INFORMATIQUE, BIOLOGIE INTÉGRATIVE ET SYSTÈMES COMPLEXES

A Temporal Logic with Event Clock Automata for Timed Hybrid Petri Nets

Sylvie Troncale [1], Jean-Paul Comet[1] [2], Gilles bernot[2]

[1] IBISC
523 place des terrasses de l'Agora
91000 Évry Cedex, France

[2] Epigenomics Project
523 place des terrasses de l'Agora
91000 Évry Cedex, France

— *Model-checking* —

RESEARCH REPORT

N$^{\mathrm{o}}$ IBISC-RR-2007-04

April 2007

IBISC, FRE 2873 CNRS - Université d'Évry Val d'Essonne, Genopole
Tour Évry 2, 523 place des terrasses de l'Agora
91000 Évry Cedex, France

Sylvie Troncale, Jean-Paul Comet, Gilles bernot

A Temporal Logic with Event Clock Automata for Timed Hybrid Petri Nets

18 p.

A Temporal Logic with Event Clock Automata for Timed Hybrid Petri Nets

Sylvie Troncale, Jean-Paul Comet, Gilles bernot

Abstract

The Hybrid Functional Petri Nets (HFPN) formalism has already shown its convenience for modelling biological systems. This class of models have been fruitfully applied in biology but the remarkable expressiveness of HFPN often leads to incomplete formal validations. In this paper, we propose a formal logical framework for Timed Hybrid Petri Nets (THPN), a sub-class of HFPN. We propose an extension of Event Clock Logic dedicated to THPN and a procedure to convert a THPN into a real-time automaton. A small biological model shows that our framework allows us to formally prove properties by a well suited model-checking procedure.

1 Introduction

Systems biology aims to a system-level understanding of the functioning of a biological system like the cell, taking into account not only molecular phenomena but also structuration of the cells, communication channels and exchanges with the outside space. This global aim is now conceivable thanks to the recent developments of genomic and postgenomic which enable identification of numerous genes and proteins. Nevertheless, the precise role of each actor stays hard to determine experimentally. Then, mathematical modelling is an essential approach to study complex biological processes. There exist numerous modelling formalisms which allow different evaluation techniques: simulation, proof, etc. The Hybrid Functional Petri Nets (HFPN) [5] formalism offers a maximum of flexibility such that modelling of discrete and continuous processes or definition of consumed or produced quantity as a function of marking. This explain why HFPN are well suited for simulation. Nevertheless, it is also necessary to verify that the global system satisfies a behavioral property. This verification step is difficult to perform on a so expressive formalism since such an expressiveness prevents the application of general formal or proof methods.

Since usual validation methods turned out to be unsuitable on HFPN, we propose an original procedure based on works of David and Alla [4] (Petri nets) and of Raskin and Schobbens [10] (satisfaction of temporal logic formulas). To tackle such a validation step, we first reduce the expressiveness of models, we focus on a sub-class of HFPN where functional aspects have been removed: the Timed Hybrid Petri Nets (THPN).

In this paper, we describe continuous traces of THPN as a particular automaton, an Event Clock automaton [2], based on a real time logic, the Event Clock logic [10]. This step needs to define precisely the continuous models, the extended Event Clock logic. THPN models can then be transcripted via the evolution graph and some manipulations and formulas in terms of Event Clock automata. We then show how the introduction of such a real time logic can be helpful in the context of biological modelling. We study a simplified model of amphibian metamorphosis regulation [6]. After having constructed the associated Event Clock automaton, we show that classical approaches of verification of Event Clock logic formulas can be applied to prove that the THPN model satisfies a particular temporal property.

This paper is organized as follows: Section 2 presents syntax and semantics of our logic. Definitions of a THPN and an evolution graph are sketched in Section 3. In Section 4, conversion algorithms of an evolution graph into an Event Clock automaton are detailed. Finally, Section 5 sketches out a biological example before we discuss our results in Section 6.

2 Continuous time logic

Syntax and semantics We define an extended syntax and semantics of Event Clock logic [10], where atoms are extended to handle continuous and discrete time executions. We call it Continuous Time Evolution Logic, CTEL for short. We first define signatures which specify variables and observable events abstracted by predicates.

Definition 1 *A signature for CTEL is a couple $\Sigma = (V, Pr)$ where V and Pr are respectively a* *set of variables and a set of predicates.*

Definition 2 *Given a CTEL signature $\Sigma = (V, Pr)$, a continuous-time model M is defined by a set $\pi \subset Pr \times \mathbb{R}^+$ and a function $\mu : (V \amalg \mathbb{R}) \times \mathbb{R}^+ \to \mathbb{R}$ (where $\amalg$ stands for the disjoint union) such that for any $r \in \mathbb{R}$, and for any $t \in \mathbb{R}^+$, $\mu(r, t) = r$. The set of models defined on the signature Σ is denoted by $Mod(\Sigma)$.*

We distinguish two kinds of atoms: instantaneous atoms α (Definition 3) and atoms (Definition 6).

Definition 3 *An instantaneous atom α is an expression of the form: $v \geq v'$, p or their negations, where $v, v' \in (V \amalg \mathbb{R})$ and $p \in Pr$.*

Definition 4 *The satisfaction relation $\models_{t_i}$ between a continuous-time model M and an instantaneous atom α at a time $t_i \in \mathbb{R}^+$ is defined as follows:*

- *$M \models_{t_i} p$ iff $(p, t_i) \in \pi$*
- *$M \models_{t_i} v \geq v'$ iff $\mu(v, t_i) \geq \mu(v', t_i)$*
- *$M \models_{t_i} \neg\alpha$, iff $M \not\models_{t_i} \alpha$*

An instantaneous atom α can be timed thanks to the use of two clocks, the history clock x_α and the prophecy clock y_α [2]. The value of a history clock x_α is the time elapsed since the last occurrence of α. The value of a prophecy clock y_α is the time to wait for the next occurrence of α. Introduction of the clocks x_α and y_α allows us to define the set of terms on the signature Σ, noted T_Σ.

Definition 5 *A term on a signature Σ is either a variable $v \in V$ or a constant $r \in \mathbb{R}$ or an expression of the form x_α (resp. y_α) where α is an instantaneous atom.*

Definition 6 *Given a signature $\Sigma = (V, Pr)$, an atom is an expression of the form $r \geq r'$, p or their negations, where $r, r' \in T_\Sigma$ and $p \in Pr$, such that if r (resp. r') is of the form x_α or y_α, the other term r' (resp. r) is necessarily an integer constant.*

Definition 7 *The set of well formed formulas on Σ, $For(\Sigma)$, is defined according to following syntaxic rules [10]. A well formed formula is composed of atoms, boolean connectives $\neg$, $\vee$, $\wedge$, qualitative temporal operators Next ($\bigcirc$), Previous ($\ominus$), Until (U) and Since (S) and of real-time operators: predicting and history operators ($\rhd$, $\lhd$):*

$$\begin{aligned}\varphi ::= a|\neg\varphi|\bigcirc\varphi| \ominus \varphi|\varphi_1 \wedge \varphi_2|\varphi_1 \vee \varphi_2| \\ \varphi_1 U \varphi_2|\varphi_1 S \varphi_2| \lhd_{\sim n} \alpha| \rhd_{\sim n} \alpha,\end{aligned}$$

where a is an atom, $\sim \in \{=, <, >, \leq, \geq\}$, φ, φ_1, $\varphi_2 \in For(\Sigma)$ and n is in $\mathbb{N} \subset \mathbb{R}$.

Events observed during the execution of a continuous time model can be expressed by a subset of well formed formulas.

Definition 8 *The set of observations on the signature Σ is the subset $Obs(\Sigma)$ of $For(\Sigma)$ defined as follows :* $\varphi ::= a|\neg\varphi|\varphi_1 \wedge \varphi_2|\varphi_1 \vee \varphi_2$ *where a is an atom, φ, φ_1 and φ_2 are observations.*

During a continuous time model execution, observations are made at different times which define a time sequence.

Definition 9 *A time sequence h is an infinite succession of times t_i which is strictly increasing and divergent.*

We now define a function $eval^h_M$ which evaluates each term of a continuous model M on a time sequence h.

Definition 10 *Given a Σ-model M and a time sequence h, the function which evaluates each term t in the model M on the time sequence h, $eval^h_M : T_\Sigma \times |h| \to \mathbb{R} \cup \{\bot\}$, where $|h|$ is the set of times in h is defined as follows :*

- $eval^h_M(v, t_i) = \mu(v, t_i)$ *where* $v \in (V \amalg \mathbb{R})$ *and* $t_i \in \mathbb{R}^+$
- $eval^h_M(x_\alpha, t_i) = \begin{pmatrix} t_i - t_j & if\ \exists t_j,\ 0 \leq t_j < t_i |\ M \models_{t_j} \alpha \\ & and\ \forall t_k,\ t_j < t_k < t_i,\ M \not\models_{t_k} \alpha \\ \bot & otherwise \end{pmatrix}$
- $eval^h_M(y_\alpha, t_i) = \begin{pmatrix} t_j - t_i & if\ \exists t_j,\ t_j > t_i |\ M \models_{t_j} \alpha \\ & and\ \forall t_k,\ t_i < t_k < t_j,\ M \not\models_{t_k} \alpha \\ \bot & otherwise \end{pmatrix}$

Definition 11 *The satisfaction relation $\models^h_{t_i} \subset Mod(\Sigma) \times For(\Sigma)$ where $t_i \in |h|$, is defined inductively as follows:*

- $M \models^h_{t_i} p$ *iff* $(p, t_i) \in \pi$
- $M \models^h_{t_i} r \ \geq\ r'$ *iff* $eval^h_M(r, t_i) \ \geq\ eval^h_M\ (r', t_i)$
- $M \models^h_{t_i} \neg\varphi$ *iff* $M \not\models^h_{t_i} \varphi$
- $M \models^h_{t_i} \varphi_1 \wedge \varphi_2$ *iff* $M \models^h_{t_i} \varphi_1$ *and* $M \models^h_{t_i} \varphi_2$
- $M \models^h_{t_i} \bigcirc\varphi$ *iff* $M \models^h_{t_{i+1}} \varphi$

  $M \models^h_{t_i} \ominus\varphi$ *iff* $i > 0$ *and* $M \models^h_{t_{i-1}} \varphi$
- $M \models^h_{t_i} \varphi_1 U \varphi_2$ *iff* $\exists t_j \geq t_i \mid M \models^h_{t_j} \varphi_2$ *and* $\forall t_k \in [t_i, t_j[,\ M \models^h_{t_k} \varphi_1$

  $M \models^h_{t_i} \varphi_1 S \varphi_2$ *iff* $\exists t_j \in [0, t_i] \mid M \models^h_{t_j} \varphi_2$ *and* $\forall t_k \in]t_j, t_i],\ M \models^h_{t_k} \varphi_1$
- $M \models^h_{t_i} \rhd_{\sim n}\varphi$ *iff* $\exists t_j > t_i\ |M \models^h_{t_j} \varphi$ *and* $\forall t_k \in]t_i, t_j[, M \not\models^h_{t_k} \varphi$ *and* $t_i - t_j \sim n$

  $M \models^h_{t_i} \lhd_{\sim n}\varphi$ *iff* $\exists t_j \in [0, t_i[\ |M \models^h_{t_j} \varphi$ *and* $\forall t_k \in]t_j, t_i[, M \not\models^h_{t_k} \varphi$ *and* $t_j - t_i \sim n$

Discrete timed traces Continuous time models are difficult to represent in an abstract formalism manipulable by a computer. We then focus on discretization of continuous time evolutions. We first define a satisfaction relation between a continuous time model and a discrete timed trace.

Definition 12 *A timed trace is of the form $\{(\varphi_i, t_i)\}_{i\in\mathbb{N}}$, where the φ_i are observations and $h_\tau = (t_i)_{i\in\mathbb{N}}$ is a time sequence. The satisfaction relation between a model M and a timed trace τ is defined by: $M \models \tau$ iff $\forall i \in \mathbb{N},\ M \models^{h_\tau}_{t_i} \varphi_i$.*

We now define the satisfaction relation between timed traces and a CTEL formula ϕ, denoted $\leftsquigarrow$.

Definition 13 *Given a timed trace τ, a position $i \in \mathbb{N}$ and a CTEL formula ϕ, τ satisifes ϕ at the position i, noted $(\tau, i) \leftsquigarrow \phi$ iff $\exists M | M \models \tau$ and $M \models^{h_\tau}_{t_i} \phi$.*

3 Formalization of THPN

We first define the THPN models [4], then present how the evolution of such models can be represented by an evolution graph.

Definition 14 *A Timed Hybrid Petri Net is a 7-tuple $(\mathcal{P}, \mathcal{T}, \zeta, Pre, Post, m_0, Tempo)$ where:*

- *$\mathcal{P}$ and $\mathcal{T}$ are disjoint sets of places and transitions,*
- *$\zeta : \mathcal{P} \cup \mathcal{T} \to \{D, C\}$ called "hybrid function," indicates for every node whether it is a discrete node or a continuous one.*

  Let T^D (resp. P^D) and T^C (resp. P^C) be the sets of discrete and continous transitions (resp. places),
- *$Pre : \mathcal{P}\times\mathcal{T} \to \mathbb{R}^+\cup\mathbb{N}$ is the input incidence application. If $T \in T^D$ then $Pre(P, T) \in \mathbb{N}$ else $Pre(P, T) \in \mathbb{R}^+$.*

  Let T be the set of places preceding the transition T and $P = \{T \in \mathcal{T} | P \in T\}$,
- *$Post : \mathcal{T} \times \mathcal{P} \to \mathbb{R}^+ \cup \mathbb{N}$ is the output incidence application. If $T \in T^D$ then $Post(T, P) \in \mathbb{N}$ else $Post(T, P) \in \mathbb{R}^+$.*

  Let T be the set of places succeeding to the transition T and $P = \{T \in \mathcal{T} | P \in T\}$,
- *$m_0 : \mathcal{P} \to \mathbb{R}^+ \cup \mathbb{N}$ is the initial marking. If $P \in P^D$ then $m_0(P) \in \mathbb{N}$ else $m_0(P) \in \mathbb{R}^+$,*
- *$Tempo$ is a function from the set $\mathcal{T}$ to the set of positive rational numbers. If $T \in T^D$, $Tempo(T)$ is a timing associated with T. It is noted $delay(T)$. If $T \in T^C$, $\frac{1}{Tempo(T)}$ represents the maximal firing speed associated with T. In the sequel, it is noted $V(T)$.*

Semantics intuition A discrete transition T is *enabled* if each place $P_i \in T$ satisfies $m(P_i) \geq Pre(P_i, T)$. If the transition T stays enabled during the time $delay(T)$, it will be fired at the end of this delay. $Pre(P_i, T)$ tokens are then removed from each place $P_i \in T$ and $Post(T, P_j)$ tokens are added to each transition $P_j \in T$. The marking can be sufficient to allow fewer simultaneous firings. The number of allowed firings defines the *enabling degree*. By definition, $T \in T^D$ is enabled if its enabling degree is not null.

A continuous transition T is *enabled* if each place $P_i \in T$ satisfies either $m(P_i) \geq Pre(P_i, T)$ if P_i is a discrete place, or $m(P_i) > 0$ if P_i is a continuous place. A continuous transition is fired to its *instantaneous firing speed* $v(T)$ such that $0 \leq v(T) \leq V(T)$. $v(T)$ corresponds to the maximal speed a transition can fire according to the current marking. By definition, $T \in T^C$ is active if its instantaneous speed is not null. A flow of $Pre(P_i, T) \times v(T)$ tokens are removed from each place $P_i \in T$ and a flow of $Post(T, P_j) \times v(T)$ tokens are added to each transition $P_j \in T$.

Evolution graph The behavior of a THPN can be represented by an evolution graph, represented by a Petri net [4]. Each place corresponds to an IB-state (invariant behavior state) and each transition is associated with an event (change of marking) whose occurrence produces a change from one IB-state to another. Such a transition can only occur if an event belonging to one of the following types takes place: the marking of a continuous place becomes zero (C1-event), a discrete transition fires (D1-event) or the enabling degree of a discrete transition changes because of the marking of a continuous place (D2-event).

Intuitively, the i^{th} transition of the evolution graph, denoted T_i^{GE} is labelled with the set $Evt(T_i^{GE})$ of occured events, with time of the event occurrence and with marking of all continuous places. IB-states are annoted by marking of all discrete transitions, by the vector of enabling degrees and by the vector of instantaneous speed.

For constructing such an evolution graph, two restrictions are imposed to THPN. First, the marking of each place $P \in \mathcal{P}$ must be bounded. This restriction guarantees the algorithm to end. Secondly, since the evolution graph represents a deterministic behavior, one has to solve conflicts which occur when the marking of a place is not sufficient to allow the different transitions to fire simultaneously. Generally, there are two ways for solving conflicts. *Sharing* proposes to share resources between transitions according to a given schema (general case: stoichiometric constants are then helpfull for determining sharing schema). And *priority* ranks transitions and gives limited resources according to the ranks (*e.g.* catalytic phenomena).

Signature For constructing the Event Clock automaton related to a THPN, let us first define the signature $sign = (V, Pr)$ of a given THPN:

V is the following set of variables. $m(P)$ represents the marking of $P \in \mathcal{P}$, $v(T)$ represents the instantaneous speed of $T \in T^C$ and $dg(T)$ represents the enabling degree of $T \in T^D$,

Pr is the following set of predicates:

- $Enable(T)$, $Act(T)$: unary predicates associated with respectively enabling of $T \in T^D$ and activation of $T \in T^C$,

- $Fire(T)$, $NulMark(P)$: unary predicates associated with respectively a D1-event and a C1-event,
- $Th(P, x)$: binary predicate associated with a D2-event, $(Threshold)$
- $NoEvt$: predicate associated with the first transition of the evolution graph when no event occurs.

4 Associated Event Clock Automaton

Let us first recall the definition of an Event Clock automaton [2].

Definition 15 *An Event Clock automaton on the signature sign is a 6-tuple $A = (L, L_0, At, \mathcal{C}, E, \mathcal{F})$ where :*

- *L is a finite set of locations and $L_0 \subseteq L$ is the subset of start locations,*
- *At is a set of atoms,*
- *$\mathcal{C}$ is a set of history or prophecy clocks,*
- *E is a finite set of edges. An edge is a triplet (l_1, ψ, l_2) where $l_1 \in L$ is the source location, $l_2 \in L$ is the target location, and $\psi \in Obs(sign)$ describes the state,*
- *$\mathcal{F} = \{F_1, ..., F_n\}$ where $F_i \subseteq L$ is a set of sets of accepting locations*

Definition 16 *A trace $\tau = \{(\varphi_i, ti)\}_{i \in \mathbb{N}}$ is recognized by an Event Clock automaton $A = (L, L_0, At, \mathcal{C}, E, \mathcal{F})$ if there exists an infinite accepted computation $\gamma = l_0 \overset{\psi_0}{\rightarrow} l_1 \overset{\psi_1}{\rightarrow} ...l_n \overset{\psi_n}{\rightarrow} ...$ where:*

- *each $l_i \in L$ and $l_0 \in L_0$,*
- *$(l_i, \psi_i, l_{i+1}) \in E$ and $(\tau, i) \leftsquigarrow \psi_i$ with $\psi_i \in Cl(\varphi_i)$ where the closure $Cl(\varphi_i)$ is the set of sub-formulas of φ_i,*
- *for every $F_i \in \mathcal{F}$, there exists infinitely many positions j such that $l_j \in F_i$.*

Let us now define the satisfaction relation between an automaton and a timed trace τ, denoted $\models\!\!\approx$.

Definition 17 *The timed language of an Event Clock automaton A, denoted $\mathcal{L}(A)$, is the set of timed traces recognized by A, $\mathcal{L}(A) = \{\tau \mid A \models\!\!\approx \tau\}$*

We now introduce a procedure to transform an evolution graph (deduced from a THPN) into an Event Clock automaton. This procedure is composed of four steps, the first and the second one constructing the set of locations, the third one determining the initial and accepting locations and the fourth one constructing edges.

1- From IB-states to locations: Each IB-state of the evolution graph gives a location of the Event Clock automaton. With each of these locations we associate an observation $\phi_1(IB_i)$ describing the THPN state during the whole time of the IB-state numbered i. $\phi_1(IB_i)$ has the following form, where val associates with a variable its current value and where $I(T^{GE})_i^{i+1}$ corresponds to the interval bounded by the value of the continuous marking at the transitions T_i^{GE} and T_{i+1}^{GE}.

$$\phi_1(IB_i) \equiv \wedge \begin{pmatrix} \bigwedge\limits_{P\in P^D} (m(P) = val(m(P))) \\ \bigwedge\limits_{T\in T^C} (v(T) = val(v(T))) \\ \bigwedge\limits_{T\in T^D} (dg(T) = val(dg(T)) \\ \bigwedge\limits_{P\in P^C} (m(P) \in I(T^{GE})_i^{i+1}) \end{pmatrix}$$

2- From transitions to locations: Each transition of the evolution graph also gives a location of the Event Clock automaton. With each of these locations we associate an observation $\phi_2(T_i^{GE})$ describing the THPN state when entering into the IB-state numbered i. $\phi_2(T_i^{GE})$ has then the following form, where val associates with a variable its current value. Note that x_{le} represents the time elapsed since the last event occurs. This last event can be either $NoEvt$, $Fire(T)$, $NulMark(P)$ or $Th(P,x)$ and Δt is the timing associated with the transition T_i^{GE}.

$$\phi_2(T_i^{GE}) \equiv \wedge \begin{pmatrix} \bigwedge\limits_{IB_i\in T_i^{GE}} \phi_1(IB_i) \\ \bigwedge\limits_{e\in Evt(T)} e \\ \bigwedge\limits_{\substack{P\in P^C \\ \Delta t = evt\ time}} (m(P) = val(m(P))) \\ \bigwedge\limits_{le\in Evt(T)} (x_{le} = \Delta t) \end{pmatrix}$$

3- Start and accepting locations: The start location is the location corresponding to the first transition T_0^{GE}. The accepting locations are determined according if the evolution graph ends. In case of deadlock, the accepting location is the location corresponding to the last IB-state. In case of loopback (cycle), each location which corresponds to a transition (T^{GE}) or to an IB-state involved in the loopback is an accepting location.

4- Edges: There is an edge between two locations if there is an arc between the corresponding IB-states or transitions in the evolution graph. Moreover, each location obtained from an IB-state loops to model the time of the IB-state. Finally, an edge outgoing from a location l is labelled by the formula of the location l.

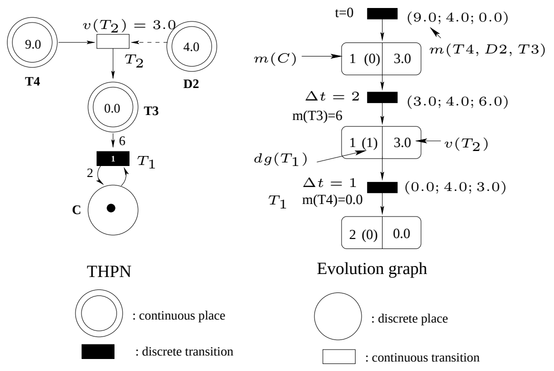

Figure 1: The THPN of cellular cycle activation in amphibian metamorphosis and its evolution graph.

5 Biological illustration

Hindlimb growth during amphibian metamorphosis is induced by triggering regulations responsible for cellular cycle activation. Such regulations are controlled by thyroid hormones [6]. The active form of thyroid hormone (TH), denoted T3 is produced from the inactive form T4 by the enzymatic action of the deiodinase of type 2, denoted D2 [3]: $D2 + T4 \rightarrow T3 + D2$.

THPN model and evolution graph Each thyroid hormone (T3 and T4) as well as the enzyme D2 are modelled by a continuous place representing their molecular concentrations (left part of Figure 1). Since the enzymatic reaction is a continuous phenomenon, the reaction allowing D2 to transform T4 into T3 is modelled by the continuous transition T_2. Since this reaction does not consume D2, a test arc (dotted arc) is used. Parameters are estimated from known kinetics of T3, T4 [9] and D2 [3].

The hindlimb growth is abstracted by the number of cells, which is represented by a discrete place (C). Initially, there is a unique cell. The discrete transition T_1 simulates cellular proliferation which occurs after mitosis time (delay 1 on T_1).

The dynamic of the previous THPN model can be extracted by constructing the evolution graph (right part of Figure 1). Only two sets of events occur: at the time $t = 2$ ($\Delta t = 2$) of the THPN execution, the continuous place T3 reaches the threshold of 6.0 enabling the discrete transition T_1 to fire and one time unit later ($\Delta t = 1$), two events simultaneously occur: the discrete transition T_1 fires and the continuous place T4 becomes empty, leading to a deadlock of the system.

Automaton construction The Event Clock automaton A_M is presented in Figure 2. It is easy to observe that traces of A_M correspond to the execution of the THPN.

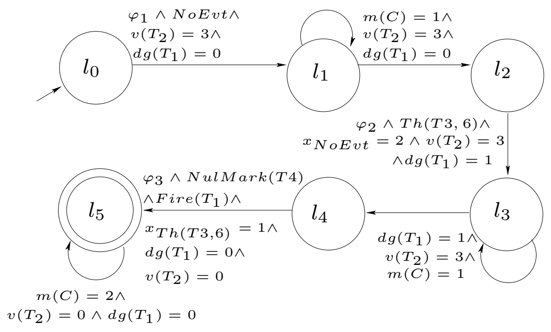
$\varphi_1 \wedge NoEvt \wedge$ $v(T_2) = 3 \wedge$ $dg(T_1) = 0$
$m(C) = 1 \wedge$ $v(T_2) = 3 \wedge$ $dg(T_1) = 0$
l_0
l_1
l_2
$\varphi_2 \wedge Th(T3,6) \wedge$ $x_{NoEvt} = 2 \wedge v(T_2) = 3$ $\wedge dg(T_1) = 1$
$\varphi_3 \wedge NulMark(T4)$ $\wedge Fire(T_1) \wedge$ $x_{Th(T3,6)} = 1 \wedge$ $dg(T_1) = 0 \wedge$ $v(T_2) = 0$
l_5
l_4
l_3
$dg(T_1) = 1 \wedge$ $v(T_2) = 3 \wedge$ $m(C) = 1$
$m(C) = 2 \wedge$ $v(T_2) = 0 \wedge dg(T_1) = 0$

Figure 2: Event Clock automaton of the THPN model, denoted A_M. $\varphi_1 \equiv (m(C) = 1) \wedge (m(T4) = 9) \wedge (m(D2) = 4) \wedge (m(T3) = 0)$, $\varphi_2 \equiv (m(C) = 1) \wedge (m(T4) = 3) \wedge (m(D2) = 4) \wedge (m(T3) = 6)$ and $\varphi_3 \equiv (m(C) = 2) \wedge (m(T4) = 0) \wedge (m(D2) = 4) \wedge (m(T3) = 3)$

Proof of a property Amound different kinds of properties, we focus here on dynamics of the cellular cycle. In this section, we consider the following property: at a moment, a minimum of three time units is necessary before the enzymatic reaction stops. This biological property enables biologists to estimate time of the metamorphosis end. It can be translated into a CTEL formula ϕ:

$$\phi \equiv \diamond \rhd_{\geq 3} (v(T_2) = 0) \equiv \neg \Box \neg \rhd_{\geq 3} (v(T_2) = 0)$$

where $\diamond$ means eventually and $\Box$ means always. The Event Clock automaton associated with the negation of the studied property, $A_{\neg\phi}$, is then constructed by using the procedure detailed in [10], see Figure 3. Traces of $A_{\neg\phi}$ represent the set of timed traces which satisfy $\neg\phi$.

The product automaton $A_p = A_M \times A_{\neg\phi}$ is drawn in Figure 4 where only accepted computations and relevant labels are indicated on edges.

The language of the product automaton A_p can be proved to be empty by constructing its region automaton as in [10, 1]. Intuitivelly, the language of the product automaton is empty if one of its traces passes through an edge labelled by $(v(T_2) = 0)$ after three time units. In fact, the history clocks x_{NoEvt} and $x_{Th(T3,6)}$ (dashed box on Figure 4) count elapsed time. The time constraints related to these clocks indicate that three time units elapse when the edge label $(v(T_2) = 0)$ is recognized by the automaton. It proves that the A_p language is empty. The Petri net then satisfies the property ϕ, *i.e.* at a moment of the biological process, more than three time units will be required to observe the end of the enzymatic reaction.

6 Discussion

Hybrid Functional Petri Nets [5] constitute a powerfull framework to define formal models of complex biological systems. Many rather large and complex systems have already been modelled using HFPN [12]. Unfortunately, *functions* (the “F” of HFPN) offer such an expressive power that they are the main obstacle to perform *formal proofs* on models defined using HFPN. Other more restricted logical frameworks without functions and generally without explicit quantitative time [11] are dedicated to precise aspects of biological systems (such as genetic regulatory networks). This kind of formalism offers automated proof procedures. Unfortunately, when defining formal models of biological systems, we often need explicit quantitative time and some functions in order to fully address the biological problem and express the biological questions in logical formulas.

Our (long term) motivation is consequently to offer automated proof procedures for a significant sub-framework of HFPN. Transitions and functions in HFPN being often continuous and quantitative, the model checking procedure of [10] based on Event Clock Logic and products of automata is promising *w.r.t* our motivation. So, the work described in this article is a first step toward our aim: it introduces a small extension of Event Clock Logic and a compatible translation of THPN models into automata, which makes it possible to perform automated reasonings on THPN models.

Future works in this vein include the development of a complete model checking procedure, extended and exhaustive definition of the set of biologically sensible strategies to translate a THPN into an automaton, and introduction of functions. For each of these three points, the main difficulties are the following.

To develop a complete model checking procedure compatible with our extension of Event Clock logic, it is necessary to accept product transitions labeled by different formulas provided that the intersection of their domain is not empty.

The construction of evolution graph depends on the resolution of conflicts as mentioned in section 3. Theoretically, this could lead to an infinite set of deduced automata, except that biologically, when a particular conflict is solved using a given rule, this rule is deduced from biochemical knowledge and has to be reused at each occurrence of this conflict.

Introduction of functions is the truly hard question. First of all, functions may hide interactions which are not shown in the graph, and this should deeply influence the construction of the automaton. Moreover, HFPN allow any form of mathematical functions and obviously, to maintain formal verification capabilities, the form of mathematical functions has to be carefully restricted.

Our approach based on Event Clock logic gives an interesting alternative to hybrid extension of classical model-checking [8, 7]. Event Clock logic seems to be well suited to add to THPN more and more sophisticated functions.

References

[1] R. Alur and D. Dill. A theory of timed automata. *Theoretical Computer Science*, 1994.

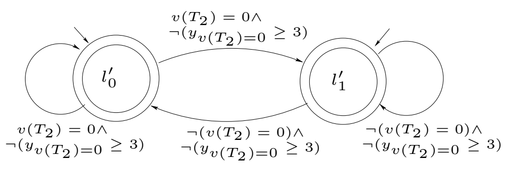
$v(T_2) = 0 \wedge$
$\neg(y_{v(T_2)=0} \geq 3)$
l'_0
l'_1
$v(T_2) = 0 \wedge$
$\neg(y_{v(T_2)=0} \geq 3)$
$\neg(v(T_2) = 0) \wedge$
$\neg(y_{v(T_2)=0} \geq 3)$
$\neg(v(T_2) = 0) \wedge$
$\neg(y_{v(T_2)=0} \geq 3)$

Figure 3: Event Clock automaton $A_{\neg\phi}$ reduced to accessible locations

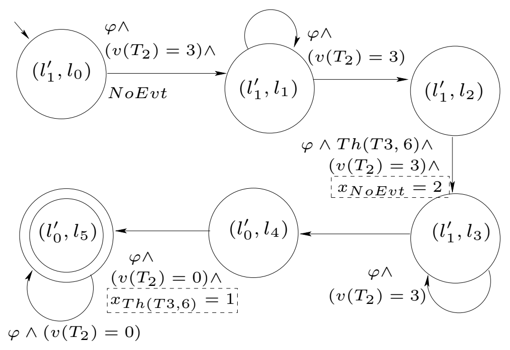

Figure 4: Event Clock automaton $A_M \times A_{\neg\phi}$. $\varphi \equiv \neg(y_{v(T_2)=0} \geq 3)$

[2] R. Alur, L. Fix, and Henzinger T.. Event-clock automata: A determinizable class of timed automata. *Theoretical Computer Science*, 1999.

[3] L. Cai and D. Brown. Expression of type 2 iodothyronine deiodinase marks the time that a tissue responds to thyroid hormone-induced metamorphosis in xenopus laevis. *Developmental Biology*, 266:87–95, 2003.

[4] R. David and H. Alla. *Discrete, Continuous, and Hybrid Petri Nets.* Springer, 2005.

[5] A. Doi, S. Fujita, H. Matsuno, M. Nagasaki, and S. Miyano. Constructing biological pathway models with hybrid functional Petri nets. *In Silico Biology*, 4:271–291, 2004.

[6] J.D. Furlow and E.S. Neff. A developmental switch induced by thyroid hormone: Xenopus laevis metamorphosis. *TRENDS in Endocrinology and Metabolism*, 17:40–47, 2006.

[7] M. Gribaudo, A. Horvath, E. Tronci, E. Ciancamerla, and M. Minichino. Model-checking based on fluid Petri nets for the temperature control system of the ICARO co-generative plant. In *Computer Safety, Reliability and Security: 21st International Conference, SAFECOMP 2002*, 2002.

[8] T. Henzinger. The theory of hybrid automata. In *IIIE Commputer Society Press*, 1996.

[9] J. Leloup and M. Buscaglia. Triiodothyronine, hormone of amphibian metamorphosis. *C.R. Acad. Sci.*, pages 2261–2263, 1977.

[10] J-F. Raskin and P-Y. Schobbens. The logic of event clocks. *Journal of Automata, Languages and Combinatorics.*, 1999.

[11] R. Thomas and R. d'Ari. Biological feedback. *CRC Press*, 1990.

[12] S. Troncale, D. Campard, F. Tahi, J. Guespin, and JP. Vannier. Modeling and simulation with hybrid functional petri nets of the role of interleukin-6 in human early haematopoiesis. In *In Pacific Symposium of Biocomputing*, 2006.

A Temporal Logic with Event Clock Automata for Timed Hybrid Petri Nets

Sylvie Troncale, Jean-Paul Comet, Gilles bernot

Abstract

The Hybrid Functional Petri Nets (HFPN) formalism has already shown its convenience for modelling biological systems. This class of models have been fruitfully applied in biology but the remarkable expressiveness of HFPN often leads to incomplete formal validations. In this paper, we propose a formal logical framework for Timed Hybrid Petri Nets (THPN), a sub-class of HFPN. We propose an extension of Event Clock Logic dedicated to THPN and a procedure to convert a THPN into a real-time automaton. A small biological model shows that our framework allows us to formally prove properties by a well suited model-checking procedure.

IBISC, FRE 2873 CNRS - Université d'Évry Val d'Essonne, Genopole
Tour Évry 2, 523 place des terrasses de l'Agora
91000 Évry Cedex, France