5^e*Conférence Francophone de MOdélisation et SIMulation*
"Modélisation et simulation pour l'analyse et l'optimisation des systèmes industriels et logistiques"
MOSIM'04 – du 1^{er} *au* 3 *septembre* 2004 - *Nantes (France)*

CONTROL AND DEADLOCK RECOVERY OF TIMED PETRI NETS USING OBSERVERS

A. Giua[1], C. Seatzu[1] et F. Basile[2]

[1] Dip. Ingegneria Elettrica ed Elettronica
Piazza D'Armi, 09123 Cagliari, Italy
{giua,seatzu}@diee.unica.it

[2] Dip. Ingegneria dell'Informazione e Ingegneria Elettrica
Via Ponte don Melillo 1, 84084 Fisciano (Salerno), Italy
fbasile@unisa.it

ABSTRACT : *In this paper we deal with the problem of controlling a timed Petri net when the marking is not known, but an estimate is computed using an observer. We show that the use of marking estimates may significantly reduce the performance of the closed-loop system and may also lead to a deadlock. We propose a solution to this problem that is based on a linear algebraic characterization of deadlock markings based on siphons analysis. More precisely, two different approaches are suggested depending on the knowledge of the timing structure of the net.*

KEY WORDS: *Petri nets, timed Petri nets, marking estimation, control, deadlock recovery.*

1. INTRODUCTION

In this paper we deal with the problem of controlling a timed Petri net whose marking cannot be measured but is estimated using an observer.

The paper summarizes in an informal manner the main results obtained by the authors, and shows via a numerical example the applicability of these results to a manufacturing system. For a more detailed description of the proposed approach and a comprehensive survey of the literature on this topic we refer to [6, 7].

1.1. Motivation

In the classical approach of Ramadge and Wonham [12] to the supervisory control of discrete event systems, the *event-feedback* control scheme shown in Figure 1.a is adopted. Here the plant spontaneously generates a word of events w. The supervisor observes the word of events generated and, given a set of legal words $\mathcal{K}$, computes at each step a suitable control pattern $f(w)$ to ensure that no illegal word be generated.

Other authors have used a different *state-feedback* control scheme, shown in Figure 1.b. Here the supervisor observes the actual plant state M and, given a set of legal states $\mathcal{L}$ computes at each step a control pattern $f(M)$ to ensure that no illegal state be reached. This scheme is particularly appealing when dealing with Petri net models of the plant [8], since the state of a net is given by an integer vector called *marking* (this explains the notation M used for the plant state in the figure) and linear algebraic techniques may be used to solve the control problem.

A slightly different scheme is shown in Figure 1.c. Here the controller observes the word of events generated and, by means of an observer, it reconstructs the actual plant state M. The observer simply duplicates the plant model, and is driven by the observed events. If the structure (assumed deterministic) and the initial state M_0 of the plant are known, the knowledge of the word generated is sufficient to reconstruct the new state that each new firing yields.

When the initial state is not completely specified a different control scheme may be used. In particular, we use Petri net models and assume that the initial marking M_0 is known to belong to a "macromarking", i.e., we know the token contents of subsets of places but not the exact token distribution. In this case we can use the control scheme shown in Figure 1.d where $\mathcal{C}(w)$ denotes the set of markings in which the system may be given the observed word w and the partial information on the initial marking. In the following we call $\mathcal{C}(w)$ the set of markings *consistent* with an observed word w.

1.2. Proposed approach

In this paper we first recall the main results in [6] where we have shown how it is possible to design a marking observer, i.e., a system that estimates the actual marking of

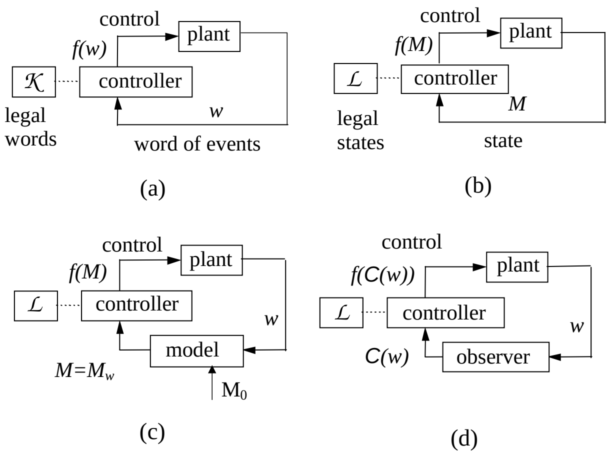

Figure 1: *Different control schemes. (a) Event–feedback. (b) State–feedback. (c) State–feedback with event observer and initial marking. (d) State–feedback with event observer and initial macromarking.*

the net based on the observation of a word of events w. In particular, the special structure of Petri nets allows us to use a simple linear algebraic formalism for estimate computation. Moreover, the set $\mathcal{C}(w)$ can easily be described in terms of the observer estimate and can be characterized as the integer solutions of a linear constraint set.

In [6] we have also shown how the estimate generated by the observer may be used to design a state feedback controller, that ensures that the controlled system never enters a set of forbidden states. We considered a special class of safety specifications that limit the weighted sum of markings in subsets of places called generalized mutual exclusion constraints (GMEC).

Clearly, the use of marking estimates, as opposed to the exact knowledge of the actual marking of the plant, leads to a worse performance of the closed-loop system. In fact, in a safety problem the aim of the controller is that of preventing all those transition firings that lead to a forbidden marking. If the actual marking is not exactly known, but is only known to belong to a given consistent set $\mathcal{C}$, the controller must forbid all transitions firing that from ”any” marking in $\mathcal{C}$ may lead to a forbidden marking and the controller becomes usually more restrictive as the cardinality of this set increases. Because of this it may be the case that the controlled system reaches a deadlock, i.e., a blocking condition, even if it is deadlock free when perfect information about the marking is available.

In [7] we shown that, using siphon analysis, the set of deadlock markings $\mathcal{M}_b$ of a structurally bounded net can be characterized as the integer solution of a linear constraint set. Siphon analysis has been already used by several authors to derive deadlock avoidance policies: see [1, 2, 3, 11]. The approach we propose in [7] is different from the above mentioned approaches in two ways. Firstly, our approach only aims to give a characterization of deadlock markings. On the contrary, the referenced approaches aim to solve a more complex problem, namely that of deriving a deadlock avoidance policy: to do this it necessary to also characterize *impending deadlock markings*, i.e., markings that are not dead but that will lead to a deadlock in a finite number of steps. Secondly, since we solve a less complex problem we are able to derive a simpler (in terms of number of constraints and number of unknowns) characterization that applies to a large class of nets (ordinary and structurally bounded), while the referenced approaches are only valid for restricted classes of nets.

Then, we focus our attention to timed Petri nets, i.e., Petri nets where a delay is associated to each transition. The delay represents the time that must elapse from the enabling of the transition until it fires. In particular, in [7] we considered two different cases.

We initially assume that a very loose information on the timing structure is available. More precisely, we assume that if no transition firing occurs within a reasonable amount of time in a controlled system — we say that the *net has timed out* — one can conclude that a deadlock has occurred and a recovery procedure should be invoked. The characterization based on siphon analysis may be used to derive a recovery procedure from deadlocks induced by the observer and to improve the marking estimate, thus providing a better characterization of the set of consistent markings.

Then, we consider the case in which the timing structure is known and propose a control algorithm that uses the previous marking estimate and control approach, but that also takes into account the knowledge of the delays and of the enabling status of each transition. This algorithm should be invoked whenever a transition has not fired for a time larger than its expected delay, i.e., when a *transition has timed out*. Thus it not only allows the supervisor to recover from total deadlocks (as in the previous case) but it allows one to detect partial deadlocks as well, and in general it improves and accelerates the convergence of the marking estimation procedure. We also show how the observer can use this information to restrict the set of consistent markings.

A drawback of the proposed approach is that it requires looking at each step for admissible solutions of a certain number of constraint sets whose variables are integer: in some cases this may hinder the implementation of the approach on on-line controllers. We show that this problem may be partially solved by simply relaxing the integer constraints we consider into linear ones.

2. BACKGROUND ON PETRI NETS

In this section we recall the formalism used in the paper. For more details on Petri nets we address to [10].

A *Place/Transition net* (P/T net) is a structure $N = (P, T, Pre, Post)$, where P is a set of m places; T is a set of n transitions; $Pre : P \times T \to \mathbb{N}$ and $Post : P \times T \to \mathbb{N}$ are the pre– and $post$– incidence functions that specify the arcs; $C = Post - Pre$ is the incidence matrix. The *preset* and *postset* of a node $X \in P \cup T$ are denoted $^\bullet X$ and $X^\bullet$ while $^\bullet X^\bullet =^\bullet X \cup X^\bullet$.

A *marking* is a vector $M : P \to \mathbb{N}$ that assigns to each place of a P/T net a non–negative integer number of tokens, represented by black dots. In the following we denote $M(p)$ the marking of place p. A *net system* $\langle N, M_0 \rangle$ is a net N with an initial marking M_0.

A transition t is *marking enabled* at M if $M \geq Pre(\cdot, t)$. In this paper we also assume that a *supervisor*, i.e., an external control agent, may forbid the occurrence of a transition specifying a marking dependent control pattern $f(t, M) : T \times \mathbb{N}^m \to \{0, 1\}$ such that $f(t, M) = 1$ if t is *control enabled*, $f(t, M) = 0$ if t is *control disabled*.

A transition t is *enabled* at M if it is marking enabled and control enabled. A transition t enabled at M may *fire*, yielding the marking $M' = M + C(\cdot, t)$.

We write $M\ [w\rangle\ M'$ to denote that the enabled sequence of transitions w may fire at M yielding M', or equivalently we use the notation $M' = w(M)$ and $M = w^{-1}(M')$. Moreover, we denote $w(M_0) = M_w$. Finally, we denote ε the sequence of null length. The set of all sequences firable in $\langle N, M_0 \rangle$ is denoted $L(N, M_0)$ (this is also called the prefix-closed free language of the net). If the firing sequence w is enabled at M_0, we also say that w is a word in $L(N, M_0)$.

A marking M is *reachable* in $\langle N, M_0 \rangle$ iff there exists a firing sequence w such that $M_0\ [w\rangle\ M$. The set of all markings reachable from M_0 defines the *reachability set* of $\langle N, M_0 \rangle$ and is denoted $R(N, M_0)$.

A nonnegative integer vector $\vec{x} \neq \vec{0}_m$ such that $\vec{x}^{\,T} \cdot C = \vec{0}_n^{\ T}$ is called a *P–invariant* (here $\vec{0}_k$ denotes a $k \times 1$ vector of zeros). A P-invariant is *minimal* if there does not exist a P-invariant $\vec{y}$ such that $\vec{y} \leq \vec{x}$.

A transition t is said to be *live* if for any $M \in R(N, M_0)$, there exists a sequence of transitions firable from M which contains t. A Petri net is said to be live if all transitions are *live*.

A marking M is a *deadlock* (or *dead*) marking if no transition $t \in T$ may fire at M. A Petri net is said to be *deadlock–free* if at least one transition is enabled at every reachable marking.

A place p is said to be *bounded* if there exists a constant k such that $M(p) \leq k$ for all $M \in R(N, M_0)$. A net system is bounded if all places are bounded. A net is *structurally bounded* if it is bounded for all initial markings.

A P/T net is called *ordinary* when all of its arc weights are 1's. A *siphon* of an ordinary net is a *non–empty* set of places $\mathcal{S} \subseteq P$ such that: $\bigcup_{p \in \mathcal{S}} {^\bullet p} \subseteq \bigcup_{p \in \mathcal{S}} p^\bullet$. A siphon is *minimal* if it is not the superset of any other siphon. In the following we denote as $\vec{s} \in \{0, 1\}^m$ the characteristic vector of $\mathcal{S}$, where $s_i = 1$ if place $p_i \in \mathcal{S}$ and $s_i = 0$ otherwise.

Definition 1. Given a net $N = (P, T, Pre, Post)$, and a subset $T' \subseteq T$ of its transitions, we define the T'*–induced subnet of* N as the new net $N' = (P, T', Pre', Post')$ where $Pre', Post'$ are the restriction of $Pre, Post$ to T'. The net N' can also be thought as obtained from N by removing all transitions in $T \setminus T'$. We also write $N' \prec_{T'} N$. ■

A deterministic *timed* P/T net is a pair (N, δ), where $N = (P, T, Pre, Post)$ is a standard P/T net, and $\delta(t) : T \to \mathbb{R}_0^+$, called release delay, assigns a non-negative fixed firing duration to each transition. A transition with a release delay equal to 0 is said to be immediate. The value of $\delta(t)$ represents the time that must elapse, starting from the time at which the transition t is enabled, until it fires. We use single server-semantics, i.e., no concurrent firings of the same transition are possible.

Finally, we conclude this section recalling a linear algebraic characterization of deadlock markings derived by the authors in [7] that will be used in the paper. Such a characterization is valid for ordinary and structurally bounded Petri nets. Note that, as discussed in the Introduction, similar linear characterizations have been independently proposed in [1, 2, 11].

Theorem 2 ([7]). Given a structurally bounded net N with m places, a marking $M \in \mathbb{N}^m$ is a deadlock marking if and only if there exists a vector $\vec{s} \in \{0, 1\}^m$ such that the following set of linear equations is satisfied:

$$\mathcal{D}(N) := \begin{cases} K_1 \cdot Pre^T \cdot \vec{s} \geq Post^T \cdot \vec{s} & (a) \\ K_2 \cdot \vec{s} + M \leq K_2 \cdot \vec{1}_m & (b) \\ \vec{s} + M \geq \vec{1}_m & (c) \\ Pre^T \cdot \vec{s} \geq \vec{1} & (d) \\ M \in \mathbb{N}^m & (e) \\ \vec{s} \in \{0, 1\}^m & (f) \end{cases} \qquad (1)$$

where $K_1 = \max_{t \in T} Post^T(\cdot, t) \cdot \vec{1}$ and K_2 is any positive integer greater or equal to the maximum structural bound of p, for any $p \in P$. ■

By virtue of the linear characterization above, we define the set of blocking markings of a net N as:

$$\mathcal{M}_b(N) = \{M \mid \exists\, \vec{s} \in \{0, 1\}^m : (M, \vec{s}) \in \mathcal{D}(N)\}. \quad (2)$$

3. MARKING ESTIMATION WITH MACROMARKINGS

In [6] we dealt with the problem of reconstructing the marking of a P/T net assuming that partial information about the initial marking is available in the form of a *macromarking*.

Definition 3 (Macromarking). Assume that the set of places P can be written as the union of $r + 1$ subsets:

$P = P_0 \cup P_1 \cup \cdots \cup P_r$ such that $P_0 \cap P_j = \emptyset$, for all $j > 0$. The number of tokens contained in P_j ($j > 0$) is known to be b_j, while the number of tokens in P_0 is unknown. For each P_j, let $\vec{v}_j$ be its characteristic vector, i.e., $v_j(p) = 1$ if $p \in P_j$, else $v_j(p) = 0$.

We call the set of markings

$$\mathcal{V}(V, \vec{b}) = \{M \in \mathbb{N}^m \mid V^T M = \vec{b}\}$$

the *macromarking defined by* $V = [\vec{v}_1, \cdots, \vec{v}_r]$ *and* $\vec{b} = [b_1, \cdots, b_r]$. ■

The notion of macromarking occurs frequently when describing systems containing a known set of resources (e.g., parts, machines) whose actual conditions (e.g., exact location of parts within the plant, state of a machine) is unknown.

We make the following assumptions.

(A1) The structure of the net $N = (P, T, Pre, Post)$ is known, while the initial marking M_0 is not.

(A2) The event occurrences (i.e., the transition firings) can be observed.

(A3) The initial marking M_0 belongs to the macromarking $\mathcal{V}(V, \vec{b})$, i.e., it satisfies the equation $V^T M_0 = \vec{b}$.

We also introduce the following notation.

Definition 4 (Set of w-consistent markings). After the word w has been observed we define the set of $w-$consistent markings as the set of all markings in which the system may be given the observed behavior and the initial marking, i.e., the set $\mathcal{C}(w) = \{M \in \mathbb{N}^m \mid \exists M_0 \in \mathcal{V}(V, \vec{b}),\ M_0[w\rangle M\}$. ■

In [6] we provided a simple algorithm to compute the estimate μ and the bound B of each actual marking M based on the observation of a word of events and on the knowledge of the initial macromarking $\mathcal{V}(V, \vec{b})$.

Algorithm 5. (Marking Estimation with Event Observation and Initial Macromarking).

```
1. Let the current observed word be
   $w = \varepsilon$ (the empty string).
2. Let the initial estimate be $\mu_\varepsilon$,
   with $\mu_\varepsilon(p) = \min\{M(p) \mid V^T \cdot M = \vec{b}\}$.
3. Let the initial bound be $B_\varepsilon = \vec{b} - V^T \cdot \mu_\varepsilon$.
4. Wait until a new transition, say $t$,
   fires.
5. Update the estimate $\mu_w$ to $\mu'_{wt}$ with
   $\mu'_{wt}(p) = \max\{\mu_w(p), Pre(p,t)\}$.
6. Let $\mu_{wt} = \mu'_{wt} + C(\cdot, t)$.
7. Let $B_{wt} = B_w - V^T \cdot (\mu'_{wt} - \mu_w)$.
8. Goto 4.                                  ■
```

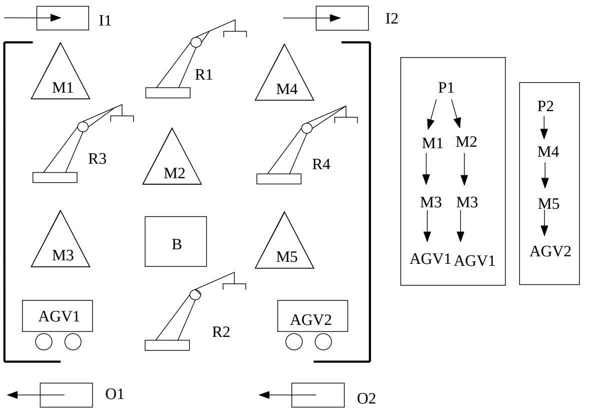

Figure 2: *Layout of the automated manufacturing system.*

In simple words, if the currently observed word is w and transition t fires, the algorithm firstly updates the current estimate from μ_w to μ'_{wt} adding the minimal number of tokens required to enable t. Secondly, the algorithm computes μ_{wt} as the marking obtained from μ'_{wt} firing t.

Definition 6. Given an estimate μ and a bound B computed using Algorithm 5, the set of (μ, B)*-consistent markings* is

$$\mathcal{M}(\mu, B) \overset{\text{def}}{=} \{ \quad M \in \mathbb{N}^n \mid M \geq \mu, \\ V^T \cdot M = V^T \cdot \mu + B\}. \tag{3}$$

■

The following important result provides a linear algebraic characterization of $\mathcal{C}(w)$.

Theorem 7 ([6]). Let us consider a net with initial macromarking $\mathcal{V}(V, \vec{b})$. Let w be an observed word, and μ_w and B_w be the corresponding estimate and bound computed using the estimation Algorithm 5.

The set of w-consistent markings coincides with the set of (μ_w, B_w)-consistent markings, i.e.,

$$\mathcal{C}(w) = \mathcal{M}(\mu_w, B_w).$$

■

3.1. A manufacturing example

We now apply the above methodology to a classical automated manufacturing system whose layout is shown in Figure 2 and whose Petri net model is shown in Figure 3 (places $C1$, $C2$, $C3$ and all connected arcs should be ignored at first). This system is similar to the one described in [4].

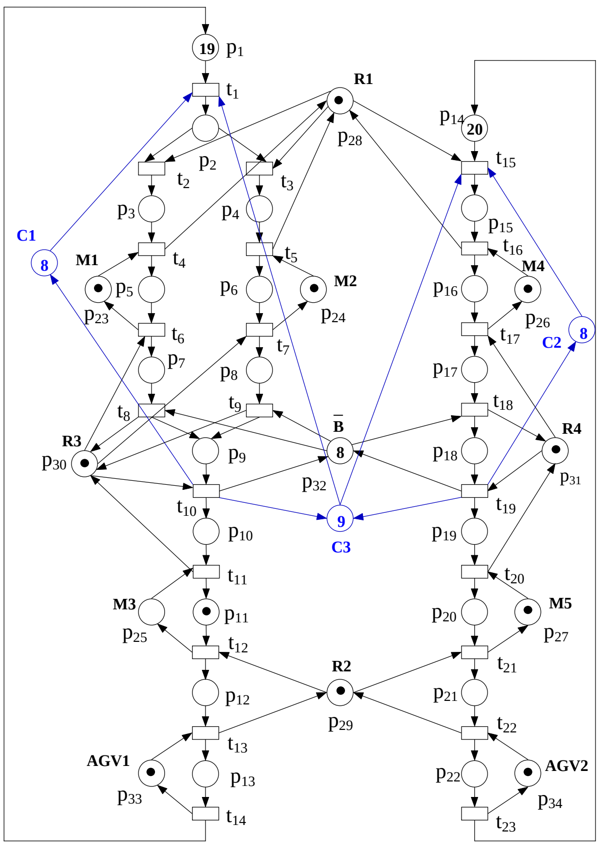

Figure 3: *Petri net model of the manufacturing system in Figure 2.*

The plant consists of five machines (M1 to M5), four robots (R1 to R4), a finite capacity buffer B, two inputs of raw parts (I1 and I2) of type1 and type2 respectively, two AGV systems (AGV1 and AGV2), and finally two outputs (O1 and O2) for the processed parts. The plant produces two different types of products from two types of raw materials. An unlimited source of raw parts is assumed. It is supposed that there are 20 pallets for each type of product.

The Petri net model is shown in Figure 3. This net has $m = 34$ places and $n = 23$ transitions. The marking of place p_{32}, the co-buffer, represents the number of free buffer slots, while the marking of places p_9 and p_{18} represent respectively the number of type1 and type2 parts present in the buffer. There exist 14 circuits, each corresponding to a P-invariant. If we assume that the initial marking of the net is that shown in Figure 3, we have (here to avoid a heavy notation we denote as M_i the marking of place p_i)

$$\left\{\begin{array}{l}
\sum_{i=1}^{13} M_i = 20 \\
\sum_{i=14}^{22} M_i = 20 \\
M_5 + M_{23} = 1 \\
M_6 + M_{24} = 1 \\
M_{11} + M_{25} = 1 \\
M_{16} + M_{26} = 1 \\
M_{20} + M_{27} = 1 \\
M_{13} + M_{33} = 1 \\
M_{22} + M_{34} = 1 \\
M_3 + M_4 + M_{15} + M_{28} = 1 \\
M_{12} + M_{21} + M_{29} = 1 \\
M_7 + M_8 + M_{10} + M_{30} = 1 \\
M_{17} + M_{19} + M_{31} = 1 \\
M_9 + M_{18} + M_{32} = 8
\end{array}\right.$$

We assume that the above set of P-invariants coincides with the macromarking, thus

$$\vec{b} = [20\ 14\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 8]^T.$$

Note that if the number of P–invariants is too high to be taken into account, we can only ”keep a subset of it”.

Now, assume that the initial marking is that shown in Figure 3, namely,

$$\begin{array}{ll}
M_0(p_1) = 19 & \\
M_0(p_{14}) = 14 & \\
M_0(p_{32}) = 8 & \\
M_0(p_i) = 0 & i = 2, \ldots, 10, 12, 13, 15, \ldots, 22, 25 \\
M_0(p_i) = 1 & i = 11, 23, 24, 26, \ldots, 31, 33, 34
\end{array}$$

In accordance to Step 2 of Algorithm 5, the initial value of the estimate is

$$\begin{array}{ll}
\mu_\varepsilon(p_{14}) = 6 & \\
\mu_\varepsilon(p_i) = 0 & i \neq 14
\end{array}$$

while the initial bound is

$$B_\varepsilon = [20\ 14\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 8]^T.$$

Assume that transition t_{12} fires, i.e., the observed word is $w = t_{12}$.

In accordance to Step 5 of Algorithm 5 the previous marking estimate is firstly updated to μ'_ε where

$$\begin{array}{ll}
\mu'_\varepsilon(p_{14}) = 6 & \\
\mu'_\varepsilon(p_i) = 1 & i = 11, 29 \\
\mu'_\varepsilon(p_i) = 0 & i \neq 11, 14, 29
\end{array}$$

In simple words this means that if transition t_{12} has fired, then at least one token should be contained in its input places p_{11} and p_{29} before its firing.

Then, (see Steps 7 and 8 of Algorithm 5) the marking estimate μ_w and bound B_w are computed, where

$$\begin{array}{ll}
\mu_w(p_{14}) = 6 & \\
\mu_w(p_i) = 1 & i = 12, 25 \\
\mu_w(p_i) = 0 & i \neq 12, 14, 25
\end{array}$$

$$B_w = [19\ 14\ 1\ 1\ 0\ 1\ 1\ 1\ 1\ 1\ 0\ 1\ 1\ 8]^T.$$

Now, assume that transition t_{13} fires. In accordance to Step 5 of Algorithm 5 the previous marking estimate is updated to μ'_w. The only difference among μ_w and μ'_w is in place p_{33}: we know for sure that if transition t_{13} has fired, at least one token was contained in p_{12} and p_{33} before its firing, but the presence of the token in p_{12} has already been detected after the firing of t_{12}.

Finally, (see Steps 7 and 8 of Algorithm 5) the marking estimate μ_w and bound B_w are computed, with $w = t_{12}t_{13}$,

$$\begin{array}{ll} \mu_w(p_{14}) = 6 & \\ \mu_w(p_i) = 1 & i = 13, 29 \\ \mu_w(p_i) = 0 & i \neq 13, 14, 29 \end{array}$$

$$B_w = [19\ 14\ 1\ 1\ 0\ 1\ 1\ 1\ 1\ 1\ 0\ 1\ 1\ 8]^T.$$

4. CONTROL USING OBSERVERS

In this section we show how the marking estimate constructed with the formalism discussed in the previous section can be used by a control agent to enforce a given specification on the plant behavior.

We make several assumptions that are briefly discussed here.

- We assume that the specification on the desired behavior is given as a set of legal markings $\mathcal{L}$.
- We consider a special type of state specifications called *generalized mutual exclusion constraints* (GMEC) that have been considered by various authors [5, 9, 13].

  Given an integer matrix $L = [\vec{l}_1 \cdots \vec{l}_q]$ with $\vec{l}_j \in \mathbb{Z}^m$ and a vector $\vec{k} = [k_1, \cdots, k_q]$ with $k_j \in \mathbb{Z}$, a GMEC $(L, \vec{k})$ defines the set of legal markings $\mathcal{L} = \{M \in \mathbb{N}^m \mid L^T \cdot M \leq \vec{k}\}$.
- The controller may disable transitions to prevent the plant from entering a forbidden marking, computing an appropriate control pattern. If the actual marking M is known, the control pattern is a function of M. However, when an observer is used in the control loop, only the set of consistent markings $\mathcal{C} \subseteq \mathbb{N}^m$ is available to the controller and the control pattern becomes a function $f(t, \mathcal{C}) : T \times 2^{\mathbb{N}^m} \to \{0, 1\}$. If $f(t, \mathcal{C}) = 0$ then t is disabled by the controller, while if $f(t, \mathcal{C}) = 1$ it is enabled.
- All transitions are controllable, i.e., can be disabled by the controller.

Thus the considered control scheme is that shown in Figure 1.d.

The control law $f(t, \mathcal{C})$ is defined as follows.

Definition 8 ([7]). Given a GMEC $(L, \vec{k})$ and a set of consistent markings $\mathcal{C} \subseteq \mathbb{N}^m$, the firing of transition t should be prevented if and only if there exists a legal consistent marking M such that the firing of t from M leads to a forbidden marking, i.e.,

$$f(t, \mathcal{C}) = \begin{cases} 0 & \text{if } (\exists M)\ M \in \mathcal{C},\ L^T \cdot M \leq \vec{k}, \\ & \quad M[t\rangle M',\ (\exists j)\ \vec{l}_j \cdot M' > k_j \\ 1 & \text{otherwise.} \end{cases}$$

■

The computation of the control pattern for a given t may be carried out looking for the existence of a $j \in \{1, \ldots, q\}$ such that the following constraint set admits a solution:

$$\begin{cases} M \in \mathcal{C} & (a) \\ L^T \cdot M \leq \vec{k} & (b) \\ M \geq Pre(\cdot, t) & (c) \\ M' = M + C(\cdot, t) & (d) \\ \vec{l}_j^T \cdot M' > k_j & (e) \\ M' \in \mathbb{N}^m & (f) \end{cases} \qquad (4)$$

If system (4) admits a solution, there exists a consistent marking M – constraint (a) – that is legal – constraint (b) – from which transition t may fire – constraint (c) – yielding a marking M' – constraint (d) – that is not legal – constraint (e) – because it holds $\vec{l}_j^T \cdot M' > k_j$. Note that, as a consequence of Theorem 7, constraint (a) is linear with respect to M.

Note that the control pattern computed using an observer may be more restrictive than the optimal state feedback computed when the actual marking is known [7]. Moreover, as shown in the following example, this may also lead to a block.

4.1. A manufacturing example (continued)

Let us consider again the manufacturing example discussed in Subsection 3.1.

To show how this net evolves in time under control let us assume the following timing structure is given: $\delta(t) = 5$ for all $t \in T \setminus \{t_{12}, t_{13}, t_{14}, t_{21}, t_{22}, t_{23}\}$, $\delta(t) = 1$ for $t \in \{t_{13}, t_{14}, t_{22}, t_{23}\}$, and $\delta(t) = 2$ for $t \in \{t_{12}, t_{21}\}$.

In this example the controller must enforce three specifications:

$$\begin{cases} \sum_{i=2}^{9} M_i \leq 8 & (a) \\ \sum_{i=15}^{18} M_i \leq 8 & (b) \\ \sum_{i=2}^{9} + \sum_{i=15}^{18} M_i \leq 9 & (c) \end{cases} \qquad (5)$$

Note that if the initial marking is completely known, the addition of the monitor places $C1$, $C2$ and $C3$ ensures the satisfaction of the linear inequality constraints (5) and the closed loop net is live (this may be easily proved following the procedure in [3]).

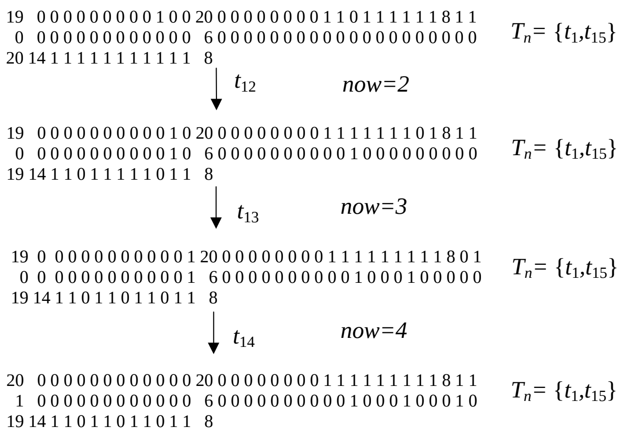

Figure 4: *Reachability graph of the net in Figure 3 under control when no deadlock recovery procedure is applied.*

On the contrary, if the marking of the plant is not measurable, an observer must be used in the control loop and this leads to a deadlock. The closed loop behaviour is that shown in Figure 4 where the first line of each node contains the real marking of the net, the second line contains the actual estimate and the third line contains the actual bound. Note that in the same figure the variable now denotes the actual value of time and, for each marking, the set $T_n = \{t \in T \mid f(t, \mathcal{C}) = 0\}$ is the set of transitions disabled by the controller.

After the sequence $w = t_{12}t_{13}t_{14}$ has fired, only two transitions t_1 and t_{15} are enabled in the net. The controller prevents the firing of both transitions even if their firing is perfectly legal and the net reaches a deadlock. This is due to the fact that there exists at least one marking in $\mathcal{C}(t_{12}t_{13}t_{14})$ that would produce the violation of one of the controller specifications if either transition t_1 or t_{15} fires. In particular, the firing of t_1 may (potentially) violate specifications (a-c), while the firing of t_{15} may violate specifications (b-c).

5. RECOVERY AND ESTIMATE UPDATE AFTER NET TIME-OUT

Let us suppose that, although we have no exact information on the timing structure of the net, we can be sure that the net is blocked if a sufficiently long time has elapsed without observing any event occurrence. Such is the case if we know that all transition delays are such that $\delta(t) \leq \Delta_{\max}, \forall t \in T$. If a time greater than $\Delta_{\max}$ elapses without observing any firing, we say that *the net has timed out*.

Proposition 9 ([7]). Assume that the net $N = (P, T, Pre, Post)$ controlled with the control pattern $f(\cdot, \mathcal{C})$ has timed out. Let us define $T' = \{t \in T \mid f(t, \mathcal{C}) = 1\}$ as the subset of T containing the transitions enabled by the controller, and let $N' \prec_{T'} N$ be the $T'-$induced subnet of N. Then the actual (unknown) marking M of the controlled net N is a deadlock marking for the uncontrolled net N', i.e., it belongs to $\mathcal{C}' = \mathcal{C} \cap \mathcal{M}_b(N')$. ■

In [7] we proposed an automatic approach that tries to exploit the information that the net has timed out to recover from this blocking condition and improve the estimate. Of course this procedure may be effective only if the deadlock has been caused by the incomplete information about the actual marking originated by the presence of the observer in the closed loop.

5.1. Deadlock recovery

The deadlock recovery procedure we proposed in [7] consists in recomputing the control pattern using appropriate constraints to capture the fact that the actual (unknown) marking M belongs to $\mathcal{M}_b(N')$ for the net N' defined in Proposition 9.

Algorithm 10. (Control Pattern Updating After Net Time-Out)

Given a net $N = (P, T, Pre, Post)$ controlled using an observer, let μ and B be the current value of estimate and bound, and define $\mathcal{C} = \mathcal{M}(\mu, B)$. Assume that the computed control pattern $f(\cdot, \mathcal{C})$ has led the net to a time-out. We can update the control pattern using the following procedure.

```
1. Let $i = 0$ and define $f_0(\cdot) \overset{\text{def}}{=} f(\cdot, \mathcal{C})$ as
   the initial control pattern.
2. Let $T_i = \{t \in T \mid f_i(t) = 1\}$ be the set of
   transitions enabled by the
   current control pattern, and let
   $N_i \prec_{T_i} N$ be the net obtained by $N$
   removing all transitions not in $T_i$.
3. Update the control pattern to $f_{i+1} =$
   $g(f_i)$, where
       $$g(f_i) \overset{\text{def}}{=} f(\cdot, \mathcal{C} \cap \mathcal{M}_b(N_i)). \qquad (6)$$
4. If $f_{i+1} = f_i$ THEN exit:  the deadlock
   recovery procedure has failed.
5. Wait until
   (a) EITHER a transition fires and
       THEN exit:  the net has recovered
       from the deadlock
   (b) OR a new net time-out occurs
       and THEN let $i = i + 1$ and go to
       2.                                  ■
```

Note that the operator $g : \{0,1\}^n \to \{0,1\}^n$ defined by (6) is a function of f_i because N_i is defined using f_i.

In this algorithm the knowledge that a time-out has occurred is used to restrict the set of consistent markings and construct a new control pattern (step 3) that is at least as permissive as the previous one [7]. If the new control pattern is still blocking and a new time-out occurs the procedure is repeated until either the net recovers from deadlock, or until we cannot update the control pattern any more and the procedure fails.

Note that in [7] we proved that Algorithm 10 always terminates in a finite number of steps.

In [7] we also provided an important characterization of those cases in which the proposed procedure is able to recover from a net time-out. More precisely, when the macromarking is such that the vectors $\vec{v}_j$ are P-invariants, we give a sufficient condition to ensure that the controlled net will never time out. Moreover, we give a sufficient condition to ensure that, even if a time-out may occur, Algorithm 10 will always successfully recover the net from a deadlock.

5.2. Improving the marking estimate

In this subsection, we discuss the possibility of using the linear algebraic characterization above not only to recover from a block, but to improve the marking estimate as well.

Assume that given an observed word w, a current estimate μ_w and bound B_w, a blocking condition occurs, and that after $\bar{\imath}$ iterations of Algorithm 10 a newly enabled transition t fires. At this point, before the firing of t, the set of consistent markings is $\mathcal{M}(\mu_w, B_w) \cap \mathcal{M}_b(N_{\bar{\imath}})$. This set corresponds to the dark area in Figure 5.

We should keep this information when computing the new set of consistent markings $\mathcal{C}(wt)$ after the firing of t. Nevertheless, this would destroy the framework that inspired the algorithm for the marking estimate computation [6], in the sense that the set of consistent markings would loose the structure given in Equation (3). Thus, we propose the following alternative solution. For each place $p_i \in P$ we solve an integer programming problem (IPP) of the form:

$$\begin{cases} \min M(p_i) \\ s.t. \\ M \in \mathcal{M}(\mu_w, B_w) \\ M \in \mathcal{M}_b(N_{\bar{\imath}}) \end{cases} \tag{7}$$

Now, we define $\mu^* = [\mu_1^* \ \cdots \ \mu_m^*]^T$ where μ_i^* is the solution of the i–th IPP and let $B^* = B_w - V^T(\mu^* - \mu_w)$ be the corresponding bound. We use μ^* and B^* as new current values of the estimate μ_w and bound B_w. This is equivalent to approximate the set of $w-$consistent markings after recovery, with the set

$$\mathcal{M}(\mu^*, B^*) = \{ \quad M \in \mathbb{N}^m \mid M \geq \mu^*, \\ V^T \cdot M = V^T \cdot \mu^* + B^* \}. \tag{8}$$

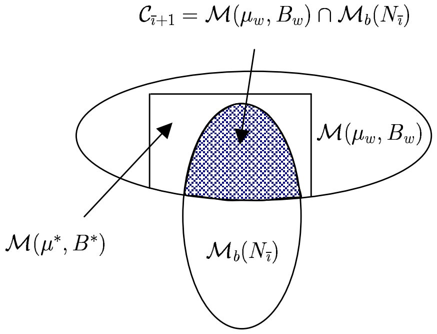
$\mathcal{C}_{\bar{\imath}+1} = \mathcal{M}(\mu_w, B_w) \cap \mathcal{M}_b(N_{\bar{\imath}})$
$\mathcal{M}(\mu_w, B_w)$
$\mathcal{M}(\mu^*, B^*)$
$\mathcal{M}_b(N_{\bar{\imath}})$

Figure 5: *Generic inclusion relationship among sets $\mathcal{M}(\mu_w, B_w)$, $\mathcal{M}(\mu^*, B^*)$ and $\mathcal{M}_b(N_{\bar{\imath}})$.*

This set is also shown in Figure 5: being $\mathcal{M}(\mu_w, B_w) \cap \mathcal{M}_b(N_{\bar{\imath}}) \subseteq \mathcal{M}(\mu^*, B^*) \subseteq \mathcal{M}(\mu_w, B_w)$ we may be losing information, but nevertheless we can keep on with a linear algebraic characterization of the set of consistent markings in the simple form specified by Equation (3).

5.3. Numerical example

Let us consider again the manufacturing system in Subsection 4.1, where the use of an observer in the closed loop may lead to a blocking condition.

In this subsection we show how the above deadlock procedure may be efficiently applied to the considered net. If we assume that the initial marking is that in Figure 3 a blocking condition occurs after the firing of the sequence $w = t_{12}t_{13}t_{14}$. The corresponding value of the marking M_w, as well as that of the estimate μ_w and bound B_w, may be seen in Figure 4.

At this point, when a time $\Delta_{\max}$ striclty greater than the maximum timing delay has elapsed ($\Delta_{\max} \geq \max_{t \in T} \delta(t)$), we apply Algorithm 10 to update the control pattern. In particular, we have that the set of transitions enabled by the control pattern is $T \setminus \{t_1, t_{15}\}$, while after only one iteration, we find out that $f = f_1 = \vec{1}$, i.e., all transitions become control enabled and the net has recovered from the observer induced deadlock. Finally, by solving $m = 34$ IPP, we may also improve the marking estimate. In particular, as shown in Figure 6, we reconstruct the marking of places p_{27}, p_{34} and we detect the presence of 9 tokens in p_{14}. Note that in Figure 6 the large grey arrow has been used to highlight that no transition firing has occurred, but the net has timed-out and the deadlock recovery procedure has been applied.

6. USING TIMING INFORMATION FOR STATE ESTIMATION

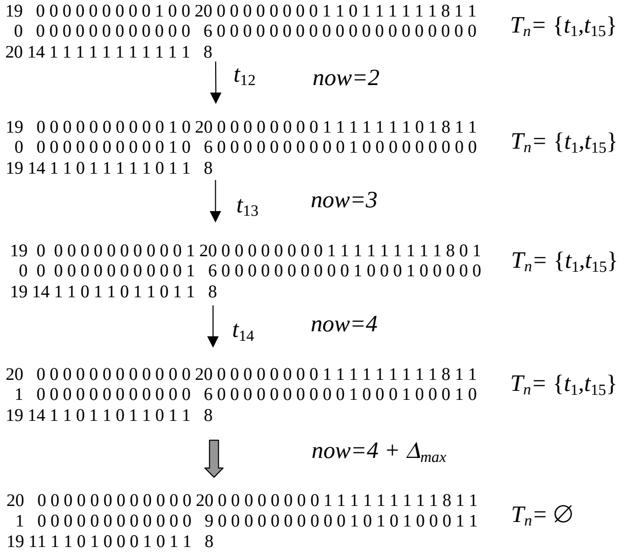

Figure 6: *Reachability graph of the net in Figure 3 under control when the deadlock recovery procedure proposed in Section 5 is applied.*

In [7] we shown how the above procedure can be modified to incorporate available information on the timing structure of the net into the state estimation process. The approach is essentially based on the linear algebraic characterization of deadlock markings given by the system of inequalities (1) that is used to restrict the set of w consistent markings. In the following this procedure is called *time-out* procedure.

To avoid repeating the formal steps of the algorithm, that are given in [7], we limit here to present the main idea that lead to its formulation. Then, we illustrate it via a numerical example.

Assume that we start observing the net at time τ and that transition t is *control* enabled during the time interval $[\tau, \tau + \delta(t)]$. Moreover, assume that the marking of the input places of t does not increase during the time interval $[\tau, \tau + \delta(t)]$. If at time $now = \tau + \delta(t)$ transition t does not fire, we can be sure that the actual marking M is such that $\neg M[t\rangle$, or equivalently t is not *marking* enabled: we say that t has *timed out* at time now.

We denote as T_{to} the set of timed out transitions.

At this point, if no transition fires we can invoke the time-out procedure. The only difference with respect to Algorithm 10 is that here T_i (see step 2 of Algorithm 10) is the set of transitions that are control enabled *and* that have timed out.

Then, as in the previous case, this information can be used to improve the marking estimate. Thus, two types of events that modify the marking estimate may occur.

— The first type of events occurs when the firing of a transition $\hat{t}$ is detected. In this case the marking estimate μ and bound B are updated following the estimation algorithm in [6]. In this step the set of timed out transitions T_{to} may eventually be updated, removing from this set all those transitions t such that ${}^\bullet t \cap \hat{t}^\bullet \neq \emptyset$, i.e., those transitions that may have been enabled by the firing of $\hat{t}$.

— The second type of events occurs when a new transition times out. In this case the set of timed out transitions is increased and we know that the actual marking must be such that the net $N_{to} \prec_{T_{to}} N$ is deadlocked, where N_{to} is the subnet of N induced by the set of the timed out transitions. We use this information to compute a new control pattern at least as permissive as the current one. We also update μ and B solving for each place an IPP of the form given by (7).

Thus, at each instant of time, it is possible to partition the set of transitions T into three subsets:

- $T_n = \{t \in T \mid f(t, \mathcal{C}) = 0\}$ is the set of transitions that are *not control enabled* given the current set of consistent markings.
- T_{to} is the set of *control enabled* transitions that have *timed out*. A transition t belongs to this set if during the time interval $[now - \delta(t), now]$ has continuously been control enabled and the marking of all its input places ${}^\bullet t$ has not increased during this same interval[1].
- T_e is the set of those control *enabled* transitions that do not belong to T_{to}.

To illustrate this procedure we apply it to the manufacturing example already considered in the previous sections.

6.1. Manufacturing example (continued)

The evolution of the net under control when the deadlock recovery procedure using timing information is applied, is reported in Figure 7. Note that in the same figure we have also reported the sets T_{to} and T_n.

The initial node of the graph is the same as in the previous case and the set of transitions disabled by the controller is $T_n = \{t_1, t_{15}\}$.

Given the actual delays, the time to wait before either applying the observer update procedure or the deadlock recovery procedure, is $\delta = 1$. In this case, after one time unit has elapsed, no transition fires. In fact, none

[1] Note that if the marking of some places in ${}^\bullet t$ has increased during the time interval $[\tau, \tau + \delta(t)]$, we can only conclude that the transition was not marking enabled at time τ, but no conclusion can be drawn on the marking enabling condition of t at time $\tau + \delta(t)$.

among the transitions t_{13}, t_{14}, t_{22} and t_{23}, whose timing delay is equal to 1, may actually fire, even if their firing is allowed by the controller. Thus, the deadlock recovery procedure is applied. We define the net N_{to} obtained from N removing all transitions not in $T_{to} = \{t_{13}, t_{14}, t_{22}, t_{23}\}$. For all $t \in T$ we compute the new control pattern $f(t, \mathcal{C} \cap \mathcal{M}_b(N_{to}))$ and we update the transition partitioning. In particular, we find out that both t_1 and t_{15} are still disabled by the controller, thus $T_n = \{t_1, t_{15}\}$, while $T_e = T \setminus (T_n \cup T_{to})$. Now, by solving 34 IPP we update the previous marking estimate and bounds. Numerical values are reported in Figure 7 where large grey arrows have been used to highlight that the deadlock procedure has been applied because some transitions have timed-out, but no transition has fired.

Now, when one more time unit has elapsed, transition t_{12} fires and the observer update procedure is applied. We update the estimate and the bound as shown in Figure 7, while the control pattern keeps the same for all transitions $t \in T$. Note that now, being ${}^\bullet t_{13} \cap t_{12}^\bullet \neq \emptyset$, the set T_{to} is updated to $T_{to} = \{t_{14}, t_{21}, t_{22}, t_{23}\}$. Moreover, $T_e = T \setminus (T_n \cup T_{to})$, where T_n is the same as in the previous step.

Then, after one more time unit t_{13} fires, and after another time unit t_{14} fires as well. The resulting marking estimate and bound are those reported in Figure 7, respectively in the fourth and fifth nodes.

At time $now = 5$ no transition fires and the deadlock recovery procedure is invoked. The new control pattern is computed and all transitions become control enabled. The marking estimate is also updated. Detailed results are reported in Figure 7.

To conclude we may observe now the closed loop net recovers from the deadlock after 5 time units. On the contrary, when we apply the procedure based on the net timeout, the net recovers from the deadlock after more than 9 units of time.

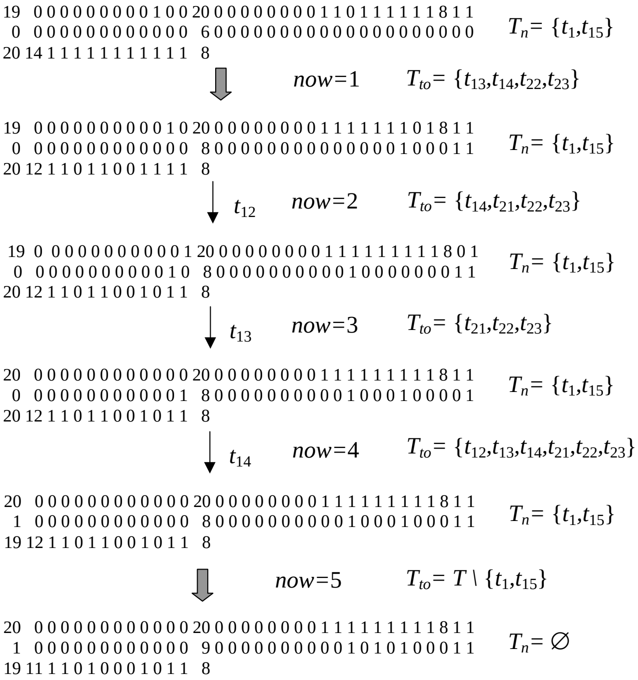

Figure 7: *Reachability graph of the net in Figure 3 under control when the deadlock recovery procedure using timing information is applied.*

8. LINEAR RELAXATION OF INTEGER PROGRAMMING

A drawback of the proposed procedures is that they require to solve at each step a certain number of integer programming problems to compute the control pattern: in some cases this may hinder the implementation of the approach on on-line controllers. This problem may be partially solved by simply relaxing the integer programming problems we consider into linear ones.

Assume that in constraint set (4) the constraints $M, M' \in \mathbb{N}^m$ are relaxed into $M, M' \in (\mathbb{R}_0^+)^m$. This yields a larger set of consistent markings $\mathcal{C}_R(w) \supseteq \mathcal{C}(w)$, i.e., we have a *relaxed observer* (R-observer) that is possibly less accurate than the previously defined observer. The control pattern computed using the R-observer is possibly suboptimal, in the sense that it is less permissive than or at most as permissive as the one computed using the observer [7]. Note, however, that the control pattern computed using the R-observer is certainly safe, i.e., it ensures that the control specifications are never violated.

Similarly, if in (2) the constraints $M \in \mathbb{N}^m$ and $\vec{s} \in \{0,1\}^m$ are relaxed into $M \in (\mathbb{R}_0^+)^m$ and $\vec{s} \in [0,1]^m$, this yields a larger set of deadlock markings. In this case the proposed recovery procedures can still be applied but the computed control patterns are, again, possibly suboptimal.

Thus, whenever necessary the control designer may take advantage of the linear relaxation trade-off that allows one to obtain a possibly suboptimal but computationally efficient solution technique.

As a final remark, it may also be possible to combine these techniques using linear programming for the on-line computation of the control patterns, and using integer programming only when applying the net time-out procedure.

As an example, in the case of the Petri net system already considered in this paper, one may verify that the on-line computation of the control patterns using the linear relaxation of (4) always yield optimal solutions. However, when a net time-out occurs, the linear relaxation is not optimal: the maximal permissive control pattern computed using the linear relaxation of (2) disables $\{t_1, t_{15}\}$ and because of this the deadlock recovery procedure may not work.

9. CONCLUSIONS

In this paper we have dealt with the problem of enforcing a set of GMEC on a timed Petri net by a state feedback control under the assumption that the system state is not measurable but can only be estimated. The use of the marking estimate instead of the actual marking may lead to a deadlock even if the controlled system is live. We propose two different solutions to this problem based on a linear algebraic characterization of the deadlock markings. The first one is applicable when no information on the timing structure of the net is available, the second one can only be used when the timing structure of the net is perfectly known.

References

[1] K. Barkaoui, A. Chaoui, B. Zouari, "Supervisory control of discrete event systems using structure theory of Petri nets," *1997 IEEE Int. Conf. on Systems, Man and Cybernetics* (Orlando, Florida), pp. 3750-3755, Oct 1997.

[2] F. Chu, X. Xie, "Deadlock analysis of Petri nets using siphons and mathematical programming," *IEEE Trans. on Robotics and Automation*, Vol. 13, No. 6, pp. 793–804, 1997.

[3] J. Ezpeleta, J.M. Colom, J. Martinez, "A Petri net based deadlock prevention policy for flexible manufacturing systems", *IEEE Trans. on Robotics & Automation*, Vol. 11, No. 2, pp. 173–184, 1995.

[4] M.C. Zhou, F. DiCesare, *Petri net synthesis for discrete event control of manufacturing systems*. Kluwer, 1993.

[5] A. Giua, F. DiCesare. M. Silva, "Generalized mutual exclusion constraints on nets with uncontrollable transitions," *Proc. 1992 IEEE Int. Conf. on Systems, Man, and Cybernetics* (Chicago, Illinois), pp. 974–979, Oct 1992.

[6] A. Giua, C. Seatzu, "Observability of place/transition nets," *IEEE Trans. on Automatic Control*, Vol. 47, No. 9, pp. 1424-1437, 2002.

[7] A. Giua, C. Seatzu, F. Basile, "Observer-based state-feedback control of timed Petri nets with deadlock recovery," *IEEE Trans. on Automatic Control*, Vol. 49, No. 1, pp. 17-29, 2004.

[8] L. E. Holloway, B. H. Krogh, A. Giua, "A survey of Petri net methods for controlled discrete event systems", *Discrete Event Systems*, Vol. 7, pp. 151-190, 1997.

[9] Y. Li, W.M. Wonham, "Control of vector discrete-event systems — part II: controller synthesis," *IEEE Trans. on Automatic Control*, Vol. 39, No. 3, pp. 512–531, 1994.

[10] T. Murata, "Petri nets: properties, analysis and applications," *Proc. IEEE*, Vol. Proc. 77, N. 4, pp. 541–580, 1989.

[11] J. Park, S.A. Reveliotis, "Deadlock avoidance in sequential resource allocation systems with multiple resource acquisitions and flexible routings," *IEEE Trans. on Automatic Control*, Vol. 46, No. 10, pp. 1572–1583, 2001.

[12] P.J. Ramadge, W.M. Wonham, "The Control of Discrete Event Systems," *Proceedings IEEE*, Vol. 77, No. 1, pp. 81–98, 1989.

[13] K. Yamalidou, J.O. Moody, M.D. Lemmon, P.J. Antsaklis, "Feedback control of Petri nets based on place invariants," *Automatica*, Vol. 32, No. 1, 1996.