation leading to proliferation and sustained activation leading to differentiation. These findings led the authors of [17] to hypothesize the existence of a positive feedback from ERK to an upstream node in the network and existence of a new network component that blocks this feedback. It was found that NGF prevents this blocking, leading to sustained activation [17]. We have formalized and implemented these topological reasonings in an ANIMO model, shown in Figure 3. The parameters used in this network are given in Table II. The accompanying experimental conditions are the addition of either EGF or NGF in sufficient quantity to fully activate their respective receptors. After 10 minutes a growth factor-neutralizing antibody is added, shutting off the input signal to the network. The evolution of the network is observed for 60 minutes from the initial treatment.

Nodes labelled as EGF, NGF, PKC (protein kinase C), RKIP (Raf kinase inhibitory protein), RAF (Raf), MEK (MAPK ERK kinase), and ERK in the ANIMO model correspond to the proteins in the network topology that was presented by Santos et al. [17]. Nodes introduction of Ab and neutralizing antibody are used to represent the introduction of growth factor-neutralizing antibody after 10 minutes from the start of the initial treatment. The reaction that activates node neutralizing antibody takes 10 minutes to complete. Finally, nodes labelled (1), (2) and (3) are phosphatases that inactivate their targets. These inactivations cause cellular signalling networks to be reset to their resting state after a signal has been processed. The feedback activation from ERK to RAF is shown as an edge from ERK to RAF in Fig. 3. In the absence of PKC activity, RKIP is active and inhibits RAF. When PKC is activated by NGF, RKIP is phosphorylated and inhibited by PKC, causing sustained activation of RAF by ERK (cf. Fig. 4b).

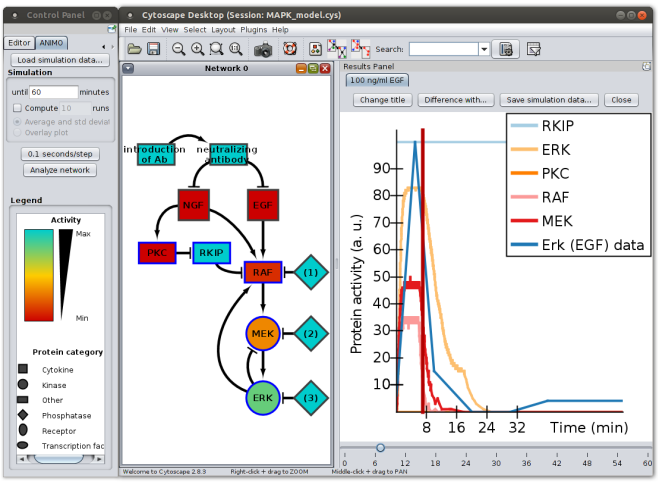


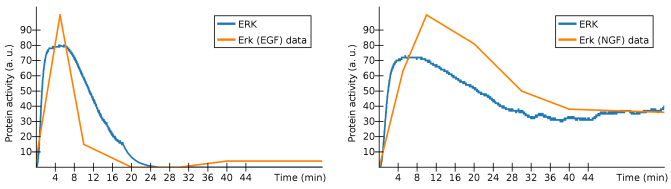
Fig. 3. The model represented in the ANIMO user interface. The *Network* central panel represents the model as the classical nodes-edges network familiar in biology. To this representation, node colours and shapes are added to represent current activity levels and protein categories, as defined in the *Legend* panel on the left. On the right, the *Results Panel* shows a graph of the activities of selected nodes during the user-chosen interval of 60 minutes. A vertical red bar, which can be moved with the underlying slider, indicates the point in the simulation trace on which the colouring of the nodes in the *Network* panel is based. The graph allows the user to visually compare simulation results with experimental data. The Erk (EGF) data series is based on experimental data from Santos et al. [17].

The dynamic behaviour of ERK in the model and from the experimental data is shown in Figure 4. The model reflects the general behaviour of ERK: transient activation results from treatment with EGF, whereas NGF treatment causes sustained activation. In this model, we deliberately left out a number of proteins, e.g. the receptors for EGF and NGF, since no experimental data were available for these nodes. In order to further refine the model, these intermediate nodes could be added to the model. In this way, the model could fit the data more closely, going beyond the scope of this example.

Due to a new addition to ANIMO, it is possible to highlight the differences in the network dynamics between two versions of the model. To this end, simulation results for one version of the model are subtracted from the results of a second version. These differences are then visualized in the network topology, by using a suitable colour scheme. This allows a visual evaluation of the influences of a change in the model on the resulting dynamic behaviour. Figure 5 shows an example, with the differences in activities between 1) treatment with

TABLE II
PARAMETER SETTINGS FOR THE MODEL IN FIGURE 3. THE SCEN. COLUMN CONTAINS THE NUMBER OF THE KINETIC SCENARIO FOR EACH REACTION (SEE SECT. III-B). $^{(*)}$ IN ORDER TO REFLECT EXPERIMENTAL TREATMENT CONDITIONS, THE SETTINGS FOR INITIAL NGF AND EGF ACTIVITY ARE TO BE CONSIDERED MUTUALLY EXCLUSIVE: IF ONE IS AT MAXIMUM ACTIVITY, THE OTHER IS SET AT 0.

Reactants			Reactions		
Name	**Levels**	**Init act.**	**Reaction**	**Scen.**	k
intr. Ab	1	1	intr. Ab $\rightarrow$ neutr. Ab	1	0.003
neutr. Ab	1	0	neutr. Ab $\rightarrow$ NGF	2	0.00375
NGF	15	$15^{(*)}$	neutr. Ab $\dashv$ EGF	2	0.0625
EGF	15	$15^{(*)}$	NGF $\rightarrow$ PKC	2	7e-4
PKC	40	0	NGF $\dashv$ RAF	2	0.005
RKIP	20	20	EGF $\rightarrow$ RAF	2	0.0125
RAF	60	0	PKC $\dashv$ RKIP	2	0.004
(1)	1	1	RKIP $\dashv$ RAF	2	0.02
MEK	60	0	(1) $\dashv$ RAF	2	0.0035
(2)	1	1	ERK $\rightarrow$ RAF	2	3.75e-4
ERK	100	0	RAF $\rightarrow$ MEK	2	0.054
(3)	1	1	(2) $\dashv$ MEK	2	0.0049
			ERK $\dashv$ MEK	2	0.0192
			MEK $\rightarrow$ ERK	2	0.075
			(3) $\dashv$ ERK	2	0.0075



(a) (b)

Fig. 4. Comparison of the model and experimental data. **(a)** Treatment with 100 ng/ml EGF results in transient ERK activation. **(b)** Treatment with 50 ng/ml NGF results in sustained ERK activation. The ERK series are computed from the model, while Erk (EGF) data and Erk (NGF) data are based on experimental data from Santos et al. [17].

NGF and 2) treatment with EGF. It can be seen that a higher activity of PKC downstream of NGF leads to inhibition of RKIP, causing sustained ERK activity. This visualization can be used to rapidly assess the effect of model changes, such as extra nodes, different initializations, altered values for reaction parameters or a new wiring of the network. This feature is particularly useful when working with large models with extensive feedback loops, when the complete impact of changes to the model becomes much harder to grasp.

V. RELATED WORK

Formal approaches to modelling biological systems can be divided into two large groups. The first includes approaches based on ordinary differential equations (ODEs), whereas the second encompasses methods based on concurrent systems. This distinction captures the main characteristic of concurrent systems, which allows one to describe a system by specifying its components in isolation, and then define the interaction rules. Methods based on ODEs on the other hand will need to explicitly account for all changes each interaction can cause to each component of the system. Even if less maintainable, models based on ODEs are usually easier to understand, because of their strong connection with the actual chemical

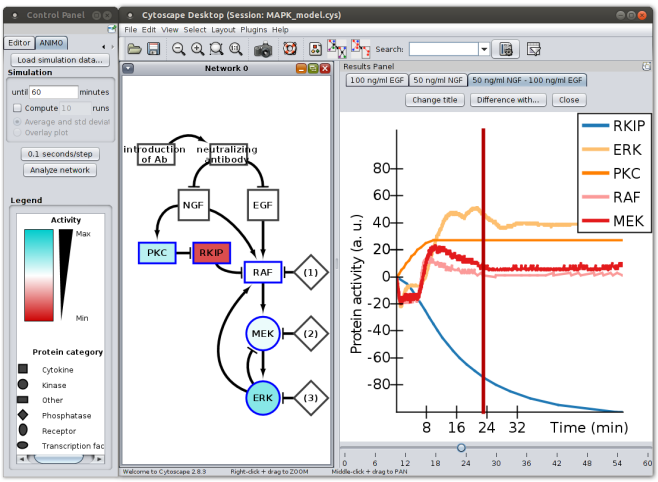


Fig. 5. Difference between the simulations obtained using 50 ng/ml NGF and 100 ng/ml EGF as input at 22 minutes (vertical red bar in right panel). In this mode, node colours are based on the difference of activity between the two configurations of the model: a cyan-coloured node is more active due to NGF treatment, while red indicates less activity and white means no difference. Note that NGF and EGF are both white, since at 22 minutes, NGF and EGF are inactivated in both conditions by the neutralizing antibody. A slider underneath the graph can be used to interactively view the differences at different time points.

reaction laws governing the evolution of a system. Approaches based on concurrency will usually add a layer on top of the canonical description of chemical reactions, requiring a user to acquire some practice before being able to fully benefit from the different paradigm.

Tools that allow to construct models based on ODEs include for example COPASI [2], E-Cell [4] and GNA [3]. These tools add to the potential of ODEs by coupling them with additional modelling approaches, such as stochastic models[1] used in COPASI and E-Cell, and by allowing for qualitative modelling, as does GNA. Other tools, such as Systems Biology Workbench [18], allow to couple high-performance analysis engines (such as roadRunner) to user-friendly interfaces (e.g. JDesigner). Finally, tools such as CellDesigner [19] provide an appealing graphical representation as an interface to powerful computational engines. Most of the cited tools are founded on parameter-intensive approaches, while the paradigm implemented in ANIMO uses less precise formulations, which requires less initial knowledge from the user.

Methods relying on concurrent systems can be either qualitative or quantitative. The distinction is based on the possibility to add numerical (quantitative) information to the model, such as reaction rates, molecular concentrations, and reaction volumes. ANIMO is a quantitative method based on concurrent systems. The use of TA to model biological signalling events has been described before by Maler and Batt [12]. ANIMO takes a less general approach, tailored for the construction of more abstract models.

[1]Stochastic modelling becomes particularly useful when some molecular species have very small concentrations, which nullifies the assumption of a well-mixed solution used in models based on ODEs.

VI. Conclusions and future work

We contribute to the modelling of biological pathways by introducing a formalization of the domain-specific language traditionally used for pathway representations into a model based on Timed Automata (TA). The ANIMO user interface makes the power of TA available to users that lack a thorough background in mathematics or computer science. This will likely contribute to to the dissemination of computational modelling in the realm of molecular cell biology and regenerative medicine. In this respect, we are applying ANIMO in a research project aimed at studying chondrocyte signalling in relation to osteoarthritis [20], [21]. The objective is to enhance cartilage tissue engineering strategies by investigating the effect of extracellular signalling molecules and cell-matrix interactions in order to mimic these signals in the development of biomaterials that provide direct support, while stimulating chondrocytes to repair the damaged cartilage tissue.

The use of TA as the underlying formalism of ANIMO lies the foundation to the use of more advanced analysis techniques to query constructed models. In particular, applying model checking techniques would allow us to perform *in silico* experiments, answering questions such as "What is the combination of inputs that leads to activity$(A) \geq 20/50$ and activity$(B) < 10/80$ in 120 minutes?". Moreover, taking advantage of statistical model checking capabilities recently introduced to UPPAAL [15], we plan to extend the current modelling paradigm adding the possibility to define stochastic behaviour. This could for instance be done by assigning exponential distributions to a set of reactions, to support probabilistic queries such as "What is the probability that activity$(A) \geq 40/50$ in 30 minutes?". Other interesting developments are aimed at further speeding up the modelling phase, in order to let the user start interrogating a model as soon as possible. We plan to include a support for parameter sensitivity analysis and automated parameter fitting to a given experimental data set. After having defined a measure of distance between the model and the experimental data, we plan to minimize that distance via user-defined parameter sweeps. We are also developing techniques based on automata learning [22] for deriving the topology of a biological network, based on a series of constraints and experimental data series.

In the future, ANIMO and related tools may lead to a new paradigm for interactive representation of biological networks. Networks in digital textbooks and articles could be displayed as animations amenable to modifications by readers. Repositories of formal descriptions of signalling modules could be used to put together executable signalling networks. In general, the process of formalizing biological knowledge will lead to a more thorough understanding of biological networks and will accelerate hypothesis-driven research.

References

[1] J. Fisher and T. A. Henzinger, "Executable cell biology." *Nature biotechnology*, vol. 25, no. 11, pp. 1239–1249, Nov. 2007.

[2] P. Mendes, S. Hoops, S. Sahle, R. Gauges, J. Dada, and U. Kummer, "Computational modeling of biochemical networks using COPASI systems biology," ser. Methods in Molecular Biology, I. V. Maly, J. M. Walker, and J. M. Walker, Eds. Totowa, NJ: Humana Press, 2009, vol. 500, ch. 2, pp. 17–59.

[3] H. de Jong, J. Geiselmann, C. Hernandez, and M. Page, "Genetic Network Analyzer: qualitative simulation of genetic regulatory networks," *Bioinformatics*, vol. 19, no. 3, pp. 336–344, Feb. 2003.

[4] M. Tomita, K. Hashimoto, K. Takahashi, T. S. Shimizu, Y. Matsuzaki, F. Miyoshi, K. Saito, S. Tanida, K. Yugi, J. C. Venter, and C. A. H. III, "E-CELL: software environment for whole-cell simulation." *Bioinformatics*, vol. 15, no. 1, pp. 72–84, Jan. 1999.

[5] F. Ciocchetta, A. Degasperi, J. Heath, and J. Hillston, "Modelling and analysis of the NF-κB pathway in Bio-PEPA," in *Transactions on Computational Systems Biology XII*, ser. Lecture Notes in Computer Science, C. Priami, R. Breitling, D. Gilbert, M. Heiner, and A. Uhrmacher, Eds. Springer Berlin / Heidelberg, 2010, vol. 5945, pp. 229–262.

[6] P. Lecca, "BlenX models of α-synuclein and parkin kinetics in neuropathology of Parkinson's disease," *Journal of Biological Systems*, vol. 19, pp. 149–181, June 2011.

[7] ANIMO, http://fmt.cs.utwente.nl/tools/animo.

[8] R. Alur and D. L. Dill, "A theory of timed automata," *Theor. Comput. Sci.*, vol. 126, pp. 183–235, April 1994.

[9] K. G. Larsen, P. Pettersson, and W. Yi, "UPPAAL in a nutshell," *International Journal on Software Tools for Technology Transfer (STTT)*, vol. 1, pp. 134–152, 1997.

[10] E. Bartocci, F. Corradini, E. Merelli, and L. Tesei, "Model checking biological oscillators," *Electronic Notes in Theoretical Computer Science*, vol. 229, no. 1, pp. 41–58, 2009, proceedings of the Second Workshop From Biology to Concurrency and Back (FBTC 2008).

[11] H. Siebert and A. Bockmayr, "Temporal constraints in the logical analysis of regulatory networks," *Theor. Comput. Sci.*, vol. 391, no. 3, pp. 258–275, Feb. 2008.

[12] O. Maler and G. Batt, "Approximating continuous systems by timed automata," in *Formal Methods in Systems Biology*, ser. Lecture Notes in Computer Science, J. Fisher, Ed. Springer Berlin / Heidelberg, 2008, vol. 5054, pp. 77–89.

[13] S. Killcoyne, G. W. Carter, J. Smith, and J. Boyle, "Cytoscape: a community-based framework for network modeling." *Methods in molecular biology (Clifton, N.J.)*, vol. 563, pp. 219–239, 2009.

[14] S. Schivo, J. Scholma, B. Wanders, R. A. U. Camacho, P. E. van der Vet, M. Karperien, R. Langerak, J. van de Pol, and J. N. Post, "Modelling biological pathway dynamics with Timed Automata," in *2012 IEEE 12th International Conference on Bioinformatics & Bioengineering (BIBE)*. Los Alamitos, CA, USA: IEEE Computer Society, 2012, pp. 447–453.

[15] A. David, K. Larsen, A. Legay, M. Mikučionis, and Z. Wang, "Time for statistical model checking of real-time systems," in *Computer Aided Verification*, ser. Lecture Notes in Computer Science, G. Gopalakrishnan and S. Qadeer, Eds. Springer Berlin / Heidelberg, 2011, vol. 6806, pp. 349–355.

[16] E. Clarke, "Model checking," in *Foundations of Software Technology and Theoretical Computer Science*, ser. Lecture Notes in Computer Science, S. Ramesh and G. Sivakumar, Eds. Springer Berlin / Heidelberg, 1997, vol. 1346, pp. 54–56.

[17] S. D. M. Santos, P. J. Verveer, and P. I. H. Bastiaens, "Growth factor-induced MAPK network topology shapes Erk response determining PC-12 cell fate," *Nature Cell Biology*, vol. 9, no. 3, pp. 324–330, Feb. 2007.

[18] F. Bergmann and H. Sauro, "SBW - a modular framework for systems biology," in *Proceedings of the Winter Simulation Conference, 2006. WSC 06.*, 2006, pp. 1637–1645.

[19] A. Funahashi, Y. Matsuoka, A. Jouraku, H. Kitano, and N. Kikuchi, "CellDesigner: a modeling tool for biochemical networks," in *Proceedings of the 38th conference on Winter simulation*, ser. WSC '06. Winter Simulation Conference, 2006, pp. 1707–1712.

[20] B. Ma, J. Leijten, L. Wu, M. Kip, C. van Blitterswijk, J. Post, and M. Karperien, "Gene expression profiling of dedifferentiated human articular chondrocytes in monolayer culture," *Osteoarthritis and Cartilage*, vol. 21, no. 4, pp. 599 – 603, 2013.

[21] J. C. H. Leijten, J. Emons, C. Sticht, S. van Gool, E. Decker, A. Uitterlinden, G. Rappold, A. Hofman, F. Rivadeneira, S. Scherjon, J. M. Wit, J. van Meurs, C. A. van Blitterswijk, and M. Karperien, "Gremlin 1, Frizzled-related protein, and Dkk-1 are key regulators of human articular cartilage homeostasis," *Arthritis & Rheumatism*, vol. 64, no. 10, pp. 3302–3312, 2012.

[22] J. Tretmans, "Model-based testing and some steps towards test-based modelling," in *Formal Methods for Eternal Networked Software Systems*, ser. Lecture Notes in Computer Science, M. Bernardo and V. Issarny, Eds. Springer Berlin / Heidelberg, 2011, vol. 6659, pp. 297–326.