Optimal control of continuous Petri nets via model predictive control

A. Giua, C. Mahulea, L. Recalde, C. Seatzu, M. Silva

Abstract— This paper addresses the optimal control problem of continuous Petri nets under infinite servers semantics. Our goal is to find a control input optimizing a certain cost function that permits the evolution from an initial marking to a desired configuration. The problem is studied through Model Predictive Control (MPC), a control method, extensively used in industrial applications. Implicit and explicit procedures are presented together with a comparison between the two schemes.

I. Introduction

DISCRETE Petri nets (PN) [10] are a mathematical formalism with an appealing graphical representation for the description of discrete-event systems, successfully used for modeling, analysis and synthesis of such systems. Its main feature is that their state space and transition firings belong to the set of non-negative integers [11].

As other models of concurrent systems, discrete PN may suffer the state explosion problem. As a consequence 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 [4], [13]. Fluid models may be studied by means of structural analysis, that does not not require the enumeration of the state space [11].

To study the performance of systems, timing can be introduced and timed PN are obtained. In this paper we consider *timed continuous Petri net systems* under infinite server semantics that, in general, provide a better approximation of a discrete model [8]. Continuous nets are subject to external control actions: we assume that the only admissible control law consists in slowing down the firing speed of transitions [13]. 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 variables it is also possible, as shown in [6], to further simplify the model into a discrete-time *linear model* with constraints on the state and input space.

Steady state optimal control of continuous PN (contPN) was studied in [7], where it has been shown that if all transitions can be controlled and the objective function is linear, the problem is solvable in polynomial time by means of Linear Programming Problems (LPP). The result of such an LPP in [7] is an optimal marking and an optimal control input in steady state. In this paper we assume that this steady state configuration is known and our problem is to reach it, from a given initial marking, by optimizing a given performance index.

The solution we propose is based on *Model Predictive Control* (MPC) [2], a control method that has become an attractive control strategy in the last years. In particular, we investigate the possibility of using both an implicit and an explicit [1] MPC control strategy. The main advantage of the explicit 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. However, as already pointed out by the authors in [1], the computational complexity of the explicit approach may become prohibitive when dealing with complex systems.

A comparison among the two procedures is proposed in the last section of the paper where the results of various numerical simulations are presented. We finally observe that although the explicit approach can be directly applied to the original piecewise linear model, however the implementation of the control design for the linearly constrained model derived in [6] is much simpler.

II. Continuous Petri nets

Definition 2.1: A 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, where $\tau = k \cdot \Theta$ and Θ is the *sampling period*. Finally, the *preset* and *postset* of a node $x \in P \cup T$ are denoted $|^\bullet x|$ and $|x^\bullet|$, respectively.

A transition $t \in T$ is *enabled* at $\boldsymbol{m}$ iff $\forall p_i \in {}^\bullet t, m_i > 0$, and its *enabling degree* is

$$enab(t, \boldsymbol{m}) = \min_{p_i \in {}^\bullet t} \left\{ \frac{m_i}{Pre(p_i, t)} \right\}.$$

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 $C = Post - Pre$ is the *token flow* matrix; this firing is also denoted $m[t(\alpha)\rangle m'$.

In general, if m is reachable from 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 $\sigma : T \to \mathbb{R}_{\geq 0}$ 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:

$$m = m_0 + C \cdot \sigma,$$

which is called the *fundamental equation*.

The basic difference between discrete and continuous PN is that the components of the markings and firing count vectors are not restricted to take values in the set of natural numbers but in the set of non-negative real numbers.

Definition 2.2: A (deterministically) *timed* contPN system is a contPN system together with a vector $\lambda : T \to \mathbb{R}_{>0}$, where λ_j is the firing rate of t_j.

Now, the fundamental equation depends on time: $m(\tau) = m_0 + C \cdot \sigma(\tau)$, where $\sigma(\tau)$ denotes the firing count vector in the interval $[0, \tau]$. Deriving it with respect to time the following is obtained: $\dot{m}(\tau) = C \cdot \dot{\sigma}(\tau)$. The derivative of the firing vector represents the *flow* of the timed model $f(\tau) = \dot{\sigma}(\tau)$. Depending on how the flow of the transition is defined many firing semantics are possible [9], [4]. This paper deals with *infinite server semantics* in which the flow of transition t_j is given by:

$$f_j = \lambda_j \; enab(t_j, m) = \lambda_j \min_{p_i \in {}^\bullet t_j} \left\{ \frac{m_i}{Pre(p_i, t_j)} \right\}.$$

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

$$s = \prod_{t \in T} |{}^\bullet t|,$$

where $|{}^\bullet t|$ denotes the cardinality of the set ${}^\bullet t$, the state space of a timed contPN can be *partitioned*[1] 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* $\Pi_k : T \times P \to \mathbb{R}$ such that:

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

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

If marking m belongs to R_k, we denote $\Pi(m) = \Pi_k$ the corresponding constraint matrix. Furthermore, the firing rate of transitions can also be represented by a diagonal matrix $\Lambda : T \times T \to \mathbb{R}_{>0}$, 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 m can be written as

$$f = \Lambda \cdot \Pi(m) \cdot m.$$

III. A LINEAR DISCRETE-TIME CONSTRAINED MODEL

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 [7].

Definition 3.1: The flow of the forced (or controlled) timed contPN will be denoted by $w(\tau) = f(\tau) - u(\tau)$, where the external control $u(\tau)$ satisfies $0 \leq u(\tau) \leq f(\tau)$. Therefore, the control input will be dynamically upper bounded by the flow of the corresponding unforced system.

The overall behavior of the system is ruled by the following system

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

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

Proposition 3.2: [6] Any piecewise linear constrained model of the form (2) can be rewritten as a *linear constrained model* of the form

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

that we call *continuous time contPN* model (or *ct-contPN* for short) where G is an appropriate constant matrix defined as follows:

$$G = \begin{bmatrix} Q & -R \end{bmatrix}, \quad Q \in \mathbb{Z}^{q \times |T|}, \quad R \in \mathbb{Z}^{q \times |P|}$$

,

$$q = \sum_{t \in T} |{}^\bullet t|,$$

and the row of Q and R relative to the generic pre arc (p_i, t_j) are

$$\begin{bmatrix} \underbrace{0 & \cdots & 0 & 1}_{j} & 0 & \cdots & 0 \end{bmatrix},$$

$$\begin{bmatrix} \underbrace{0 & \cdots & 0 & \dfrac{\lambda_j}{Pre(p_i, t_j)}}_{i} & 0 & \cdots & 0 \end{bmatrix},$$

respectively. The initial value of the state of this system is $m(0) = m_0 \geq 0$.

The system in eq. (3) is a linear system with a *state-matrix* equal to 0 and an *input matrix* equal to the *token flow matrix*
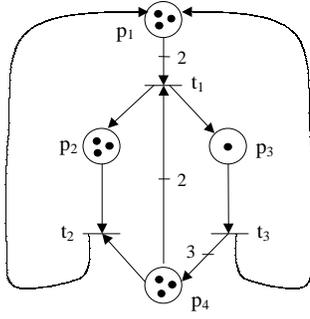
Fig. 1. Continuous PN system.

of the contPN. There is still a dynamic constraint on the system inputs that depends on the value of the system state m. The continuous-time system (3) can be time-discretized, thus obtaining a discrete-time "equivalent" model.

Definition 3.3: 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:

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

The initial value of the state of this system is $\boldsymbol{m}(0) = \boldsymbol{m_0} \geq \boldsymbol{0}$.

Example 3.4: Let us consider the net system in Fig. 1 with $\Theta = 1$ and $\boldsymbol{\lambda} = [1\ 1\ 1]^T$. Then the discrete-time representation is given by:

$$
\begin{cases}
\boldsymbol{m}(k+1) = \boldsymbol{m}(k) + \boldsymbol{C}\boldsymbol{w}(k) \\
w_1(k) - \frac{\lambda_1}{2} \cdot m_1(k) \leq 0 \\
w_1(k) - \frac{\lambda_1}{2} \cdot m_4(k) \leq 0 \\
w_2(k) - \lambda_2 \cdot m_2(k) \leq 0 \\
w_2(k) - \lambda_2 \cdot m_4(k) \leq 0 \\
w_3(k) - \lambda_3 \cdot m_3(k) \leq 0 \\
\boldsymbol{w}(k), \boldsymbol{m}(k+1) \geq \boldsymbol{0}
\end{cases} \tag{5}
$$

and

$$
\boldsymbol{G} = \begin{bmatrix}
1 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} \\
0 & 1 & 0 & 0 & -1 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & 1 & 0 & 0 & -1 & 0
\end{bmatrix} \tag{6}
$$

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 the initial marking $\boldsymbol{m_0}$ is positive, because the flow of a transition goes to zero whenever one of the input places is empty [12].

In a dt-contPN, this is not always true. Let us consider the net in Fig. 1, with

$$
\boldsymbol{m_0} = [0.1\ 1.9\ 1.9\ 0.5\ 0\ 0\ 0.5\ 0\ 0]^T,
$$

$$
\boldsymbol{\lambda} = [5\ 1\ 1\ 1\ 1\ 1]^T,
$$

and $\Theta = 0.5$. Assume transition t_2, t_3, t_4, t_5, t_6 are stopped ($w_2(0) = w_3(0) = w_4(0) = w_5(0) = w_6(0) = 0$), then $m_1(1) = m_1(0) - \Theta \cdot w_1(0) = 0.1 - 0.5 \cdot w_1(0)$. But $w_1(0)$ is upper bounded by $\lambda_1 \cdot m_1(0) = 5 \cdot 0.1 = 0.5$. If the maximum value is chosen, then $m_3(1)$ will be negative!!! So some "spurious solutions" may be added by means of time-discretization!

This can be avoided if the sampling period is small enough [6]. In particular, the following result holds.

Proposition 3.5: [6] Let $\langle \mathcal{N}, \boldsymbol{\lambda}, \boldsymbol{m_0}, \Theta \rangle$ be a dt-contPN system with $\boldsymbol{m_0} \geq \boldsymbol{0}$. Let Θ be the sampling period such that for all $p \in P$ it holds

$$
\sum_{t_j \in p^\bullet} \lambda_j \Theta < 1. \tag{7}
$$

Any marking reachable from $\boldsymbol{m_0}$ is non negative.

IV. IMPLICIT AND EXPLICIT MPC

Steady state optimal control of contPN was studied in [7] and 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 configuration is known and our problem is to reach it (from a given $\boldsymbol{m_0}$) in a finite time by optimizing a given performance index.

Model Predictive Control (MPC) [2], also referred as *moving horizon control* or *receding horizon control*, is a control method that has become an attractive feedback strategy, especially for linear processes. In the last years, many research groups have also worked on MPC of nonlinear systems. In the next sessions we will show how these results can be implemented in the case of contPN under infinite servers semantics.

The basic idea of MPC is the following: at every time step, the control action is chosen solving an optimal control problem, minimizing a performance criterion over a future 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. This is an on-line procedure and in many cases it is difficult (or even impossible) to implement because the on-line solution of a *linear* or *quadratic* program (LP or QP, respectively), depending on the performance index, is required.

Various MPC algorithms use different cost functions to obtain the control action. In this paper we consider the following standard form:

$$
\begin{aligned}
J(\boldsymbol{m}(k)&, \boldsymbol{w}(k), N) = \\
&\{(\boldsymbol{m}(k+N) - \boldsymbol{m}_f)' \cdot \boldsymbol{Z} \cdot (\boldsymbol{m}(k+N) - \boldsymbol{m}_f) \\
&+ \sum_{j=0}^{N-1} [(\boldsymbol{m}(k+j) - \boldsymbol{m}_f)' \cdot \boldsymbol{Q} \cdot (\boldsymbol{m}(k+j) - \boldsymbol{m}_f) + \\
&(\boldsymbol{w}(k+j) - \boldsymbol{w}_f)' \cdot \boldsymbol{R} \cdot (\boldsymbol{w}(k+j) - \boldsymbol{w}_f)]\}
\end{aligned} \tag{8}
$$

where $\boldsymbol{Z}$, $\boldsymbol{Q}$ and $\boldsymbol{R}$ are positive definite matrices.

The constraints for the LP or QP are derived from the dt-contPN definition, and at every step the new marking should respect the set of eq. (4). Thus, at each step the following problem need to be solved:

$$
\begin{aligned}
\min\ & J(\boldsymbol{m}(k), \boldsymbol{w}(k), N) \\
\text{s.t.}:\ & \boldsymbol{m}(k+j+1) = \boldsymbol{m}(k+j) + \Theta \cdot \boldsymbol{C} \cdot \boldsymbol{w}(k+j), \\
& \qquad\qquad\qquad\qquad j = 0, \ldots, N-1 \\
& \boldsymbol{G} \cdot \left[\begin{array}{c} \boldsymbol{w}(k+j) \\ \boldsymbol{m}(k+j) \end{array} \right] \leq 0, j = 0, \ldots, N-1 \\
& \boldsymbol{w}(k+j) \geq 0, j = 0, \ldots, N-1
\end{aligned}
\tag{9}
$$

An alternative to *implicit* MPC has been proposed in [1] by Bemporad *et al.*, where the authors present a technique to compute *off-line* an *explicit* solution of the MPC control problem, based on *multi-parametric linear* programming (mp-LP) or *quadratic* programming (mp-QP). They split the maximum controllable set (i.e., all states that are controllable) into polytopes described by linear inequalities[2] in which the control command is described as a piecewise affine function of the state. Thus, the control law results in a *state feedback* control law.

In [1] 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, and in order to avoid repeating concepts already reported in other papers, we do not provide these results here.

Note that the explicit solution in [1] deals with performance indices of the form (8) where the length N of the prediction horizon may either be finite or infinite: using the standard notation in [1], we denote these cases as *finite time optimal control* (FTOC) and *infinite time optimal control* (ITOC), respectively.

The main advantage of the explicit approach is that the most burdensome part of the procedure is performed off-line, while the on-line part of the procedure simply consists in establishing in which region the current state is. However, its applicability to real size cases is limited by two important facts. Firstly, the computational complexity of the off-line part highly increases with the length of the prediction horizon and with the order of the state space, becoming prohibitive for certain values of these parameters (see the examples in the next section). Moreover, the number of regions highly increases under the same circumstances, constituting a serious limitation to the on-line part of the procedure (because it makes it difficult to establish which control law should be applied).

Our goal in this paper is that of investigating the possibility of using both implicit and explicit MPC to optimally control contPN.

V. Numerical examples

In this section we consider two different numerical examples and make a detailed comparison among the results obtained with the above approaches.

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

The explicit solution has been computed using the Multi-Parametric Toolbox called MPT [5], a free and user-friendly MATLAB toolbox for design, analysis and deployment of optimal controllers for constrained linear and hybrid systems.

Implicit MPC has been implemented using GAMS [3] and MATLAB. MATLAB has been used to write the optimization problem in the required form and to compute the system evolution. GAMS has been used to solve the optimization problem. In particular, the MINOS solver is utilized and the results of the optimization have also been compared with the results of other solvers.

We perform various numerical simulations using different sampling periods, namely $\Theta = 0.01$, 0.05, 0.1, all satisfying the inequality (7), and different values of N.

In order to compare the resulting evolutions, we compute the infinite time horizon index multiplied by Θ, namely

$$
\begin{aligned}
\bar{J}(\boldsymbol{m}(0), \Theta) = \Theta \cdot \sum_{j=0}^{\infty} \ & [(\boldsymbol{m}(j) - \boldsymbol{m}_f)' \cdot \boldsymbol{Q} \cdot (\boldsymbol{m}(j) - \boldsymbol{m}_f) + \\
& (\boldsymbol{w}(j) - \boldsymbol{w}_f)' \cdot \boldsymbol{R} \cdot (\boldsymbol{w}(j) - \boldsymbol{w}_f)]
\end{aligned}
\tag{10}
$$

where $\boldsymbol{Q}$ are $\boldsymbol{R}$ are the same weighting matrices used to compute the MPC controller.

A. First example

Let us consider the net system in Fig. 1 with $\boldsymbol{\lambda} = [1\ 1\ 1]^T$.

Assume that the steady state (final) marking and control input are equal to

$$
\boldsymbol{m}_f = [2.50\ 3.25\ 1.25\ 2.50]^T
$$

and

$$
\boldsymbol{w}_f = [1.25\ 1.25\ 1.25]^T
$$

respectively.

We consider a quadratic performance index of the form (8) where $\boldsymbol{R} = 0.01 \cdot \boldsymbol{I}$, $\boldsymbol{Q} = \boldsymbol{I}$ and $\boldsymbol{Z} = 100 \cdot \boldsymbol{I}$.

1) Implicit MPC: Tables 1 and 2 summarize the results obtained in the case of implicit MPC and initial marking equal to $\boldsymbol{m_0} = [3\ 3\ 1\ 3]^T$ and $\boldsymbol{m_0} = [2\ 4\ 1\ 4]^T$, respectively. Note that the computational time is the average time in [sec] required to solve one QP problem, while $\bar{J}$ is the cost defined as in eq. (10). In this case simulations have been carried out on an Intel Pentium 4 at 3.20 GHz.

Fig. 2 shows the system's evolution in the case of $\boldsymbol{m_0} = [2\ 4\ 1\ 4]^T$, $\Theta = 0.05$ and $N = 1$. As it can be noted the system reaches the desired configuration.

From these, and other results that have not been reported here for sake of brevity, we can draw the following conclusions.

Firstly, the cost $\bar{J}$ is practically the same[3] for all values of N, hence it is not necessary to increase the timing horizon to improve the solution. Note that for sufficiently large values of N this is not surprising. In fact, as well known from classical systems' theory, there exists a $\bar{N}$ such that for any initial state and any $N \geq \bar{N}$, the finite horizon controller is equal to the infinite horizon controller.

[3] Except in the case $\Theta = 0.05$ when passing from $N = 1$ to $N = 2$.
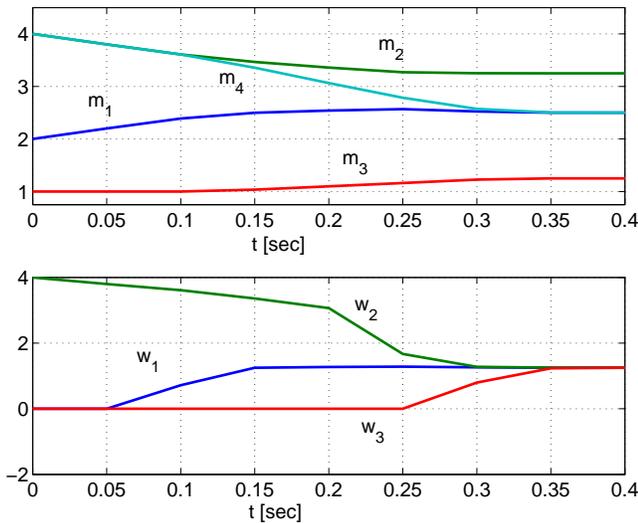
Fig. 2. Marking and flow evolution of the contPN system in Fig. 1 for $\Theta = 0.05$, N=1 and $\boldsymbol{m_0} = [2\ 4\ 1\ 4]^T$.

Fig. 3. State space partition for the net system in Fig. 1 in the case of $N = 1$ and $\Theta = 0.05$.

Fig. 4. State space partition for the net system in Fig. 1 in the case of $N = 2$ and $\Theta = 0.05$.

Secondly, the cost decreases when the sampling period decreases. This can be interpreted in the sense that optimal control is computed more frequently and consequently it is also applied more frequently.

Finally, we observe that, while for $\Theta = 0.1$ all the solutions are practically implementable on this computer, this is no more true in the other cases. In fact, the computational time to solve the QP problem becomes larger than the sampling period if N exceeds certain values. Some improvements can be done in order to reduce the computational times, e.g., rewriting the optimization problem as in [1], but these solutions have not been investigated here.

2) Explicit MPC: The same numerical simulations have also been performed using the explicit approach. As already discussed above, in such a case we need to compute off-line an appropriate state space partition. In Fig. 3 and 4 we have reported the state space partitions relative to the case of $\Theta = 0.05$ and $N = 1$ and 2, respectively.

Two important remarks should be done in order to well interpret these figures. Firstly, we observe that the controller is defined in a two-dimensional space even if the marking of the net is a four-dimensional vector. This is due to the presence of the P-semiflows

$$m_1 + m_2 + m_3 = 7$$

and

$$m_1 + 4m_3 + m_4 = 10$$

that reduce to two the number of state variables that vary independently, i.e., the order of the system.

Secondly, we depict with the same color all those regions to which it correspond the same control law. As it can be noted, it may be the case that the union of these regions may not be a polytope. However, since these regions are defined as the union of a certain number of polytopes this does not increase the complexity of the on-line phase of the

procedure: at each step the controller should determine in which polytope the state lies.

Similar state space partitions have been obtained in the case of $\Theta = 0.01$ and $\Theta = 0.1$.

Table 3 summarizes the main parameters relative to such partitions, namely the time in [sec] necessary to compute them, and the number n_P of polytopes. Note that in this case simulations have been carried out on a Pentium III 450 MHz.

What is important to underline is that using these partitions the resulting controller guarantees stability and constraint satisfaction for all time. Moreover it also covers all controllable states.

In the case of $\Theta = 0.01$ or $\Theta = 0.05$ and $N \geq 3$, as well as in the case of ITOC, we have not been able to compute the explicit MPC controller. Indeed, when computing the state space partition using the toolbox MPT [5] unfeasible solutions are obtained. We get into analogous problems in the case of $\Theta = 0.1$ and $N \geq 6$, as well as in the case of ITOC. Note however that even if for $\Theta = 0.1$ and $N = 3, 4, 5$, we have been able to compute the state space partitions: these

Θ = 0.1			Θ = 0.05			Θ = 0.01		
N	$\bar{J}$	computational time [sec]	N	$\bar{J}$	computational time [sec]	N	$\bar{J}$	computational time [sec]
1	0.0773	0.04	1	0.06578	0.04	1	0.0449	0.04
2	0.0782	0.04	2	0.0584	0.04	2	0.0448	0.04
10	0.0774	0.06	10	0.0581	0.06	10	0.0450	0.05
20	0.0774	0.09	20	0.0581	0.10	20	0.0453	0.10

TABLE I

THE RESULTS OF IMPC APPLIED TO CONTPN SYSTEM IN FIG. 1 WITH $\boldsymbol{m_0} = [3\ 3\ 1\ 3]^T$.

Θ = 0.1			Θ = 0.05			Θ = 0.01		
N	$\bar{J}$	computational time [sec]	N	$\bar{J}$	computational time [sec]	N	$\bar{J}$	computational time [sec]
1	0.4641	0.04	1	0.3785	0.04	1	0.3186	0.04
2	0.4644	0.04	2	0.3797	0.04	2	0.3186	0.04
10	0.4639	0.06	10	0.3782	0.06	10	0.3205	0.05
20	0.4639	0.09	20	0.3782	0.10	20	0.3212	0.09

TABLE II

THE RESULTS OF IMPC APPLIED TO CONTPN SYSTEM IN FIG. 1 WITH $\boldsymbol{m_0} = [2\ 4\ 1\ 4]^T$.
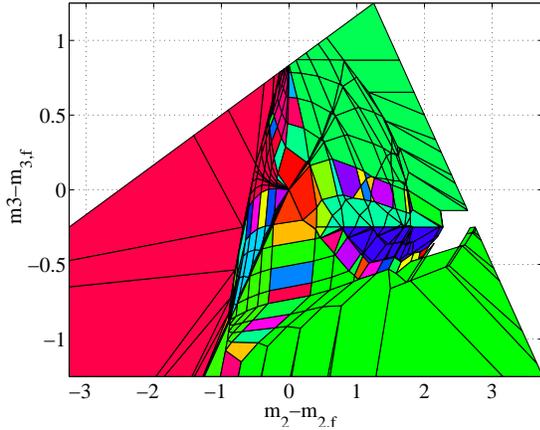
Fig. 5. State space partition for the net system in Fig. 1 in the case of $N = 5$ and $\Theta = 0.1$.

Fig. 6. Timed continuous Marked Graph system.

do not cover the set of controllable states. As an example, in Fig. 5 we have reported the state space partition relative to the case of $\Theta = 0.1$ and $N = 5$.

Also in the case of explicit controller (when applicable, i.e., for $N = 1$ and 2) we have repeated the same numerical simulations as above. Obviously, apart from negligible numerical differences, we get the same results obtained using the implicit controller that have been reported in Tables 1 and 2, and in Fig. 2.

B. Second example

Let us consider the net system in Fig. 6 with $\boldsymbol{\lambda} = [1\ 1\ 1\ 1\ 1\ 1]^T$.

This net system has been extensively studied proving that it has an infinite number of equilibrium points for the same control action (see [7] for details).

Assume the steady state (final) marking and control input

be:

$$\boldsymbol{m}_f = [0.25\ 0.75\ 0.7\ 0.25\ 0.25\ 0.25\ 0.25\ 0.75\ 0.8]^T$$

and

$$\boldsymbol{w}_f = [0.25\ 0.25\ 0.25\ 0.25\ 0.25\ 0.25]^T$$

respectively.

We consider a quadratic performance index of the form (8) where $\boldsymbol{R} = 0.01 \cdot \boldsymbol{I}$, $\boldsymbol{Q} = \boldsymbol{I}$ and $\boldsymbol{Z} = 100 \cdot \boldsymbol{I}$. We perform various numerical simulations using different sampling periods ($\Theta = 0.01$, 0.05, 0.1, all satisfying the inequality (7)) and different values of N. The initial marking has been taken equal to $\boldsymbol{m_0} = [1\ 1\ 1\ 0.5\ 0\ 0\ 0.5\ 0\ 0]^T$.

Analogous considerations as in the previous example can be repeated here in the case of implicit MPC: negligible differences in terms of infinite time cost occur for different values of N; results are better for small values of Θ. Finally, not all these solutions are implementable. As an example, when $\Theta = 0.01$ the time necessary to solve on-line the optimization problem is larger that the sampling period.

We also try to apply the explicit MPC controller, but when computing the state space partition using the toolbox MPT

Θ = 0.1			Θ = 0.05			Θ = 0.01		
N	computational time [sec]	n_P	N	computational time [sec]	n_P	N	computational time [sec]	n_P
1	3.96	24	1	3.96	24	1	3.85	24
2	15.49	86	2	15.49	86	2	13.79	76

TABLE III

THE RESULTS RELATIVE TO THE SPATE SPACE PARTITION IN THE CASE OF THE eMPC APPLIED TO THE CONTPN SYSTEM IN FIG. 1.

Θ = 0.1			Θ = 0.05			Θ = 0.01		
N	Cost	computational time [sec]	N	Cost	computational time [sec]	N	Cost	computational time [sec]
1	2.6630	0.0391	1	2.6055	0.0394	1	2.5602	0.0389
2	2.6634	0.0411	2	2.6062	0.0404	2	2.5602	0.0403
10	2.6630	0.0639	10	2.6056	0.0622	10	2.5604	0.0580
20	2.6630	0.1188	20	2.6056	0.1106	20	2.5604	0.0955

TABLE IV

THE RESULTS OF iMPC APPLIED TO CONTPN SYSTEM IN FIG. 6 WITH $\boldsymbol{m_0} = [1, 1, 1, 0.5, 0, 0, 0.5, 0, 0]^T$.

[5] unfeasible solutions are obtained.

C. A comparison among the two approaches

From numerical simulations carried out we can draw the following conclusions. If the order of the system is low, the explicit MPC is surely the best solution, provided that the FTOC gives satisfactory results also for small values of N. In fact, whenever applicable, the evolution in the implicit and in the explicit case are coincident. However, in the case of explicit MPC the most burdensome part of the procedure is performed off-line, while in the case of implicit MPC we need to solve on-line an optimization problem. Moreover, when applying the explicit MPC controller, the closed loop stability is guaranteed.

On the contrary, in the case of more complex systems, or when large values of N are considered, the computation of the explicit controller may be prohibitive and the implicit MPC is the unique solution. Note that, since in theory, the explicit MPC should be computable for any finite value of N, and thus also for $N = \infty$, this problem may be overcome optimizing the routines for its calculation. We do not address this problem here and we limit to use the toolbox MPT.

VI. 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 optimal control laws based on MPC. In particular, we investigated the possibility of using both implicit and explicit control. The main advantage of the explicit approach is that it provides a state feedback control law and the most burdensome part of the procedure is performed off-line. Nevertheless, when the order of the system is high, or the length of the prediction horizon is too large, the complexity of the explicit controller becomes prohibitive or the results of the simulations, carried out on the MPT toolbox of MATLAB [5], are unfeasible. The same holds in the case of IFOC.

Therefore, we can conclude that at present in many real size cases the implicit MPC is the only solution.

REFERENCES

[1] A. Bemporad, M. Morari, V. Dua, and E.N. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3–20, 2002.

[2] A. Bemporad, F.D. Torrisi, and M. Morari. Performance analysis of piecewise linear systems and model predictive control systems. In *Proc. of the 39th IEEE Conference on Decision and Control*, pages 4957–4962, Sydney, Australia, 2000.

[3] R.F. Boisvert, S.E. Howe, and D.K. Kahaner. Gams: A framework for the management of scientific software. *ACM Transactions on Mathematical Software*, 11(4):313–355, 1985.

[4] R. David and H. Alla. *Discrete, Continuous and Hybrid Petri Nets*. Springer-Verlag, 2004.

[5] M. Kvasnica, P. Grieder, and M. Baotić. Multi-Parametric Toolbox (MPT), 2004.

[6] C. Mahulea, A. Giua, L. Recalde, C. Seatzu, and M. Silva. On sampling continuous timed PNs: reachability "equivalence" under infinite servers semantics. In 2nd *IFAC Conf. on Analysis and Design of Hybrid Systems*, Alghero, Italy, June 2006.

[7] C. Mahulea, A. Ramírez, L. Recalde, and M. Silva. 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, June 2005. J.M. Colom, S. Sreenivas and T. Ushio, eds.

[8] C. Mahulea, L. Recalde, and M. Silva. On performance monotonicity and basic servers semantics of continuous petri nets. In *WODES'06: 8th International Workshop on Discrete Event Systems*, Ann Arbor, USA, July 2006.

[9] L. Recalde and M. Silva. Petri Nets fluidification revisited: Semantics and steady state. *APII-JESA*, 35(4):435–449, 2001.

[10] M. Silva. Introducing Petri nets. In *Practice of Petri Nets in Manufacturing*. Chapman & Hall, 1993.

[11] M. Silva, J. M. Colom, and J. Campos. Linear algebraic techniques for the analysis of Petri nets. In *Recent Advances in Mathematical Theory of Systems, Control, Networks, and Signal Processing II*, pages 35–42. Mita Press, 1992.

[12] M. Silva and L. Recalde. Unforced continuous petri nets and positive systems. In A. De Santis L. Benvenuti and L. Farina, editors, *Positive Systems, Proceedings of the First Multidisciplinary International Symposium on Positive Systems: Theory and Applications (POSTA 2003)*, number 294. Springer–Verlag, 2003.

[13] M. Silva and L. Recalde. On fluidification of Petri net models: from discrete to hybrid and continuous models. *Annual Reviews in Control*, 28(2):253–266, 2004.