Coloured hybrid Petri nets for systems biology

Mostafa Herajy[1*], Fei Liu[2*] and Christian Rohr[3*]

[1] Department of Mathematics and Computer Science, Faculty of Science, Port Said University, 42521 - Port Said, Egypt
[2] Control and Simulation Center, Harbin Institute of Technology, Postbox 3006, 150080, Harbin, China
[3] Computer Science Institute, Brandenburg University of Technology Postbox 10 13 44, 03013 Cottbus, Germany

Abstract. Coloured Petri nets are imperative for studying bigger biological models, particularly, those which expose repetition of components. Such models can be easily scaled by minor changes of a few parameters. Similarly, studying certain biological phenomena necessitates the existence of discrete and continuous variables as well as continuous and stochastic processes in one and the same model. Thus, hybrid modelling and simulation is indispensable to deal with these challenges. In this paper we introduce a generic Petri net class, Coloured Generalised Hybrid Petri Nets ($\mathcal{GHPN}^{\mathcal{C}}$) by combining coloured Petri nets and Generalised Hybrid Petri Nets ($\mathcal{GHPN}$), which integrates discrete and continuous places as well as stochastic and deterministic transitions on the coloured level. Moreover, we present a case study which illustrates a possible and typical application where such a Petri net class is highly required.

Keywords: coloured Petri nets; hybrid Petri nets; stochastic and continuous simulation

1 Introduction

Petri nets have been widely used for modelling and analysis of biological systems, see e.g.,[2,10,18,21,31,33,34,35,36,39,40]. Their intuitive graphical representation makes them easily approachable by biologists. Furthermore, Petri nets possess well-established mathematical notations which may render them as the future de facto standard for modelling biological systems compared to other graphical languages currently in use. Nevertheless, with the rapid progress of systems biology, standard place/transition Petri nets become insufficient to accommodate the needs of systems biologists to study larger models. Therefore, many different extensions have been adapted for their potential contribution to systems biology. Among such promising extensions are hybrid Petri nets ($\mathcal{HPN}$) [1,6] and coloured Petri nets ($\mathcal{PN}^{\mathcal{C}}$) [23].

* corresponding authors

M. Heiner (Ed.): BioPPN 2014, a satellite event of PETRI NETS 2014,
CEUR Workshop Proceedings Vol. 1159, 2014.

Hybrid Petri nets [1,6] have been increasingly motivated for their contribution to systems biology [17,18,35,36]. On the one hand (biological) systems may occur at different time scales: slow and fast [14,25]. Using one modelling paradigm (i.e., discrete or continuous paradigm alone) tends to be inefficient for studying multi-timescale biological models [18]. On the other hand the integration of discrete and continuous variables as well as deterministic and stochastic processes in the same model is necessary in certain application scenarios to implement particular model semantics [18,36]. For instance, in [21] discrete places and immediate transitions were employed to implement the part related to cell division of the eukaryotic cell cycle while continuous and stochastic transitions were used to represent and quantitatively simulate biological reactions. Additionally, the $\mathcal{HPN}$ formalism provides a different approach to control the accuracy and the speed of biological models during simulation [18]. Thus, they provide the modeller with a tool to make a trade-off between the simulation's efficiency and the result's accuracy.

Another, yet powerful extension of standard Petri nets is coloured Petri nets ($\mathcal{PN}^{\mathcal{C}}$) [23,31,40], where a group of similar components are represented by one component, each of which is defined as and thus distinguished by a colour [27]. $\mathcal{PN}^{\mathcal{C}}$ are very useful for modelling larger biological systems where low-level Petri nets do not scale well, while a (biological) system modelled via coloured Petri nets can easily be scaled with minor modifications of certain coloured variables. Furthermore, modelling of biological systems is moving from single to multiple scales (multi-scale modelling), which introduces a series of challenges such as repetition of components, communication between components, organisation of components, and pattern formation of components [27]. All these challenges potentially could be tackled by coloured Petri nets rather than standard Petri nets.

Nevertheless, with the rapid change of type and size of models of biological systems which have to be analysed, a Petri net class that integrates all the features of hybrid Petri nets and coloured Petri nets is highly required. The high-level representation of coloured Petri nets can be used to model larger biochemical networks and stochastic and continuous components can be simultaneously used to facilitate the efficient simulation of multi-timescale models.

Thus, in this paper we introduce a class of Petri nets, Coloured Generalised Hybrid Petri Nets ($\mathcal{GHPN}^{\mathcal{C}}$) by combining all features of Generalised Hybrid Petri Nets ($\mathcal{GHPN}$) [18] and the merits of Coloured Petri Nets ($\mathcal{PN}^{\mathcal{C}}$) [27]. The new class supports a rich set of transition types as well as discrete and continuous places. Moreover, it permits the full interplay between stochastic and deterministic processes at the coloured level. All the feature of $\mathcal{GHPN}^{\mathcal{C}}$ are implemented in Snoopy [15] – a unifying Petri net tool which is available free of charge for academic use.

The paper is structured as follows: first we briefly discuss the related work of hybrid Petri nets and coloured Petri nets followed by a brief background of each of those net types. Next, we present the formal definition of the new Petri net class followed by its semantic as well as an outline of the simulation algorithm used to produce the dynamic of $\mathcal{GHPN}^{\mathcal{C}}$. Afterwards, we present one possible and typical

application of $\mathcal{GHPN}^{\mathcal{C}}$ in the context of systems biology, the repressilator model, which is an engineered synthetic system encoded on a plasmid. We conclude by summarizing possible extensions of Coloured Generalised Hybrid Petri Nets.

2 Related Work

In this section, we will briefly describe related work on hybrid and coloured Petri nets for systems biology.

2.1 Hybrid Petri Nets for Systems Biology

Hybrid Petri nets have been introduced in [1] to deal with situations where discrete and continuous entities exist in the same model. The motivation given in early publications was in modelling technical systems whereby discrete places are used to model the states of switch-like components while continuous places are used to represent their fluid counterparts. The pioneer work of introducing hybrid Petri nets to systems biology was done by Matsuno et al., in [36]. They had noticed that using equal inflow and outflow in the fluid part of hybrid Petri nets is not natural to use them to model biological processes. Thus, they introduced Hybrid Functional Petri Nets (HFPN) [36,35]. Many successful models have been implemented using HFPN (e.g., see [8,34]).

In [18], we have used Hybrid Petri nets in a different way whereby stochastic transitions are used to model slow biological processes, while fast processes are modelled via continuous transitions. In [21], this approach is used to model the eukaryotic cell cycle.

2.2 Coloured Petri Nets for Systems Biology

The early applications of coloured Petri nets for systems biology were limited, which, to our knowledge, can almost only be seen in [2,3,5,11,26,38,40,41,42]. These studies are rather small and usually resort to `Design/CPN` [4] or its successor `CPN Tools` [23]. However these tools were not specifically designed with the requirements of systems biology in mind. Thus they are not suitable in many aspects, e.g., they do not directly support stochastic or continuous modelling, nor stochastic or deterministic simulation.

Since coloured Petri nets were introduced to our Petri net modelling tool, Snoopy, we have extensively explored the application of coloured Petri nets for (multiscale) systems biology. For example, in [29], we used coloured stochastic Petri nets to model and analyse stochastic membrane systems, where each compartment is encoded as a colour. In [30], we described the multiscale modelling of coupled Ca^{2+} channels using coloured stochastic Petri nets by considering two levels: Ca^{2+} release sites and Ca^{2+} channels. In [10], Coloured stochastic and coloured continuous Petri nets are used for multiscale modelling and analysis of Planar Cell Polarity in the Drosophila wing, and the built model consists of more than 800 cells. In [37], a case study of phase variation in bacterial colony growth

is given, in which cells are distributed on a two-dimensional grid represented by both Cartesian and polar coordinate systems.

3 Background

3.1 Generalised Hybrid Petri Nets

In [18] we have introduced a special class of Hybrid Petri nets called Generalised Hybrid Petri Nets ($\mathcal{GHPN}$). The main two objectives of $\mathcal{GHPN}$ are: to provide systems biologists with a convenient and flexible graphical tool to model and simulate biological processes with different time scales and to facilitate the process of constructing hybrid models where stochastic and deterministic (i.e, continuous) processes are tightly coupled. To fulfil such objectives, $\mathcal{GHPN}$ contain a rich set of element types: places (discrete, and continuous), transitions (stochastic, immediate, deterministic, stochastic, and continuous), and arcs (standard, read, inhibitor, equal, and reset) [18,17]. Figure 1 depicts the graphical representation of the different element types of $\mathcal{GHPN}$.

Discrete places (single line circle) hold non-negative integer numbers which may represent the number of molecules of a given species (tokens in Petri net notions). On the other hand, continuous places - which are represented by the shaded line circle - hold non-negative real numbers which represent the concentration of a certain species. Please note that, except when otherwise mentioned, the number which a place p_i holds, also called its marking, is referred to by $m(p_i)$.

Furthermore, $\mathcal{GHPN}$ offer five transition types: stochastic, immediate, deterministically delayed, scheduled, and continuous transitions [18]. Stochastic transitions, which are drawn in Snoopy as a single line square, fire randomly with an exponentially distributed random delay. The user can specify a set of firing rate functions that determine the random firing delay. The transitions' pre-places can be used to define the firing rate functions of stochastic transitions. Immediate transitions (black bar) fire with zero delay, and have always highest priority in the case of conflicts with other transitions. They may carry weights (which can also be defined by state-dependent functions) that specify the relative firing frequency in the case of conflicts between immediate transitions. Deterministically delayed transitions (represented as black squares) fire after a specified constant time delay. Scheduled transitions (grey squares) fire at user-specified absolute time points. Continuous transitions (shaded line square) fire continuously in the same way as in continuous Petri nets. Their semantics are governed by a set of ordinary differential equations (ODEs) which define the changes in the transitions' pre- and post-places. More details about the biochemical interpretation of deterministically delayed, scheduled, and immediate transitions can be found in [16].

The connection between those two types of nodes (places and transitions) takes place using a set of different arcs. $\mathcal{GHPN}$ offer six types of arcs: standard, inhibitor, read, equal, reset, and modifier arcs. Standard arcs connect transitions with places or vice versa. They can be discrete, i.e., carry non-negative integer-valued weights (stoichiometry in the biochemical context), or continuous, i.e.,

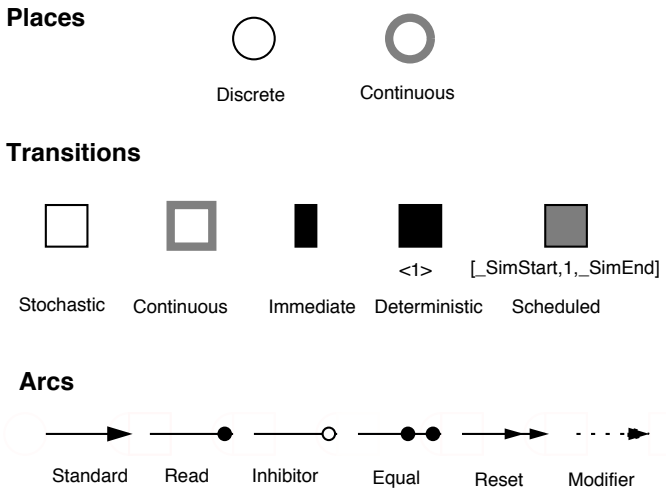

Fig. 1. Graphical representation of the $\mathcal{GHPN}$ elements [18]. Places are classified as discrete and continuous; transitions as continuous, stochastic, immediate, deterministically delayed and scheduled; and arcs as standard, inhibitor, read, equal, reset, and modifier.

carry non-negative real-valued weights. In addition to their influence on the enabling of transitions, they also affect the place marking when a transition fires by adding (removing) tokens from the transition's post-places (pre-places).

Extended arcs such as inhibitor, read, equal, reset, and modifier arcs can only be used to connect places to transitions, and not vice versa. A transition connected with an inhibitor arc is enabled (with respect to the corresponding pre-place) if the marking of the pre-place is less than the arc weight. In contrast, a transition connected with a read arc is enabled if the marking of the pre-place is greater than or equal to the arc weight. Similarly, a transition connected using an equal arc is enabled if the marking of the pre-place is equal to the arc weight.

The other two remaining arcs do not affect the enabling of transitions. A reset arc is used to reset a place marking to zero when the corresponding transition fires. Modifier arcs permit one to include any place in the transitions' rate functions and simultaneously preserve the net structure restriction. Besides, the markings of places connected using read, inhibitor, equal, or modifier arcs does not change when the corresponding transition fires.

3.2 Coloured Petri nets

Coloured Petri nets [22] consist, as standard Petri nets, of places, transitions and arcs that connect places and transitions. Additionally, a coloured Petri net is characterised by a set of finite colour sets. Each place gets assigned a colour set and may contain distinguishable tokens; each token has a colour of this colour set.

As there can be several tokens of the same colour on a given place, the tokens on a place define a multiset over the place's colour set.

Each transition gets a guard, which is a Boolean expression over variables, constants, functions, etc. The guard of a transition must be evaluated to true for the enabling of the transition. The trivial guard "true" is usually not explicitly given.

Each arc gets assigned an expression, rather than an integer number in standard Petri nets; the result type of this expression is a multiset over the colour set of the connected place.

4 Coloured Generalised Hybrid Petri Nets

In this section we present the definition of coloured generalised hybrid Petri nets. $\mathcal{GHPN}^{\mathcal{C}}$ reuse all elements of $\mathcal{GHPN}$, however on the coloured level. We start off by defining $\mathcal{GHPN}^{\mathcal{C}}$. Afterwards, we present the semantics of $\mathcal{GHPN}^{\mathcal{C}}$ that governs the execution of models constructed by it.

4.1 Formal Definition

Definition 1 *A Coloured Generalised Hybrid Petri Net $\mathcal{GHPN}^{\mathcal{C}}$ is a nine-tuple $N =< P, T, A, \sum, C, F, V, G, m_0 >$[27,18], where:*

- $P = P_{disc} \cup P_{cont}$ *whereby P_{disc} is the set of discrete places and P_{cont} is the set of continuous places.*
- $T = T_{stoch} \cup T_{im} \cup T_{timed} \cup T_{scheduled} \cup T_{cont}$ *with:*
  1. T_{stoch} *is the set of stochastic transitions, which fire stochastically after an exponentially distributed waiting time.*
  2. T_{im} *is the set of immediate transitions, which fire with waiting time zero; they have higher priority compared with other transitions.*
  3. T_{timed} *is the set of deterministically delayed transitions, which fire after a deterministic time delay.*
  4. $T_{scheduled}$ *is the set of scheduled transitions, which fire at predefined time points.*
  5. T_{cont} *is the set of continuous transitions, which fire continuously over time.*
- $A = A_{disc} \cup A_{cont} \cup A_{inhibit} \cup A_{read} \cup A_{equal} \cup A_{reset} \cup A_{modifier}$ *is the set of directed arcs, with:*
  1. $A_{disc} \subseteq ((P \times T) \cup (T \times P))$ *defines the set of discrete arcs.*
  2. $A_{cont} \subseteq ((P_{cont} \times T) \cup (T \times P_{cont}))$ *defines the set of continuous arcs.*
  3. $A_{read} \subseteq (P \times T)$ *defines the set of read arcs.*
  4. $A_{inhibit} \subseteq (P \times T)$ *defines the set of inhibit arcs.*
  5. $A_{equal} \subseteq (P_{disc} \times T)$ *defines the set of equal arcs.*
  6. $A_{reset} \subseteq (P \times T^D)$ *defines the set of reset arcs,*
  7. $A_{modifier} \subseteq (P \times T)$ *defines the set of modifier arcs.*

  where $T^D = T_{stoch} \cup T_{im} \cup T_{timed} \cup T_{scheduled}$ is the set of discrete transitions.

- $\sum$ *is a finite, non-empty set of colour sets.*
- $C : P \to \sum$ *is a colour function that assigns to each place* $p \in P$ *a colour set* $C(p) \in \sum$.
- $F : A \to EXP$ *is an arc function that assigns to each arc* $a \in A$ *an arc expression of a multiset type* $C(p)_{MS}$*, where* p *is the place connected to the arc* a.
- V *is a set of functions* $V = \{g, d, w, f\}$ *where :*
  1. $g : T_{stoch} \to H_s$ *is a function which assigns a stochastic hazard function* h_{s_t} *to each stochastic transition instance* $t_j \in T_{stoch}$*, whereby* $H_s = \{h_{s_t} | h_{s_t} : \mathbb{R}_0^{|^\bullet t_j|} \to \mathbb{R}_0^+, t_j \in T_{stoch}\}$ *is the set of all stochastic hazard functions, and* $g(t_j) = h_{s_t}, \forall t_j \in T_{stoch}$.
  2. $w : T_{im} \to H_w$ *is a function which assigns a weight function* h_w *to each immediate transition instance* $t_j \in T_{im}$*, such that* $H_w = \{h_{w_t} | h_{w_t} : \mathbb{R}_0^{|^\bullet t_j|} \to \mathbb{R}_0^+, t_j \in T_{im}\}$ *is the set of all weight functions, and* $w(t_j) = h_{w_t}, \forall t_j \in T_{im}$.
  3. $d : T_{timed} \cup T_{scheduled} \to \mathbb{R}_0^+$*, is a function which assigns a constant time to each deterministically delayed and scheduled transition instance representing the (relative or absolute) waiting time.*
  4. $f : T_{cont} \to H_c$ *is a function which assigns a rate function* h_c *to each continuous transition instance* $t_j \in T_{cont}$*, such that* $H_c = \{h_{c_t} | h_{c_t} : \mathbb{R}_0^{|^\bullet t_j|} \to \mathbb{R}_0^+, t_j \in T_{cont}\}$ *is the set of all rates functions and* $f(t_j) = h_{c_t}, \forall t_j \in T_{cont}$.
- $G : T \to EXP$ *is a guard function that assigns to each transition* $t \in T$ *a guard expression of the Boolean type.*
- $m_0 : P \to EXP$ *is an initialisation function that assigns to each place* $p \in P$ *an initialisation expression of a multiset type* $C(p)_{MS}$.

Here, $\mathbb{R}_0^+$ *denotes the set of non-negative real numbers,* $^\bullet t_j$ *denotes the set of pre-places of a transition* t_j.

□

4.2 Semantics

The formal semantics of $\mathcal{GHPN}^{\mathcal{C}}$ is defined by unfolding the coloured hybrid Petri nets into the equivalent low level one. Thus we firstly discuss how $\mathcal{GHPN}^{\mathcal{C}}$ can be unfolded into $\mathcal{GHPN}$. Then, we extend the formal semantics which has been presented in [18] and apply it to $\mathcal{GHPN}^{\mathcal{C}}$.

Unfolding Uncoloured Petri nets can be folded into coloured Petri nets, if partitions of the place and transition sets are given. Vice versa, coloured Petri nets with finite colour sets can be automatically unfolded into uncoloured Petri nets, which then allows the use of all of the existing powerful standard Petri net analysis techniques. The conversion between uncoloured and coloured Petri nets

changes the style of representation, but does not change the actual net structure of the underlying biological reaction network.

In [32], we have presented an efficient unfolding method for coloured Petri nets with finite coloured sets, in which we provide two approaches to compute transition instances. For a transition, if the colour set of each variable in its guard has a finite integer domain, a constraint satisfaction approach is used to obtain all valid transition instances. Otherwise, a general unfolding algorithm is adopted, in which some optimisation techniques are used like partial binding–partial test and pattern matching [27,32].

Hybrid semantics The semantics of $\mathcal{GHPN}^{\mathcal{C}}$ is given in terms of the discrete and continuous transitions. To harmonise the mathematical notations, let $T^C = T - T^D = T_{cont}$ denote the set of continuous transitions. Before we proceed, we assume that the coloured expression which is assigned to an arc is evaluated to a numeric value. Likewise the initial marking.

Definition 2 (Enabling condition) *Let* $N =< P, T, A, \sum, C, F, V, G, m_0 >$ *be a coloured generalised hybrid Petri net and* m *be the marking of* N *at time* τ*. A transition* $t_j \in T$ *is enabled in the marking* m*, denoted by* $m[t_j\rangle$*, iff* $\forall p_i \in {}^\bullet t_j$*:*

- $m(p_i) \geq F(p_i, t_j)$, *if* $(p_i, t_j) \in A_{cont} \cup A_{disc} \wedge t_j \in T^D$,
- $m(p_i) > 0$, *if* $(p_i, t_j) \in A_{cont} \wedge t_j \in T^C$,
- $m(p_i) \geq F(p_i, t_j)$, *if* $(p_i, t_j) \in A_{read}$,
- $m(p_i) < F(p_i, t_j)$, *if* $(p_i, t_j) \in A_{inhibit}$,
- $m(p_i) = F(p_i, t_j)$, *if* $(p_i, t_j) \in A_{equal}$.
- $G(t_j) = true$.

□

Definition 3 (Firing rule of discrete transitions) *Let* $N =< P, T, A, \sum, C, F, V, G, m_0 >$ *be a coloured generalised hybrid Petri net,* m *a marking of* N*, and* $t_j \in T^D$ *a transition enabled in the marking* m*,* $m[t_j\rangle$*, at time* τ*. The transition* t_j *can fire and reach a new marking* m'*, denoted by* $m[t_j\rangle m'$*, at time* $\tau + d_j$ *if it is still enabled at that new time, with:*

- $\forall p_i \in {}^\bullet t_j$

$$m'(p_i) = \begin{cases} m(p_i) - F(p_i, t_j) & \text{if } (p_i, t_j) \in A_{cont} \cup A_{disc} \\ 0 & \text{if } (p_i, t_j) \in A_{reset} \\ m(p_i) & \text{else} \end{cases}$$

- $\forall p_i \in t_j^\bullet$

$$m'(p_i) = m(p_i) + F(t_j, p_i)$$

where

$$d_j = \begin{cases} d(t_j) & \text{if } t_j \in T_{timed} \\ \tau + d(t_j) & \text{if } t_j \in T_{scheduled} \\ d_{stoch}(t_j) & \text{if } t_j \in T_{stoch} \\ 0 & \text{if } t_j \in T_{im} \end{cases}$$

is a delay which is associated to the discrete transition t_j and $d_{stoch}(t_j)$ is the random firing delay with negative exponential probability density function calculated for each stochastic transition t_j using its rate $g(t_j)$.

□

According to the above enabling and firing definitions, discrete transitions follow a policy which is called an *enabling memory policy* [24].

Moreover, when multiple immediate transitions are concurrently enabled, conflicts are solved by computing the relative firing frequencies of each enabled immediate transition. That is if an immediate transition t_j is enabled in the current marking m, then it fires with the probability given by (1).

$$\frac{w(t_j)(m)}{\sum_{t_k \in T_{im} \wedge isEnabled(t_k,m)} w(t_k)(m)} \tag{1}$$

where w is the weight assigned to immediate transitions.

Firing of continuous transitions The semantics of continuous transitions is analogue to the ones in continuous Petri nets with maximal firing speeds depending on time as introduced in [6] and tailored to the specific needs in systems biology. The transitions' current firing rates (instantaneous firing speeds) depend on the current marking of their pre-places (i.e., species concentrations). In what follows, the firing semantics of continuous transitions is formally given.

We introduce the following notation. Let $v_j(\tau)$ represent the current firing rate of a transition $t_j \in T^C$ at time τ, $m_i(\tau) = m(p_i)$ denotes the current marking of a place p_i at time τ, and $f_j(\tau)$=$f(t_j)$ denotes the maximal firing rate of a transition t_j at time τ, then:

$$v_j(\tau) = \begin{cases} f_j(\tau) & \text{if } t_j \text{ is enabled} \\ 0 & \text{else} \end{cases} \tag{2}$$

Equation (2) implies that a continuous transition can fire with its maximal rate if it is enabled or its rate will be zero otherwise.

When a continuous transition is enabled, it fires as soon as possible and its effect on the connected places can be given by the following definition.

Definition 4 (Firing of continuous transitions) *Let $N =< P, T, A, \sum, C, F, V, G, m_0 >$ be a coloured generalised hybrid Petri net, m a marking of N, $t_j \in T^C$ a transition enabled in the marking m, $m[t_j\rangle$, at time τ, and $v_j(\tau)$ denotes the current firing rate of the transition t_j. The transition t_j fires with:*

– $\forall p_i \in {}^\bullet t_j$

$$m_i(\tau + d\tau) = m_i(\tau) - F(p_i, t_j) \cdot v_j(\tau) d\tau \tag{3}$$

– $\forall p_i \in t_j^\bullet$

$$m_i(\tau + d\tau) = m_i(\tau) + F(t_j, p_i) \cdot v_j(\tau) d\tau \tag{4}$$

□

Equations (3) and (4) are called outflow and inflow of a place p_i, respectively, due to the firing of a transition t_j [6].

Generation of the corresponding ODEs For a given transition $t_j \in T^C$, the functions $read(u, p_i)$, $inhibit(u, p_i)$ are defined as follows:

$$read(u, m(p_i)) = \begin{cases} 1 & \text{if } m(p_i) \geq u \\ 0 & \text{else} \end{cases}$$

with $u = F(p_i, t_j) \wedge (p_i, t_j) \in A_{read}$, and

$$inhibit(u, m(p_i)) = \begin{cases} 1 & \text{if } m(p_i) < u \\ 0 & \text{else} \end{cases}$$

with $u = F(p_i, t_j) \wedge (p_i, t_j) \in A_{inhibit}$.

Then the ODE corresponding to each continuous place in $\mathcal{GHPN}^{\mathcal{C}}$ can be generated using (5)

$$\begin{aligned} \frac{dm\,(p_i)}{d\tau} = & \sum_{t_j \in {}^\bullet p_i} F\,(t_j, p_i) \cdot v_j\,(\tau) \cdot read(u, m(p_i)) \cdot inhibit(u, m(p_i)) - \\ & \sum_{t_j \in p_i {}^\bullet} F\,(p_i, t_j) \cdot v_j\,(\tau) \cdot read(u, m(p_i)) \cdot inhibit(u, m(p_i)) \end{aligned} \tag{5}$$

4.3 Simulation

The semantics of $\mathcal{GHPN}^{\mathcal{C}}$ which we have discussed in Section 4.2 can be produced via simulation. In simulating $\mathcal{GHPN}^{\mathcal{C}}$ models, the following two main issues need be considered: the partitioning of the net's transitions into stochastic and continuous ones, and the synchronisation between the stochastic and continuous regimes.

The partitioning of the net elements has an important influence on both the accuracy and the efficiency of $\mathcal{GHPN}^{\mathcal{C}}$ simulation. The more continuous transitions we use, the faster the simulation speed we get. However, this will have a high impact on the result accuracy. Thus, it is important to appropriately

partition the model transitions into stochastic and continuous ones. This can be done using either static or dynamic partitioning. In static partitioning, the process of assignment of each transition to either the stochastic or continuous category is done off line (i.e., before the simulation starts). Transitions with high firing rate will be considered continuously while transitions with low firing rates are considered stochastically. Additionally, the number of tokens of the transition's pre-places plays a role in determining the transition type. An important merit of this approach is that there is no additional overhead to the simulation due to the partitioning process. However, transition rates can dramatically change during the simulation. This motivates the use of dynamic partitioning. Using dynamic partitioning, transitions can change their type during simulation from stochastic to continuous or vice versa. Such change is usually based on either the transition rate and/or the number of tokens in the transition's pre-paces. Nevertheless, dynamic partitioning will result in additional overhead due to the repeated check of fulfilment of the partitioning conditions. Therefore, if the gain due to the use of dynamic partitioning is less than the partitioning overhead, static partitioning should be used instead. A detailed discussion about this issue can be found in [18].

Similarly, the synchronisation between stochastic and continuous regimes is vital for the accuracy of the simulation result. At the end, stochastic and continuous transitions are not isolated from each other. They mutually affect each other. Thus, we need a mechanism to better perform the synchronisation. One choice is to use (6) to detect the occurrence of a stochastic event while the numerical integration is progressing [12],

$$g(\mathbf{x}) = \int_{t}^{t+\tau} a_0^s(\mathbf{x})dt - \xi = 0\,, \tag{6}$$

where ξ is a random number exponentially distributed with a unit mean, and $a_0^s(\mathbf{x})$ is the cumulative rate of all stochastic transitions.

Once we agreed on a selection of a partitioning algorithm and a synchronisation approach we can execute the $\mathcal{GHPN}^{\mathcal{C}}$ model by repeating a set of steps. In [18], we have presented a hybrid simulation algorithm based on (6) using both static and dynamic partitioning. The main steps are outlined here while the details can be found in [18].

Starting from an initial marking, the simulation algorithm computes state changes over the time which are regarded as the current marking at each simulation step. Initially, the rates of stochastic and continuous transitions are calculated. Next, the accumulated rate function of stochastic transitions is calculated. When the model contains one or more continuous transitions, the simulation algorithm constructs an ODE for each place. ODE construction is done via (5). Afterwords, the simulator numerically integrates the resulting set of ODEs simultaneously with (6) until an event occurs. The event can be one of the following: the enabling of an immediate transition, the enabling of a deterministically delayed transition, a deterministically delayed transition has finished its delay, or a stochastic event has occurred (i.e., Equation (6) is satisfied).

In all cases the ODE integrator is interrupted and the appropriate action is taken (e.g., by firing a discrete transition). After that the ODE integrator is restarted with the new marking to account for the discontinuity which is due to the firing of a discrete transition.

When there is no continuous transition in the system being simulated, the simulation of $\mathcal{GHPN}^{\mathcal{C}}$ model is simplified to the simulation of stochastic Petri nets which can be carried out using, e.g., Gillespie's stochastic simulation algorithm [13].

5 Implementation

All features of $\mathcal{GHPN}^{\mathcal{C}}$ which have been discussed in this paper are implemented in Snoopy [15] – a unifying Petri net tool that supports the construction and execution of different Petri net classes among them are: standard, extended, continuous, stochastic, and hybrid Petri nets. Those classes are available both on the uncoloured and coloured level. Snoopy is platform-independent and provided free of charge for academic purposes.

Snoopy also supports the automatic unfolding of $\mathcal{GHPN}^{\mathcal{C}}$. In Snoopy, we can explicitly unfold a coloured stochastic/continuous/hybrid Petri net to its counterpart, a stochastic/continuous/hybrid Petri net, or we can do this implicitly during the execution of the Petri net model.

Furthermore, we have recently released a server-based version of the Snoopy simulator called S^4 (Snoopy Steering and Simulation Server) [19,20]. S^4 also supports the simulation of $\mathcal{GHPN}^{\mathcal{C}}$. Besides, an $\mathcal{GHPN}^{\mathcal{C}}$ model submitted to S^4 can be steered (i.e., key simulation parameters can be changed on the fly), and collaboratively executed by different users. Snoopy and S^4 can be downloaded from `http://www-dssz.informatik.tu-cottbus.de/snoopy.html`.

6 Applications

In this section we present one case study as a typical application of Coloured Generalised Hybrid Petri Nets.

We will demonstrate the $\mathcal{GHPN}^{\mathcal{C}}$ using an example of a synthetic circuit – the repressilator, which is an engineered synthetic system encoded on a plasmid, and designed to exhibit oscillations [7]. The repressilator system is a regulatory cycle of three genes, denoted by, e.g., gene_a, gene_b and gene_c, where each gene represses its successor, namely, gene_a inhibits gene_b, gene_b inhibits gene_c, and gene_c inhibits gene_a. This negative regulation is realised by the repressors, protein_a, protein_b and protein_c, generated by the genes gene_a, gene_b and gene_c, respectively.

The 1-bounded places as determined by P-invariant analysis and the related transitions as determined by T-invariant analysis are kept discrete. The unbounded places and related transitions are approximated by continuous places and transitions, respectively. That is, places gene_i and blocked_i, and transitions

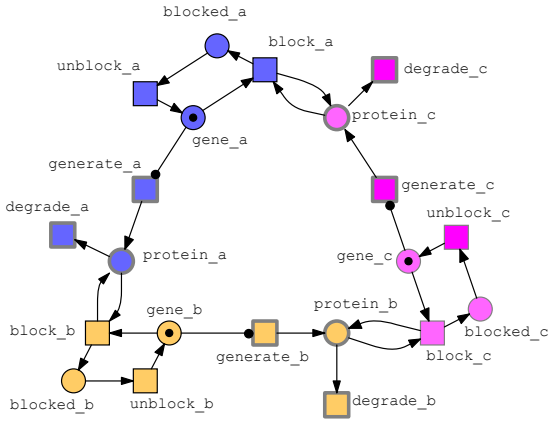

Fig. 2. The $\mathcal{GHPN}$ model of the repressilator, where the continuous places (transitions) are represented by shaded line circles (squares), which is taken from [31].

Table 1. Rate functions for the $\mathcal{GHPN}$ repressilator model. $MA(c)$ denotes the mass action function, where c is a kinetic parameter. See last column for the explicit rate functions for gene a.

transition class	kinetic parameter c	rate function pattern	example: gene a
generate	0.1	$MA(0.1)$	$0.1 * gene\_a$
block	1.0	$MA(1.0)$	$1.0 * gene\_a * protein\_c$
unblock	0.0001	$MA(0.0001)$	$0.0001 * blocked\_a$
degrade	0.001	$MA(0.001)$	$0.001 * protein\_a$

block_i and unblock_i are treated as discrete, where i=a,b,c, and all other nodes as continuous. To distinguish between discrete and continuous nodes, we choose different graphical representations, see Figure 2. The rate functions of the $\mathcal{GHPN}$ model are given in Table 1.

For example, if we assign the rates in Table 1 to the hybrid model and consider them as stochastic or deterministic rates, depending on the transition type, hybrid simulation yields plots as illustrated in Figure 3.

Figure 4 gives an $\mathcal{GHPN}^{\mathcal{C}}$ for the repressilator model in Figure 2. A colour set GeneSet is defined with three colours, a, b and c, to distinguish the three genes. Each place gets assigned this colour set GeneSet. A multiset expression, 1‘all()=1‘a++1‘b++1‘c, is assigned to the place *protein*. In Figure 4, we define a variable x of $GeneSet$, which is used in arc expressions. The predecessor operator “-” in the arc expression $-x$ returns the predecessor of x in an ordered finite

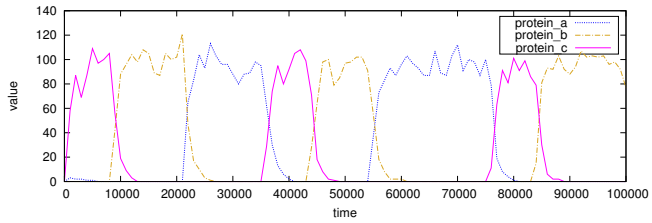

Fig. 3. Plot of one hybrid simulation run for the repressilator. For rate functions, see Table 1. This plot suggests that $\mathcal{GHPN}$ are able to capture the oscillation. Repeated runs look differently; thus stochasticity is captured as well.

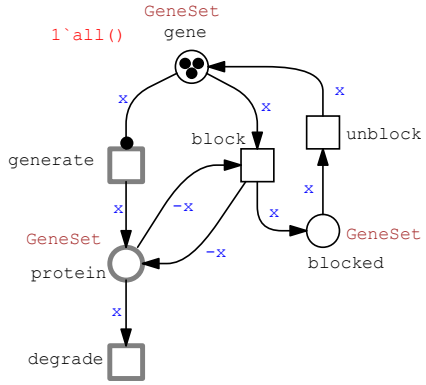

Fig. 4. A colored Petri net model for the repressilator. The declarations: colorset GeneSet = enum with a,b,c, and variable x: GeneSet. With this colour set, this model corresponds exactly to the Petri nets in Figure 2

colour set. For example, if $x = b$, then $-x$ returns a. If x is the first colour, then it returns the last colour. For example, if $x = a$, then $-x$ returns c.

The coloured Petri net model in Figure 4 when unfolded yields the same uncoloured Petri net model in Figure 2. That is, the $\mathcal{GHPN}^{\mathcal{C}}$ model reduces the size of the original stochastic Petri net model to one third, which is a big advantage of coloured Petri nets.

7 Conclusions and Future Work

In this paper we have introduced a new class of coloured Petri nets called Coloured Generalised Hybrid Petri Nets ($\mathcal{GHPN}^{\mathcal{C}}$). $\mathcal{GHPN}^{\mathcal{C}}$ are particularly tailored to systems biologists' needs to model and analyse multiscales models. Moreover, $\mathcal{GHPN}^{\mathcal{C}}$ provide the interplay between stochastic and continuous regimes on the coloured level which eventually provide a tool to control both the accuracy and the speed of running simulation.

In [21] we have proved that marking-dependent arc weights are necessary to model certain biological systems. However, this feature is still not implemented in Snoopy for $\mathcal{GHPN}^{\mathcal{C}}$. Thus we plan to add it in order to widen the types of models that can be constructed via $\mathcal{GHPN}^{\mathcal{C}}$.

Furthermore, the current simulation of $\mathcal{GHPN}^{\mathcal{C}}$ is carried out by firstly unfold the coloured model into an uncoloured one, then it is simulated on that level. Although the unfolding process is fully automated, it may take a considerable amount of time, particularly for bigger models. Thus if an error is observed at the (beginning) of the simulation due to modelling mistakes, the entire unfolding process will completely be repeated. Therefore, simulating $\mathcal{GHPN}^{\mathcal{C}}$ directly on the coloured level [9] will increase the productivity and eventually save a lot of time.

Finally, the presented case study in this paper aims to demonstrate how $\mathcal{GHPN}^{\mathcal{C}}$ works. More sophisticated biological models can be constructed in the future using $\mathcal{GHPN}^{\mathcal{C}}$. For example, we are working on using $\mathcal{GHPN}^{\mathcal{C}}$ to model diffusion-reaction systems described by partial differential equations, where we first discretise the space into a set of grid cells, and then we obtain a $\mathcal{GHPN}^{\mathcal{C}}$ model by representing each grid cell as a colour [28].

References

1. Alla, H., David, R.: Continuous and hybrid Petri nets. J. Circ. Syst. Comp. 8(1), 159 –188 (1998)
2. Banks, R., Steggles, L.J.: A high-level Petri net framework for genetic regulatory networks. Journal of Integrative Bioinformatics 4(3), 1–12 (2007)
3. Chaouiya, C., Remy, E., Thieffry, D.: Qualitative Petri net modelling of genetic networks. In: Transactions on Computational Systems Biology. pp. 95–112. LNCS 4220, Springer (2006)
4. Christensen, S., Jørgensen, J.B., Kristensen, L.M.: Design/CPN - a computer tool for coloured Petri nets. In: Proc. of the Third International Workshop on Tools and Algorithms for Construction and Analysis of Systems. pp. 209–223. LNCS 1217, Springer (1997)
5. Comet, J., Klaudel, H., Liauzu, S.: Modeling multi-valued genetic regulatory networks using high-level Petri nets. In: Proc. of International Conference on Application and Theory of Petri Nets. pp. 208–227 (2005)
6. David, R., Alla, H.: Discrete, Continuous, and Hybrid Petri Nets. Springer (2010)
7. Elowitz, M.B., Leibler, S.: A synthetic oscillatory network of transcriptional regulators. Nature 403(6767), 335–338 (2000)
8. Fujita, S., Matsui, M., Matsuno, H., Miyano, S.: Modeling and simulation of fission yeast cell cycle on hybrid functional Petri net. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences E87-A(11), 2919–2927 (2004)
9. Gaeta, R.: Efficient discrete-event simulation of colored Petri nets. IEEE Transactions on Software Engineering 22(9), 629–639 (1996)
10. Gao, Q., Gilbert, D., Heiner, M., Liu, F., Maccagnola, D., Tree, D.: Multiscale Modelling and Analysis of Planar Cell Polarity in the Drosophila Wing. IEEE/ACM Transactions on Computational Biology and Bioinformatics 10(2), 337–351 (2013)

11. Genrich, H., Küffner, R., Voss, K.: Executable Petri net models for the analysis of metabolic pathways. International Journal on Software Tools for Technology Transfer 3(4), 394–404 (2001)
12. Gillespie, D.: Markov processes: an introduction for physical scientists. Academic Press (1991)
13. Gillespie, D.: Stochastic simulation of chemical kinetics. Annual review of physical chemistry 58(1), 35–55 (2007)
14. Haseltine, E., Rawlings, J.: Approximate simulation of coupled fast and slow reactions for stochastic chemical kinetics. J. Chem. Phys. 117(15), 6959–6969 (2002)
15. Heiner, M., Herajy, M., Liu, F., Rohr, C., Schwarick, M.: Snoopy – a unifying Petri net tool. In: Haddad, S., Pomello, L. (eds.) Proc. PETRI NETS 2012. LNCS, vol. 7347, pp. 398–407. Springer (2012)
16. Heiner, M., Lehrack, S., Gilbert, D., Marwan, W.: Extended stochastic Petri nets for model-based design of wetlab experiments. In: Transactions on Computational Systems Biology XI, pp. 138–163. Springer, Berlin, Heidelberg (2009)
17. Herajy, M.: Computational Steering of Multi-Scale Biochemical Networks. Ph.D. thesis, BTU Cottbus, Dep. of CS (January 2013)
18. Herajy, M., Heiner, M.: Hybrid representation and simulation of stiff biochemical networks. J. Nonlinear Analysis: Hybrid Systems 6(4), 942–959 (November 2012)
19. Herajy, M., Heiner, M.: A Steering Server for Collaborative Simulation of Quantitative Petri Nets. In: Ciardo, G., Kindler, E. (eds.) Proc. PETRI NETS 2014. LNCS, vol. to appear. Springer (June 2014)
20. Herajy, M., Heiner, M.: Petri net-based collaborative simulation and steering of biochemical reaction networks. Fundamenta Informatica (129), 49–67 (2014)
21. Herajy, M., Schwarick, M., Heiner, M.: Hybrid Petri Nets for Modelling the Eukaryotic Cell Cycle. ToPNoC VIII, LNCS 8100 pp. 123–141 (2013)
22. Jensen, K.: Coloured Petri nets and the invariant-method. Theoretical Computer Science 14(3), 317–336 (1981)
23. Jensen, K., Kristensen, L.: Coloured Petri Nets. Springer (2009)
24. Kartson, D., Balbo, G., Donatelli, S., Franceschinis, G., Conte, G.: Modelling with generalized stochastic Petri nets. John Wiley & Sons, Inc., 1st edn. (1994)
25. Kiehl, T., Mattheyses, R., Simmons, M.: Hybrid simulation of cellular behavior. Bioinformatics 20, 316–322 (February 2004)
26. Lee, D., Zimmer, R., Lee, S., Park, S.: Colored Petri net modeling and simulation of signal transduction pathways. Metabolic Engineering 8(2), 112–122 (2006)
27. Liu, F.: Colored Petri Nets for Systems Biology. Ph.D. thesis, Department of Computer Science, Brandenburg University of Technology Cottbus (2012)
28. Liu, F., Blätke, M., Heiner, M., Yang, M.: Modelling and simulating reaction-diffusion systems using coloured petri nets. Computers in Biology and Medicine p. under review (2014)
29. Liu, F., Heiner, M.: Modeling membrane systems using colored stochastic Petri nets . Nat. Computing 12(4), 617 – 629 (2013)
30. Liu, F., Heiner, M.: Multiscale modelling of coupled Ca^{2+} channels using coloured stochastic Petri nets. IET Systems Biology 7(4), 106 – 113 (August 2013)
31. Liu, F., Heiner, M.: Petri Nets for Modeling and Analyzing Biochemical Reaction Networks, chap. 9, pp. 245–272. Springer (2014)
32. Liu, F., Heiner, M., Yang, M.: An efficient method for unfolding colored Petri nets. In: Proc. of the Winter Simulation Conference. IEEE (2012)
33. Marwan, W., Rohr, C., Heiner, M.: Petri nets in Snoopy: A unifying framework for the graphical display, computational modelling, and simulation of bacterial regulatory networks, Methods in Molec. Biol., vol. 804, chap. 21, pp. 409–437 (2012)

34. Matsui, M., Fujita, S., Suzuki, S., Matsuno, H., Miyano, S.: Simulated cell division processes of the xenopus cell cycle pathway by genomic object net. Journal of Integrative Bioinformatics p. 0001 (2004)
35. Matsuno, H., Nagasaki, M., Miyano, S.: Hybrid Petri net based modeling for biological pathway simulation. Natural Computing 10(3), 1099–1120 (2011)
36. Matsuno, H., Tanaka, Y., Aoshima, H., Doi, A., Matsui, M., Miyano, S.: Biopathways representation and simulation on hybrid functional Petri net. In silico biology 3(3) (2003)
37. Parvu, O., Gilbert, D., Heiner, M., Liu, F., Saunders, N.: Modelling and Analysis of Phase Variation in Bacterial Colony Growth. In: Gupta, A., Henzinger, T. (eds.) Proc. CMSB 2013. LNCS/LNBI, vol. 8130, pp. 78–91. Springer (September 2013)
38. Peleg, M., Gabashvili, I.S., Altman, R.B.: Qualitative models of molecular function: Linking genetic polymorphisms of trna to their functional sequelae. Proceedings of the IEEE 90(12), 1875–1886 (2002)
39. Reddy, V., Mavrovouniotis, M., Liebman, M.: Petri net representations in metabolic pathways. In: Proceedings of the 1st International Conference on Intelligent Systems for Molecular Biology. pp. 328–336 (1993)
40. Runge, T.: Application of coloured Petri nets in systems biology. In: Proc. of the 5th Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools. pp. 77–95. University of Aarhus (2004)
41. Täubner, C., Mathiak, B., Kupfer, A., Fleischer, N., Eckstein, S.: Modelling and simulation of the tlr4 pathway with coloured Petri nets. In: Proc. of the 28th Annual International Conference of the IEEE Engineering in Medicine and Biology Society. pp. 2009–2012. IEEE (2006)
42. Voss, K., Heiner, M., Koch, I.: Steady state analysis of metabolic pathways using Petri nets. In Silico Biology 3, 0031 (2003)