AN ADAPTIVE LEARNING APPROACH TO PARAMETER ESTIMATION FOR HYBRID PETRI NETS IN SYSTEMS BIOLOGY

Peter Vieting[1,2], Rodrigo C. de Lamare[1], Lukas Martin[3], Guido Dartmann[4] and Anke Schmeink[2]

[1]Center for Telecommunication Studies, Pontifical Catholic University of Rio de Janeiro, Brazil
[2]Research Area ISEK, RWTH Aachen University, Germany
[3]Department of Intensive Care and Intermediate Care, University Hospital RWTH Aachen, Germany
[4]Research Area Distributed Systems, Trier University of Applied Sciences, Environmental Campus, Trier, Germany
{peter.vieting, guido.dartmann, anke.schmeink}@rwth-aachen.de, delamare@cetuc.puc-rio.br, lmartin@ukaachen.de

ABSTRACT

In this work, we investigate adaptive learning techniques in hybrid Petri nets (HPNs) that can model biological systems. In particular, based on a state space formulation we develop a decision-aided adaptive gradient descent (DAAGD) algorithm capable of cost-effectively estimating the parameters used in an HPN model. Contrary to standard gradient descent techniques, the DAAGD algorithm does not require prior knowledge, i.e., information about the discrete transitions' firing instants. Simulations of a gene regulatory network assess the performance of the proposed DAAGD algorithm against standard gradient descent algorithms with full, imperfect and no prior knowledge.

***Index Terms*—** Bioinformatics and genomics, statistical learning, machine learning and pattern recognition, detection and estimation theory, adaptive signal processing

1. INTRODUCTION

Petri nets (PNs) play an important role in modeling methods for systems biology [1]. Among other benefits they have the advantage of providing a precise mathematical description of a system while still being illustrative. Besides various medical and biological [2], [3] as well as industrial [4] and smart home [5] applications, hybrid Petri net (HPN) models have been developed to represent gene regulatory networks [6] and biopathways [7].

Knowing about PNs' modeling capabilities, the need for parameter estimation techniques arises, in particular those which can identify a network's structure [8]. Previous work on parameter estimation has made use of different metaheuristics such as simulated annealing, genetic algorithms and evolutionary programming [8]. Attempts to improve the computational efficiency were made by decomposing a model into smaller parts and estimating their parameters separately [9]. Furthermore, [10] adapts a data assimilation approach using particle filtering to approximate the Bayes estimators.

Especially for medical applications, the proposition of learning HPNs is very promising due to its white box modeling concept. Classical machine learning concepts such as neural networks are black box models and, therefore, not very predictable and reliable in this field. Yet, there is a strong need for clinical decision support systems [11], [12], [13], [14], [15]. Our approach can target two possible use cases: (i) individual medicine with reliable models of patient-adapted pathophysiological processes and (ii) drug design with models of in vitro experiments.

Contribution: The system model considered in this work especially aims at investigating the possibilities of combining deterministic and stochastic PN models. The rationale is that knowledge about a disease process can be modeled using deterministic PNs. Simultaneously, there are many complex and unknown processes occurring in the human body that could be modeled stochastically. Thus, the interaction between these two parts becomes a research topic of potential for investigation. We present a system model which combines the deterministic and the stochastic part of a PN with an interconnecting PN (IPN). We propose a decision-aided adaptive gradient descent (DAAGD) algorithm for learning the IPN's arc weights from (noisy) data of measured markings based on the mean squared error (MSE) criterion. A case where the discrete transitions' firing instants are given and a second case with no such prior knowledge are distinguished.

This paper is structured as follows: In Section 2, we present the system model and the problem definition. On the basis of the state space formulation from Section 3, a standard gradient descent approach is established in Section 4. A learning scheme for more general cases is developed by introducing the DAAGD algorithm in Section 5. Simulations of an exemplary HPN model are carried out and presented in Section 6. Finally, Section 7 concludes the paper.

2. SYSTEM MODEL

The considered HPN model is depicted in Fig. 1. It consists of three separate parts, a stochastic Petri net (SPN), a continuous Petri net (CPN) and the IPN representing the interaction between both parts. We denote the total number of SPN places and transitions as $P^{\mathcal{S}}$ and $T^{\mathcal{S}}$, respectively, SPN places as $p_1^{\mathcal{S}}, \ldots, p_{P^{\mathcal{S}}}^{\mathcal{S}}$ and SPN transitions as $\tau_1^{\mathcal{S}}, \ldots, \tau_{T^{\mathcal{S}}}^{\mathcal{S}}$. For CPN and IPN (only transitions), an analogous notation holds using the superscripts $\mathcal{C}$ and $\mathcal{I}$ instead of $\mathcal{S}$. The two subnets (SPN and CPN) are ordinary as known from the literature, e.g., [16], whereas the IPN consists of transitions only. While these are fully connected to all places in the two subnets in general, essentially most of the connections have a weight equal to zero.

To simplify the considerations, we assume the IPN to be pure, i.e., no place in the HPN is both an input and output place for an IPN transition (no self-loops). Furthermore, we assume that only one discrete transition can fire per time instant. This is equivalent to having a sufficiently high sampling rate.

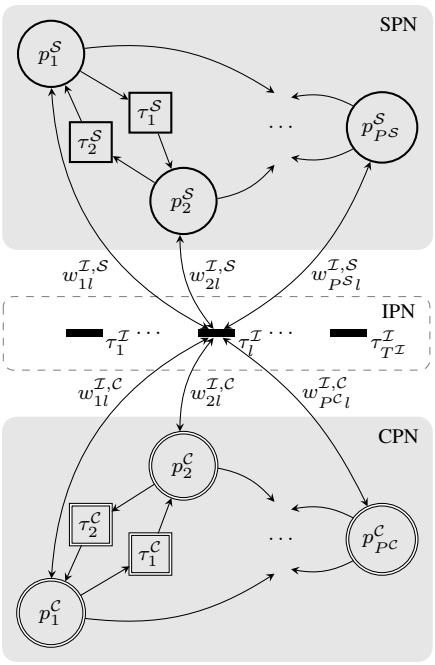


Fig. 1. System model of the HPN under consideration

2.1. Problem Definition

The task is to find the unknown IPN transition arc weights based on measured markings of all places over time for some realization of the HPN. We assume that both subnets are known, i.e., their topology and the transitions' kinetics and rates are given.

3. STATE SPACE FORMULATION OF AN HPN

In this section, we first show how to discretize an HPN and then formulate a state space representation of the discretized HPN.

3.1. Discretization of the Hybrid Petri Net

In a CPN, the marking m of a discrete place $p^{\mathcal{C}}$ can be described by the ordinary differential equation (ODE)

$$\begin{aligned}\dot{m}(p^{\mathcal{C}}) &= \sum_{\tau^{\mathcal{C}} \in \mathbb{T}^{\mathcal{C}}} \mathrm{Post}^{\mathcal{C}}(p^{\mathcal{C}}, \tau^{\mathcal{C}}) \cdot \nu(\tau^{\mathcal{C}}) - \sum_{\tau^{\mathcal{C}} \in \mathbb{T}^{\mathcal{C}}} \mathrm{Pre}^{\mathcal{C}}(p^{\mathcal{C}}, \tau^{\mathcal{C}}) \cdot \nu(\tau^{\mathcal{C}}) \\ &= \sum_{\tau^{\mathcal{C}} \in \mathbb{T}^{\mathcal{C}}} w^{\mathcal{C}}_{p\tau} \cdot \nu(\tau^{\mathcal{C}}), \end{aligned} \tag{1}$$

with the CPN's output and input applications $\mathrm{Post}^{\mathcal{C}}$ and $\mathrm{Pre}^{\mathcal{C}}$, the CPN transition arc weights $w^{\mathcal{C}}_{p\tau} = \mathrm{Post}^{\mathcal{C}}(p,\tau) - \mathrm{Pre}^{\mathcal{C}}(p,\tau) = \left[\mathbf{A}^{\mathcal{C}}\right]_{p\tau}$ as the entries of the incidence matrix $\mathbf{A}^{\mathcal{C}} \in \mathbb{R}^{P^{\mathcal{C}} \times T^{\mathcal{C}}}$, the firing rate function $\nu(\tau^{\mathcal{C}})$ and the set of all continuous transitions $\mathbb{T}^{\mathcal{C}}$ [16]. Writing (1) into vector notation yields $\dot{\mathbf{m}}^{\mathcal{C}} = \mathbf{A}^{\mathcal{C}} \cdot \boldsymbol{\nu}^{\mathcal{C}}$, where $\mathbf{m}^{\mathcal{C}} = \left[m(p^{\mathcal{C}}_1), \ldots, m(p^{\mathcal{C}}_{P^{\mathcal{C}}})\right]^{\mathrm{T}} \in \mathbb{R}^{P^{\mathcal{C}}}$ and $\boldsymbol{\nu}^{\mathcal{C}} = \left[\nu(\tau^{\mathcal{C}}_1), \ldots, \nu(\tau^{\mathcal{C}}_{T^{\mathcal{C}}})\right]^{\mathrm{T}} \in \mathbb{R}^{T^{\mathcal{C}}}$. To describe the HPN using a state equation, we can discretize the CPN by converting the ODE into discrete time steps. When doing so, we have to consider the discrete step size Δt in the instantaneous firing speeds $\mathbf{v}^{\mathcal{C}} = \Delta t \cdot \boldsymbol{\nu}^{\mathcal{C}}$. Thus, with the set of discrete time instants $\mathbb{K} = \{1, 2, \ldots, K\}$ we obtain $\mathbf{m}^{\mathcal{C}}_{k+1} = \mathbf{m}^{\mathcal{C}}_k + \mathbf{A}^{\mathcal{C}} \cdot \mathbf{v}^{\mathcal{C}}_k, \quad k \in \mathbb{K}$.

3.2. State Equation of the Hybrid Petri Net

For the stochastic transitions in the SPN and IPN, the next step is not deterministic. For a determinized inspection (considering an arbitrary but fixed network realization) we introduce the control vector $\mathbf{u}$ [17]. Each element represents one transition of the HPN, thus $\mathbf{u} = \left[u(\tau^{\mathcal{S}}_1), \ldots, u(\tau^{\mathcal{S}}_{T^{\mathcal{S}}}), u(\tau^{\mathcal{C}}_1), \ldots, u(\tau^{\mathcal{C}}_{T^{\mathcal{C}}}), u(\tau^{\mathcal{I}}_1), \ldots, u(\tau^{\mathcal{I}}_{T^{\mathcal{I}}})\right]^{\mathrm{T}}$. $u_k(\tau) \in \{0, 1\}$ specifies whether transition τ fires at time instant k. As this fully describes the change between two time instants, the instantaneous firing speeds of the stochastic transitions $\mathbf{v}^{\mathcal{S}}_k$ and $\mathbf{v}^{\mathcal{I}}_k$ can be set to $\mathbf{1}\ \forall k$, respectively. For completeness, we state that the definition of $\mathbf{v}^{\mathcal{C}}_k$ remains unchanged while $\mathbf{u}^{\mathcal{C}}_k = \mathbf{1}\ \forall k$ since the CPN's transitions fire continuously.

To extend the notation to the whole HPN, we define

$$\mathbf{m} = \begin{bmatrix} \mathbf{m}^{\mathcal{S}} \\ \mathbf{m}^{\mathcal{C}} \end{bmatrix}, \mathbf{A} = \left[\begin{array}{c|c|c} \mathbf{A}^{\mathcal{S}} & \mathbf{0} & \\ \hline \mathbf{0} & \mathbf{A}^{\mathcal{C}} & \mathbf{A}^{\mathcal{I}} \end{array}\right], \mathbf{v} = \begin{bmatrix} \mathbf{v}^{\mathcal{S}} \\ \mathbf{v}^{\mathcal{C}} \\ \mathbf{v}^{\mathcal{I}} \end{bmatrix}$$

and $\mathbf{u}$ analogously to $\mathbf{v}$ as well as the notation $\boldsymbol{\Lambda}^{\mathcal{X}}_{\mathbf{v}_k} = \mathrm{diag}(\mathbf{v}^{\mathcal{X}}_k)$. Using the control vector notation to extend $\mathbf{m}^{\mathcal{C}}_{k+1}$ accordingly yields

$$\begin{aligned}\mathbf{m}_{k+1} &= \mathbf{m}_k + \mathbf{A} \cdot \boldsymbol{\Lambda}_{\mathbf{v}_k} \cdot \mathbf{u}_k \\ &= \mathbf{m}_k + \begin{bmatrix} \mathbf{A}^{\mathcal{S}} \cdot \boldsymbol{\Lambda}^{\mathcal{S}}_{\mathbf{v}_k} \cdot \mathbf{u}^{\mathcal{S}}_k \\ \mathbf{A}^{\mathcal{C}} \cdot \boldsymbol{\Lambda}^{\mathcal{C}}_{\mathbf{v}_k} \cdot \mathbf{u}^{\mathcal{C}}_k \end{bmatrix} + \mathbf{A}^{\mathcal{I}} \cdot \boldsymbol{\Lambda}^{\mathcal{I}}_{\mathbf{v}_k} \cdot \mathbf{u}^{\mathcal{I}}_k \\ &= \mathbf{m}_k + \mathbf{A}^{\mathcal{S},\mathcal{C}} \cdot \boldsymbol{\Lambda}^{\mathcal{S},\mathcal{C}}_{\mathbf{v}_k} \cdot \mathbf{u}^{\mathcal{S},\mathcal{C}}_k + \mathbf{A}^{\mathcal{I}} \cdot \boldsymbol{\Lambda}^{\mathcal{I}}_{\mathbf{v}_k} \cdot \mathbf{u}^{\mathcal{I}}_k. \end{aligned} \tag{2}$$

In case that the time instants of the firings are unknown, we have to adopt a different point of view. Calculating the true value of the next marking is not possible as we deal with a random process. However, the time until the next firing of an enabled stochastic transition is a random variable with exponential distribution [16] and we can therefore evaluate the probability that a transition fires before the next time instant. Consequently, we could adapt the notation and incorporate these probabilities in the speed vectors $\mathbf{v}^{\mathcal{S}}_k$ and $\mathbf{v}^{\mathcal{I}}_k$ whereas the definition of $\mathbf{v}^{\mathcal{C}}_k$ remained unchanged. Meanwhile, $u_k(\tau) \in \{0, 1\}$ would specify whether the transition τ is enabled at time instant k for both stochastic and continuous transitions.

Using the stated changes in the variable definitions, the probabilistic state equation takes the same form as its determinized counterpart (2). It is crucial to mention that in this case, however, not only the incidence matrix $\mathbf{A}^{\mathcal{I}}$ but also the IPN's speed and control vectors depend on the weights.

4. ADAPTIVE LEARNING APPROACH

In this section, we propose an adaptive learning approach to find the IPN arc weights which are contained in $\mathbf{A}^{\mathcal{I}}$ [18]. We assume to have access to the markings' measurements influenced by the additive white Gaussian noise (AWGN) $\boldsymbol{\eta}_k \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$, $\boldsymbol{\eta}_k \in \mathbb{R}^P$, $k \in \mathbb{K}$ according to $\tilde{\mathbf{m}}_k = \mathbf{m}_k + \boldsymbol{\eta}_k$.

4.1. Formulation of an Objective Function

When estimating the weights, we aim to minimize the difference between the measured markings $\tilde{\mathbf{m}}$ and the model prediction $\hat{\mathbf{m}}$. In the literature, a common way of achieving this is via gradient-based and

least-squares approaches [8]. As the objective function, we therefore consider the MSE which is defined as

$$\mathrm{MSE} = \frac{1}{K}\sum_{k=1}^{K}\widehat{\mathrm{MSE}}_k, \tag{3}$$

where we employ the instantaneous MSE estimate that equals the MSE observed at time instant k

$$\widehat{\mathrm{MSE}}_k = \frac{1}{P}\cdot(\hat{\mathbf{m}}_k-\tilde{\mathbf{m}}_k)^{\mathrm{T}}\cdot(\hat{\mathbf{m}}_k-\tilde{\mathbf{m}}_k), \tag{4}$$

with the model prediction

$$\hat{\mathbf{m}}_k = \tilde{\mathbf{m}}_{k-1} + \mathbf{A}^{\mathcal{S},\mathcal{C}}\cdot\mathbf{\Lambda}^{\mathcal{S},\mathcal{C}}_{\mathbf{v}_{k-1}}\cdot\mathbf{u}^{\mathcal{S},\mathcal{C}}_{k-1} + \hat{\mathbf{A}}^{\mathcal{I}}_{k-1}\cdot\mathbf{\Lambda}^{\mathcal{I}}_{\mathbf{v}_{k-1}}\cdot\mathbf{u}^{\mathcal{I}}_{k-1} \tag{5}$$

similar to the expression in (2) and $\hat{\mathbf{A}}^{\mathcal{I}}_{k-1}$ as the estimate of $\mathbf{A}^{\mathcal{I}}$ at time instant $k-1$.

4.2. Determinized Hybrid Petri Net Approach

To apply an adaptive learning approach, we calculate the instantaneous gradient of the objective function

$$\nabla_{\hat{\mathbf{A}}^{\mathcal{I}}_{k-1}}\widehat{\mathrm{MSE}}_k = \frac{2}{P}\cdot(\hat{\mathbf{m}}_k-\tilde{\mathbf{m}}_k)\cdot\left(\mathbf{u}^{\mathcal{I}}_{k-1}\right)^{\mathrm{T}}\cdot\mathbf{\Lambda}^{\mathcal{I}}_{\mathbf{v}_{k-1}}. \tag{6}$$

We can therefore update the weights' estimates according to

$$\begin{aligned}\hat{\mathbf{A}}^{\mathcal{I}}_{k} &= \hat{\mathbf{A}}^{\mathcal{I}}_{k-1} - \mu'\cdot\nabla_{\hat{\mathbf{A}}^{\mathcal{I}}_{k-1}}\widehat{\mathrm{MSE}}_k\\ &= \hat{\mathbf{A}}^{\mathcal{I}}_{k-1} - \mu\cdot(\hat{\mathbf{m}}_k-\tilde{\mathbf{m}}_k)\cdot\left(\mathbf{u}^{\mathcal{I}}_{k-1}\right)^{\mathrm{T}}\cdot\mathbf{\Lambda}^{\mathcal{I}}_{\mathbf{v}_{k-1}},\end{aligned} \tag{7}$$

where the convergence factor of the adaptive algorithm is $\mu = \mu'\cdot\frac{2}{P}$.

The intuitive interpretation of the result is as follows. If there is no IPN firing, the weights do not affect this time instant and are thus not updated. This is incorporated by the multiplication with the control vector $\mathbf{u}^{\mathcal{I}}_k$. However, if there is an IPN firing, the prediction for the next step is distorted exactly by the weights' error. Thus, we update the weights' estimates by the difference between the prediction and the observed marking, i.e., $(\hat{\mathbf{m}}_k-\tilde{\mathbf{m}}_k)$.

4.3. Probabilistic Hybrid Petri Net Approach

The drawback of the approach presented above is the requirement to know the control vector at all instants of time. We could think about deriving a similar approach using the probabilistic state equation which does not require this knowledge. Since the IPN's speed and control vectors depend on the weights in that case, the gradient takes a different form than in (6). Additionally, calculating these vectors involves the unit step function θ which is non-smooth. A smooth approximation of θ can be deployed but as we will see in Section 6, the approach using approximations is not successful. That is why the need for a new learning technique arises.

5. DECISION-AIDED ADAPTIVE GRADIENT DESCENT

One of the major insights of Section 4 is the fact that learning only occurs when an IPN transition fires because their effect can only then be observed which is why these instants are crucial to any learning approach. This inspires a new algorithm that is capable of learning the weights without prior knowledge, i.e., given knowledge of the IPN firing times. The key idea is to estimate the control vector, i.e., to detect the firing of discrete transitions from the measured data samples at every time instant instead of taking it as given. Subsequently, the gradient descent approach is applied.

5.1. Maximum Likelihood Estimation of Firings

At every time instant k we find an estimate $\hat{\mathbf{u}}^{\mathcal{S},\mathcal{I}}_{k-1}\in\mathbb{U}=\{0,1\}^{T^{\mathcal{D}}}$ of the control vector with the number of discrete transitions $T^{\mathcal{D}} = T^{\mathcal{S}}+T^{\mathcal{I}}$ given the observed markings $\tilde{\mathbf{m}}_k$ and $\tilde{\mathbf{m}}_{k-1}$ using the maximum likelihood estimator (MLE)

$$\begin{aligned}\hat{\mathbf{u}}^{\mathcal{S},\mathcal{I}}_{k-1} &= \underset{\mathbf{u}^{\mathcal{S},\mathcal{I}}_{k-1}\in\mathbb{U}}{\arg\max}\ \Pr\left(\tilde{\mathbf{m}}_k|\tilde{\mathbf{m}}_{k-1},\mathbf{u}^{\mathcal{S},\mathcal{I}}_{k-1}\right)\\ &= \underset{\mathbf{u}^{\mathcal{S},\mathcal{I}}_{k-1}\in\mathbb{U}}{\arg\max}\ \Pr\left(\boldsymbol{\eta}_k-\boldsymbol{\eta}_{k-1}=\Delta\tilde{\mathbf{m}}_{k-1}-\hat{\mathbf{A}}^{\mathcal{S},\mathcal{I}}_{k-1}\cdot\mathbf{u}^{\mathcal{S},\mathcal{I}}_{k-1}\right)\\ &= \underset{\mathbf{u}^{\mathcal{S},\mathcal{I}}_{k-1}\in\mathbb{U}}{\arg\max}\ -\left(\Delta\tilde{\mathbf{m}}_{k-1}-\hat{\mathbf{A}}^{\mathcal{S},\mathcal{I}}_{k-1}\cdot\mathbf{u}^{\mathcal{S},\mathcal{I}}_{k-1}\right)^{\mathrm{T}}\cdot\mathbf{\Sigma}^{-1}\cdot\\ &\qquad\left(\Delta\tilde{\mathbf{m}}_{k-1}-\hat{\mathbf{A}}^{\mathcal{S},\mathcal{I}}_{k-1}\cdot\mathbf{u}^{\mathcal{S},\mathcal{I}}_{k-1}\right),\end{aligned} \tag{8}$$

where we use the notations $\Delta\tilde{\mathbf{m}}_{k-1} = \tilde{\mathbf{m}}_k-\tilde{\mathbf{m}}_{k-1}-\Delta\tilde{\mathbf{m}}^{\mathcal{C}}_{k-1}$, $\Delta\tilde{\mathbf{m}}^{\mathcal{C}}_{k-1} = \begin{bmatrix}\mathbf{0}\\ \mathbf{A}^{\mathcal{C}}\end{bmatrix}\cdot\mathbf{\Lambda}^{\mathcal{C}}_{\mathbf{v}_{k-1}}\cdot\mathbf{u}^{\mathcal{C}}_{k-1}$ and $\hat{\mathbf{A}}^{\mathcal{S},\mathcal{I}}_{k-1} = \left[\begin{array}{c:c}\mathbf{A}^{\mathcal{S}} & \\ \hdashline \mathbf{0} & \hat{\mathbf{A}}^{\mathcal{I}}_{k-1}\end{array}\right]$ as well as $\hat{\mathbf{u}}^{\mathcal{S},\mathcal{I}}_{k-1} = \left[(\hat{\mathbf{u}}^{\mathcal{S}}_{k-1})^{\mathrm{T}}\ (\hat{\mathbf{u}}^{\mathcal{I}}_{k-1})^{\mathrm{T}}\right]^{\mathrm{T}}$. The well-known distribution of the AWGN is used for the transformations in (8).

5.2. Event-Triggered Firing Detection

The MLE in (8) essentially depends on a good estimate of the incidence matrix. While $\mathbf{A}^{\mathcal{S}}$ is given, the structure of the IPN is initially unknown and only discovered during the process. Hence, the MLE allows a learning approach that works generally but requires assistance at specific time instants, i.e., the instants where a previously undiscovered IPN transition fires.

To resolve this, an approach resembling event-triggered control can be deployed. With event-triggered control, actuator values are updated only when a triggering condition is satisfied instead of periodically [19]. Here, a new IPN transition will be initialized at each triggering time. These time instants are determined by monitoring a triggering condition that is fulfilled when the prediction's MSE exceeds some threshold $\mathrm{MSE}_{\mathrm{th}}$, i.e.,

$$(\hat{\mathbf{m}}_k-\tilde{\mathbf{m}}_k)^{\mathrm{T}}\cdot(\hat{\mathbf{m}}_k-\tilde{\mathbf{m}}_k) > P\cdot\mathrm{MSE}_{\mathrm{th}}, \tag{9}$$

with the prediction $\hat{\mathbf{m}}_k = \tilde{\mathbf{m}}_{k-1}+\hat{\mathbf{A}}_{k-1}\cdot\mathbf{\Lambda}_{\mathbf{v}_{k-1}}\cdot\hat{\mathbf{u}}_{k-1}$, where $\mathbf{v}^{\mathcal{S}}_k$ and $\mathbf{v}^{\mathcal{I}}_k$ are set to $\mathbf{1}$, $\hat{\mathbf{u}}^{\mathcal{S}}_k$ and $\hat{\mathbf{u}}^{\mathcal{I}}_k$ are computed together according to (8) while $\mathbf{v}^{\mathcal{C}}_k$ is the vector of instantaneous firing speeds and $\mathbf{u}^{\mathcal{C}}_k = \mathbf{1}$ as usual. The rationale is that if an undiscovered IPN transition fires, the MLE cannot detect it and the prediction error is expected to be particularly large.

5.3. Proposed DAAGD Algorithm

The previously introduced tools are incorporated in the following algorithm. We initialize the number of IPN transitions $T^{\mathcal{I}}$ to be zero and start iterating through the time instants $k\in\mathbb{K}$ calculating the MLE of the control vector for each k. If the triggering condition (9) is met, $T^{\mathcal{I}}$ is increased by one and a new IPN transition is initialized. This is done by a learning step according to the standard gradient descent approach from (7) with $\mu = 1$, the reason being that this is the best possible estimate maximizing the MLE's chances of recognizing the transition upon its next firing. In case of no triggering, a normal learning step is performed. A pseudo-code of the DAAGD algorithm can be found in Algorithm 1.

Algorithm 1 Decision-Aided Adaptive Gradient Descent

$T^{\mathcal{I}} \leftarrow 0$
for all $k \in \mathbb{K}$ **do**
  Calculate maximum likelihood estimate $\hat{\mathbf{u}}_{k-1}$ using (8)
  Estimate next marking $\hat{\mathbf{m}}_k = \tilde{\mathbf{m}}_{k-1} + \hat{\mathbf{A}}_{k-1} \cdot \mathbf{\Lambda}_{\mathbf{v}_{k-1}} \cdot \hat{\mathbf{u}}_{k-1}$
  if Error>Threshold according to (9) **then**
    $T^{\mathcal{I}} \leftarrow T^{\mathcal{I}} + 1$
    Initialize new IPN transition using (7) with $\mu = 1$
  else
    Perform learning step using (7)
  end if
end for

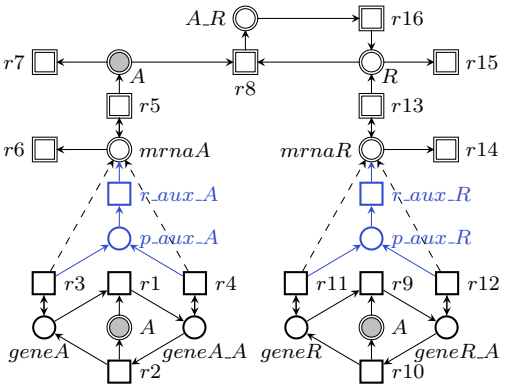


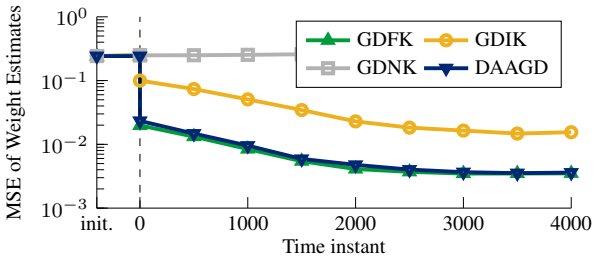
Fig. 2. A (simplified) HPN model of a gene regulatory network of a circadian clock mechanism from [1]. Originally with the dashed arrows, the modification replaces these by the parts marked in blue.

6. SIMULATION RESULTS

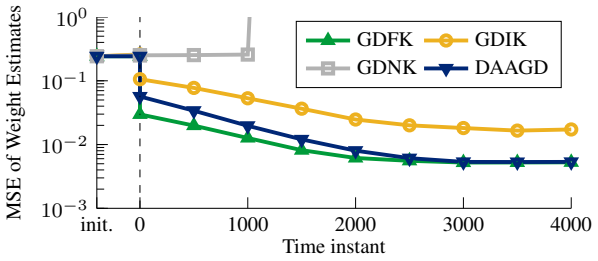
To validate the DAAGD algorithm and to put it into context, we adopt a (simplified) model of a gene regulatory network of a circadian clock mechanism from [1] depicted in Fig. 2. It can be separated into a continuous and a stochastic part interconnected by stochastic transitions. Thus, the considered system model shown in Fig. 1 is capable of representing this network in general. However, it makes use of read arcs contradicting the restriction of a pure IPN from Section 2. Based on the property that all impure PNs can be converted into pure PNs [16], we propose a modification of the network that leaves the model behavior unchanged while moving the read arc connected transitions from the IPN into the respective subnets where the restriction does not apply (see Fig. 2).

To verify and compare the different approaches, we carried out numerical simulations of 100 realizations of the exemplary HPN model from Fig. 2 and applied the standard gradient descent approach with full knowledge (GDFK), imperfect knowledge (GDIK) and no knowledge (GDNK) of the firing times as well as the DAAGD algorithm to estimate the arc weights. Note that the GDNK uses the probabilistic state equation. Fig. 3 shows the resulting MSE of the learning schemes. We chose a convergence factor of $\mu = 0.3$ and a bit error rate of $u_{\mathrm{BER}} = 10^{-3}$ for the imperfect prior knowledge. The respective noise powers and threshold values are given in the captions. To achieve the depicted figures, at each time instant the resulting curves were averaged over all transitions according to

$$\mathrm{MSE}_k = \frac{1}{P \cdot T^{\mathcal{I}}} \cdot \mathrm{tr}\left(\mathbf{R}_k^{\mathrm{T}} \cdot \mathbf{R}_k\right), \tag{10}$$



(a) $\sigma_{\boldsymbol{\eta}}^2 = 0.01, \mathrm{MSE}_{\mathrm{th}} = 0.15$



(b) $\sigma_{\boldsymbol{\eta}}^2 = 0.015, \mathrm{MSE}_{\mathrm{th}} = 0.2$

Fig. 3. MSE for the learning schemes under different parameters.

with $\mathbf{R}_k = \hat{\mathbf{A}}_k^{\mathcal{I}} - \mathbf{A}_k^{\mathcal{I}}$ and subsequently the average over all realizations was computed. Special care has to be taken when the DAAGD algorithm's number of detected IPN transitions $T_{\mathrm{det}}^{\mathcal{I}}$ is incorrect, i.e., $T_{\mathrm{det}}^{\mathcal{I}} \neq T^{\mathcal{I}}$. In this case, we compare each detected transition τ_{det} to the transition that truly fired in the initialization time instant of τ_{det}. If there was no firing, we compare it to $\mathbf{0}$. It is worth noting that the starting point of the curves in Fig. 3 is the MSE at initialization. Before averaging, the curves are synchronized for all transitions such that $k = 0$ always corresponds to the time instant of the respective transition's first firing, which allows comparison of different realizations. Furthermore, in order to make it a fair comparison, the standard gradient descent approaches are modified to use $\mu = 1$ in the first learning step of each transition just like the DAAGD algorithm.

We can observe that the GDNK fails to provide any information about the weights and even increases the error while the GDFK yields errors in the order of magnitude of 10^{-2} to 10^{-3}. When only imperfect knowledge is present, we can see similar learning behavior at a higher error level. It is clear that the DAAGD algorithm yields the same estimates as the GDFK if the MLE identifies all control vectors correctly. This is confirmed in Fig. 3a, where their performance is almost the same. For an increased noise level, $\mathrm{MSE}_{\mathrm{th}}$ has to be adapted to ensure that $T_{\mathrm{det}}^{\mathcal{I}}$ will be correct. Fig. 3b shows that in this case the learning rate is slower because the IPN transitions are not necessarily detected immediately at the first firing, however, the MSE still converges to the same level as for the GDFK.

7. CONCLUSION

We have derived an adaptive gradient descent approach for learning parameters of an HPN model based on a state space description of the HPN. Aided by a decision mechanism, the DAAGD algorithm allows learning of parameters in HPNs where no knowledge about the firing times is available. Simulations showed that the approaches are capable of identifying parameters in a gene regulatory network.

8. REFERENCES

[1] M.A. Blätke, M. Heiner, and W. Marwan, "Biomodel engineering with Petri nets," in *Algebraic and Discrete Mathematical Methods for Modern Biology*, pp. 141–192. Elsevier, 2015.

[2] M. Herajy and M. Heiner, "Hybrid representation and simulation of stiff biochemical networks," *Nonlinear Analysis: Hybrid Systems*, vol. 6, no. 4, pp. 942–959, 2012.

[3] M. Herajy, M. Schwarick, and M. Heiner, "Hybrid Petri nets for modelling the eukaryotic cell cycle," in *Transactions on Petri Nets and Other Models of Concurrency VIII*, pp. 123–141. Springer, 2013.

[4] A. Giua, M.T. Pilloni, and C. Seatzu, "Modelling and simulation of a bottling plant using hybrid Petri nets," *International journal of production research*, vol. 43, no. 7, pp. 1375–1395, 2005.

[5] M.R. Jongerden, J. Hüls, B.R. Haverkort, and A. Remke, "Assessing the cost of energy independence," in *IEEE International Energy Conference (ENERGYCON)*. IEEE, 2016, pp. 1–6.

[6] H. Matsuno, A. Doi, M. Nagasaki, and S. Miyano, "Hybrid Petri net representation of gene regulatory network," in *Biocomputing 2000*, pp. 341–352. World Scientific, 1999.

[7] H. Matsuno, Y. Tanaka, H. Aoshima, M. Matsui, S. Miyano, et al., "Biopathways representation and simulation on hybrid functional Petri net," *In silico biology*, vol. 3, no. 3, pp. 389–404, 2003.

[8] J. Sun, J.M. Garibaldi, and C. Hodgman, "Parameter estimation using metaheuristics in systems biology: a comprehensive review," *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, vol. 9, no. 1, pp. 185–202, 2012.

[9] G. Koh, H.F.C. Teong, M.-V. Clément, D. Hsu, and P.S. Thiagarajan, "A decompositional approach to parameter estimation in pathway modeling: a case study of the Akt and MAPK pathways and their crosstalk," *Bioinformatics*, vol. 22, no. 14, pp. e271–e280, 2006.

[10] M. Nagasaki, R. Yamaguchi, R. Yoshida, S. Imoto, A. Doi, Y. Tamada, H. Matsuno, S. Miyano, and T. Higuchi, "Genomic data assimilation for estimating hybrid functional Petri net from time-course gene expression data," *Genome Informatics*, vol. 17, no. 1, pp. 46–61, 2006.

[11] A.E.W. Johnson, M.M. Ghassemi, S. Nemati, K.E. Niehaus, D.A. Clifton, and G.D. Clifford, "Machine learning and decision support in critical care," *Proceedings of the IEEE*, vol. 104, no. 2, pp. 444–466, 2016.

[12] C.R.M. Leite, D.L. Martin, G.R.M.A. Sizilio, K.E.A. dos Santos, B.G. de Araújo, R.A. de M. Valentim, A.D.D. Neto, J.D. de Melo, and A.M.G. Guerreiro, "Modeling of medical care with stochastic Petri nets," in *32nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2010, pp. 1336–1339.

[13] T.M. Hemmerling, F. Cirillo, and S. Cyr, "Decision support systems in medicine - anesthesia, critical care and intensive care medicine," in *Decision Support Systems*. InTech, 2012.

[14] Y. Dong, N.W. Chbat, A. Gupta, M. Hadzikadic, and O. Gajic, "Systems modeling and simulation applications for critical care medicine," *Annals of intensive care*, vol. 2, no. 1, pp. 18, 2012.

[15] S.M. Pappada, T.J. Papadimos, et al., "Clinical decision support systems: From medical simulation to clinical practice," *International Journal of Academic Medicine*, vol. 3, no. 1, pp. 78–8, 2017.

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

[17] T. Murata, "State equation, controllability, and maximal matchings of Petri nets," *IEEE Transactions on Automatic Control*, vol. 22, no. 3, pp. 412–416, 1977.

[18] P.S.R. Diniz, *Adaptive Filtering: Algorithms and Practical Implementation*, Kluwer Academic Publishers, Norwell, MA, USA, 2nd edition, 2002.

[19] W.P.M.H. Heemels, K.H. Johansson, and P. Tabuada, "An introduction to event-triggered and self-triggered control," in *IEEE 51st Annual Conference on Decision and Control (CDC)*, 2012, pp. 3270–3285.