Gazi University Journal of Science
GU J Sci
25(4):901-908 (2012)

ORIGINAL ARTICLE

An Application for the Control of Stochastic Petri nets via Fluidification Approach

Hanife APAYDIN ÖZKAN[1,♠]

[1]*Anadolu University, Department of Electrical and Electronical Engineering, 26555, Eskişehir, TURKEY*

Received: 29.03.2012 Accepted:18.09.2012

Abstract

Petri nets are frequently used for modeling and analysis of discrete event systems. Similar to other modeling formalisms for discrete systems, it suffers from state explosion. Fluidification can be used to overcome this difficulty yielding fluid approximation of original Petri nets in the sense of behaviours and properties. This models are called continuous Petri nets. In this work, stochastic Petri nets and their fluid approximation timed continuous Petri nets is considered. One of the main advantages of timed continuous Petri nets is to be able to design a controller by using more analytical techniques. But it is important to come back to a reasonable design or control in the original discrete setting. In this work, a target state control strategy of timed continuous Petri nets will be interpreted for the control of underlying Stochastic Petri nets. The efficiency of this interpretation will be studied on a table factory system.

Key Words: *Timed continuous Petri nets, Stochastic Petri nets, control*

1. INTRODUCTION

Petri Nets (PNs) are powerful graphical and mathematical tools for modelling, analysis and synthesis of discrete event systems [1,2]. In the real world, almost every event is time related. The necessity for including timing variables in the PN models of dynamic systems is apparent since these systems are real time in nature. A realistic way to introduce time in PNs is to assume that all transitions are timed according to some given probability density function, namely Stochastic Petri Nets (SPNs) which have proven to be a popular and useful tool for modelling and performance analysis of complex stochastic systems. Unfortunatelly, in the case of large scale systems, SPNs suffer from state explosion problem similarly to most formalisms for DESs.

Fluidification is one of the most useful relaxation techniques to overcome this state explosion problem and to reduce the computational complexity of the analysis and synthesis of PNs. For PNs, fluidification was introduced in [3,4] aiming to give fluid (continuous) approximation of the original PN in the sense of behaviours and properties, and these models are called timed continuous Petri nets (contPN).

One of the most important advantages of contPN is to be able to use more analytical techniques for the analysis of some interesting properties, like controllability and the synthesis of controllers, such as optimal steady-state [5] or dynamic controllers for reaching a desired marking [6-9].

Many works have been proposed for control of continuous Petri nets in the literature [5-9]. Steady state optimal control of continuous Petri nets was studied in [5], while the transitory control problem is also solved by means of implicit and explicit Model Predictive Control (MPC) strategy in [6]. The step tracking problem, i.e. design of control laws to drive the system states to target references, was considered and a Lyapunov-function-based dynamic control algorithm was proposed for the problem [7]. In

Corresponding author, e-mali: hapaydin1@anadolu.edu.tr

[8], an efficient heuristics for minimum time control of contPNs, which aims at driving the system from an initial state to a target one through a piecewise linear trajectory is developed.

Assuming that good off-line designs or dynamic controls are obtained for the continuous relaxation, it is important to come back to a reasonable design or control in the original discrete setting.

In this work, controlling SPN by applying a control law designed for the corresponding contPN approximation will be considered. The aim of this paper in particular is to analyze and illustrate the efficient usability of the target marking control strategy for contPNs developed in the author's previous work [8], for the control of mean value of underlying SPNs. For this purpose, the scheme provided in [9] will be used. This scheme was developed for the interpretation of a control law designed for a contPN system into the underlying SPN one.

The remainder of the paper is organized as follows. Section 2 briefly introduces the required concepts of PNs, contPNs and SPNs while Section 3 introduces the formulation of applied control. In Section 4, implementation of the control to SPN via contPN will be examined. A table factory system is taken as a case study to illustrate the efficiency of implementation in Section 5. Finally, Section 6 summarizes the main conclusions of the work.

2. PETRİ NETS

This section introduces the main concepts related to Petri nets, stochastic Petri nets and timed continuous Petri nets.

2.1. Petri nets

Definition: A discrete Petri net (PN) system is a pair $\langle N, \boldsymbol{m}_0 \rangle$ where $N = \langle P, T, \boldsymbol{Pre}, \boldsymbol{Post} \rangle$ is a net structure where $P = \{p_1, p_2, ..., p_{|P|}\}$ and $T = \{t_1, t_2, ..., t_{|T|}\}$ are the sets of places and transitions, respectively; $\boldsymbol{Pre}$, $\boldsymbol{Post} \in \mathbb{N}^{|P|\times|T|}$ are pre and post matrices connecting places and transitions; $\boldsymbol{m}_0 \in \mathbb{N}^{|P|}$ is initial marking (state).

In the graphical representation of a PN, places are represented by circles and transitions are represented by bars. Places and transitions are connected by directed arcs. $Pre(p_i, t_j)$ or Pre_{ij} denotes the weight of arc directed from place $p_i \in P$ to transition $t_j \in T$ and $Post(p_i, t_j)$ or $Post_{ij}$ denotes the weight of arc directed from transition $t_j \in T$ to place $p_i \in P$. For a node (place or transition) $v \in P \cup T$ the sets of its input and output nodes are denoted by ${}^{\bullet}v$ and $v^{\bullet}$, respectively.

$\boldsymbol{m} \in \mathbb{N}^{|P|}$ is a marking vector, $m(p_i)$ or m_i indicates the number of tokens in place $p_i \in P$. A transition $t_j \in T$ is enabled at $\boldsymbol{m}$ iff $m_i \geq Pre_{ij}$, $\forall p_i \in {}^{\bullet}t_j$ and its enabling degree $enab(t_j, \boldsymbol{m})$ is the value $\min_{p_i \in {}^{\bullet}t_j} \{m_i / Pre_{ij}\}$ rounded to nearest lower integer. An enabled transition t_j can fire at marking $\boldsymbol{m}$ in a certain amount $\beta \in \mathbb{N}_{>0}$ such that $\beta \leq enab(t_j, \boldsymbol{m})$.

If $\boldsymbol{m}$ is reachable from $\boldsymbol{m}_0$ through a finite sequence $\sigma = ``t_{i1}, t_{i2}, t_{i3} ... t_{ik}"$, the state equation is satisfied:

$$\boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma} \qquad (1)$$

where $\boldsymbol{C} = \boldsymbol{Post} - \boldsymbol{Pre}$ is the token flow matrix or incidence matrix, $\boldsymbol{\sigma} \in \mathbb{N}_{\geq 0}^{|T|}$ is the firing count vector, i.e., σ_j is the cumulative amount of firings of t_j in the sequence σ.

2.1. Stochastic Petri nets

PNs were originally proposed without any notion of time or probability. But for many practical applications, the addition of time is a necessity.

Since the transitions represent activities and activities take time in most timed PN models, time is associated with transitions commonly. If the delays are probabilistically specified which is more appropriate for real applications, such PN model is called as stochastic net.

For timed PNs, to make analysis tractable typically only a restricted set of probability distributions is allowed. A way to introduce time in discrete PNs is to assume that all transitions are timed with an exponential probability density function (pdf) which is a one parameter continuous distribution such that:

$$g(x) = \lambda \cdot e^{-\lambda \cdot x}, \quad x \geq 0 \qquad (2)$$

Definition: A stochastic Petri net (SPN) system is a tuple $\langle N, \Delta, \boldsymbol{m}_0 \rangle$ where $N = \langle P, T, \boldsymbol{Pre}, \boldsymbol{Post} \rangle$ is the net structure where P and T are the sets of places and transitions respectively; $\boldsymbol{Pre}$ and $\boldsymbol{Post}$ are the pre and post matrices; $\Delta = [d_1 \quad d_2 ... \quad d_{|T|}] \in \mathbb{N}_{>0}^{|T|}$ is delay vector where d_j is delay of transition t_j, $\boldsymbol{m}_0$ is the initial marking (state).

In this work, we assume that all transitions are timed with exponential pdf. In SPNs with exponentially distributed random variable, if a transition models infinite servers, time to fire the transition t_j at a marking $\boldsymbol{m}$ is exponentially distributed random variable with parameter $\lambda_j \cdot enab(t_j, \boldsymbol{m})$ whose mean value is $1/(\lambda_j \cdot enab(t_j, \boldsymbol{m}))$. If a transition models finite servers, time to fire the transition t_j at a marking $\boldsymbol{m}$ is exponentially distributed random variable with parameter λ_j whose mean value is $1/\lambda_j$.

SPNs suffer from state explosion problem, like in all formalisms for DESs. One possible way to overcome this difficulty is fluidification (or continuization) as a classical relaxation technique. Fluidification of PNs allows the use of linear programming techniques (that can be solved in polynomial time) instead of integer programming techniques for the analysis and synthesis of the systems,

moreover the resulting contPN systems may be studied by means of several analytical techniques.

2.2. Timed Continuous Petri nets

Definition 2: A continuous Petri net system is a pair $\langle N, \boldsymbol{m}_0 \rangle$ where $N = \langle P, T, \boldsymbol{Pre}, \boldsymbol{Post} \rangle$ is a net structure defined in Definition 1; $\boldsymbol{m}_0 \in \mathbb{R}^{|P|}$ is initial marking (state).

In continuous Petri nets, markings, $\boldsymbol{m}$, are not restricted to be integer, that is $\boldsymbol{m} \in \mathbb{R}^{|P|}_{\geq 0}$. A transition $t_j \in T$ is enabled at $\boldsymbol{m}$ iff $\forall p_i \in {}^{\bullet}t_j, m_i > 0$. That is, the enabling condition of continuous systems is the same as the enabling condition of discrete systems. As differ from the discrete case, the enabling degree is not limited to a natural number:

$$enab(t_j, \boldsymbol{m}) = \min_{p_i \in {}^{\bullet}t_j} \left\{ \frac{m_i}{Pre_{ij}} \right\} \qquad (3)$$

An enabled transition t_j can fire in any real amount α, k with $0 < \boldsymbol{\alpha} \leq enab(t_j, \boldsymbol{m})$ leading to a new state

$$\boldsymbol{m}' = \boldsymbol{m} + \boldsymbol{\alpha} \cdot C(\cdot, t_j) \qquad (4)$$

that the delays associated to the firing of transitions can be approximated by their mean values. This leads continuous and deterministic ``approximated'' model [11].

Definition 2.2 A timed continuous Petri net (contPN) system $\langle N, \boldsymbol{\Lambda}, \boldsymbol{m}_0 \rangle$ is a continuous Petri net system together with a vector $\boldsymbol{\Lambda} = [\lambda_1 \quad \lambda_2 \quad \dots \quad \lambda_{|T|}] \in \mathbb{R}^{|T|}_{>0}$ where λ_j is the firing rate of t_j.

As in untimed continuous Petri nets, state equation summarizes the way the marking evolves along time. The state equation of a contPN has an explicit dependence on time $\boldsymbol{m}[\tau] = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}[\tau]$ where τ is global time. But, in continuous systems, the marking is continuously changing, so we may consider the derivative of $\boldsymbol{m}$ with respect to time. By this way $\dot{\boldsymbol{m}}[\tau] = \boldsymbol{C} \cdot \dot{\boldsymbol{\sigma}}[\tau]$ is obtained. Here, $\dot{\boldsymbol{\sigma}}$ is flow through transitions and it is denoted by $\boldsymbol{f}[\tau] = \dot{\boldsymbol{\sigma}}[\tau]$. Hence, the state equation is

$$\dot{\boldsymbol{m}}[\tau] = \boldsymbol{C} \cdot \boldsymbol{f}[\tau] \qquad (5)$$

For the sake of simplicity τ is omitted in the rest of the paper.

Different semantics have been defined for continuous timed transitions [12,13]. It has been proven that the continuous model under infinite server semantics provides a better approximation of the original discrete model under some general conditions. Hence, this paper is focused on infinite server semantics. Since, we use a first order (or deterministic) approximation of SPNs, the firing of transition takes $1/\lambda_j \cdot enab(t_j, \boldsymbol{m})$ time units. The flow of transition t_j, $f(t_j)$ or f_j is defined as:

$$f_j = \lambda_j \cdot enab(t_j, \boldsymbol{m}) = \lambda_j \cdot \min_{p_i \in {}^{\bullet}t_j} \left\{ \frac{m_i}{Pre_{ij}} \right\} \qquad (6)$$

Under this semantics the flow vector is given as,

$$\boldsymbol{f} = \boldsymbol{\Lambda} \cdot \boldsymbol{\Pi}[\boldsymbol{m}] \cdot \boldsymbol{m} \qquad (7)$$

where $\boldsymbol{\Lambda} = \mathrm{diag}\{\lambda_1, \dots, \lambda_{|T|}\}$ is the firing rate matrix and $\boldsymbol{\Pi}[\boldsymbol{m}]$ is the constraint matrix at marking $\boldsymbol{m}$ defined by elements:

$$\Pi[\boldsymbol{m}]_{ji} = \left\{ \begin{array}{ll} \dfrac{1}{Pre_{ij}}, & \text{if } \dfrac{m_i}{Pre_{ij}} = \min_{p_h \in {}^{\bullet}t_j} \left\{ \dfrac{m_h}{Pre_{hj}} \right\} \\ 0, & \text{otherwise} \end{array} \right\} \qquad (8)$$

Note that the value of $\boldsymbol{\Pi}[\boldsymbol{m}]$ changes when the system switches its *configuration*: a configuration assigns to each transition one place that will control its firing rate (i.e. it is constraining that transition). The number of configurations is upper bounded by $\gamma = \prod_{j=1}^{|T|} |{}^{\bullet}t_j|$.

Example 1: Consider the contPN in Figure 1. Assume $\lambda_1 = 3, \lambda_2 = \lambda_3 = 1$. The system dynamics is described as follows:

$$\dot{m}_1 = -2f_1 + f_2 + f_3 = -6 \cdot \min\{m_1/2, m_4/2\} - \min\{m_2, m_4\} + m_3$$
$$\dot{m}_2 = f_1 - f_2 = 3 \cdot \min\{m_1/2, m_4/2\} - \min\{m_2, m_4\}$$
$$\dot{m}_3 = f_1 - f_3 = 3 \cdot \min\{m_1/2, m_4/2\} - m_3$$
$$\dot{m}_4 = -2f_1 - f_2 + 3 \cdot f_3 = -3 \cdot \min\{m_1/2, m_4/2\} - \min\{m_2, m_4\} + 3 \cdot m_3$$

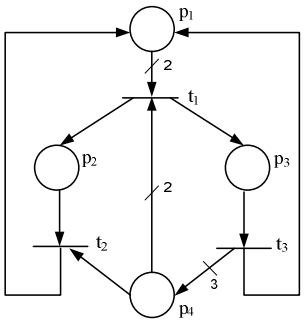

Figure 1. A contPN [6]

3. CONTROL of contPN

In contPNs, the only control action we consider is to brake it down. Hence, the only action that can be applied to contPN is to reduce the flow of transitions [14]. If a transition can be controlled (its flow can be reduced or even stopped), we will say that it is a controllable transition [5]. In this work, it is assumed that all transitions are controllable.

Definition 3: The controlled flow, $\boldsymbol{w}$, of a contPN is

defined as $\boldsymbol{w}[\tau] = \boldsymbol{f}[\tau] - \boldsymbol{u}[\tau]$ with $0 \le \boldsymbol{u}[\tau] \le \boldsymbol{f}[\tau]$, where $\boldsymbol{f}$ is the flow of the uncontrolled system, i.e., defined as in (4), and $\boldsymbol{u}$ is the control action. Therefore, the control input $\boldsymbol{u}$ is dynamically upper bounded by the flow $\boldsymbol{f}$ of the corresponding unforced system. Under these conditions, the overall behaviour of the system in which all transitions are controllable is ruled by the following system:

$$\begin{aligned} &\dot{\boldsymbol{m}} = \boldsymbol{C}\cdot(\boldsymbol{f} - \boldsymbol{u}) = \boldsymbol{C}\cdot\boldsymbol{w} \quad \text{(a)} \\ &0 \le \boldsymbol{w} \le \boldsymbol{\Lambda}\cdot\boldsymbol{\Pi}[\boldsymbol{m}]\cdot\boldsymbol{m} \quad \text{(b)} \end{aligned} \tag{9}$$

In this work, we consider the control strategy developed in [8] aiming at driving the system from initial state to the target one through a linear or a piecewise linear trajectory by minimizing time. In the following, we explain how to compute the corresponding control action $\boldsymbol{u}$ that drives the contPN system from an initial marking, $\boldsymbol{m}_0$, to a desired target marking $\boldsymbol{m}_f$ through a linear trajectory. The procedure consists of assigning constant flows, $\boldsymbol{w}$, that satisfies dynamic upper bounds in (9)(b). $\boldsymbol{m}_0$ and $\boldsymbol{m}_f$ are assumed to be strictly positive. The assumption $\boldsymbol{m}_0$ that is positive ensures that the system can move at $\tau = 0$ in the direction of $\boldsymbol{m}_f$ [15]; the assumption that $\boldsymbol{m}_f$ is positive ensures that $\boldsymbol{m}_f$ can be reached in finite time [5].

Notice that in order to be reachable, $\boldsymbol{m}_f$ necessarily satisfies the state equation: $\boldsymbol{m}[\tau] = \boldsymbol{m}_0 + \boldsymbol{C}\cdot\boldsymbol{\sigma}[\tau]$. As differ from [8], here we will drive the system only through a linear trajectory and will not distinguish the cases $\boldsymbol{m}_0$ and $\boldsymbol{m}_f$ are in the same or in different regions. This way, by the cost of reaching time, calculation of controlled flow is simplified since solving only one LPP is enough.

The programming problem (10) computes a constant controlled flow vector that drives the system from $\boldsymbol{m}_0$ and $\boldsymbol{m}_f$ in minimum time:

$$\begin{aligned} \min_{\boldsymbol{w}} \quad & \tau_f \\ \text{s.t.} \quad & \boldsymbol{m}_f = \boldsymbol{m}_0 + \boldsymbol{C}\cdot\boldsymbol{w}\cdot\tau_f \quad \text{(a)} \\ & 0 \le w_j \le \lambda_j \cdot \min\left\{\frac{m_{0i}}{Pre_{ij}}, \frac{m_{fi}}{Pre_{ij}}\right\} \\ & \forall j \in \{1,..,|T|\} \text{ where i satisfies } Pre_{ij} \neq 0 \quad \text{(b)} \end{aligned} \tag{10}$$

The equations correspond to: (a) the time dependent equation of the straight line connecting $\boldsymbol{m}_0$ to $\boldsymbol{m}_f$ (b) the flow constraints in (9)(b). Notice that (10)(b) is a linear constraint because m_{0i} and m_{fi} are known. The product $\boldsymbol{w}\cdot\tau_f$ makes (10) a BLP. But it can be transformed into a LPP by defining a new vector of variables $\boldsymbol{s} = \boldsymbol{w}\cdot\tau_f$. The resulting LPP is:

$$\begin{aligned} \min_{s} \quad & \tau_f \\ \text{s.t.} \quad & \boldsymbol{m}_f = \boldsymbol{m}_0 + \boldsymbol{C}\cdot\boldsymbol{s} \quad \text{(a)} \\ & 0 \le s_j \le \lambda_j \cdot \min\left\{\frac{m_{0i}}{Pre_{ij}}, \frac{m_{fi}}{Pre_{ij}}\right\}\cdot\tau_f \\ & \forall j \in \{1,..,|T|\} \text{ where i satisfies } Pre_{ij} \neq 0 \quad \text{(b)} \end{aligned} \tag{11}$$

4. IMPLEMENTATION OF CONTROL to SPN via contPN

One of the most important advantages of contPNs is to be able to use more analytical techniques for the analysis of some interesting properties. But after the control law is designed for a contPN, it is important to come back to a reasonable design or control in the original discrete setting. In this section, implementation of control designed for a contPN to the underlying SPN will be examined.

The approximation of the SPN by means of contPN was studied in [16]. There, contPN is analyzed in discrete time, and the following difference equation is obtained:

$$\boldsymbol{m}_{k+1} = \boldsymbol{m}_k + \boldsymbol{C}\cdot\boldsymbol{\lambda}\cdot\boldsymbol{\Pi}(\boldsymbol{m}_k)\cdot\boldsymbol{m}_k\cdot\Delta\tau - \boldsymbol{C}\cdot\Delta\tau\cdot\boldsymbol{u}_k \tag{12}$$

where $\Delta\tau$ is a small enough sampling period. In that work it was proved that when initial states are same, the marking of contPN system whose evolution is described by (12) without input approximates the mean value of the marking of SPN during the time interval $(\tau_0, \tau_0 + n\cdot\Delta\tau)$ for live contPN for any time step k in the interval $(\tau_0, \tau_0 + n\cdot\Delta\tau)$. The approximation can be improved when the probability that the transitions of SPN are all enabled is near one or the probability that the marking is outside the region of initial state is near zero. By using the work in [16], a scheme has been provided in [9] for the interpretation of a control law designed for a contPN system with infinite server semantics into the corresponding SPN. The resulting scheme constitutes a tool for controlling the mean value of a SPN system applying additional delays to the controllable transitions. The scheme is explained below.

Considering the state equation in (9), the controlled flow of a contPN is denoted by

$$w(t_j) = [1 - \alpha(u(t_j), \boldsymbol{m})]\cdot\lambda_j\cdot enab(t_j, \boldsymbol{m}) \tag{13}$$

where $\alpha(u(t_j), \boldsymbol{m})$ is a function takes values in the interval $[0\ \ 1]$. Therefore, the applied control law imposes to t_j an additional delay of:

$$delay(t_j) = \frac{1}{[1 - \alpha(u(t_j), \boldsymbol{m})]\cdot\lambda_j} - \frac{1}{\lambda_j} \tag{14}$$

If additional delays are defined for all the controllable transitions in the same way, and they are added to the corresponding mean time delays of the SPN system, then the mean value of its marking will still be approximated by the marking of the contPN. The application of the control law designed for contPN to SPN is described in the block diagram in Figure 2. In this block diagram, block C2D computes such additional delays, so, according to the previous equation and substituting α, the output of C2D at

time step k is defined as:

$$delay_k(t_j) = \frac{enab(t_j, \boldsymbol{m}_k)}{enab(t_j, \boldsymbol{m}_k)\cdot\lambda_j - u_k(t_j)} - \frac{1}{\lambda_j} \qquad (15)$$

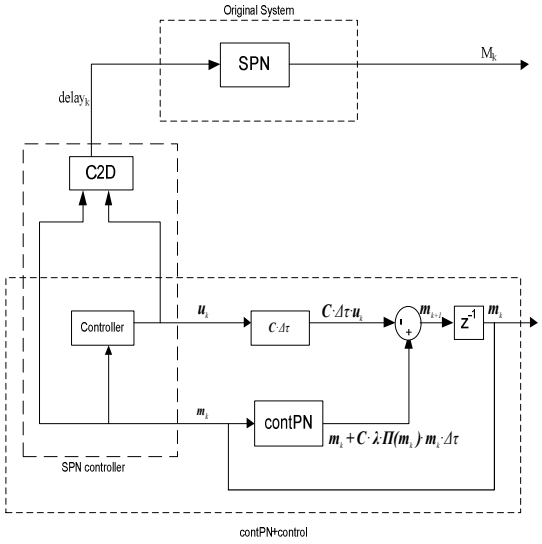

Figure 2. Block diagram of implementation

Example 2: Let us consider the control law design for the contPN in Example 1. Asuume the same firing rates. Let $\boldsymbol{m}_0 = [13 \quad 3 \quad 1 \quad 10]^T$ and $\boldsymbol{m}_f = [12 \quad 3 \quad 2 \quad 7]^T$. First we design the controller to drive the system from $\boldsymbol{m}_0$ to $\boldsymbol{m}_f$ through a linear trajectory by minimizing the time. The constraints of LPP (11) are:

$$\begin{aligned} 12 &= 13 - 2\cdot s_1 + s_2 + s_3 \\ 3 &= 3 + s_1 - s_2 \\ 2 &= 1 + s_1 - s_3 \\ 10 &= 7 - 2\cdot s_1 - s_2 + 3\cdot s_3 \end{aligned} \qquad (a) \qquad (12)$$

$$\begin{aligned} 0 \le s_1 &\le 1.5\cdot\min\{m_{01}, m_{f1}, m_{04}, m_{f4}\}\cdot\tau_f \\ 0 \le s_2 &\le \min\{m_{02}, m_{f2}, m_{04}, m_{f4}\}\cdot\tau_f \\ 0 \le s_3 &\le \min\{m_{03}, m_{f3}\}\cdot\tau_f \end{aligned} \qquad (b)$$

By solving this LPP, the optimal solution is calculated as $s_1 = s_2 = 1, \quad s_3 = 0$ and $\tau_f = 1$. That is, $w_1 = w_2 = 1$ and $w_3 = 0$. And the resulting control law is

$$u(\tau) = \begin{cases} \begin{bmatrix} 1.5\cdot m_4(\tau) \\ m_2(\tau) - 1 \\ m_3(\tau) \end{bmatrix}, & if \quad 0 < \tau \le 1 \end{cases} \qquad (13)$$

The convergence of the markings of contPN under the designed control law is illustrated in Figure 2.

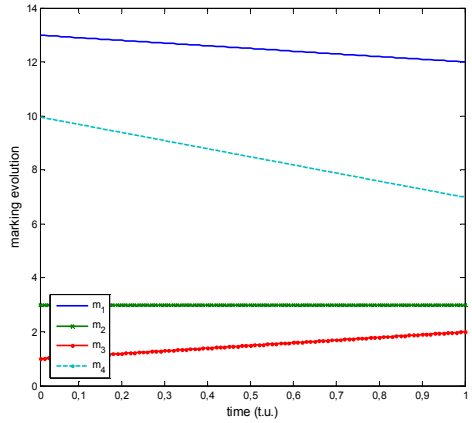

Figure 3. Marking evolutions of contPN for Example 2

In order to implement the control (13) obtained for the contPN to the underlying original SPN, we apply the control scheme of the block diagram in Figure 2. The evolutions of mean values of markings (with 1000 repetition) of the SPN is given in Figure 4 and 5.

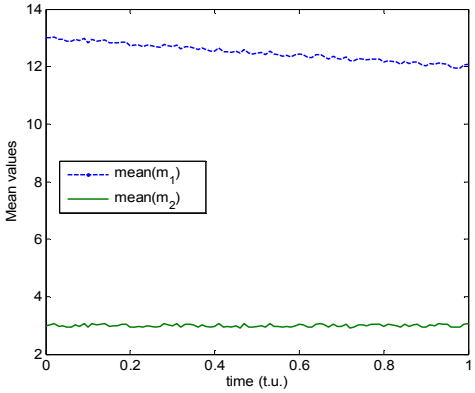

Figure 4: Evolutions of mean values of m_1 and m_2 for Example 2

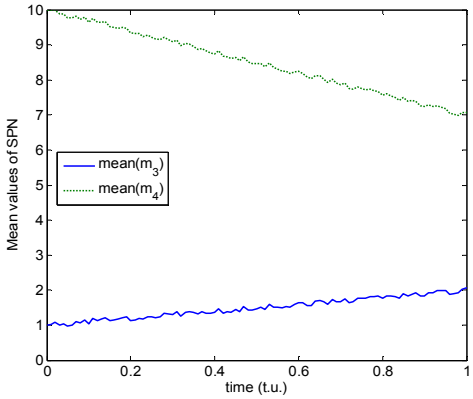

Figure 5: Evolutions of mean values of m_3 and m_4 for Example 2

5. CASE STUDY

Let us consider the contPN sketched in Figure 6 (taken from [17]) which models a table factory system. This system, consists of two different machines to make table-legs (t_1 and t_2), a machine to produce the table boards (t_3), a machine to assemble four legs and a board (t_5), a big painting line which paints two tables at once t_6. More unpainted tables are sent (t_4) from another factory. The places p_1, p_2, p_3 and p_4 are work orders; while p_5, p_6 and p_7 are devoted to the storage of table-legs, boards and unpainted tables, respectively (see [17] for details).

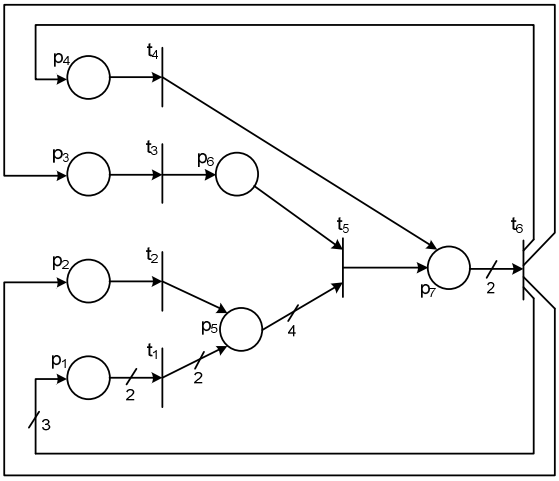

Figure 6: ContPN model of a table factory system

Suppose in the initial marking $m_{01} = m_{03} = 2$, $m_{02} = m_{05} = m_{06} = m_{07} = 1$ and $m_{04} = 3$. For the final state work orders are desired as $m_{f1} = m_{f2} = m_{f4} = 2$, $m_{f3} = 3$. We want to increase the number of stored table legs, i.e. $m_{f5} = 4$, and keep the number of stored boards and unpainted tables, i.e. $m_{f6} = m_{f7} = 1$ in minimum time. By solving LPP(11), the corresponding control action is obtained as:

$$u(\tau) = \begin{Bmatrix} \lceil 0.5 \cdot \mathrm{m}_{01} \text{-} 0.75 \rceil \\ \mathrm{m}_{02} \\ \mathrm{m}_{03} \\ \mathrm{m}_{04} \text{-} 1 \\ 0.25 \cdot \mathrm{m}_{05} \\ \lceil 0.5 \cdot \mathrm{m}_{07} \text{-} 0.5 \rceil \end{Bmatrix}, \quad if \quad 0 < \tau \le 2 \qquad (16)$$

In order to implement the control (13) obtained for the contPN to the underlying original SPN, we apply the control scheme of the block diagram in Figure 2. The markings of contPN and the mean values (with 1000 repetition) of the SPN at time $\tau = 2$ are given in Table 1. The evolutions of some markings (at contPN) and the mean values of these markings (at SPN) are given in Figures 7-10.

Table 1: The markings of the contPN and the mean values of the SPN at time $\tau = 2$

	contPN (exact value)	SPN (meanvalue)
m_1	2	2.03
m_2	2	2.02
m_3	3	2.98
m_4	2	2.01
m_5	4	3.97
m_6	1	1.04
m_7	1	0.97

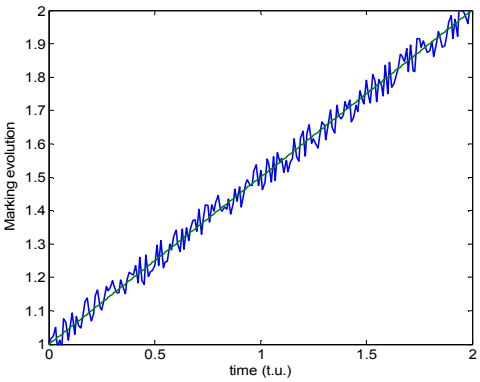

Figure 7: Evolution of m_2 for the case study

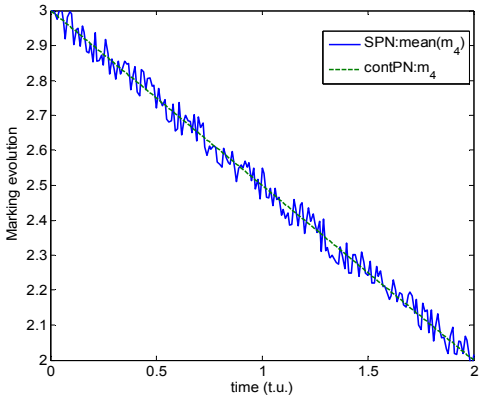

Figure 8: Evolution of m_4 for the case study

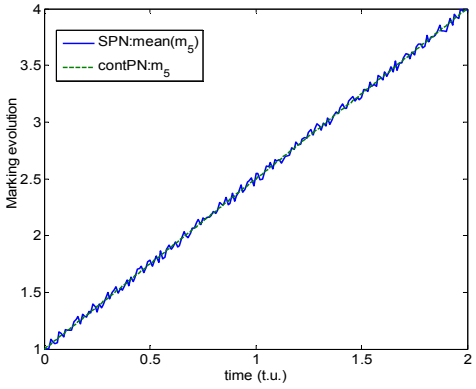

Figure 9: Evolution of m_5 for the case study

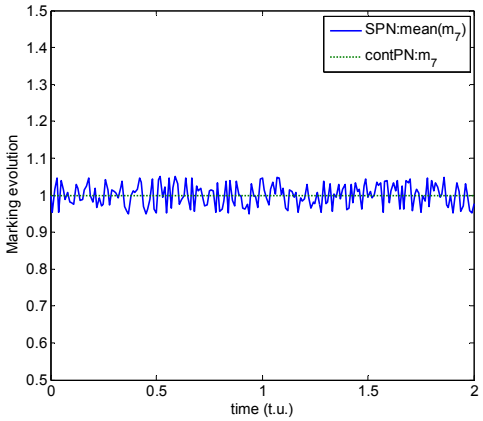

Figure 10: Evolution of m_7 for the case study

6. CONCLUSION

In this work, efficient usability of the target marking control strategy for contPNs developed in the author's previous work [8] is analyzed and illustrated for the control of mean values at the underlying SPNs. For this purpose, the scheme developed in [9] is used for the interpretation of the control law designed for a contPN system into the underlying SPN. The efficiency of this interpretation is studied on a table factory system and satisfactory results are obtained, showing that the target state controller designed for contPN system can be used for the control of underlying SPN system efficiently.

REFERENCES

[1] D. Mandrioli, A. Morzenti, M. Pezze, P. Pietro S.,and S. Silva. A Petri net and logic approach to the specification and verification of real time systems. In Formal Methods for Real Time Computing. John Wiley &Sons Ltd.

[2] T. Murata. Petri nets: properties, analysis and applications. Proceedings of the IEEE, 77:541–580, 1989.

[3] R. David and H.Alla. Continuous Petri nets. In 8th European Workshop on Application and Theory of Petri Nets, Zaragoza, Spain, 1987.

[4] M. Silva and L. Recalde. Petri nets and integrality relaxations a view of continuous Petri nets. IEEE Trans on Systems Man and Cybernetics, 32(4):314-327, 2002.

[5] C. Mahulea, A. Ramirez, L. Recalde, M.Silva. Steady state control reference and token conservation laws in continuous Petri net systems. IEEE Transactions on Automation Science and Engineering, 2008, 5(2): 307–320.

[6] C. Mahulea, A. Giua, L. Recalde, C. Seatzu, M. Silva. Optimal model predictive control of timed continuous Petri nets. IEEE Transactions on Automatic Control, 2008, 53(7): 1731 – 1735.

[7] J. Xu, L. Recalde, M. Silva. Tracking control of join-free timed continuous Petri net systems under infinite servers semantics. Discrete Event Dynamic Systems, 2008, 18(2): 263–288.

[8] H. Apaydın - Özkan, J. Julvez, C. Mahulea, M. Silva. Approaching minimum time control of Timed continuous petri nets. Nonlinear Analysis: Hybrid Systems, 2011, 5(2): 136 – 148.

[9] C.R. Vázquez, M. Silva, "Performance Control of Markovian Petri Nets via Fluid Models: A Stock-Level Control Example", In 5th IEEE Conference on Automation Science and Engineering (IEEE CASE). (Session In Memoriam of prof. Laura Recalde: Advances in Petri Net theory.), Bangalore, India, aug 2009.

[10] L. Recalde, E. Teruel, and M. Silva. On linear systems. Journal of Circuits Systems and Computers, 8(1):223–265, 1998.

[11] L. Recalde, E. Teruel, and M. Silva. Autonomous continuous P/T systems. In S. Donatelli and J. Kleijn, editors, Application and Theory of Petri Nets, 1639:107–126. Springer, 1999.

[12] L. Recalde and M. Silva. Petri nets fluidification revisited semantics and steady state. Eoropean Journal of Automation *APII JESA*, 35(4):435–449, 2001.

[13] H.Alla and R.David, A modeling and Analysis tool for discrete event systems-Continuous Petri net. Performance Evaluation, 1998, 33(1),175-199.

[14] M. Silva, L. Recalde. On fluidification of Petri net models: from discrete to hybrid and continuous models.

[15] J. Julvez, L. Recalde, and M. Silva. On reachability in autonomous continuous Petri net systems. In 24th International Conference on Application and Theory of Petri Nets, pages 221–240, Eindhoven, Netherlands, June2003. Springer

[16] C.R. Vazquez, L. Recalde, and M. Silva. Stochastic–continuous state approximation of

markovian petri net systems. In CDC 2008: 47th IEEE Conference on Decision and Control, Cancun, Mexico, December 2008.

[17] E. Teruel, J. M. Colom, M. Silva. Choice-free Petri nets: A model for deterministic concurrent systems with bulk services and arrivals. IEEE Transactions on Systems, Man, and Cybernetics, 1997, **1**(27): 73–83.