ComPlexUs
ORIGINAL RESEARCH PAPER

Complexus 2006;3:231–251 DOI: 10.1159/000110010
Received: November 11, 2006
Accepted after revision: June 21, 2007
Published online: October 31, 2007

Hybrid Modelling and Dynamical Analysis of Gene Regulatory Networks with Delays

Jamil Ahmad[a] Gilles Bernot[b, c] Jean-Paul Comet[c]
Didier Lime[a] Olivier Roux[a]

[a]IRCCyN (ECN & CNRS 6597), BP 92101, Nantes, [b]Programme d'épigénomique, Evry, and [c]Laboratoire I3S (UNSA & CNRS UMR 6070), Les Algorithmes, Sophia Antipolis, France

Key Words
Networks, analysis and modelling · Bioinformatics · Computer analysis · Gene regulatory network

Abstract
René Thomas' discrete modelling of gene regulatory networks (GRN) is a well-known approach to study the dynamics resulting from a set of interacting genes. It deals with some parameters which reflect the possible targets of trajectories. Those parameters are *a priori unknown*, but they may generally be deduced from a well-chosen set of biologically observed trajectories. Besides, it neglects the time delays for a gene to pass from one level of expression to another one. The purpose of this paper is to show that we can account for time delays of increasing or decreasing expression levels of genes in a GRN, while preserving powerful enough computer-aided reasoning capabilities. We designed a more accurate abstraction of GRN where *delays are now supposed to be non-null*

Simplexus
Almost nothing in cell biology can be understood without considering the detailed interactions of multiple genes and proteins within complex genetic regulatory networks. It is the highly non-linear feedbacks within such networks that control and direct communications between cells, internal changes in response to varying environmental conditions, such as temperature, or the synthesis of new materials from basic molecular building blocks in food sources. Built on biochemistry, and chemical reactions that proceed continuously, regulatory networks nevertheless give a digital character to biology, supporting the discrete switching behaviour that enables cells to orchestrate abrupt transitions in their own interests.

For 50 years, of course, biologists have been building up an understanding of genetic regulatory networks, especially the simpler examples, starting with the famous lac operon, important in metabolic control in bacteria. But contemporary research is rapidly pushing far beyond anything in the past, in part because of the availability of complete genomic and proteomic information for a wide number of organisms. As new experimental techniques make it possible to identify large networks of genes, proteins and their interactions, however, understanding also requires new methods for dissecting the logical dynamics of network feedbacks, and for creating theoretical models capable of producing insight from masses of data. How should regulatory networks be modeled?

Classic work in this area has explored two very different approaches. On the one hand, some theorists have worked with greatly oversimplified models, representing genes within networks as Boolean variables, for example, which have only two states – on or off. Such work has elucidated basic theoretical properties expected in regulatory networks, such as

Prof. Dr. Olivier Roux, IRCCyN (UMR CNRS 6597)
1, rue de la Noé, BP 92101
FR–44321 Nantes Cedex 03 (France)
Tel. +33 2 40 37 69 79, Fax +33 2 40 37 69 30
E-Mail olivier.roux@irccyn.ec-nantes.fr

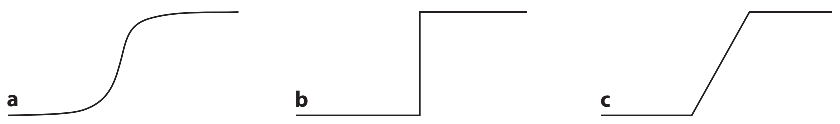


Fig. 1. A sigmoid curve (**a**), its logical caricature (**b**), and its piecewise linear caricature (**c**).

unknown new parameters. We show that such models, together with hybrid model-checking algorithms, make it possible to obtain some results about the behaviour of a network of interacting genes, since dynamics depend on the respective values of the parameters. The characteristic of our approach is that, among possible execution trajectories in the model, we can automatically find out both viability cycles and absorption in capture basins. As a running example, we show that we are able to discriminate between various possible dynamics of mucus production in the bacterium *Pseudomonas aeruginosa*.

1 Introduction

Work about modelling for simulation and analysis of gene regulatory networks (GRN) was initiated by Jacob and Monod [1]. As de Jong [2] writes, the idea of Sugita [3] of interpreting gene interactions as logical systems led to further developments with *boolean networks* and their generalization to *discrete networks* [4, 5]. Theoretical models of GRN are useful when several feedback loops of gene interactions generate complex behaviours. In such cases, understanding the causality chains which induce some observed phenotype is impossible without computer assistance. Using a computer to perform simulations can help biologists to explore some possible behaviours, however only performing simulations is unsatisfactory and insufficient to establish properties of GRN and to validate biological hypotheses. So, it becomes necessary to give a *formal* definition of GRN, as introduced in Bernot et al. [6], in such a way that the computer can perform *model checking* with respect to logical formulae expressing biological knowledge. We propose here the enhancement of this formalization that we call *temporal networks* in order to take into account delays of gene activation and/or degradation of their products.

According to Thomas and Kaufman [7], *regulation can be defined as the processes that adjust the rate of production and decay of the elements of a system to the state of the system itself and to relevant environmental constraints.* Thus, Thomas and Kaufman argued for the logical caricature (fig. 1b), but thus, they had to consider an ‘asynchronous’ option which leads to the consideration of different dynamics corresponding to different choices of the fastest variable change.

One of the main advantages of the discrete asynchronous approach of Thomas is that it has been formalized so that biological hypotheses can be encoded into logical temporal properties and *automatically* validated or refuted using model checking techniques [6]. The price to pay is the logical caricature (fig. 1b) of the sigmoid curve (fig. 1a), which consequently ignores activation delays (and degradation delays for decreasing sigmoids).

Our approach intends to design a new modelling formalism to account for time delays in GRN. Intuitively, we simply propose to caricature the sigmoidal shape of interaction (fig. 1a) by a piecewise linear

the nature of limit cycles, and how the number of such cycles grows with the size of the network. Alternatively, other researchers have started instead from the more detailed level of basic biochemistry, modeling the interactions among genes, proteins and other molecules with differential equations that follow the evolution of chemical concentrations. This work has led to surprisingly realistic and powerful models for addressing practical matters – even the likely influence of drugs on the human heart.

But each of these extremes, focusing either on the discrete or fully continuous dynamics, has its limitations. While the former leaves out many details of the real workings of gene expression, the latter includes so many details that computational demands preclude their application to many larger systems. To move forward, researchers are increasingly introducing more sophisticated computational methods from computer science, such as so-called ‘hybrid’ techniques, which aim to mould ideas from both continuous and discrete modeling together into one paradigm. The idea is to use each modeling style where it seems most appropriate and efficient, and thereby to produce models that match the mixed continuous and discrete character of real biology more realistically.

In this paper, Jamil Ahmad and colleagues use this hybrid strategy to generalize an influential class of models proposed by René Thomas, which have proven useful for logical reasoning about the properties of genetic regulatory networks. In such models, a regulatory network is a feedback network, with positive or negative influences linking the various elements, as illustrated in the authors’ figure 2. Each element in such a network can be expressed at one of a number of discrete levels, in which case it exerts excitatory or inhibitory influence on those elements to which it is linked. In this mathematical framework, a system may curve as in figure 1c. We show that model checking remains possible for some properties, using the model checking tool HyTech [8]. Our aim has been to be able to automatically verify the properties commonly considered by biologists. Experimentally, the properties which can be observed by 'wet biology' are usually equilibrium states or periodic behaviours. Transitory states are generally difficult to catch experimentally. Consequently, stable states, basins of attraction, circular trajectories and reachability are the main properties we have considered. Another capability of HyTech is to manage unknown parameters: they are replaced by abstract symbols, and then, HyTech provides the constraints on these symbols which are necessary and sufficient to prove a given property. Exact delays being often not precisely known, this symbolic treatment of our framework is a great advantage.

The proposed framework is based on *hybrid system* modelling. These hybrid systems are such that discrete events are combined with continuous differential equations to capture the switching behaviour of the real life that is observed in phenomena inside the cells. The idea we follow is that the thresholds of corresponding protein (or RNA) expression levels of the genes are no longer instantaneously triggered. Since a threshold crossing actually takes some time, the expression levels may begin to increase (or decrease), and then be suspended and afterwards, the previous dynamics can be resumed or opposed, and so on. This reflects the 'feature of *accumulation*'. Furthermore, differences in time delay values determine some precise trajectories of the system, which reduce the non-determinism induced by the asynchronous semantics of René Thomas. These features have to be considered so that we can be able to find out the precise parameter constraints that are consistent with the observed dynamics of gene expression levels.

Other works might be related to ours, either because the authors use the same idea of hybrid modelling and analysis of biological networks, but not with time delays [9–14] or because they are concerned with time delays but they adopt a completely different approach [15–17]. Finally, other authors have an approach close to ours: Adélaïde and Sure [18] make a parametric abstraction of GRN using hybrid automata but they have a restriction in assuming a uniform degradation rate, and Siebert and Bockmayr [19] deal with time delays using timed automata (which is a more restrictively expressive model than hybrid automata, since it does not deal with non-increasing variables). The time delays incorporated into the logical analysis of GRN [19] are somehow similar to ours, but it is not possible to deal with accumulation as we do (see section 4.4).

The paper is organized as follows. We recall in section 2 the principles of discrete modelling for genetic regulatory networks. General principles of hybrid modelling are explained in section 3, and our specific approach of hybrid modelling for GRN is discussed in section 4. We show in section 5 the analysis which can be made on the models, and we come to a conclusion. Throughout the paper, we show how our approach can be applied to the example of mucus production in *Pseudomonas aeruginosa*.

2 Discrete Modelling for Genetic Regulatory Networks

In this section we present René Thomas' modelling of genetic regulatory networks with the example of the mucus production system in *P. aeruginosa*, an opportunistic pathogen, which is often encountered in chronic lung diseases such as cystic fibrosis. These bacteria are commonly present in the environment and secrete mucus only in lungs affected

step along a trajectory of states as the elements change their discrete levels at the same time.

These models are highly schematic, however, and do not include, for example, the influence of time delays in the switching of elements from one state of expression to another. Ahmad et al. try to include such effects, at least in a first approximation, by using hybrid models. Their idea is to preserve the discrete variables that refer to distinct qualitative states having differing expression levels, while adding further continuous variables to follow the smaller variations in expression that ultimately trigger transitions between states. For a regulatory network in the bacterium *Pseudomonas aeruginosa*, they show how this hybrid approach can be used to build more powerful computational models for real genetic regulatory networks, and offer a method for systematically building up theoretical models that are consistent with known biological reality.

As an illustrative example, the authors focus on a relatively simple genetic regulatory network, the mucus production system of *P. aeruginosa*. Here the main regulatory molecule, AlgU, interacts with an operon leading to two primary effects – the stimulus of its own production and also that of a protein which then acts to inhibit AlgU production. Following a number of technical definitions (definitions 1 through 4), the authors illustrate how the network can be modeled using both discrete and continuous variables. Here the discrete variables, x and y, reflect the discrete (qualitative) expression levels of various genes or proteins (in this case, AlgU and its associated inhibiting protein). In reality, of course, expression levels are continuous; treating them within a discrete approximation reflects the biological fact that levels tend to be roughly consistent in any biological state (such an equilibrium). The aim of the continuous variables is to improve on

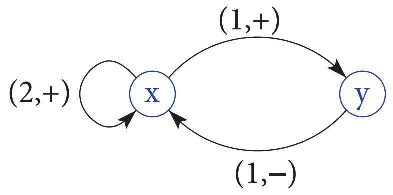


Fig. 2. Regulatory graph for mucus production in *P. aeruginosa.*

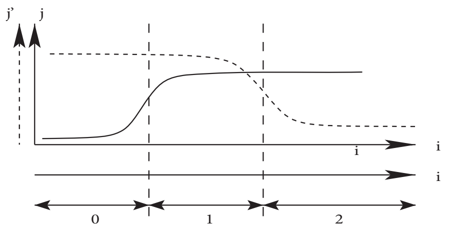


Fig. 3. The discretization is supervised by the thresholds of actions on targets.

by cystic fibrosis. As this mucus increases the respiratory deficiency of the patient, it is the major cause of mortality. The regulatory network which controls the mucus production has been elucidated [20]. The main regulator for the mucus production, AlgU, supervises an operon which is made of 4 genes among which one codes for a protein that is an inhibitor of AlgU. Moreover AlgU favours its own synthesis. The mucus production regulatory network can then be simplified into the regulatory graph of figure 2, where node x represents AlgU, and node y its inhibitor.

Formally the biological regulatory graphs represent interactions between biological entities (genes via their proteins for instance) which are depicted by vertices. Each edge is labelled with the sign of the interaction: '–' for an inhibition and '+' for an activation. Each node abstracts the role of a biological entity and is usually called a variable.

The sigmoid nature of interactions (see fig. 3) allows us to abstract continuous expression levels by qualitative abstract expression levels: for each positive interaction of *i* on *j*, if the variable *i* has an expression above (respectively below) a certain threshold (inflection point of the sigmoid), then the variable *j* is (respectively is not) influenced by *i* and symmetric for negative interactions. Figure 3 assumes that the variable *i* acts positively on *j* and negatively on *j*'; each curve represents the expression of the target *after* a sufficient *delay* for the regulator *i* to act on it. Three regions are relevant: the first one corresponds to the situation where *i* neither activates *j* nor inhibits *j*', the second to the situation where *i* activates *j* and does not inhibit *j*', and the last one corresponds to the one where *i* activates *j* and inhibits *j*'. This justifies the discretization of the expression associated with *i* into three abstract levels (0, 1 and 2) corresponding to the previous regions and constituting the only relevant information from a qualitative point of view.

For a variable which has ν targets (itself possibly included), $\nu + 1$ abstract levels have to be considered if all thresholds are distinct, but possibly less in the case where two or more thresholds are equal.

Definition 1

A biological regulatory graph is a labelled directed graph $G = (V, E)$ *where each vertex i of V is provided with a boundary* $\beta_i \in \mathbb{N}$ *less or equal to the out-degree (the number of out-going arcs) of i, except if the out-degree is* 0 *in which case* $\beta_i = 1$. *Each edge* $(i \rightarrow j)$ *is labelled with a pair* $(\tau_{ij}, \varepsilon_{ij})$ *where* $\varepsilon_{ij} \in \{-, +\}$ *is the sign of the interaction and* τ_{ij}, *called threshold, is a natural number between* 1 *and* β_I.

For our running example, no information is available on thresholds of interactions x → y and x → x. In particular we have to consider both situations $\tau_{xy} > \tau_{xx}$ and $\tau_{xy} < \tau_{xx}$, but for simplicity we explain the modelling on the second one.

this approximation by including some of the dynamical features, in particular those associated with transitions between states.

As the authors state, their aim is to develop an automatic approach for building up theoretical models by 'determining all parameters which lead to dynamics compatible with the specified properties.' Their strategy is to use computation (a specific program called SMBioNet) to see which, of all the conceivable mathematical networks of the hybrid type, have dynamics that can match the key features of biological reality. The number of such models grows rapidly with the size of the network, because of the combinatorially many different states possible, and the many different ways the elements can influence one another with positive, negative interactions. The number of possibilities in the continuous sector of the model is essentially infinite; but for simplicity the authors restrict their models to involve only piecewise continuous dynamics for such variables, using them to replace and improve the discrete step function approximation of the sigmoid shape (figure 1).

For the genetic regulatory network controlling mucus production in *P. aeruginosa*, the authors in this way discovered eight dynamical models consistent with the known biological facts (mainly, cycles of expression or fixed expression levels in equilibrium states, as measured in the lab). This example illustrates the power of this approach to go beyond the original mathematics of Rene Thomas, by including a little information on the transients of changing expression levels.

Explained more fully in Section 4.1, the hybrid approach yields models with greater flexibility. Without the continuous dynamics, for example, the model can only map one state to a unique following state. With such variables, the dynamics is considerably richer. As the authors explain in Section 4.2, there are

The interaction graph is not sufficient for describing the dynamics of the system and then to explain the mucus production. To introduce dynamics, we first define the qualitative states of the system.

Definition 2

A state of a biological graph is a tuple $n = (n_1, n_2, ..., n_p)$, where p is the number of vertices and n_i is the abstract expression level of i with $n_i \in \mathbb{N}$ and $n_i \leq \beta_i$.

The threshold and sign of an interaction combined with the current state allows us to know if the regulator has an influence on its targets. Comparing threshold τ_{ij} of interaction $(i \rightarrow j)$ and a state, one determines if *i* stimulates *j*: for a positive interaction, if variable *i* has an abstract level below τ_{ij}, the interaction is not active and *j* is not stimulated, otherwise, it is. For negative interactions, the conditions are symmetrical. But to define the dynamics, we need to determine towards which abstract levels the targets are attracted. To answer this question, one has to know for each state *n* which regulators are actually effective on the considered target *i*, in other words, which are the '*resources*' of *i* in the state *n*.

Definition 3

Given a biological regulatory graph $G = (V, E)$ and a state $n = (n_1, n_2, ..., n_p)$, the set of resources of i is the set

$$R_i(n) = \Big\{ j \in V \;\Big|\; (j \xrightarrow{(\tau,+)} i) \in E \text{ and } (n_j \geq \tau), \\ \text{or } (j \xrightarrow{(\tau,-)} i) \in E \text{ and } (n_j < \tau) \Big\}.$$

$R_i(n)$ contains the activators of *i* whose abstract level is above the threshold and the inhibitors of *i* whose abstract level is below the threshold. A resource is either the presence of an activator or the *absence* of an inhibitor.

The abstract level towards which a variable *i* is attracted when its resource set is ω, denoted by $k_{i,\omega}$, is called the *focal point* of *i* for resources ω. The values of focal points define in a straightforward manner the *synchronous state graph* which represents a first dynamics of the system. The synchronous state graph is obtained by setting for the unique possible successor of state $n = (n_1, n_2, ..., n_p)$ the state towards which the system is attracted: $n' = (k_{1,R_1(n)}, k_{2,R_2(n)}, ... k_{p,R_p(n)})$ where $R_i(n)$ is the set of resources of *i* at state *n*. This definition has at least two drawbacks:

- First, it allows two or more variables to change simultaneously, while the probability that several variables pass through their respective thresholds at the same time is negligible in vivo. But we do not know which one will pass through its threshold first.
- Second, it does not prevent that a variable passes directly two or more thresholds, which is not realistic because an abstract expression level should evolve gradually.

An improved semantics is defined in terms of an *asynchronous state graph* which:

- replaces each diagonal transition of the synchronous state graph (transition with 2 or more variables changing their expression levels) by the collection of transitions each of them modifying only one of the involved variables,
- replaces a transition of length greater or equal to 2 (which passes two or more thresholds at once) by a transition of length 1 in the same direction.

To formally define the asynchronous state graph, we now introduce the evolution operator $\mapsto$: for $x, k \in \mathrm{N}$, $x \mapsto k$ is equal to $x - 1$ if $x > k$, to $x + 1$ if $x < k$ and to x if $x = k$.

Definition 4

Let $G = (V, E)$ be a regulatory graph with p variables. Its asynchronous state graph, denoted $(S, \rightarrow)$, is defined as follows:

now two kinds of transitions: 1) a continuous transition, wherein the continuous variables evolve so as to cross important thresholds, triggering meaningful biological change, and 2) discrete transitions, in which the discrete variables abruptly change and the continuous variables get 'reset' and begin evolving again. This leads to the possibility that a trajectory passing through the state (1,0) may go on to either the state (1,1) or to the state (2,0), depending on what happens to the continuous variables. A completely discrete model is probabilistic, in other words, while the hybrid model is more deterministic; or, as the authors argue at the end of section 4, this more detailed model naturally distinguishes dynamics that would appear identical in the Thomas scheme.

The authors go on in Section 5 to show how such modeling can yield results of real biological relevance, which may seem surprising given that this work often looks more like computer science than biology. But that's where biology is heading and for good reason; biology is, in many respects, computation.

Mark Buchanan

– *The set of vertices is the set of states* $\Pi_{i \in V}\{0, ..., \beta\}$
– *There is a transition from the state* $n = (n_1, ..., n_p)$ *to* $m = (m_1, ..., m_p)$ *iff*

- there exists a unique $i \in [1, p]$

  such that $\begin{cases} m_i \neq n_i \\ m_i = (n_i \Rsh k_{i,R_i(n)}) \end{cases}$

- or $\begin{cases} m = n \text{ and} \\ \forall i \in [1, p], n_i = k_{i,R_i(n)} \end{cases}$

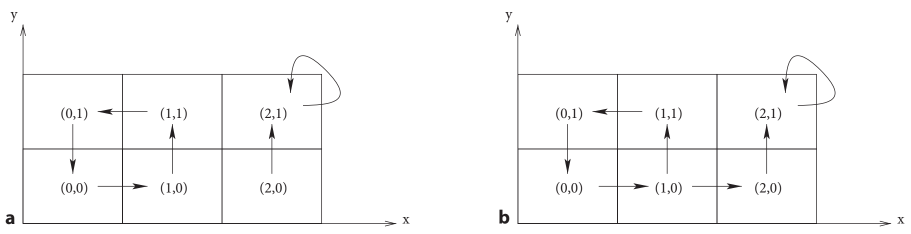


Fig. 4. Two asynchronous state graphs for mucus production in *P. aeruginosa.* Presence (**a**) or absence (**b**) of validation of the CTL specification is shown.

The parameters $k_{i,\omega}$ play a major role for the dynamics of the model. Unfortunately most often they are not experimentally measurable. Indeed finding suitable classes of those parameters constitutes a major issue of the modelling activity. Hopefully, several kinds of constraints can be taken into consideration in order to reduce the set of possible instantiations of these parameters. If Thomas' approach is seen as a discretization of a particular class of continuous differential equation systems, then the parameters $k_{i,\omega}$ reflect a discretization of sums of ratios of positive constants [21]. In such a case, the family of $k_{i,\omega}$ has to verify the lattice constraints: $k_{v,0} = 0$ and $\omega \subseteq \omega' \Rightarrow k_{v,\omega} \leq k_{v,\omega'}$.

Some biological knowledge or hypotheses about the behaviour can also be used as indirect criteria: homeostasis (respectively multistationarity) is experimentally observable and it indicates that a negative (respectively positive) circuit is functional [22–26], or in other words, is responsible for the associated behaviour. Necessary conditions for functionality of a circuit constrain the values of parameters (notion of characteristic states in [23]).

Finally, temporal properties formalizing biological knowledge or hypotheses can also be used [6]. The behavioural properties of the system are first translated into a formal language like computation tree logic (CTL)[27], then all models are formally checked against those properties, memorizing only models which satisfy them.

The interaction graph abstracting the mucus production system in *P. aeruginosa* contains a positive feedback circuit. This makes possible a dynamic with two stationary states for some values of the parameters which would allow, from a biological point of view, an epigenetic change (stable change of phenotype without mutation) from the non-mucoid state to the mucoid one. The dynamical hypothesis (epigenetic hypothesis) can be translated into a CTL specification [28]: stability of mucoid state (respectively non-mucoid state) is expressed by $(x = 2) \Rightarrow AXAF(x = 2)$ (respectively $(x = 0) \Rightarrow AG(\neg(x = 2))$). These formulae mean that, if x is equal to 2, then it will finally be equal to 2 in the future, and that, if x is equal to 0, then it will never be equal to 2.

We developed an automatic approach for determining all parameter values which lead to dynamics compatible with the specified properties: it consists in enumerating all the possible dynamics and selecting by model checking only those which are consistent with the properties. This intensive model checking approach, implemented in SMBioNet[1] [6], formally proves that the epigenetic hypothesis is consistent because 8 models satisfy these formulae from which one is presented in figure 4a.

The other model, presented in figure 4b, does not satisfy the CTL specification since, from any state, it is always possible to go from a state with a low level of x $(x = 0)$ to a state with a saturated level of x $(x = 2)$.

The latter model actually abstracts two different kinds of dynamics depending on the precise nature of the abstract cycle $C = (0, 0) \rightarrow (1, 0) \rightarrow (1, 1) \rightarrow (0, 1) \rightarrow (0, 0)$. On the one hand, this cycle can represent an attractive cycle in the continuous phase space, that is, a cycle where trajectories starting from a certain domain tend to a limit cycle or possibly to a single point (see fig. 5a). If the initial state is in that domain (α), then trajectories never leave it and they go closer and closer to the *limit cycle*. In that case, some other trajectories (β), those that start too far from the limit cycle, leave the abstract cycle C and go into the domain associated with the state $(2, 1)$. On the other hand, the cycle can derive from a discretization of outgoing spiral trajectories around an unstable steady state (see fig. 5b). In that case, if the initial point is the steady state, the system does not evolve (γ), but at the first tiny change, the system goes away from the steady state and (δ) trajectories will finally reach the domain $(2, 1)$. The previous specifications

[1] http://smbionet.lami.univ-evry.fr/

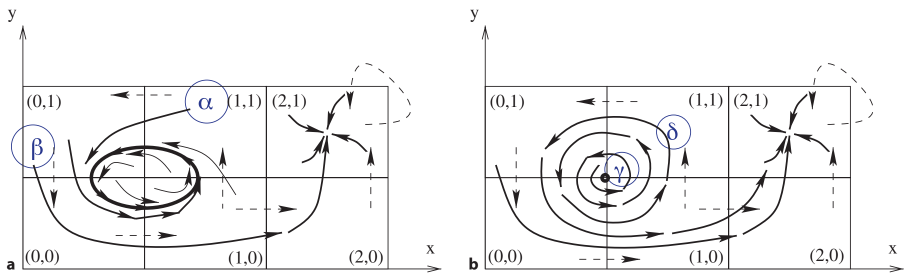


Fig. 5. Continuous trajectories compatible with asynchronous state graphs of figure 4b: cyclic trajectory (**a**) and divergent trajectory (**b**).

are satisfied only in cases (α) and (γ). Since the model of figure 4b abstracts different situations that do not all satisfy the specifications, it is not selected by the intensive model checking approach. The modelling of Thomas, because it does not consider time, is too coarse to distinguish these different situations.

We introduce in this paper delays in the modelling to separate these different classes of behaviours. It is expected that *under certain constraints on delay parameters*, the model presents both a cyclic trajectory, representing the mucoid state, and one attractive basin representing the non-mucoid state. Under other constraints, the system presents only the attractive basin. Biological knowledge is then taken into consideration to choose between the models with or without cyclic trajectories.

3 Principles of Hybrid Modelling

Hybrid models allow the joint representation of both discrete and continuous dynamics. They have been successfully used in the last decades, in particular for the verification of embedded and real-time systems.

One of the simplest and most natural formalisms for hybrid systems is the timed automaton formalism [29]. In this modelling framework, the state of the system is described by a discrete location and the values of continuous variables which evolve synchronously with time. These variables, called *clocks*, can be tested for the satisfaction of given constraints and reset when passing from one discrete location to another. This modelling framework has been extensively studied in the last decade for it enjoys nice properties of decidability and complexity. In particular, the study of the state space of hybrid systems relies heavily on the *reachability problem*, which allows one to say if, starting from an initial state, some given state can be reached by executing the semantics of the model. The reachability problem is decidable in PSPACE for timed automata [29].

However, the relative simplicity of the analysis of timed automata comes at the price of some limitations in the expressive power. We will add fixed rates in such a way that the values of variables either stay constant, or decrease synchronously with the elapsing of time. By doing this, we are not in the class of timed automata, but in the more general framework of linear hybrid automata (LHA) for which reachability (and most of the other interesting properties) is undecidable [30].

We now give a more formal definition for our underlying modelling framework: we define it as a timed automaton augmented with the possibility of having 0, 1 or –1 as derivatives for 'clocks' (and these may be negative). This is actually a subclass of the LHA [30].

Given a set of variables X, let $C(X)$ be the set of conjunctions of 'simple' constraints i.e. of the form $x - y \# c$ or $x \# c$ with $c \in \mathbb{Q}$, $x, y \in X$ and $\# \in \{=, <, \leq, \geq, >\}$.

Definition 5 (Linear Hybrid Automaton) [30, 31]

A Linear Hybrid Automaton *is a 6-tuple* (L, l_0, X, E, Inv, Dif) *where*

- *L is a finite set of* locations.
- *l_0 is the* initial location.
- *X is a finite set of real-valued variables.*
- *$E \subset L \times C(X) \times 2^X \times L$ is a finite set of edges. If $e = (l, \gamma, R, l') \in E$, e is the edge between the locations l and l', with the guard γ and the set of variables to be reset R.*
- *$Inv \in (C(X))^L$ maps an* invariant *to each location.*
- *$Dif \in (\mathbb{Z}^X)^L$ maps an* evolution rate *to each (continuous) variable in each location, dX/dt being the set of derivatives of the variables wrt. time. $dX/dt = (Dif(l, x))_{x \in X}$.*

For short, given a location l, a continuous variable x and $n \in \mathbb{Z}$, we will denote $Dif(l, x) = n$ by $dx/dt = n$ when the location considered is not ambiguous. Note that in our GRN models, $Dif(l, x)$ will always be 0, 1 or –1.

To illustrate the use of a hybrid model, we give the toy example below.

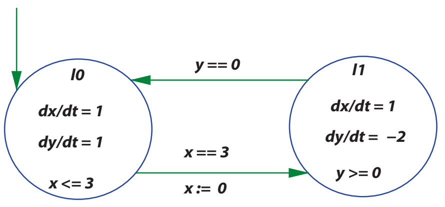


Fig. 6. A toy example of hybrid automaton.

$L = \{l0, l1\}$, $X = \{x, y\}$, $E = \{(l0, \{x == 3\}, \{x\}, l1), (l1, \{y == 0\}, \{\}, l0)\}$,

$Inv = \{(l0, \{x \leq 3\}), (l1, \{y \geq 0\})\}$, $Dif = \{(l0, dx/dt = 1, dy/dt = 1), (l1, dx/dt = 1, dy/dt = -2)\}$.

From such a model, we are interested in the executions which are the sequences of states $q \in Q$ (see definition 6) such that $q = (l_i, (x, y))$ where $i \in \{0, 1\}$ and $(x, y) \in \mathbb{R}^2$. The relation between these states, one with its successor, is either a discrete or a continuous transition, which are defined as follows.

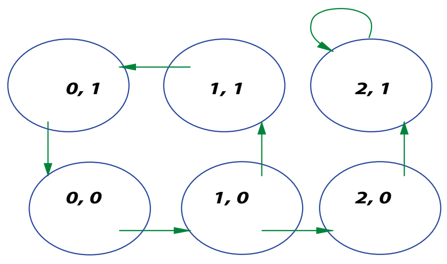


Fig. 7. Example of a GRN discrete model (see fig. 4b).

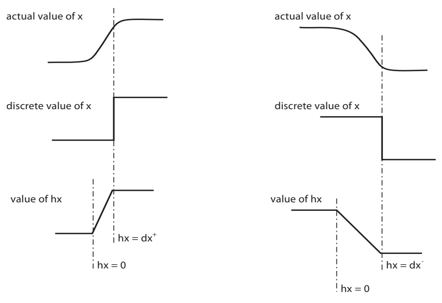


Fig. 8. Modelling of time delays.

Definition 6 (Semantics of an LHA)

The semantics of an LHA H is defined as a Timed Transition System $S_H = (Q, q_0, \rightarrow)$ *where* $Q = L \times \mathbb{R}^X$, q_0 *is the initial state and* $\rightarrow$ *is defined, for* $t \in \mathbb{R}^{\geq 0}$, by:

- *discrete transitions:* $(l, \nu) \rightarrow (l', \nu')$ *iff* $\exists(l, \gamma, R, l') \in E$ *such that*

$$\begin{cases} \gamma(\nu) = \mathtt{true}, & (\gamma \text{ is the guard that must be true for the value } \nu) \\ \nu'(x) = \nu(x) \text{ if } x \notin R, 0 \text{ otherwise}, & (\text{keep the value } \nu \text{ of } x\text{, except if it must be reset}) \\ Inv(l')(\nu') = \mathtt{true} & (\text{the invariant must be true in the target location}) \end{cases}$$

- *continuous transitions:*

$$(l, \nu) \xrightarrow{t} (l, \nu') \text{ iff } \begin{cases} \nu' = \nu + \frac{dX}{dt} \times t, \\ \forall t' \in [0, t], Inv(l)(\nu + \frac{dX}{dt} \times t') = \mathtt{true} \end{cases}$$

Note that, even if $t \in R^{\geq 0}$, the values ν of the variables may be positive or negative according to the signs of dX/dt.

Among the possible executions in the toy example of figure 6, it can be checked that the following (closed) path is an infinite cycle:

$$(\mathtt{l0}, (1, 0)) \xrightarrow{2} (\mathtt{l0}, (3, 2)) \longrightarrow (\mathtt{l1}, (0, 2)) \xrightarrow{1} (\mathtt{l1}, (1, 0)) \longrightarrow (\mathtt{l0}, (1, 0)) \xrightarrow{2} \ldots$$

A Linear Hybrid Model Checker: HyTech

For the implementation and analysis of LHA, a suitable model checker plays an important role. HyTech [8] is a general tool for LHA that was originally developed for the formal verification of embedded or real-time systems. HyTech provides a handy set of *analysis commands* (a manipulation language), which are sufficient for the analysis algorithms we want to implement. We also make use of the *parameter synthesis* peculiarity of HyTech for the core results of this article. Other model checking tools exist, either offering better reachability algorithms than HyTech but having insufficient analysis command language [32], or limiting LHA to the restricted class of timed automata [33].

4 Hybrid Modelling for GRN

The aim of this section is to show how we take into account the *delayed threshold triggering* (according to the schema of fig. 8), in order to go from the purely discrete modelling of Thomas et al. [23] to a hybrid modelling taking into account the temporal behaviour (with durations) in the expression space. Principles are illustrated with the example featuring the mucus production in *P. aeruginosa*, which was introduced in section 2.

4.1 From Expression Space to Temporal Zones

The discrete modelling of the running example, as we already explained (see section 2), is given in the graph of figure 7 where each discrete states of the system stands for the values of the couple of genes (x,y). For example, the state (1,0) represents the situation where, unlike x, y has not yet triggered the threshold 1. Green arrows indicate the transitions between the states.

4.1.1 Delays and Clocks

Since we now consider that the delays for a gene to actually move from the value n to the value n + 1 is not null, we have to switch from a model based on an expression space to a new one which superposes it. This space deals with time intervals. These additional intervals are those of the *clocks* which measure time instead of expression levels. A value belonging to such an interval measures the time elapsed since the last change of discrete expression level.

In figure 8, h_x is the time elapsed since this last change and d_x is the boundary delay after which the discrete expression changes again. There are in fact two delays d^+_x and d^-_x, respectively, for production or degradation of the product of the variable x. The *delay* for x to increase up to the next discrete level is a real parameter d^+_x (>0) and similarly, the delay for x to decrease is $|d^-_x|$ (with $d^-_x < 0$, in order to avoid the confusion between a tendency to increase and a tendency to decrease).

As a matter of fact, for each discrete state (x,y), we expand x and y: x is expanded in the red segment of length $|d^-_x| + |d^+_x|$ which is the model of h_x (the clock of x) (fig. 9a).

The relation between h_x and x is: if the discrete value of x is equal to 1, when x increases, h_x evolves from 0 to d^+_x; when x decreases, h_x evolves from 0 down to d^-_x.

Since y is expanded in the same way, we get (fig. 9b):

The state (1, 0) in the standard Thomas discrete state space is now equipped with a zone (the blue rectangle) which is the (1, 0) location of the temporal model. Thus, the point at the intersection of these red segments stands for the state $(h_x, h_y) = (0, 0)$ in the zone (x, y) = (1, 0).

Hence, we are now mostly concerned with the (h_x, h_y) *temporal zones* and secondarily with the (x, y) abstract expression space. *The consequence is that we rather deal with the linear dynamics of the* h_x*-clocks than with the non-linear dynamics of the expressions of* x.

4.1.2 Temporal Zones

As we mentioned, rectangles represent no longer only a unique symbolic state (e.g. (1, 0), etc.). They stand for a set of states and they are called *temporal zones*, i.e. some regions where time elapses until one of the blue lines is reached (which means that the discrete value of one variable is changed).

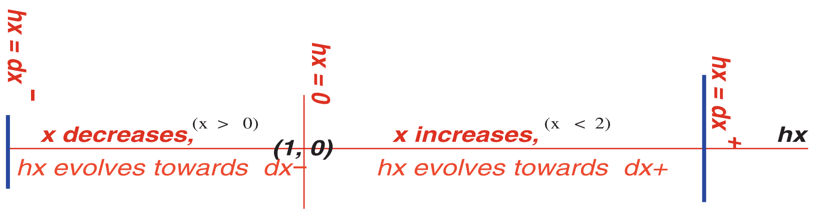
Then, when crossing one threshold (e.g. that of x) and thus entering a new location, there is an immediate jump to the red segment origin of the corresponding clock (e.g. $h_x := 0$, the values of the other clocks are kept), and the dynamics of the clocks are the new ones associated with the new location (these dynamics will be detailed below).

The coordinates of any given point P in a zone are the values (hx, hy) of the two clocks which measure times to reach the limit values $d^-_{x_{10}}$ or $d^+_{x_{10}}$ for h_x and, respectively, $d^-_{y_{10}}$ or $d^+_{y_{10}}$ for h_y.[2]

For example, in figure 10, the point $P = ((1, 0), h_{x_P}, h_{y_P})$ represents a state of the system where $0 < h_{x_P} < d^+_{x_{10}}$ and $0 < h_{y_P} < d^+_{y_{10}}$. It means that, in the future (i.e. for the successors P' of P), either $h_{x_{P'}}$ will reach $d^+_{x_{10}}$ or $h_{y_{P'}}$ will reach $d^+_{y_{10}}$, leading to the following location where the state (x, y) is either (2, 0) or (1, 1).

Hence, we will now discuss the dynamics inside zones, since it is deterministic for the prediction of the future of a point *P*.

9a



9b

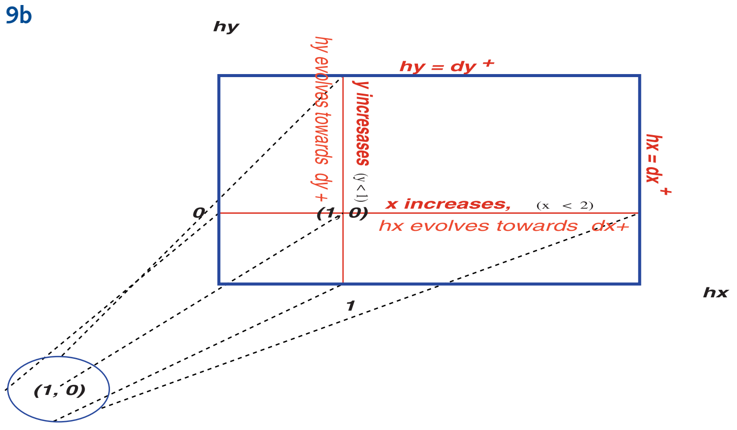


Fig. 9. Expansion of the location (x, y) = (1, 0) in the temporal zone (h_x, h_y).

[2]Indices of d_x and d_y are related to the name of the location (1, 0).

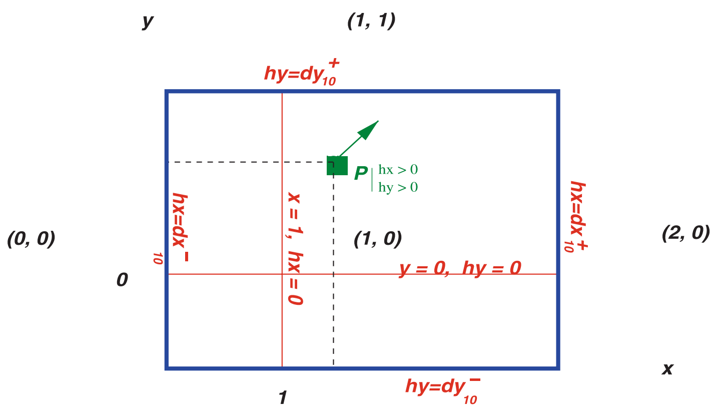


Fig. 10. Example of 'temporal' modelling for a GRN (partial sight).

4.1.3 Dynamics in the Temporal Zones

According to the LHA definition (see definition 6 in section 3), the hybrid automaton (L, l_0, X, E, Inv, Dif), which is the model of the regulatory network, is built from the discrete automaton $(S, \rightarrow)$ in the following way.

The locations l of the former automaton are the states s of the latter ($L = S$). X is the set of real-valued variables h_i for each gene i. Thus, if n is the number of genes, the discrete space is the set L of locations s, the *temporal space* is the set of points $P_s = h_{i\in[1 \dots n]} \in \mathbb{R}^n$ in each location s.

Dynamics of the clocks in each location s give a view of the tendency for each gene in this location. They are given by the values of dh_i/dt and they belong to the set $\{-1, 0, +1\}$, which means that the clock of the gene i is either decreasing, or staying at the same level, or increasing. This tendency takes into account the next discrete transitions, each of which reflecting the closer future dynamics for each gene. Indeed, since these dynamics were abruptly desynchronized for the discrete modelling (so that only the most immediate change was kept), we now have to look several steps ahead in order to get the whole actual tendency of all the genes. This means that we have to take care of the next n transitions of the discrete model (with priority for the closer transitions).

Formally, if $S \subset L$ is any set of locations, $post(S)$ is the set of the next locations from any location of S (the target locations of each outcoming transition of all the states of S) and S^k is the set $post^k(S) = post(post^{k-1}(S))$. We note $S = \{s\}$ the set with the single location s, the derivatives of the clock variables (vector $\overrightarrow{dh}/dt$) in s are given by:

for (k := n − 1 down to 0) do:

$$\{\forall (s_k \in S^k \text{ and } s_{k+1} \in post(S^k) \text{ such that } s_k \rightarrow s_{k+1}): dh_i/dt \leftarrow (s_{k+1}[i] - s_k[i])\} \quad (1)$$

where $s_k[i]$ denotes the i-th component of vector s_k, i.e. the abstract expression level of gene i associated with the location s_k (and similarly for $s_{k+1}[i]$).

The edges between locations are the transitions of the discrete model together with their respective guards (the reached time delay) and the reset statements (the corresponding clock reset). Thus, for any edge $e = (s, \gamma, R, s')$ in E:

$\exists i \in [1..n]$ such that
$(s[i] = k \text{ and } s'[i] = k\ \alpha\ 1_{(\alpha \in \{+,-\})})$:

$$\begin{cases} \gamma \text{ is : } & h_i == d^{\alpha}_{i_s} \\ and & \\ R \text{ is : } & h_i \leftarrow 0 \end{cases}$$

The whole space of the temporal zones, together with the speed directions of clocks, is thus supplemented in figure 11, where the direction in the zone (2, 1) is the null vector (drawn as a green asterisk).

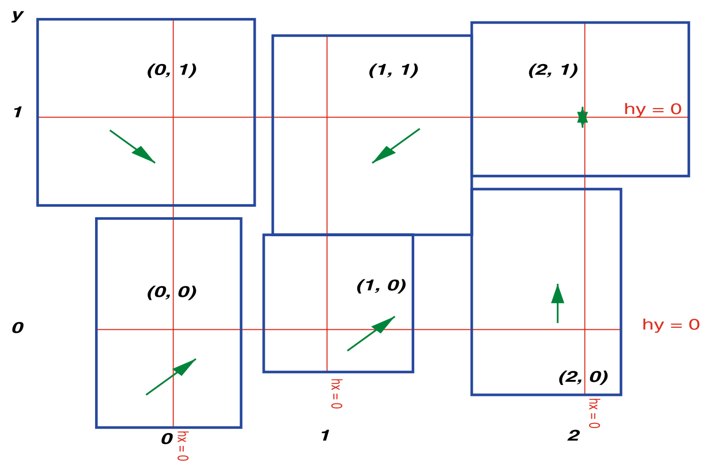


Fig. 11. Example of 'temporal' modelling for a GRN (full sight)

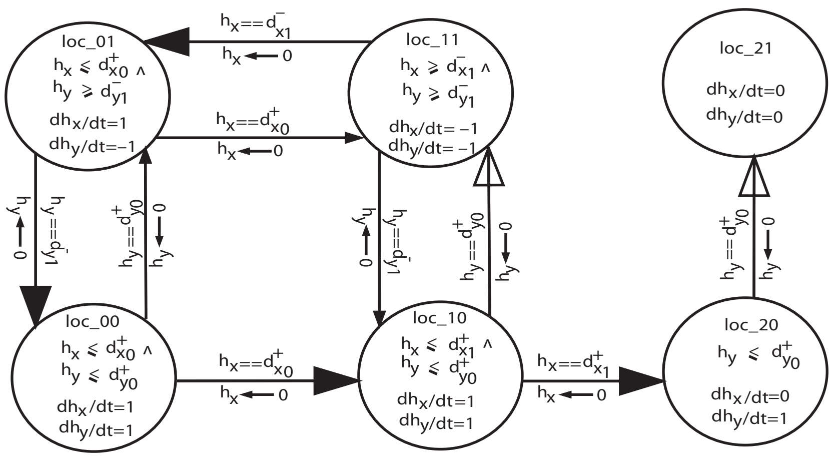


Fig. 12. Hybrid model for the example of *P. aeruginosa* (bold arrows represent the transitions of the discrete model).

Definition 7 (Temporal State Space)

The set of all the temporal zones derived from the discrete model of a GRN is called the temporal state space of a GRN (fig. 11).

4.2 Hybrid Model of the Running Example

We now come back again to the regulatory network of the example of *P. aeruginosa* (see section 2, fig. 2) and we show the hybrid model it leads to, according to the procedure explained above.

As we have already explained, we associate a clock h_v to each variable $v \in \{x, y\}$ of the example. The clock speeds in the locations and the discrete transitions are set according to the rules (1) and (2) given in the previous section. More precisely, the vector of derivatives of the clocks is found in the discrete transition system in figure 7. For example, in location (0, 1), since we know that the next two discrete transitions are respectively (in the increasingly close order) such that x increases and y decreases, we get (according to the formula (1)) that: $dh_x/dt = 1$ and $dh_y/dt = -1$. Also, in location (1, 0), since there is an immediate tendency for x and y to increase, we get that: $dh_x/dt = 1$ and $dh_y/dt = 1$. Finally, in location (2, 1), since the tendency is to stay in this location, we have: $dh_x/dt = 0$ and $dh_y/dt = 0$.

Note that, in figure 12, there might be some indeterminism if, for example in loc_10, h_x becomes equal to $d^+_{x_{10}}$ at the same moment when hy becomes equal to $d^+_{y_{10}}$. There can be either the translation towards loc_20 or towards loc_11.

4.3 Trajectories

The dynamics of expression levels are made of a sequence of transitions in the temporal state space. It can be explained in the following way. There are two types of progression which alternatively take place:

- *a continuous transition*, which stands for time elapsing in a zone (along one direction which is the same for any point in the zone), until one of the blue borders of the zone is reached
- *a discrete transition*, which stands for the instantaneous change of zone, and which leads to the appropriate clock reset (and which gives the new values of the discrete expression levels).

Any sequence of points related by such transitions makes up what is called the *trajectory*.

An example of trajectory is given in figure 13 (layout is thick purple) to show the whole behaviour from a given point P in the zone (1,0). The point P is unspecified but arbitrarily such that $0 < h_{x_P} < d^+_{x_{10}}$ and $0 < h_{y_P} < d^+_{y_{10}}$. The time elapsing (according to the diagonal segment (plain line)) at first brings to the crossing of the threshold of y with passage in the zone (1,1) and jump to the 0_{h_y}-axis

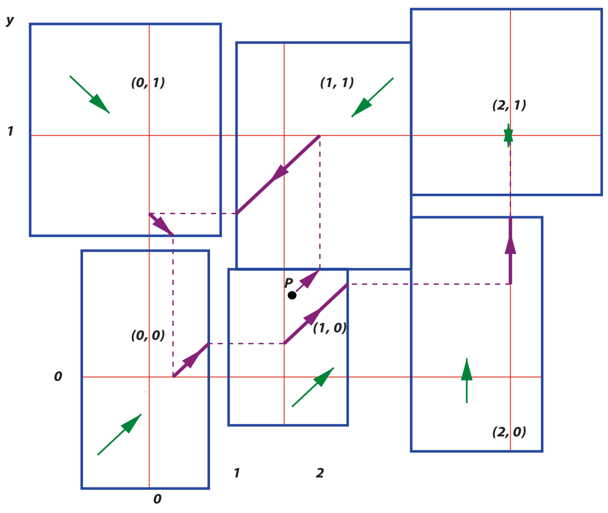


Fig. 13. Trajectory starting from a temporal point P and showing its temporal behaviour.

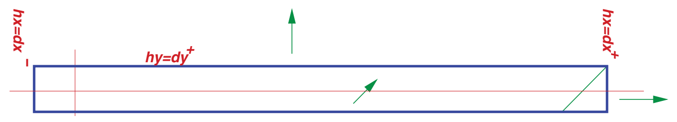
(with resetting of h_y): instantaneous transition drawn as a dashed line, and so on (fig. 13).

This alternation of continuous temporal transitions with instantaneous discrete transitions (which correspond to the change of zone) is actually underlying an LHA in which locations are temporal zones, and continuous variables embodies the expressions which are modeled by *'clocks'* that can be either suspended or going onwards or backwards, and measure the times needed to reach the related thresholds.

4.3.1 Various Types of Behaviours

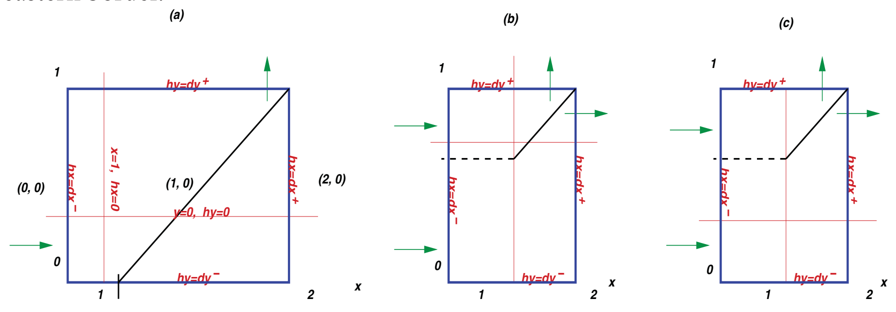
As one can see in figure 13, some trajectories passing through the zone (1, 0) will exit through the zone (1, 1), and some other trajectories will exit through the zone (2, 0). Indeed, since they depend upon the a priori unknown delay parameters, we neither know the shapes of the blue rectangles nor their dimensions nor the exact position of these rectangles with respect to the red lines. However, these shapes, dimensions and positions have a direct impact on the trajectories, i.e. on the dynamics of the modelled system and this is therefore a major clue in the analysis of the system. Some issues have then to be discussed:

- Even if the dynamics inside a rectangle are diagonal, the fact that this rectangle be very elongated (horizontally or vertically) may cause trajectories much more likely to exit through one zone or through the other zone:



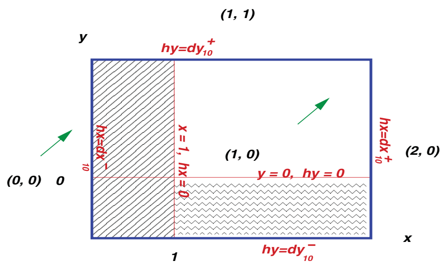
- Moreover, in many zones, the position of the diagonal frontier reaching some specific corner of the zone is a deterministic feature and several cases may occur (various combinations of which the most interesting ones are recapitulated below[3]):
  - *the black diagonal (and the dashed line continuation) intersects the southern border of the zone (case* (a)*):* all the trajectories will leave by the northern side
  - *the black diagonal (and the dashed line continuation) intersects the western border of the zone*
    - * *either in the part* $h_y < 0$ *(case* (b)*)*
    - * *or in the part* $h_y > 0$ *(case* (c)*)*

In these cases, some trajectories will exit through the northern side, others by the eastern border.



The only difference between case (b) and case (c) in the figure above is that, in the situation (b), the trajectory *must exit* by the northern side if h_y is non-negative, and it *may exit* by the eastern side, otherwise (situation (c)).

- Finally, some parts of a blue rectangle may be unreachable. For example, in the zone (1, 0):



The left subzone of the rectangle is not reachable for both reasons: (1) the only entry into the zone (1, 0) is from the west and it was done together with an immediate switch to the $h_x = 0$ red segment, and (2) the dynamics are such that h_x is rising (as well as h_y).

Also, even if the GRN does not indicate a decrease in a substance on the lowest level, there is nevertheless a part where $h_y < 0$, but obviously, this subzone will never be reached (since h_y increases and it was not negative in the previous zone).

4.3.2 Divergences and Invariance Kernels

The aim is now to look at some interesting features of the possible trajectories. Indeed, as it was introduced at the end of section 2, we are interested in trajectories which are either cyclic or diverging. Figure 14 illustrates these different types of trajectories on slightly different models (the difference lying in the values of the parameters d_i^{α}). We observe now that some behaviours are *cyclic* (the green ones), i.e. they are confined in what we call a *closed path* (more precisely, they go infinitely often and sequentially in the

[3] Without loss of generality, we only look here at a zone such as (1, 0), with the frontier reaching the upper right corner, and the entry in this zone is done by the leftmost side.

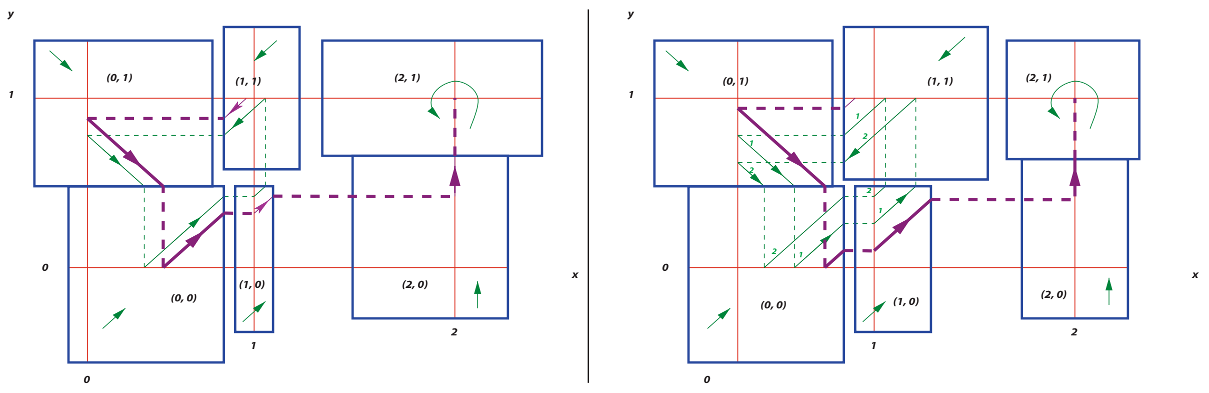


Fig. 14. Invariant trajectories (thin green) and divergent trajectory (thick purple).

states (1, 1), (0, 1), (0, 0), and (1, 0)). Other behaviours escape from this cycle to move away eventually definitively towards the state (2, 1) (the purple ones). In this last case, we speak of *divergence*, with respect to a cycle.

As we will see in section 5 on automatic analysis of the trajectories, we are able to establish these specific sets of trajectories. More precisely, the greatest set of points such that any trajectory, starting from one of these points, remains in this set is of great interest and it is called the '*invariance kernel*' [34]. This kernel is expressed as a constraint on the delay parameters, which can be algorithmically synthesized. In contrast, the outside of this kernel is composed of divergent trajectories which lead towards another region, which we call a '*capture basin*'. These phenomena are illustrated with the examples of figure 14, and they are further analyzed in subsection 5.1.

4.4 Discussion

We come now to a short discussion about our approach of hybrid modelling for temporal networks. A biologically important advantage of our approach, which has been a technically rather difficult feature to preserve, is that it takes into account all accumulated increasing and decreasing phases in the behaviour of a system.

Accumulation

An important feature of our modelling approach is that it records the accumulations along trajectories. These accumulations can be monotone or not. It means that one can successively pass through a set of states leading to the

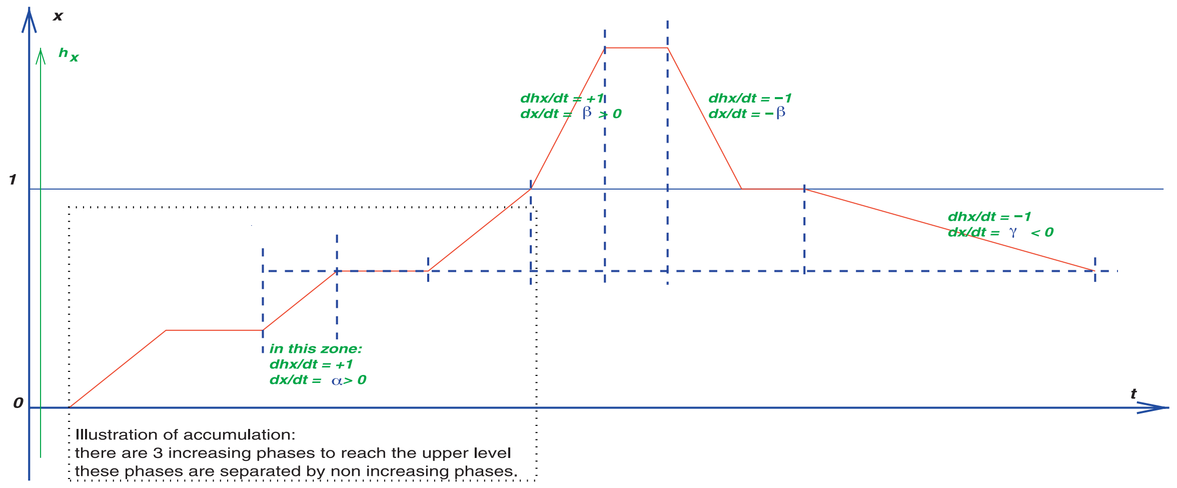


Fig. 15. Accumulations for one clock associated with one specific gene (activation/decay rates α, β, γ are intermediate values which are deduced from the delays and thresholds).

progress of a variable in one direction, then immediately afterwards in another direction, or temporarily stable (see fig. 15). This corresponds to the fact that, when x has a given value, it is in one of the various zones which correspond to this value. But it can pass through several zones because of the changes of other variable values, and thus, it may have an behaviour which is not always the same (e.g. in fig. 15: at first, h_x increases, then it remains stable, and then it increases again). Anyway, the reached expression level is not lost.

It can be noted, however, that the compensation of a beginning of synthesis (thus, a decrease) is not done at the appropriate speed of degradation, since it is indeed at exactly the opposite speed of this previous synthesis. This issue actually constitutes a kind of small inaccuracy.

5 Analysis

In this section, we show that the major advantage of modelling regulatory interactions between genes with delays is that it makes it possible to algorithmically analyze more precisely the behaviours of a system. We first present the notion of invariance kernel along with an algorithm to find such kernels. Then, we give the results of the analysis of the hybrid model of the running example of *P. aeruginosa*. These results are the computed intervals of initial conditions such that a trajectory starting from any initial region eventually converges to a cycle.

For the sake of simplicity, from now on, we only deal with fewer delay parameters, assuming that all $d^{\alpha}_{x_{ij}}$ are equal, whatever the actual value of j is, and similarly for all $d^{\alpha}_{y_{ij}}$, whatever the actual value of i is. The major consequence is that, from now on, zones are adjacent (see adjacent blue rectangles in fig. 16 and 17).

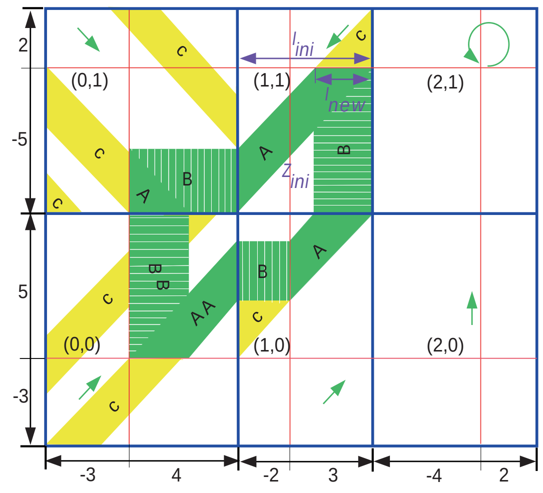


Fig. 16. Phase portrait for invariance kernel.

5.1 Cycles and Invariance Kernels

The introduction of delay parameters and clocks in the GRN modelling can help to find the stable and unstable cycles, the results of which have been shown [35] where delays have not been taken into account. Thomas and D'Ari [36] affirm: 'the set of initial conditions that lead to stay in the cycle plays a vital role in the dynamics of the network', and it was the former work of these conditions to stay in a cycle. The solution of this generalized problem of finding conditions for the infinite cycle was addressed later [34, 37], when the authors introduced the idea of *invariance kernel* which is the set of initial conditions of which all executions remain in the constrained set forever.

The initial conditions of the clock variables in the temporal state space might not be in the cycle. It is therefore important to analyze which initial values can lead the system into the cycle, to determine the constraints on the time delays and clocks which will effectively allow the system to enter the cycle and then to verify that these constraints are compatible with the conditions for remaining in the cycle.

These studies lead us to the theory of viability.

5.1.1 Viability and Invariance

Viability theory is an area of mathematics concerned with the viable behaviour of controlled dynamic systems [38]. A system execution is considered viable if the system trajectory remains within a prescribed region, the *viability domain* [34]. Roughly speaking, the viability domain is a set of states such that there exits at least one execution, from all the initial conditions, that remains viable in this set. The basic problem that viability theory attempts to solve is whether a control strategy exists that prevents the system from leaving the viability domain. The invariance kernel [39] for such a problem is the set of all initial conditions, for which such a strategy exists.

Definition 8 (Invariance Kernel)

Let K be a subset of the temporal state space of a GRN. A set K is invariant if for any $x \in K$, every trajectory starting in x is viable in K. Given a set K, its largest invariant subset is called the invariance kernel of K.

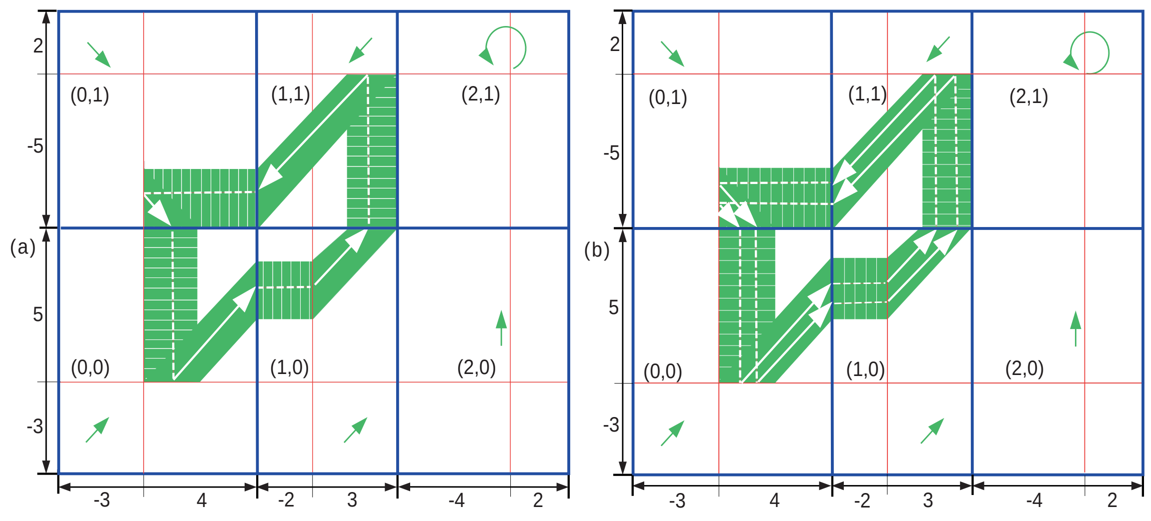


Fig. 17. Plain and nested cycles within the phase portrait for invariance kernel.

5.1.2 An Algorithm to Find the Invariance Kernel

We define here two sets of conditions (constraints upon the coordinates of origin points) for the trajectories leading to infinite cycles: the *necessary conditions* and *the sufficient conditions*. These sets of conditions define specific regions.

First, the set of necessary conditions is any region which is a subset of a temporal zone associated with a location of a closed path (see section 4.3.2), because some trajectories having origins outside this set will neither remain invariant nor converge to an infinite cycle. Of course, it is not a sufficient condition, since additionally, some trajectories, starting from some points in this zone, may not remain within the invariance kernel. The set of sufficient conditions is therefore a subset of the previous set, such that all the trajectories from all the points in this set always converge to invariant cycles. It can be any arbitrarily selected subregion of the chosen initial zone as soon as it is greater or equal (w.r.t. inclusion) than the searched region of the invariance kernel in this zone, but some regions may not intersect with the invariance kernel. It is a good idea to take as initial region a set of states which is orthogonal to the direction of the reset of the immediate previous discrete transition. For example, in figure 16, the region where $h_y = 0$ in the zone (1, 1) is appropriate since h_y has just been reset.

Algorithm 1 (below) finds the set of conditions that leads to a region standing

Algorithm 1. Set of conditions that leads to a cyclic region

```
1:   $Z_{ini} := $ <Arbitrary initial zone>
2:   $Z_{new} := Z_{ini}$
3:   repeat
4:     $Z_{old} := Z_{new}$
5:     $Z_{new} := pre(Z_{new})$
6:   until $(Z_{new} = Z_{old}) \vee empty(Z_{new})$
7:   if not $empty(Z_{new})$ then
8:     $I_{ini} := $ <arbitrary region in the initial zone>
9:     $I_{new} := I_{ini}$
10:    $P_{reg} := I_{new}$
11:    repeat
12:      $I_{old} := I_{new}$
13:      repeat
14:        $P_{reg} := pre(P_{reg})$
15:      until (not empty $(I_{ini} \cap P_{reg})) \vee empty(P_{reg})$
16:      $P_{reg} := P_{reg} \cap I_{ini}$
17:      $I_{new} := I_{new} \cap P_{reg}$
18:    until $(I_{new} = I_{old}) \vee empty(P_{reg})$
19:  end if
```

for an invariance kernel. Thus, it proceeds in the following way. First, it checks the necessary condition for one arbitrarily chosen zone Z_{ini}, and then, it iteratively finds the predecessor regions, P_{reg}, of the initial region. The fixpoint, I_{new}, is therefore a subregion that stands for the *sufficient conditions* for an invariance kernel. Any trajectory from a point in this subregion converges to an infinite cycle.

Illustration of the Algorithm Execution

Let us illustrate the execution of algorithm 1, according to its HyTech implementation (see Appendix 7.2).

We arbitrarily choose the initial zone to be (1, 1) and, according to the first part of the algorithm (lines 3–6), the closed path reveals to be: $(1, 1) \rightarrow (0, 1) \rightarrow (0, 0) \rightarrow (1, 0) \rightarrow (1, 1)$ ... The results of these computations are given in table 1.

Then, we arbitrarily set the initial region to be: $h_y = 0 \wedge (-2 \leq h_x \leq 3)$ and the computations of the nested iterations (lines 11–18) are recorded in table 2.

Here, C1 = (not empty($I_{ini} > P_{reg}$) $\vee$ empty(P_{reg})) and C2 = (($I_{new} = I_{old}$) $\vee$ empty(P_{reg})).

Now we have an algorithm to find out cycles, we are interested in the generalization to get the set of all the cycles which is called the *phase portrait for invariance kernel.*

Definition 9 (Phase Portrait for Invariance Kernel)

A phase portrait is a plot of multiple trajectories corresponding to different initial conditions leading to cycles.

As a matter of fact, a phase portrait for invariance kernel is the union of all trajectories which have their origins in the largest initial interval. We show in section 5.3 the results of invariance kernel computations from some initial regions, each of which is the greatest interval in some arbitrary direction in the different zones.

The green area (with labels *A*s and *B*s) of figure 16 shows the phase portrait for the invariance kernel. Here, it appears to be a non-convex polygon with a hole in the centre. Different subregions of the phase portrait are labelled with letters *A* and *B*, which respectively show the continuous (inclined) and discrete (rectangular) subregions within the phase portrait. Hence, in figure 16, *A* followed by *B* demonstrates that continuous transitions are followed by discrete transitions and vice versa. Regions labelled with the letter *C* show the region from where the trajectories will converge to the invariance kernel.

5.2 Types of Cycle Trajectories

It appears that invariance kernels consist of two exclusive types of cycles. We called these cycles *plain* or *nested* cycles.

Definition 10 (Rotation)

A rotation is a closed path in the temporal zone space which begins in any zone and later comes back for the first time in this zone (not necessarily at the same point).

The difference between a rotation and a cycle in the temporal state space is that rotation does not take clock values into account.

An example of a rotation is the piece of trajectory: $(1, 1) \rightarrow (0, 1) \rightarrow (0, 0) \rightarrow (1, 0) \rightarrow (1, 1)$, whatever the values of hx and hy are in the first and in the second passage in the zone (1, 1).

Definition 11 (Plain Cycle and Nested Cycle)

A plain cycle is a single rotation trajectory that starts from a point in a phase portrait and then finally arrives at the same point.

A nested cycle is a more than one rotation trajectory that starts from a point in a phase portrait and then finally arrives at the same point.

Figure 17a shows a plain cycle, and figure 17b shows a nested cycle in the phase portrait of the invariance kernel.

5.3 Model Checking of the Running Example

We do the HyTech [8] analysis of the hybrid model of figure 12 by using algorithm 1 to first check that the chosen initial zone is a necessary condition for the invariance kernel and then to find the sufficient condition for the invariance kernel in the form of clocks and delay constraints. The HyTech file of the model in Appendix 7.1 and the HyTech file of the analysis commands in Appendix 7.2 show the implementations of the hybrid automaton and the algorithm, respectively.

The results obtained when choosing the initial interval in different zones of the temporal model are shown in the following two sections.

5.3.1 Results with Unvalued Delay Parameters

As already mentioned, *parameter synthesis* is one of the peculiarities of HyTech. Parameters in HyTech are treated as unknown constants. The HyTech results, as shown in the 'constrained region' column of table 3, have been synthesized with delays as parameters.

These rough results may seem hard to understand, but they give some hints which are of interest about the achieved relations between the delay parameters that cycles exist. They become more legible when we know the values of at least some parameters (see subsection 5.3.2).

The columns 'Z_{ini}' and 'I_{ini}' of table 3 give some initial zones and some initial regions, respectively. These initial conditions were arbitrarily selected. The 'constrained region in the phase portrait' column shows the constraints obtained when selecting some initial intervals in the different zones. The constraints represent a region which is a set of initial conditions leading to trajectory cycles of the invariance kernel. The last column of table 3 shows the execution time in seconds on a Pentium(R)-4 3.20 GHz machine.

Table 1. An example of the execution of the first loop of algorithm 1 for $Z_{ini} = (1,1)$

Pass	Z_{old}	$Z_{new} = Pre(Z_{new})$	$(Z_{new} = Z_{old}) \vee \text{empty}(Z_{new})$
1	{(1,1)}	{(1,0)}	false
2	{(1,0)}	{(0,0), (1,0)}	false
3	{(0,0), (1,0)}	{(0,1), (0,0), (1,0)}	false
4	{(0,1), (0,0), (1,0)}	{(0,1), (0,0), (1,0), (1,1)}	false
5	{(0,1), (0,0), (1,0), (1,1)}	{(0,1), (0,0), (1,0), (1,1)}	true

Table 2. An example of the execution of the second loop of algorithm 1 for $I_{ini} = (h_y = 0 \wedge (-2 \le h_x \le 3))$ and $Z_{ini} = (1,1)$

Pass	I_{old}	P_{reg}	C1	I_{new}	C2
1	loc_11 : $h_y = 0 \wedge (-2 \le h_x \le 3)$	loc_10 : $h_y \le 5 \wedge h_y \ge h_x + 2 \wedge h_y \le h_x + 7$	false	loc_11 : $h_y = 0 \wedge (1 \le h_x \le 3)$	false
		loc_00 : $h_x \le 4 \wedge h_y \le h_x + 1 \wedge h_x \le h_y + 2$	false		
		loc_01 : $h_y + 5 \ge 0 \wedge 0 \ge h_x + h_y + 3 \wedge h_x + h_y + 6 \ge 0$	false		
		loc_11 : $hx + 2 \ge 0 \wedge h_x \le h_y + 3 \wedge h_x \ge h_y + 1$	true		
2	loc_11 : $h_y = 0 \wedge (1 \le h_x \le 3)$	loc_10 : $h_y \le 5 \wedge h_y \ge h_x + 2 \wedge h_y \le h_x + 4$	false	loc_11 : $h_y = 0 \wedge (1 \le h_x \le 3)$	true
		loc_00 : $h_x \le 4 \wedge h_y \le h_x \wedge h_x \le h_y + 2$	false		
		loc_01 : $h_y + 5 \ge 0 \wedge 0 \ge h_x + h_y + 3 \wedge h_x + h_y + 5 \ge 0$	false		
		loc_11 : $h_x + 2 \ge 0 \wedge h_x \le h_y + 3 \wedge h_x \ge h_y + 1$	true		

Table 3. Table of constraints (unvalued parameters)

Z_{ini}	I_{ini}	Constrained region in the phase portrait	Time
(0,0)	$h_y = 0 \wedge d^-_{x_0} \le h_x \le d^+_{x_0}$	$h_y = 0 \wedge h_x \ge 0 \wedge h_x + d^+_{y_0} + d^-_{y_1} \le d^+_{x_0} + d^-_{x_1} \wedge h_x + d^-_{x_0} + d^+_{y_0} + d^-_{y_1} \le d^+_{x_0} + d^-_{x_1} \wedge$ $h_x + d^-_{y_1} \le d^-_{x_1} \wedge h_x + d^-_{y_1} \le 0 \wedge h_x + d^+_{y_0} \le d^+_{x_0} + d^+_{x_1} \wedge d^-_{x_1} \le h_x + d^+_{y_0} + d^-_{y_1} \wedge$ $d^-_{x_1} \le h_x + d^+_{x_1} + d^-_{y_1} \wedge d^+_{x_0} + d^-_{x_1} \le h_x + d^+_{y_0} \wedge d^+_{x_0} \le h_x + d^+_{y_0}$	0.79
(0,1)	$h_x = 0 \wedge d^-_{y_1} \le h_y \le d^+_{y_1}$	$h_x = 0 \wedge h_y \le 0 \wedge d^+_{x_0} + d^-_{x_1} + d^-_{y_1} \le h_y + d^+_{y_0} \wedge h_y + d^+_{y_0} \le d^+_{x_0} + d^-_{x_1} \wedge h_y \le d^+_{x_0} + d^-_{y_1} \wedge$ $h_y + d^+_{y_0} \le d^+_{x_0} + d^+_{x_1} + d^-_{y_1} \wedge h_y \le d^-_{x_1} \wedge d^+_{x_0} + d^-_{x_1} + d^-_{y_1} \le h_y + d^+_{y_0} + d^+_{y_1} \wedge$ $d^+_{x_0} + d^-_{y_1} \le h_y + d^+_{y_0} \wedge d^-_{x_1} \le h_y + d^+_{x_1} \wedge d^-_{x_1} \le h_y + d^+_{y_0}$	0.76
(1,0)	$h_x = 0 \wedge d^-_{y_0} \le h_y \le d^+_{y_0}$	$h_x = 0 \wedge h_y \ge 0 \wedge h_y + d^-_{x_1} \le d^+_{x_0} + d^+_{y_0} + d^-_{y_1} \wedge h_y + d^-_{y_0} + d^-_{x_1} \le d^+_{x_0} + d^+_{y_0} + d^-_{y_1} \wedge$ $h_y + d^-_{x_1} \le d^+_{x_0} + d^+_{x_1} + d^-_{y_1} \wedge h_y \le d^+_{x_0} \wedge h_y + d^-_{x_1} \le d^+_{y_0} \wedge d^+_{x_0} + d^-_{y_1} \le h_y + d^-_{x_1} \wedge$ $d^+_{x_0} + d^-_{y_1} \le h_y \wedge d^+_{y_0} + d^-_{y_1} \le h_y + d^-_{x_1} \wedge d^+_{y_0} \le h_y + d^+_{x_1}$	0.69
(1,1)	$h_y = 0 \wedge d^-_{x_1} \le h_x \le d^+_{x_1}$	$h_y = 0 \wedge h_x \ge 0 \wedge h_x + d^+_{x_0} + d^-_{y_1} \le d^+_{y_0} + d^-_{x_1} \wedge h_x + d^+_{x_0} + d^-_{y_1} \le d^+_{y_0} \wedge h_x \le d^+_{y_0} \wedge$ $h_x + d^-_{y_1} \le d^-_{x_1} \wedge d^+_{y_0} \le h_x + d^+_{x_0} \wedge d^-_{x_1} \wedge h_x + d^+_{x_0} + d^-_{y_1} \wedge d^+_{y_0} + d^-_{x_1} \le h_x + d^+_{x_0} + d^+_{x_1} + d^-_{y_1}$	0.64
(2,0)	$h_x = 0 \wedge d^-_{y_0} \le h_y \le d^+_{y_0}$	$\emptyset$: (2,0) is not within a closed path	0.46

Table 4. Table of constraints (valued parameters)

Z_{ini}	I_{ini}	Constrained region in the phase portrait	Time
(0,0)	$h_y = 0 \wedge d^-_{x0} \leq h_x \leq d^+_{x0}$	$h_y = 0 \wedge 0 \leq h_x \leq 2$	0.28
(0,1)	$h_x = 0 \wedge d^-_{y1} \leq h_y \leq d^+_{y1}$	$h_x = 0 \wedge -5 \leq h_y \leq -3$	0.26
(1,0)	$h_x = 0 \wedge d^-_{y0} \leq h_y \leq d^+_{y0}$	$h_x = 0 \wedge 2 \leq h_y \leq 4$	0.26
(1,1)	$h_y = 0 \wedge d^-_{x1} \leq h_x \leq d^+_{x1}$	$h_y = 0 \wedge 1 \leq h_x \leq 3$	0.25
(2,0)	$h_x = 0 \wedge d^-_{y0} \leq h_y \leq d^+_{y0}$	Ø: (2,0) is not within a closed path	0.28

5.3.2 Results with Delay Values

In this section, we present in table 4 the HyTech results with valued delay parameters, in order to give a clear picture of the work presented in this paper. The delay values are shown in the following table

d^-_{x0}	–3	d^-_{y0}	–3
d^+_{x0}	+4	d^+_{y0}	+5
d^-_{x1}	–2	d^-_{y1}	–5
d^+_{x1}	+3	d^+_{y1}	+2

(the values of the delays $d^-_{x_2}$ and $d^+_{x_2}$ are not useful since we know that there is no invariance kernel involving the zones (2, 0) and (2, 1)). *They correspond to the exact height and width of the rectangles in* figure 16.

The phase portrait, as shown in figure 16, is drawn according to the results of table 4. The meaning of columns Z_{ini}, I_{ini}, 'Constrained region in the phase portrait' and 'Time' of table 4 is explained in the previous section.

6 Conclusion

Our modelling framework is a modest contribution to the theory of GRN as introduced by René Thomas (based on discrete mutivalued expression levels and with purely qualitative predictions). We have shown how to go further than the formal modelling framework proposed by Bernot et al. [6], while entirely preserving the computer-aided method of qualitative mathematical model discovery. Our modelling approach takes into account delays in gene interactions. One of our main goals was to preserve the ability to perform automated model checking (using HyTech), with a sensible treatment of delays in addition to the usual treatment of expression levels. Linear approximations have been necessary to achieve this goal; nevertheless, the proposed approach constitutes a valuable improvement since it provides more accurate abstractions of biological phenomena: some models which are neglected by the purely discrete approach without delays have been shown to be acceptable models. We illustrated the complete process through the example of *P. aeruginosa* and we got some interesting results for discriminating between different behaviours: infinite cycles or absorption inside a capture basin.

Other case studies have been done [40], among them the well-known Enterobacteria phage λ example which deals with 4 genes and leads to a model of 48 zones. Using the hybrid modelling depicted in this paper, we were able to establish conditions regarding the delays which are associated with the 'lysogenic pathway' or with the 'lytic cycle'. Another experiment about the evolution of *Escherichia coli* bacteria is at present being performed, which reinforces the biological relevance of the model.

The major feature of our modelling approach is 3-fold: (1) it is consistent with that of René Thomas (it leads even strictly to the same model if we set the values of the delay parameters appropriately either to zero or to infinity); (2) we can determine more precise trajectories of the expression levels than with a discrete model, since it takes care of the relations between the production or decay delays, and (3) furthermore, we can account for accumulations of these productions and decays.

7 Appendix

In the subsequent sections we present the HyTech file. The HyTech file consists of two parts: (1) the automaton part that implements the hybrid model of figure 12 and (2) the analysis command part that implements algorithm 1.

7.1 Hybrid Automaton

```
-- pseudomonas.hy
--Valued parameters
define(dpx0, 4)
define(dnx0,-3)
define(dpy0, 5)
define(dny0,-3)
define(dpx1, 3)
define(dnx1,-2)
define(dpy1, 2)
define(dny1,-5)
var
hx,hy :analog; -- Clock variables
k,n: discrete; -- Discrete variables which are used to describe  the discrete  model of pseudomonas
--The define statements must be removed or commented while using  unvalued parameters
--dpx0,dnx0,dpy0,dny0,dpx1,dnx1,dpy1,dny1: parameter; -- unvalued parameters
--/ Hybrid model
-- Discrete transition takes place in the discrete model when the same guard is satisfied in the following
-- temporal  model. Here, this transition is identified by assigning a discrete value to the variable k.
automaton auto
synclabs: ;
initially loc_11;
-----------------------------
--/ for the location 00
 loc loc_00: while hx<=dpx0 & hy<=dpy0 wait {dhx=1,dhy=1}
 when hx=dpx0 do {hx'=0,k'=k+1} goto loc_10;
 when hy=dpy0 do {hy'=0} goto loc_01;
-----------------------------
--/ for the location 10
 loc loc_10: while hx<=dpx1 & hy<=dpy0 wait {dhx=1,dhy=1}
 when hx=dpx1 do {hx'=0,k'=k+1} goto loc_20;
 when hy=dpy0 do {hy'=0,k'=k+1} goto loc_11;
-----------------------------
--/ for the location 01
 loc loc_01: while hx<=dpx0 & hy>=dny1 wait {dhx=1,dhy=-1}
 when hx=dpx0 do {hx'=0 } goto loc_11;
 when hy=dny1 do {hy'=0,k'=k+1} goto loc_00;
-----------------------------
--/ for the location 11
 loc loc_11: while hx>=dnx1 & hy>=dny1 wait {dhx=-1,dhy=-1}
 when hx=dnx1 do {hx'=0,k'=k+1} goto loc_01;
 when hy=dny1 do {hy'=0} goto loc_10;
-----------------------------
--/ for the location 20
 loc loc_20:  while  hy<=dpy0 wait {dhx=0,dhy=1}
  when hy=dpy0 do {hy'=0,k'=k+1} goto loc_21;
-----------------------------
--/ for the location 21

 loc loc_21: while asap wait {dhx=0,dhy=0}
-----------------------------
end -- of automaton
```

7.2 Analysis commands

```
-- Analysis commands
var
ini_zone,new_zone,old_zone,pre_zone,ini_reg,I_old,I_new,I_ini,pre_reg: region;
-------------------------------------------------------------------------------------
-- To find  a region of the initial conditions that lead to a cyclic region :
--  * we suppose the initial region to be a delay axis  in the initial  zone
--  * we verify the initial zone to be the location of a closed path
--  * and finally, we find the intersection of the initial region with the
--    predecessor images of this region until the fixpoint of iteration
-------------------------------------------------------------------------------------
--ini_zone and ini_reg represent the initial zone and the initial region respectively
--ini_zone:=loc[auto]=loc_00;
--ini_reg:=loc[auto]=loc_00 & hy=0 & hx>=dnx0 & hx<=dpx0;--horizontal axis in the zone (0,0)
--ini_zone:=loc[auto]=loc_01;
--ini_reg:=loc[auto]=loc_01 & hx=0 & hy>=dny1 & hy<=dpy1;--vertical axis in the zone (0,1)
--ini_zone:=loc[auto]=loc_10;
--ini_reg:=loc[auto]=loc_10 & hx=0 & hy>=dny0 & hy<=dpy0;
ini_zone:=loc[auto]=loc_11;
ini_reg:=loc[auto]=loc_11 & hy=0 & hx>=dnx1 & hx<=dpx1;
--ini_zone:=loc[auto]=loc_20;
--ini_reg:=loc[auto]=loc_20 & hx=0 & hy>=dny0 & hy<=dpy0;
I_ini:=ini_reg; -- initial interval is equal to initial region
old_zone:=ini_zone;
pre_zone:=hide k,n in hull (pre(ini_zone  & k=n) & ~k=n) endhide;
new_zone:=pre_zone;
while  not empty(new_zone) and not new_zone = old_zone do
   old_zone:=new_zone;
   pre_zone:=hide k,n in hull(pre(new_zone & k=n) & ~k=n) endhide;
   new_zone:=(new_zone | pre_zone);
endwhile;
-- To verify that the initial zone is accessible from itself
if not empty (new_zone & ini_zone) then
   -- if accessible
   I_new:=I_ini;
   pre_reg:=hide k,n in hull(pre(I_new & k=n) & ~k=n ) endhide;
   I_old:=I_ini & ~I_ini; --empty region initialization
   while  not empty(pre_reg) and not I_new=I_old do
      I_old:=I_new;
      while  not empty(pre_reg) and empty(pre_reg & I_ini) do
         pre_reg:= hide k,n in hull(pre(pre_reg & k=n ) & ~k=n) endhide;
      endwhile;
      pre_reg:=hull(pre_reg & I_ini);
      I_new:=hull(pre_reg & I_new);
      pre_reg:=hide k,n in hull(pre(I_new & k=n) & ~k=n ) endhide;
   endwhile;
   if not empty (I_new) then
      prints "============================================================";
      prints "Region of initial conditions for the invariance kernel";
      print I_new;
      prints "============================================================";
   else
      prints "No initial condition exists in the defined initial region";
   endif;
else
   -- if not accessible
   prints "The initial zone is not accessible from itself therefore ";
   prints "there is no initial condition that leads to an invariance kernel.";
endif;
```

References

1 Jacob F, Monod J: Genetics regulatory mechanisms in the synthesis of proteins. J Mol Biol 1961;3:318–356.

2 de Jong H: Modélisation et simulation qualitative de réseaux de régulation génique; 'Habilitation à diriger des recherches', Joseph Fourier University, Grenoble, 2004.

3 Sugita M: Functional analysis of chemical systems in vivo using a logical circuit equivalent. II. The idea of a molecular automaton. J Theor Biol 1963;4: 179–192.

4 Kauffman SA: Metabolic stability and epigenesis in randomly constructed genetic nets. J Theor Biol 1969;22:437–467.

5 Thomas R: Boolean formalization of genetic control circuits. J Theor Biol 1973;42:563–585.

6 Bernot G, Comet J-P, Richard A, Guespin J: Application of formal methods to biological regulatory networks: extending Thomas' asynchronous logical approach with temporal logic. J Theor Biol 2004; 229:339–347.

7 Thomas R, Kaufman M: Multistationarity, the basis of cell differentiation and memory. II. Logical analysis of regulatory networks in terms of feedback circuits. Chaos 2001;11:180–195.

8 Henzinger T-A, Ho P-H, Wong-Toi H: HyTech: a model checker for hybrid systems. Int J Software Tools Technol Transfer 1997;1:110–122.

9 Lincoln P, Tiwari A: Symbolic systems biology: hybrid modeling and analysis of biological networks. 7th International Workshop, Hybrid Systems: Computation and Control (HSCC 2004). Lecture Notes Comput Sci 2004;2993:660–672.

10 Ghosh R, Tomlin CJ: Lateral inhibition through delta-notch signaling: a piecewise affine hybrid model. 4th International Workshop on Hybrid Systems: Computation and Control (HSCC 2001). Lecture Notes Comput Sci 2001;2034:232–246.

11 Alur R, Belta C, Ivancic F, Kumar V, Mintz M, Pappas G, Rubin H, Schug J: Hybrid modeling and simulation of biomolecular networks. 4th International Workshop on Hybrid Systems: Computation and Control (HSCC 2001). Lecture Notes Comput Sci 2001;2034:19–32.

12 de Jong H, Gouzé J-L, Hernandez C, Page M, Sari T, Geiselmann J: Hybrid modeling and simulation of genetic regulatory networks: a qualitative approach. 6th International Workshop, Hybrid Systems: Computation and Control (HSCC 2003). Lecture Notes Comput Sci 2003;2623:267–282.

13 Belta C, Schug J, Dang T, Kumar V, Pappas GJ, Rubin H, Dunlap P: Stability and reachability analysis of a hybrid model of luminescence in the marine bacterium *Vibrio fischeri*. 40th IEEE Conference on Decision and Control (CDC 2001), Orlando, 2001, pp 869–874.

14 Belta C, Finin P, Habets LC, Halasz A, Imielinski M, Kumar V, Rubin H: Understanding the bacterial stringent response using reachability analysis of hybrid systems. 7th International Workshop, Hybrid Systems: Computation and Control (HSCC 2004). Lecture Notes Comput Sci 2004;2993:111–125.

15 Fowler AC, Mackey MC: Relaxation oscillations in a class of delay differential equations. SIAM J Appl Math 2002;63:299–323.

16 Wu FX, Zhang WJ, Kusalik AJ: State-space model for gene regulatory networks with time delays. IEEE Computational Systems Bioinformatics Conference (CSB 2004). IEEE Computer Society, 2004, pp 454–455.

17 Maranas CD, Dasika MS, Gupta A: A mixed integer linear programming (MILP) framework for inferring time delay in gene regulatory networks. Pacific Symposium on Biocomputing, Hawaii, 2004, pp 474–485.

18 Adélaïde M, Sutre G: Parametric analysis and abstraction of genetic regulatory networks. Proc 2nd Workshop on Concurrent Models in Molecular Biology (BioCONCUR'04). Electronic Notes in Theor Comp Sci. Amsterdam, Elsevier, 2004.

19 Siebert H, Bockmayr A: Incorporating time delays into the logical analysis of gene regulatory networks. International Conference on Computational Methods in Systems Biology (CMSB'06), Trento, Italy, LNBI 4210. Berlin, Springer, 2006, pp 169–183.

20 Guespin-Michel J, Kaufman M: Positive feedback circuits and adaptive regulations in bacteria. Acta Biotheor 2001;49:207–218.

21 Snoussi EH, Thomas R: Logical identification of all steady states: the concept of feedback loop characteristic states. Bull Math Biol 1993;55:973–991.

22 Thomas R: On the Relation between the Logical Structure of Systems and Their Ability to Generate Multiple Steady States or Sustained Oscillations. Springer Ser Synerg 1980;9:180–193.

23 Thomas R, Thieffry D, Kaufman M: Dynamical behaviour of biological regulatory networks. I. Biological role of feedback loops and practical use of the concept of the loop-characteristic state. Bull Math Biol 1995;57:247–276.

24 Demongeot J, Kaufman M, Thomas R: Positive feedback circuits and memory. CR Acad Sci III 2000;323:69–79.

25 Cinquin O, Demongeot J: Positive and negative feedback: striking a balance between necessary antagonists. J Theor Biol 2002;216:229–241.

26 Soulé C: Graphical requirements for multistationarity. ComPlexUs 2003;1:123–133.

27 Emerson EA: Handbook of theoretical computer science, vol B: Formal Models and Semantics. Cambridge, MIT Press, 1990, pp 995–1072.

28 Filopon D, Merieau A, Bernot G, Comet J-P, Leberre R, Guery B, Polack B, Guespin-Michel J: Epigenetic acquisition of inducibility of type III cytotoxicity in *P. aeruginosa*. BMC Bioinformatics 2006;7:272–281.

29 Alur R, Dill DL: A theory of timed automata. Theor Comput Sci 1994;126:183–235.

30 Henzinger TA, Kopke PW, Puri A, Varaiya P: What's decidable about hybrid automata? J Comput Syst Sci 1998;57:94–124.

31 Henzinger T: The theory of hybrid automata. Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science (LICS '96), New Brunswick, 1996, pp 278–292.

32 Frehse G: Phaver: algorithmic verification of hybrid systems past HyTech; in HSCC. Berlin, Springer, 2005, pp 258–273.

33 Bengtsson J, Larsen K, Larsson F, Pettersson P, Yi W, Weise C: New generation of UPPAAL, 1998.

34 Asarin E, Schneider G, Yovine S: Towards computing phase portraits of polygonal differential inclusions; in HSCC 2002, number 2289 in LNCS. Stanford, Springer, 2002, pp 49–61.

35 Bernot G, Cassez F, Comet J-P, Delaplace F, Müller C, Roux O, Roux O: Semantics of biological regulatory networks; in Danos V, Laneve C (eds): BioConcur 2003. ENTCS series. Marseille, Elsevier, 2003.

36 Thomas R, D'Ari R: Biological Feedback. Boca Raton, CRC Press, 1990.

37 Schneider G: Computing invariance kernels of polygonal hybrid systems. Nordic J Comput 2004;11: 194–210.

38 Aubin JP: A survey of viability theory. SIAM J Control Optim 1990;28:749–788.

39 Aubin JP: Viability kernels and capture basins of sets under differential inclusions. SIAM J Control Optim 2001;40:853–881.

40 Ahmad J, Roux O, Bernot G, Comet JP, Richard A: Analysing formal models of genetic regulatory networks with delays. Int J Bioinformatics Res Appl 2007.