Optimal control of timed continuous Petri nets via explicit MPC *

A. Giua[2], C. Mahulea[1], L. Recalde[1], C. Seatzu[2], and M. Silva[1]

[1] Dep. de Informática e Ingeniería de Sistemas, Universidad de Zaragoza, Spain, {cmahulea,lrecalde,silva}@unizar.es
[2] Dip. Ingegneria Elettrica ed Elettronica, Università di Cagliari, Italy, {giua,seatzu}@diee.unica.it

Abstract. In this paper we deal with the problem of controlling *timed continuous Petri nets* in order to reach a final (steady state) configuration while minimizing a quadratic performance index. The formulation of a discrete-time *linear positive* model with dynamic (or state-based) constraints on the control input, enables us to design a *state-feedback* control law based on *explicit model predictive control* (eMPC). The eMPC partitions the state space into polytopes: an affine state-feedback control law is uniquely associated to each polytope, while the on-line phase of the approach consists in evaluating the current region and consequently the optimal control law.

1 Introduction

Petri nets (PN) are a mathematical tool with an appealing graphical representation very adequate for modeling discrete event systems. Its main feature is that their state space belongs to the set of non-negative integers [1].

In many real size applications the number of reachable states may be very high thus the analysis and optimization of these systems require large amount of computational efforts, thus leading to analytically and computationally untractable problems. One way to tackle this difficulty consists in the relaxation of the original integrity constraints, giving a *fluid* (i.e., continuous) approximation of the discrete event dynamics [2, 3]. Fluid models may be studied by means of structural analysis, an efficient technique that does not require the enumeration of the state space [1].

In this paper we consider *timed continuous Petri net systems* under infinite server semantics and subject to external control actions: we assume that the only admissible control law consists in slowing down the firing speed of transitions [3]. Such a system can be represented by a particular *hybrid positive* model: a *piecewise linear* positive model with autonomous switches and with constraints on the state and control input space. By a suitable change of

* This work was partially supported by project CICYT and FEDER DPI2003-06376 in which Alessandro Giua and Carla Seatzu are also participating.

variables it is also possible, as shown in [4], to further simplify the model into a discrete-time *linear model* with constraints on the state and input space: in particular, the positiveness of the system results from these constraints. This is the model that will be considered in this paper.

The optimal control of constrained systems has received a lot of attention in the literature, and one of the most general and interesting approaches is the so-called *explicit model predictive control* (eMPC) [5]. The objective of this paper is that of showing how eMPC can be effectively applied to the control of timed continuous Petri nets. Note that although the eMPC approach can be directly applied to the original piecewise linear model, the implementation of the control design for the linear model derived in [4] is much simpler.

The particular problem considered in this paper is that of reaching (from a given initial state) a final steady state in a finite time, while minimizing a given quadratic performance index. The main advantage of the proposed solution is that it provides a state-feedback control law whose closed-loop stability and constraint satisfaction are guaranteed, while the most burdensome part of the procedure is performed off-line.

2 Continuous Petri nets

Definition 1. *A* continuous *PN (contPN) system is a pair* $\langle \mathcal{N}, \boldsymbol{m_0} \rangle$*, where:* $\mathcal{N} = \langle P, T, \boldsymbol{Pre}, \boldsymbol{Post} \rangle$ *is the net structure with set of places* P*, set of transitions* T*, pre and post incidence matrices* $\boldsymbol{Pre}, \boldsymbol{Post} : P \times T \to \mathbb{N}$*;* $\boldsymbol{m_0} : P \to \mathbb{R}_{\geq 0}$ *is the initial marking.*

We denote $\boldsymbol{m}(\tau)$ the marking at time τ and in discrete time we denote $\boldsymbol{m}(k)$ the marking at sampling instant k, $\tau = k \cdot \Theta$. Finally, the *preset* and *postset* of a node $x \in P \cup T$ are denoted ${}^{\bullet}x$ and $x^{\bullet}$, respectively.

A transition $t_j \in T$ is *enabled* at $\boldsymbol{m}$ iff $\forall p_i \in {}^{\bullet}t_j, m_i > 0$, and its *enabling degree* is $enab(t_j, \boldsymbol{m}) = \min\limits_{p_i \in {}^{\bullet}t_j} \left\{ \frac{m_i}{Pre(p_i,t_j)} \right\}$. An enabled transition t can fire in any real amount $0 \leq \alpha \leq enab(t, \boldsymbol{m})$ leading to a new marking $\boldsymbol{m}' = \boldsymbol{m} + \alpha \boldsymbol{C}(\cdot, t)$, where $\boldsymbol{C} = \boldsymbol{Post} - \boldsymbol{Pre}$ is the *token flow* matrix; this firing is also denoted $\boldsymbol{m}[t(\alpha)\rangle \boldsymbol{m}'$.

In general, if $\boldsymbol{m}$ is reachable from $\boldsymbol{m_0}$ through a sequence $\sigma = t_{r_1}(\alpha_1) t_{r_2}(\alpha_2) \ldots t_{r_k}(\alpha_k)$, and we denote by $\boldsymbol{\sigma} : T \to \mathbb{R}_{\geq 0}$ the *firing vector* whose component associated to a transition t_j is $\sigma_j = \sum_{h \in H(\sigma, t_j)} \alpha_h$, where $H(\sigma, t_j) = \{h = 1, \ldots, k \mid t_{r_h} = t_j\}$, we can write: $\boldsymbol{m} = \boldsymbol{m_0} + \boldsymbol{C} \cdot \boldsymbol{\sigma}$, which is called the *fundamental equation*.

Definition 2. *A (deterministically) timed contPN system is a contPN system together with a vector* $\boldsymbol{\lambda} : T \to \mathbb{R}_{>0}$*, where* λ_j *is the firing rate of* t_j*.*

Now, the fundamental equation depends on time: $\boldsymbol{m}(\tau) = \boldsymbol{m_0} + \boldsymbol{C} \cdot \boldsymbol{\sigma}(\tau)$, where $\boldsymbol{\sigma}(\tau)$ denotes the firing count vector in the interval $[0, \tau]$. Deriving it wrt time the following is obtained: $\dot{\boldsymbol{m}}(\tau) = \boldsymbol{C} \cdot \dot{\boldsymbol{\sigma}}(\tau)$. The derivative of firing vector represents the *flow* of the timed model $\boldsymbol{f}(\tau) = \dot{\boldsymbol{\sigma}}(\tau)$. Depending on how the flow of the transition is defined several firing semantics are possible [6, 2]. This paper deals with *infinite server semantics* in which the flow of transition t_j is given by: $f_j = \lambda_j \min\limits_{p_i \in {}^\bullet t_j} \left\{ \frac{m_i}{Pre(p_i,t_j)} \right\}$.

Because the flow of a transition depends on its enabling degree which is based on the minimum function, a timed contPN under infinite servers semantics is a piecewise linear system. In fact, if we define $s = \prod\limits_{t \in T} |{}^\bullet t|$, where $|{}^\bullet t|$ denotes the cardinality of the set ${}^\bullet t$, the state space of a timed contPN can be *partitioned*[3] as follows: $R_1 \cup \cdots \cup R_s$, where each set R_k (for $k = 1, \ldots , s$) denotes a *region* where the flow is limited by the same subset of places (one for each transition). For a given region R_k, we can define the *constraint matrix* $\boldsymbol{\Pi_k} : T \times P \to \mathbb{R}$ such that:

$$\boldsymbol{\Pi_k}(t_j, p_i) = \begin{cases} \dfrac{1}{Pre(p_i,t_j)} \text{ if } (\forall \boldsymbol{m} \in R_k)\ \dfrac{m_i}{Pre(p_i,t_j)} = \min\limits_{p_h \in {}^\bullet t_j} \left\{ \dfrac{m_h}{Pre(p_h,t_j)} \right\}; \\ \quad 0 \qquad \text{otherwise.} \end{cases} \tag{1}$$

If marking $\boldsymbol{m}$ belongs to R_k, we denote $\boldsymbol{\Pi}(\boldsymbol{m}) = \boldsymbol{\Pi}_k$ the corresponding constraint matrix. Furthermore, the firing rate of transitions can also be represented by a diagonal matrix $\boldsymbol{\Lambda} : T \times T \to \mathbb{R}_{>0}$, where $\Lambda(t_j, t_h) = \lambda_j$ if $j = h$, and 0, otherwise. Using this notation, the non-linear flow of the transitions at a given marking $\boldsymbol{m}$ can be written as $\boldsymbol{f} = \boldsymbol{\Lambda} \cdot \boldsymbol{\Pi}(\boldsymbol{m}) \cdot \boldsymbol{m}$.

3 A linear constrained model of timed contPN

In this section we consider net systems subject to external control actions, and assume that the only admissible control law consists in slowing down the firing speed of transitions, that are assumed to be all controllable [3, 7].

Definition 3. *The flow of the forced (or controlled) timed contPN will be denoted by* $\boldsymbol{w}(\tau) = \boldsymbol{f}(\tau) - \boldsymbol{u}(\tau)$*, with* $\boldsymbol{0} \leq \boldsymbol{u}(\tau) \leq \boldsymbol{f}(\tau)$.

Therefore, the control input will be dynamically upper bounded by the flow of the corresponding unforced system. The behavior of the system is ruled by

$$\begin{cases} \dot{\boldsymbol{m}}(\tau) = \boldsymbol{C} \cdot [\boldsymbol{\Lambda} \cdot \boldsymbol{\Pi}(\boldsymbol{m}(\tau)) \cdot \boldsymbol{m}(\tau) - \boldsymbol{u}(\tau)] \\ 0 \leq \boldsymbol{u}(\tau) \leq \boldsymbol{\Lambda} \cdot \boldsymbol{\Pi}(\boldsymbol{m}(\tau)) \cdot \boldsymbol{m}(\tau) \end{cases} \tag{2}$$

This is a particular hybrid system: a piecewise linear system with autonomous switches and dynamic (or state-based) constraints in the input.

Proposition 1. *[4] Any piecewise linear constrained model of the form* (2) *can be rewritten as a* linear constrained model *of the form*

[3] These partitions are disjoint except possibly on the borders.

$$\begin{cases} \dot{\boldsymbol{m}}(\tau) = \boldsymbol{C} \cdot \boldsymbol{w}(\tau), \quad \boldsymbol{m}(0) = \boldsymbol{m_0} \geq \boldsymbol{0} \\ \boldsymbol{G} \cdot \begin{bmatrix} \boldsymbol{w}(\tau) \\ \boldsymbol{m}(\tau) \end{bmatrix} \leq \boldsymbol{0} \\ \boldsymbol{w}(\tau) \geq \boldsymbol{0} \end{cases} \tag{3}$$

that we call continuous time contPN *model (or* ct-contPN *for short) where* $\boldsymbol{G}$ *is an appropriate constant matrix defined as follows:* $\boldsymbol{G} = \begin{bmatrix} \boldsymbol{Q} & -\boldsymbol{R} \end{bmatrix}$, $\boldsymbol{Q} \in \mathbb{Z}^{q \times |T|}$, $\boldsymbol{R} \in \mathbb{Z}^{q \times |P|}$, $q = \sum_{t \in T} |^{\bullet}t|$*, and the the row of* $\boldsymbol{Q}$ *and* $\boldsymbol{R}$ *relative to the generic pre arc* (p_i, t_j) *are respectively*

$$\begin{bmatrix} \underbrace{0 \;\; \cdots \;\; 0 \;\; 1}_{j} & 0 & \cdots & 0 \end{bmatrix}, \qquad \begin{bmatrix} \underbrace{0 \;\; \cdots \;\; 0 \;\; \frac{\lambda_j}{Pre(p_i, t_j)}}_{i} & 0 & \cdots & 0 \end{bmatrix}.$$

The system in equation (3) is a linear system with a *state-matrix* equal to $\boldsymbol{0}$ and an *input matrix* equal to the *token flow matrix* of the contPN. There is still a dynamic constraint on the system inputs that depends on the value of the system state $\boldsymbol{m}$. The continuous-time system (3) can be discretized, thus obtaining a discrete-time "equivalent" model.

Definition 4. *Consider a ct-contPN model of the form* (3) *and let* Θ *be a sampling period. A model can be given in terms of a* discrete-time contPN *or* dt-contPN *as follows:*

$$\begin{cases} \boldsymbol{m}(k+1) = \boldsymbol{m}(k) + \Theta \cdot \boldsymbol{C} \cdot \boldsymbol{w}(k), \quad \boldsymbol{m}(0) = \boldsymbol{m_0} \geq \boldsymbol{0} \\ \boldsymbol{G} \cdot \begin{bmatrix} \boldsymbol{w}(k) \\ \boldsymbol{m}(k) \end{bmatrix} \leq \boldsymbol{0} \\ \boldsymbol{w}(k) \geq \boldsymbol{0}. \end{cases} \tag{4}$$

Example 1. Let us consider the net system in Fig. 1 with $\Theta = 1$ and $\boldsymbol{\lambda} = [5 \; 1]^T$. The discrete-time representation is given by $\boldsymbol{m}(k+1) = \boldsymbol{m}(k) + \boldsymbol{C}\boldsymbol{w}(k)$ where $\boldsymbol{C} = [-1 \; 1; 1 \; -1; -1 \; 1]$, and: $w_1(k) - \frac{\lambda_1}{2} \cdot m_1(k) \leq 0$, $w_1(k) - \lambda_1 \cdot m_3(k) \leq 0$, $w_2(k) - \lambda_2 \cdot m_1(k) \leq 0$, $w_2(k) - \lambda_2 \cdot m_2(k) \leq 0$, $\boldsymbol{w}(k),\ \boldsymbol{m}(k+1) \geq \boldsymbol{0}$.

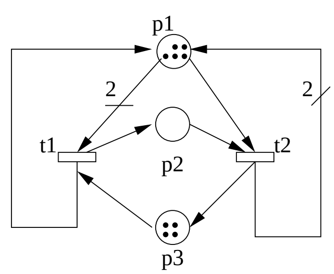

Fig. 1. Continuous PN system.

Note that even if $m = |P| = 3$, the number of state variables that vary independently is equal to one, being $m_1 + m_2 = 5$ and $m_2 + m_3 = 4$. ■

It is important to stress that, although the evolution of a dt-contPN occurs in discrete steps, as was the case for an untimed system, *discrete time evolutions* and *untimed evolutions* are not the same. In fact, while an untimed net system can be seen evolving sequentially, executing a single transition firing at each step, a dt-contPN may evolve in concurrent steps where more than one transition can fire. In a ct-contPN under infinite servers semantics, the positiveness of the marking is ensured if $\boldsymbol{m_0}$ is positive, because the flow of a transition goes to zero whenever one of input places is empty [8]. In a dt-contPN, this is not sufficient. We extensively addressed this problem in [4] where we shown that it can be avoided choosing Θ small enough.

4 Optimal control via explicit MPC

Steady state optimal control of contPN was studied in [7]: if all transitions can be controlled the problem can be solved in polynomial time. The result of LPP in [7] is an optimal marking and an optimal control input in steady state. In this paper we assume that this steady state marking is known and our problem is to reach it (from a given $\boldsymbol{m_0}$) in a finite time by optimizing a quadratic performance index of the form

$$J(\boldsymbol{m}(\tau), \boldsymbol{w}(\tau), N) = \left\{ r \cdot ||\boldsymbol{m}(N) - \boldsymbol{m}_f||_2^2 + \sum_{k=0}^{N-1} \left[q(k) \cdot ||\boldsymbol{m}(k) - \boldsymbol{m}_f||_2^2 + r(k) \cdot ||\boldsymbol{w}(k) - \boldsymbol{w}_f||_2^2 \right] \right\} \tag{5}$$

where: r represents the weights on the final state (the desired state after N time horizon); $q(k)$ and $r(k)$ represent the penalty on the intermediate trajectory and the penalty on the control effort, respectively.

The solution we propose is based on MPC. MPC, also referred to as *moving horizon control* or *receding horizon control*, is an advanced control method that has become an attractive feedback strategy, both in the case of linear and nonlinear systems [9]. In this section we show how MPC can be effectively used to control contPN under infinite servers semantics.

The basic idea of MPC, going back to the 70's, is the following: at every time step, the control action is chosen solving an optimal control problem that minimizes a performance criterion over a future (sliding) horizon. Only the first control command will be applied and after one time step other measurements will be got and the optimization problem is repeated. The applicability of the (implicit) MPC approach is limited by the requirement of solving on-line a linear (or quadratic) programming problem.

A possible solution to overcome this difficulty has been firstly given by Bemporad *et al.* in [5] where the *explicit MPC* has been proposed to compute *off-line* the explicit state-feedback solution to the linear quadratic optimal control problem subject to state and input constraints. More precisely, the eMPC approach moves all the burdensome computations off-line and partitions the state space into polytopes described by linear inequalities[4]. An affine state-feedback control law is uniquely associated to each polytopic region. The on-line phase of the approach consists in evaluating the current region and consequently the optimal value of the control law. Thus, the resulting control law is a *piecewise continuous affine state-feedback* control law.

In [5] Bemporad *et al.* have shown in detail how the state space partition and the affine control laws can be computed by means of multiparametric quadratic programming. For sake of brevity we do not provide these results here. Moreover, the explicit solution can be easily computed thanks to the Multi-Parametric Toolbox called MPT [10], a MATLAB toolbox for design and analysis of optimal controllers for constrained linear and hybrid systems.

As already discussed in [5], we remark that computing eMPC may lead to controllers with prohibitive complexity, both in running time and number of polytopes. In particular, there are three aspects which are important in this respect: performance, closed-loop stability and constraint satisfaction. The MPT toolbox provides several possibilities to compute the controller and the partition of the state space, which are specified below.

Finite Time Optimal Control (FTOC) yields the finite time optimal controller, i.e., the performance will be N-step optimal but may not be infinite horizon optimal. The complexity of the controller highly increases with the prediction horizon N. Within this method, the MPT toolbox provides two different modes.

— *probstruct:Tconstraint=0.* No guarantees on stability or closed-loop constraint satisfaction is given. As N is increased the feasible set of states will converge to the maximum controllable set (i.e., all states that are controllable) from the outside-in, i.e., the controlled set will get smaller as N increases.

— *probstruct:Tconstraint=1.* The resulting controller will guarantee stability and constraint satisfaction for all time, but will only cover a subset of the maximum controllable set of states. By increasing the prediction horizon, the controllable set of states will converge to the maximum controllable set from the inside-out, i.e., the controlled set will grow larger as N increases.

Infinite Time Optimal Control (ITOC) yields the infinite time optimal controller, i.e., the best possible performance for the control problem. Asymptotic stability and constraint satisfaction are guaranteed and the maximum controllable set will be covered by the resulting controller.

The main goal of this paper is that of investigating via some numerical simulations, carried out using the Multi-Parametric Toolbox of MATLAB [10], the possibility of using eMPC to control contPN. As a result, the following conclusions can be drawn.

— In the case of FTOC all the obtained results are reliable. The only limitation of the approach, as already pointed out in [5], is that the controller's complexity and the computational time become prohibitive when the order of the state space, as well as the length of the prediction horizon N, increases.

[4] A bounded polyhedron $\mathcal{P} \subset \mathbb{R}^n$, $\mathcal{P} = \{\boldsymbol{x} \in \mathbb{R}^n \mid \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{B}\}$ is called a *polytope*.

— In the case of ITOC, we run into some difficulties in tuning the parameters, and following the guidelines in [10], we obtained unreliable results. Therefore, we prefer not to handle this case in the rest of the paper.

Example 2. Let us consider the cont-PN in Fig. 1 with $\boldsymbol{\lambda} = [5,\ 1]^T$. Assume that the steady state (final) configuration is given by $\boldsymbol{m}_f = [2.5\ 2.5\ 1.5]^T$ and $\boldsymbol{w}_f = [2.5\ 2.5]^T$. Finally, let $\Theta = 0.05$. We consider a FTOC of the form (5) where $r = 10$, $q(k) = 1$ and $r(k) = 0.01 \cdot \boldsymbol{I}$ for all $k = 0, 1, \ldots, N-1$.

In Table 1 we summarized the results obtained in the case of FTOC with *probstruct:Tconstraint=0* and *probstruct:Tconstraint=1*, and different values of N. First, we observe that for all values of N the number of regions (see column 2) is the always same in the case of *probstruct:Tconstraint=0* and *probstruct:Tconstraint=1* and increases with N. In columns 3 to 5 we reported the results relative to *probstruct:Tconstraint=0*, while in the last three are reported the results relative to *probstruct:Tconstraint=1*. In the first case the controlled set keeps unaltered; in the second case it grows larger when N grows. In both cases the computational time increases with the horizon N. All simulations have been carried out on a Pentium III 450 MHz.

		probstruct:Tconstraint=0			*probstruct:Tconstraint=1*		
N	number of regions	computational time [sec]	controlled set	J	computational time [sec]	controlled set	J
1	5	0.94	$[0,4]$	14.63	0.87	$[1.57, 3.10]$	–
2	8	1.32	$[0,4]$	14.61	1.32	$[1.10, 3.18]$	–
3	11	1.92	$[0,4]$	14.60	1.87	$[0.52, 3.27]$	–
4	14	2.58	$[0,4]$	14.60	2.91	$[0, 3.30]$	–
5	16	3.24	$[0,4]$	14.60	3.35	$[0, 3.40]$	–
10	19	13.89	$[0,4]$	14.60	14.28	$[0, 3.75]$	14.60
15	24	122.04	$[0,4]$	14.60	126.00	$[0,4]$	14.60
20	26	648.12	$[0,4]$	14.60	565.46	$[0,4]$	14.60

Table 1. The results of Example 2.

In Table 1 we have also reported the value of the quadratic performance index assuming $m_2(0) = 3.5$. We may observe that in this case the value of the cost is not strongly dependent on the value of N. In the case of *probstruct:Tconstraint* $= 1$ the cost has not been evaluated for $N = 1, \ldots, 5$: indeed in such cases the controlled set was not large enough to cover the state space of interest and a control law was not available.

Finally, Fig. 2 presents some details on the controlled system's behaviour in the case of *probstruct:Tconstraint=1* and $N = 10$: in fig. (a) we have reported the markings' evolution wrt time; fig. (b) shows the behaviour of the control variables w_1 and w_2 wrt time; fig. (c) shows the current index of the convex region (the polytope) that uniquely determines the piecewise affine control law. As we may observe, the system reaches the desired steady state configuration: $\boldsymbol{m}_f = [2.5\ 2.5\ 1.5]^T$, $\boldsymbol{w}_f = [2.5\ 2.5]^T$. ■

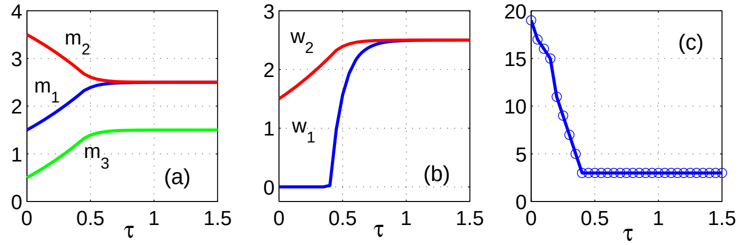

Fig. 2. The results of Example 2 when *probstruct:Tconstraint=1* and $N = 10$.

5 Conclusions

We considered *timed contPN* under infinite server semantics. On the basis of a constrained discrete-time positive linear model of the system, we derived a *state-feedback* optimal control law based on eMPC. The results of some numerical simulations carried out on the MPT toolbox of MATLAB [10] are discussed in the case of finite time optimal control with different finite horizons. We do not provide the results of numerical simulations carried out in the case of infinite time optimal control because we found out some inconsistencies when following the guidelines in [10] for the parameters tuning.

References

1. Silva, M., Colom, J.M., Campos, J.: Linear algebraic techniques for the analysis of Petri nets. In: Recent Advances in Mathematical Theory of Systems, Control, Networks, and Signal Processing II. Mita Press (1992) 35–42
2. David, R., Alla, H.: Discrete, Continuous and Hybrid Petri Nets. Springer-Verlag (2004)
3. Silva, M., Recalde, L.: On fluidification of Petri net models: from discrete to hybrid and continuous models. Annual Reviews in Control **28** (2004) 253–266
4. Mahulea, C., Giua, A., Recalde, L., Seatzu, C., Silva, M.: On sampling continuous timed PNs: reachability "equivalence" under infinite servers semantics. In: 2^{nd} IFAC C. on Analysis and Design of Hybrid Systems, Alghero, Italy (2006)
5. Bemporad, A., Morari, M., Dua, V., Pistikopoulos, E.: The explicit linear quadratic regulator for constrained systems. Automatica **38** (2002) 3–20
6. Recalde, L., Silva, M.: Petri Nets fluidification revisited: Semantics and steady state. APII-JESA **35** (2001) 435–449
7. Mahulea, C., Ramírez, A., Recalde, L., Silva, M.: Steady state control, zero valued poles and token conservation laws in continuous net systems. In: Workshop on Control of Hybrid and Discrete Event Systems, Miami, USA, J.M. Colom, S. Sreenivas and T. Ushio, eds. (2005)
8. Silva, M., Recalde, L.: Unforced continuous petri nets and positive systems. In L. Benvenuti, A.D.S., Farina, L., eds.: Positive Systems, Proceedings of the First Multidisciplinary International Symposium on Positive Systems: Theory and Applications (POSTA 2003). Number 294, Springer–Verlag (2003)
9. Bemporad, A., Torrisi, F., Morari, M.: Performance analysis of piecewise linear systems and model predictive control systems. In: 39th IEEE Conference on Decision and Control, Sydney, Australia (2000) 4957–4962
10. Kvasnica, M., Grieder, P., Baotić, M.: Multi-Parametric Toolbox (MPT) (2004)