Validation of biological models with Temporal Logic and Timed Hybrid Petri Nets

Sylvie Troncale, Jean-Paul Comet and Gilles Bernot

***Abstract*— The Hybrid Functional Petri Nets (HFPN) formalism has shown its convenience for modelling biological systems. This class of models has been fruitfully applied in biology but the remarkable expressiveness of HFPN often leads to incomplete validations. In this paper, we propose a 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.**

I. INTRODUCTION

Systems biology is aiming 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 remains hard to determine experimentally. Then, mathematical modelling and abstraction methods are essential approaches to bridge the gap of incomplete knowledge and to study complex biological processes. There exist numerous modelling formalisms which allow different validation techniques *w.r.t* biological knowledge: simulation, proof, etc. The Hybrid Functional Petri Nets (HFPN) [1] formalism offers a maximum of flexibility such as modelling of discrete and continuous processes, or definition of consumed or produced quantities as functions of marking and this explains why HFPN are well suited for simulation in biology. Nevertheless, simulations are not sufficient to formally validate or refute a model, that is, to confront the model with known behavioural properties. Such a step of "model checking" (*i.e.* checking if a model satisfies a property) enables one to select only Petri net models satisfying a set of known biological properties. Nevertheless, "model checking" is impossible to perform in a computer aided manner on a so expressive formalism. One of the obvious reasons is that functions of HFPN induce some *implicit* use of the system states.

Since usual validation methods turned out to be intractable on HFPN, we propose an original procedure based on works of David and Alla [2] (Petri nets) and of Raskin and Schobbens [3] (satisfaction of temporal logic formulas). To tackle a powerful validation ability, we need to reduce the expressiveness of HFPN, we focus on a sub-class of HFPN: the Timed Hybrid Petri Nets (THPN) [2]. THPN enable the construction of models of a large range of complex biological systems [4].

In this paper, we describe continuous traces of THPN as a particular automaton, an Event Clock automaton [5], based on a real time logic, the Event Clock logic [3]. This step requires to define precisely the continuous models and the extended Event Clock logic. THPN models can then be transcribed via the evolution graph and some manipulations and formulas in terms of Event Clock automata. We then show how the introduction of 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 organised as follows: Section II presents syntax and semantics of our logic. Definitions of a THPN and an evolution graph are reminded in Section III. In Section IV, we describe our conversion algorithms of an evolution graph into an Event Clock automaton. Finally, Section V sketches out a biological example before we discuss our results in Section VI.

II. CONTINUOUS TIME LOGIC

In this section, we briefly recall the way we have extended the classical Event Clock Logic [3]. More detailed definitions can be found in [7].

A. Syntax and semantics

We define a slightly extended syntax and semantics of Event Clock logic [3], 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. A continuous-time model M is then 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 real number value $v \in \mathbb{R}$, and for any $t \in \mathbb{R}^+$, $\mu(v, t) = v$.*

We distinguish two kinds of atoms: instantaneous atoms (Definition 2) and general atoms (Definition 4).

Definition 2 *An instantaneous atom α is an expression of one of the two following forms:*

- *a predicate $p \in Pr$, in which case a model M satisfies α at a time t iff $(p,t) \in \pi$,*
- *an inequality $v \geq v'$, where $v, v' \in (V \amalg \mathbb{R})$ in which case M satisfies α at a time t iff $\mu(v,t) \geq \mu(v',t)$.*

Lastly, a model M satisfies $\neg\alpha$ at a time t iff it does not satisfy α at this time.

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 [5]. 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, which in turn allows us to define the set of general atoms.

Definition 3 *A term on a signature Σ is either a variable (resp. a constant value) v belonging to $V \amalg \mathbb{R}$ or an expression of the form x_α (resp. y_α) where α is an instantaneous atom.*

Definition 4 *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 an integer.*

Definition 5 *Following [3], a well formed formula is composed of atoms, connectives $\neg$, $\vee$, $\wedge$, $\Rightarrow$, temporal operators Next ($\bigcirc$), Previous ($\ominus$), Until (U) and Since (S) and of real-time operators: predicting and history operators ($\rhd$, $\lhd$):*

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

where a is an atom, $\sim$ is a comparison belonging to $\{=, <, >, \leq, \geq\}$, φ, φ_1, φ_2 are formulas and n is a natural number.

For example, assume that we study the cell cycle then G_1, G_2, S and $Mitosis$ would be predicates of Pr in the signature. So, the formula "$G_1 \Rightarrow (\rhd_{=y_{\neg G_1}} G_2 \wedge \rhd_{\leq 12} G_2)$" is an example of well formed formula. It means that "*if the cell is in the G_1 phase, then the phase which comes at the end of G_1 is G_2 and G_2 comes before 12 hours.*"

We have chosen the logic introduced by Raskin and Schobbens due to its remarkable expression power. Let us remark that it includes in particular classical temporal operators such as always ($\Box$) and eventually ($\diamond$) (see [3]).

Properties observed during the execution of a continuous time model are observed at a given top of horloge. They concern the current state of the system at this time, thus they cannot involve past or future events. Consequently, they can be expressed by the subset of well formed formulas defined below.

Definition 6 *An observation on the signature Σ is a formula of the form:*

$$\varphi ::= a|\neg\varphi|\varphi_1 \wedge \varphi_2|\varphi_1 \vee \varphi_2|\varphi_1 \Rightarrow \varphi_2$$

where a is an instantaneous atom, φ, φ_1 and φ_2 are observations.

For example, the previous cell cycle formula is not an observation as $\rhd$ and y appear in the formula. On the contrary, "$G_1 \Rightarrow \neg G_2 \wedge \neg S \wedge \neg Mitosis$" means that the phase G_1 excludes any other phases and is an observation at any given top of horloge.

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

Definition 7 *A time sequence h is an infinite succession of times t_i, where $i \in \mathbb{N}$, which is strictly increasing and divergent.*

To consider a given time sequence h allows us to actually *compute* the value of any term in T_Σ. This value is defined *via* the $eval$ function.

Definition 8 *Given a Σ-model M, a time sequence h and a time t_i belonging to h, the evaluation of a term r is defined as follows:*

- *If r is reduced to a symbol $v \in (V \amalg \mathbb{R})$, then $eval^h_M(v, t_i) = \mu(v, t_i)$.*
- *If r is of the form x_α, then the evaluation fails with a conventional value $\perp$ if α has never been satisfied before t_i in the model M, otherwise it is equal to the time elapsed since the last occurrence of α.*
- *Similarly, if r is of the form y_α, then the evaluation fails with a conventional value $\perp$ if α will never be satisfied after t_i in the model M, otherwise it is equal to the time to wait for the next occurrence of α.*

Evaluation of terms being defined, it becomes straightforward to verify whether a formula is satisfied in a model M at a time t_i of a time sequence h, simply by applying the truth tables of the connectors.

B. Discrete timed traces

A typical wet experiment consists in putting a biological system into an imposed initial state and observing it at some well chosen intervals of time. Consequently, it seems natural to validate a model or a property (formula) with respect to the experimental observations made at the chosen tops of horloge. The notion of timed traces is precisely designed to input those experimental observations into our technical stuff.

Definition 9 *A timed trace is defined by $\tau = \{(\varphi_i, t_i)\}_{i \in \mathbb{N}}$, where the φ_i are observations and $h_\tau = (t_i)_{i \in \mathbb{N}}$ is a time sequence. A model M satisfies a trace τ if for any natural number i, the observation φ_i is satisfied according to h_τ at time t_i.*

Assume that some model M_0 has been defined to model a given biological system and assume that τ accumulates the

successive observations during a wet experiment. If τ is not satisfied by M_0 then we can say that the model M_0 has been refuted experimentally.

Biologists can also perform experiments in order to check an hypothesis. Then, assuming that the hypothesis has been expressed through a CTEL formula ϕ, we have to check whether τ is compatible with this formula, denoted by $\tau \backsim \phi$.

Definition 10 *Let us consider a timed trace τ, a natural number i and a CTEL formula ϕ. A trace τ is compatible with ϕ at the position i, noted $(\tau, i) \backsim \phi$ if and only if there exists a model M which satisfies the trace τ and such that M satisfies the formula ϕ_i at the time t_i according to the time sequence h_τ.*

III. REMINDER OF THE THPN DEFINITION

Following the work of David and Alla [2]:

Definition 11 *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,*
- *$\zeta : \mathcal{P} \cup \mathcal{T} \rightarrow \{D, C\}$ 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 continuous transitions (resp. places),
- *$Pre : \mathcal{P} \times \mathcal{T} \rightarrow \mathbb{R}^+ \cup \mathbb{N}$ is the input incidence application. If $T \in T^D$ then $Pre(P,T) \in \mathbb{N}$ else $Pre(P,T) \in \mathbb{R}^+$.*
- *$Post : \mathcal{T} \times \mathcal{P} \rightarrow \mathbb{R}^+ \cup \mathbb{N}$ is the output incidence application. If $T \in T^D$ then $Post(T,P) \in \mathbb{N}$ else $Post(T,P) \in \mathbb{R}^+$.*
- *$m_0 : \mathcal{P} \rightarrow \mathbb{R}^+ \cup \mathbb{N}$ 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 $V(T)$.*

We note $^\circ T$ (resp. $^\circ P$) the set of places (resp. transitions) preceding the transition T (resp. the place P) and we note T° (resp. P°) the set of transitions (resp. places) succeeding to the transition T (resp. to the place P).

A. Semantic intuition

A discrete transition T is *enabled* if each place $P_i \in {}^\circ T$ 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 $P_i \in {}^\circ T$ and $Post(T, P_j)$ tokens are added to each transition $P_j \in T^\circ$. The marking can be sufficient to allow fewer simultaneous firings. The number of possible successive firing allowed by a given marking is 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 $P_i \in {}^\circ T$ 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 $P_i \in {}^\circ T$ and a flow of $Post(T, P_j) \times v(T)$ tokens are added to each transition $P_j \in T^\circ$.

B. Evolution graph

The behavior of a THPN can be represented by an evolution graph, which is a classical Petri net [2]. 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 occurred events, with time of the event occurrence and with marking of all continuous places. IB-states are annotated 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 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 helpful for determining sharing schema). And *priority* ranks transitions and gives limited resources according to the ranks (*e.g.* catalytic phenomena).

C. Signature

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

V is the following set of variables:

- for every $P \in \mathcal{P}$, we need a variable which denotes the marking of P. Conventionally, this variable will be denoted $m(P)$,
- for every $T \in T^C$, the variable $v(T)$ denotes the instantaneous speed of $T \in T^C$,
- for every $T \in T^D$, the variable $dg(T)$ denotes 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.

IV. ASSOCIATED EVENT CLOCK AUTOMATON

Owing to the extension of Event Clock Logic proposed in Section II, we can extract an Event Clock automaton from a THPN model. This will allow us to prove properties on THPN. Let us first recall the definition of an Event Clock automaton [5].

Definition 12 *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 of the THPN,*
- *$\mathcal{F} = \{F_1, ..., F_n\}$ where $F_i \subseteq L$ is a set of sets of accepting locations*

Definition 13 *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 $l_0 \in L_0$,*
- *$(l_i, \psi_i, l_{i+1}) \in E$ with $(\tau, i) \leftsquigarrow \psi_i$*
- *for every $F_i \in \mathcal{F}$, there exists infinitely many positions j such that $l_j \in F_i$.*

Definition 14 *The timed language of an Event Clock automaton A, denoted $\mathcal{L}(A)$, is the set of timed traces recognized by A.*

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 construct the set of locations, the third one determines the initial and accepting locations and the fourth one constructs 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 all along time the IB-state numbered i is true. $\phi_1(IB_i)$ 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 values of the continuous marking at the transitions T_i^{GE} and T_{i+1}^{GE}.

$$\phi_1(IB_i) \equiv \wedge \left(\begin{array}{l} \bigwedge_{P \in P^D} (m(P) = val(m(P))) \\ \bigwedge_{T \in T^C} (v(T) = val(v(T))) \\ \bigwedge_{T \in T^D} (dg(T) = val(dg(T)) \\ \bigwedge_{P \in P^C} (m(P) \in I(T^{GE})_i^{i+1}) \end{array} \right)$$

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. 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}.

$$\phi_2(T_i^{GE}) \equiv \wedge \left(\begin{array}{l} \bigwedge_{IB_i \in {}^\circ T_i^{GE}} \phi_1(IB_i) \\ \bigwedge_{e \in Evt(T)} e \\ \bigwedge_{\substack{P \in P^C \\ \Delta t = evt\ time}} (m(P) = val(m(P))) \\ \bigwedge_{le \in Evt({}^\circ\circ T)} (x_{le} = \Delta t) \end{array} \right)$$

3- Start and accepting locations: The start location is the location corresponding to the first transition T_0^{GE}. The accepting locations are the ones such that 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 represent the time of the IB-state. Finally, an edge outgoing from a location l is labelled by the formula of the location l.

V. BIOLOGICAL ILLUSTRATION

Most amphibians undergo numerous morphological changes at the tadpole stage, a biological process called metamorphosis. Amphibian metamorphosis can be divided into three periods. During premetamorphosis the feeding tadpole grows. During prometamorphosis hindlimbs grow and differentiate. Finally, tail resorption characterizes metamorphic climax. All these modifications are under control of thyroid hormone, denoted TH [8]. It is relevant to distinguish two molecular forms of TH [9]. Thyroxine (tetraiodothyronine, T4) corresponds to the major form secreted by the thyroid gland, it is an “inactive” form of TH and is considered

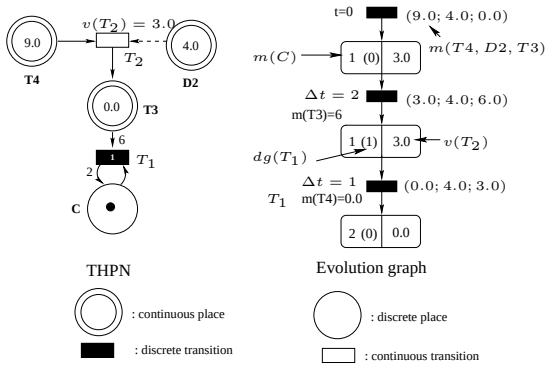

Fig. 1. The THPN of cellular cycle activation in amphibian metamorphosis and its evolution graph.

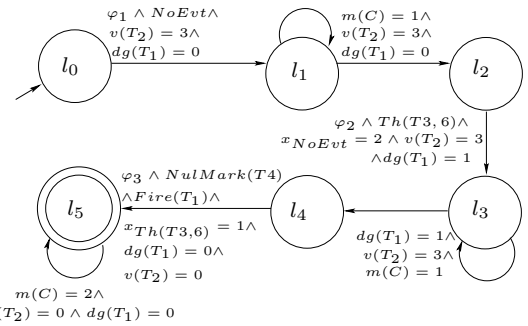

Fig. 2. Event Clock automaton of the THPN model, denoted A_M. $\varphi_1 \equiv (m(C) = 1) \wedge (m(T4) = 9) \wedge (m(D2) = 4) \wedge (m(T3) = 0)$, $\varphi_2 \equiv (m(C) = 1) \wedge (m(T4) = 3) \wedge (m(D2) = 4) \wedge (m(T3) = 6)$ and $\varphi_3 \equiv (m(C) = 2) \wedge (m(T4) = 0) \wedge (m(D2)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

as a pro-hormone. Triiodothyronine (T3) is the biologically active form but it is secreted in smaller quantity [6].

Among all the changes related to metamorphosis, we were particularly interested in the regulatory mechanisms responsible for hindlimb growth.

Hindlinb growth is induced when the concentration of plasmatic TH is minimal [8]. This morphological modification is nevertheless triggered thanks to the type 2 iodothyronine deiodinase, denoted D2 [10] which transforms the inactive form T4 into the active form T3: $D2 + T4 \rightarrow T3 + D2$. The enzymatic action enables the limb cells to reach the T3 concentration necessary to activate the cell cycle. Growth of hindlimbs are observed.

A. 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 [8] and D2 [10].

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 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 waiting for the external blood flow to fill T4.

B. Automaton construction

The Event Clock automaton A_M is presented in Figure 2. Traces of A_M correspond to the execution of the THPN.

C. Proof of a property

Among 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: $\quad \diamond \rhd_{\geq 3} (v(T_2) = 0)$
or equivalently: $\quad \neg\Box\neg \rhd_{\geq 3} (v(T_2) = 0)$
where $\diamond$ means eventually and $\Box$ means always. The first formula means that at a given instant a minimum of three time units will be required to the enzymatic reaction stops ($v(T_2) = 0$) . The second formula (formally equivalent to the first one) means that the following property is wrong: "the end of the enzymatic reaction is globally observed before three time units elapsed".

The Event Clock automaton associated with the negation of the studied property, $A_{\neg\phi}$, is then constructed by using the procedure defined by Raskin and Schobbens in [3], 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 [3], [11]. Since traces of $A_{\neg\phi}$ guarantee the end of the enzymatic reaction $(v(T_2) = 0)$ always occurs before three time units elapsed, the language of the product automaton is then intuitively empty if one of its traces passes through an edge labelled by $(v(T_2) = 0)$ after three time units

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.

VI. DISCUSSION

Hybrid Functional Petri Nets [1] constitute a powerful framework to define computable models of complex biological systems. Many rather large and complex systems have already been modelled using HFPN [12]. Reasoning about those models, in a computer aided manner, is consequently of first interest. Unfortunately, *functions* (the "F" of HFPN) offer such an expressive power that they are the main obstacle to perform *proofs* on models defined using HFPN. Other more restricted logical frameworks without functions and generally without explicit quantitative time [13] are dedicated to precise aspects of biological systems such as genetic regulatory networks. This kind of formalism offers automated proof procedures [14]. 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 [3] 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, but fortunately in biology, 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 and intrinsically 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 validation 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 [15], [16]. We are convinced that Event Clock logic is well suited to add to THPN more and more sophisticated functions.

REFERENCES

[1] 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.

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

[3] J-F. Raskin and P-Y. Schobbens. The logic of event clocks. *JALC*, 1999.

[4] H. Matsuno, A. Doi, M. Nagasaki, and S. Miyano. Hybrid Petri net representation of gene regulatory network. In *Pacific Symposium of Biocomputing*, volume 5, pages 338–349, 2000.

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

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

[7] S. Troncale, J.-P. Comet, and G. Bernot. Verification of Timed Hybrid Petri Nets with temporal logic. Technical report, IBISC, 2007.

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

[9] C. Rose. Integrating ecology and developmental biology to explain the timing of frog metamorphosis. *TRENDS in Ecolgy and Evolution*, 20:129–135, 2005.

[10] 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.

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

[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 haematopoiesis. In *PSB*, 2006.

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

[14] G. Bernot, J.-P. Comet, A. Richard, and J. Guespin. Application of formal methods to biological regulatory networks: Extending thomas'asynchronous logical apporoach with temporal logic. *J.T.B*, 2004.

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

[16] M. Gribaudo, A. Horvath, E. Tronci, E. Ciancamerla, and M. Minichino. Model-checking based on fluid Petri nets. In *Computer Safety, Reliability and Security*, 2002.

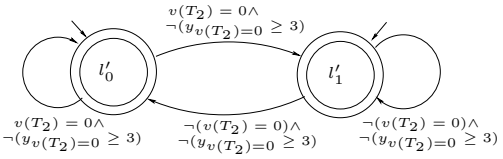

Fig. 3. Event Clock automaton $A_{\neg\phi}$ reduced to accessible locations

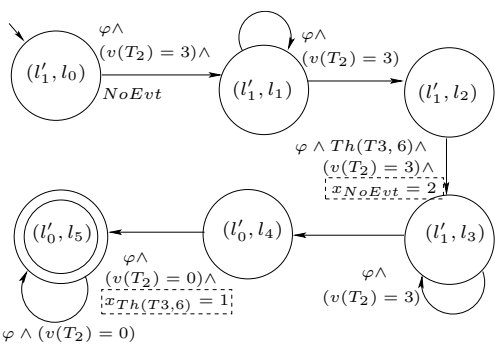

Fig. 4. Event Clock automaton $A_M \times A_{\neg\phi}$. $\varphi \equiv \neg(y_{v(T_2)=0} \geq 3)$