Reduction of Constraints for Controller Synthesis based on Safe Petri Nets
Abbas Dideban, Hassane Alla

To cite this version:

HAL Id: hal-00333246
https://hal.archives-ouvertes.fr/hal-00333246
Submitted on 22 Oct 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Reduction of Constraints for Controller Synthesis based on Safe Petri Nets
Abbas DIDEBAN\textsuperscript{a}, Hassane ALLA\textsuperscript{b}

Abstract
In this paper, we present an efficient method based on safe Petri nets to construct a controller. A set of linear constraints allows forbidding the reachability of specific states. The number of these so-called forbidden states and consequently the number of constraints are large and lead to a large number of control places. A systematic method to reduce the size and the number of constraints for safe Petri Nets is offered. By using a method based on Petri nets invariants, maximal permissive controllers are determined.

Key words: Discrete Event Systems (DES), Petri Nets, Supervisory control, Controller synthesis, Forbidden states

1. Introduction
Supervisory control theory is essentially a theory for restricting the behavior of the plant to satisfy a "safety specification" that specifies which evolutions of the plant should not be allowed. The theory of Ramadge and Wonham (1987; 1989) is based on the modeling of the systems using formal languages and finite automata. However, the great number of states representing the behavior of system, and the lack of structure in the model, limit the possibility of developing an effective algorithm for the analysis and the synthesis of real systems. To solve these problems, several methods of controller synthesis based on Petri Nets (PNs) were proposed. PNs are a suitable tool to study Discrete Event Systems (DES) due to its capability in modeling and its mathematical properties. Very active research in the field of the controller synthesis for DES was born during the last decade (Roussel and Giua 2005; Giua et Xie 2005; Basile et al. 2006).

In (Yamalidou and Moody 1996), (Moody and Antsaklis 2000) and (Basile et al. 2006), the authors use the marking invariants to determine algebraically the incidence matrix of the supervisor PNs model. This method is very simple to be used. However, if some transitions are uncontrollable, it does not give the maximal permissive solution. In the method presented in (Basile et al. 2006) the authors used the structural controllability condition which is only a sufficient condition for having a controllable model. This technique presents two other disadvantages: 1) it is not always possible to describe the specifications by constraints and, 2) the number of constraints can be very large.

The control synthesis consists in preventing from forbidden states. These states may be deduced from specifications and can also be deadlock states. A method to minimize the addition of PN places is proposed in (ZhiWu and Zhou 2004), it is based on elementary siphons. There are some drawbacks in their study. Firstly, one can see that it is based on the computation of minimal siphons and secondly the proposed method is not generally optimal. A third problem is that uncontrollable transitions cannot be considered. In (Uzam 2002; Ghaffari 2003b), the authors proposed a method for solving the problems of forbidden states by the theory of regions. The advantage of this method is its generality for non-safe PNs. However, there are some drawbacks for this method, too:
- Generally, the number of control places is close to the number of border forbidden states.
- The computation time for solving the set of integer equations can be very large.

In (Giua et al. 1992), it is shown that it is possible to use linear constraints to specify forbidden states for safe and conservative PNs. The proposed approach is based on the equivalence between the set of forbidden states and the set
of linear constraints deduced from it. Using the invariants technique presented in (Yamalidou and Moody 1996), allows building a set of control places, which constitutes the optimal controller. However, the number of forbidden states, and consequently, the number of constraints, are large and leads to a large number of control places. In (Giuia et al. 1992), it is also shown that some constraints can be replaced by a single one; however, there is no systematic method to calculate the simplified constraints in a general case. The method comes from the linear constraints, which can be simplified taking the PNs structural properties into account.

In (Dideban and Alla 2005), a systematic method has been presented to reduce the number of constraints for safe and conservative PNs. The equations deduced from P-invariants properties in conservative PNs are used for simplification. This method needs to construct the set of possible states which is more expensive than the set of reachable states.

In this paper, we relax the property of conservative PNs. Then, a method is proposed to reduce the number of linear constraints for safe PNs. The advantage of this method is that the time and memory space for simplification are less than those presented in (Dideban and Alla 2005). In our approach, we use constraints which are equivalent to forbidden states. These constraints can be calculated in two different ways. They can be given directly as specifications or they can be deduced thanks to the Kumar approach (Kumar and Holloway, 1996).

In this paper, the important concept of over-state will be defined. This concept corresponds to a set of markings which has the same property. This idea will help us to build the simplest constraints, which forbid a greater number of states. A property for the existence of the maximal permissive controller will be analytically proved. In some very particular cases of non conservative PNs, the optimal solution does not exist. We show that this approach allows highlighting this problem in a simple way. This important concept can be used in other approaches.

In our approach, as in (Dideban and Alla 2005), we use the Reducibility Graph (RG) as an intermediate step for calculating the controller. Although the complexity of the computation of RG is exponential, this calculation is performed off-line. Moreover, the implemented final controller is a PN model, whose size is very close to the initial model. Generally, few control places are added.

The rest of this paper is organized as follows: In Section 2, the motivation and the fundamental definitions will be presented and illustrated via an example. In Section 3, the idea of passage from forbidden states to the linear constraints will be introduced. The concept of over-state and the basic idea of the simplification will be presented in Section 4. The calculation of the maximal permissive controller will be described in Sections 5. Finally, the conclusion is given in the last section.

2. Preliminary presentation

In this paper, it is supposed that the reader is familiar with the PNs basis (David and Alla 2005) and the theory of supervisory control (Ramadge and Wonham 1987; 1989).

In this section, we present only the notations and definitions which will be used later.

A PN is represented by a quadruplet \( R = \{ P, T, W, M_0 \} \) where \( P \) is the set of places, \( T \) is the set of transitions, \( W \) is the incidence matrix and \( M_0 \) is the initial marking. This PN is assumed to be safe; the marking of each place is Boolean.

**Definition 1:** The set \( \{0,1\}^N \) represents all the Boolean vectors of dimension \( N \).

A marking of a safe PN containing \( N \) places is a vector of the set \( \{0,1\}^N \).

The set of the marked places of a marking \( M \) is given by a function support defined as below:

**Definition 2:** The function \( \text{Support}(X) \) of a vector \( X \in \{0,1\}^N \) is:

\[
\text{Support}(X) = \text{the set of marked places in } X. 
\]

The support of vector \( M_1 = [1, 0, 1, 0, 0, 1, 0] \) is:

\[
\text{Support}(M_1) = \{P_1, P_3, P_6\} \text{ or more simply:} \]

\[
\text{Support}(M_1) = P_1P_3P_6.
\]

To simplify the notation of the formal expressions, we will use the support of a marking instead of its corresponding vector.

\( M_0 \) denotes the set of PN reachable markings. In \( M_0 \), two subsets could be distinguished: the set of authorized states \( M_A \) and the set of forbidden states \( M_F \). The set of forbidden states correspond to two groups: 1) the set of reachable states \( M_P \), which either do not respect the specifications or are deadlock states, 2) the set of states for which the occurrence of uncontrollable events leads to states in \( M_F \).

The set of authorized states are the reachable states without the set of forbidden states:

\[
M_A = M_0 \setminus M_F
\]

Among the forbidden states, an important subset is constituted by the border forbidden state denoted as \( M_B \).

**Definition 3:** Let \( M_B \) be the set of border forbidden state:

\[
M_B = \{ M_e \in M_F \mid \exists \sigma \in \Sigma_e \text{ and } \exists M_f \in M_A, M_e \xrightarrow{\sigma} M_f \}
\]

Where \( \Sigma_e \) is the set of controllable transitions.

We will use the following example in order to illustrate the definitions and the results developed in this paper.

Consider a system composed of two machines \( M_1 \) and \( M_2 \), which can work independently. The starting and the end of the tasks on these machines are respectively realized by uncontrollable events \( c_1 \) and \( c_2 \), and by uncontrollable events \( f_1 \) and \( f_2 \). When machine \( M_1 \) ends its task on a part, it stays available for a new task while machine \( M_2 \) has to
transfer its produced part in a buffer before beginning a new task (event \( b_2 \)). Both machines are activated simultaneously (event \( \text{start} \)) but each of them can be inactivated separately (events \( sp_1 \) and \( sp_2 \)). The specifications impose a sequence of the events \( f_1 \) and \( b_2 \). An elementary production is a result of a process on a part by \( M_{A_1} \) followed by another process by \( M_{A_2} \). This production is repeated in a cyclic way. The system can be started by a start command and can be stopped by a stop command. At the end, the production process on a part must be completed. For restart, we need to initialization of the controller. The process and specifications models are represented in Figure 1. They are non conservative PNs.

![Fig. 1. PN model of the a) Process b) specification](image)

The synchronous composition between the models of process and the model of specifications is given by a safe PN in Figure 2.

![Fig. 2. PN model of the system coupled with its specification](image)

The existence of uncontrollable events leads to the existence of forbidden states. For example when the system is in state \( M_s \), it is possible to fire the uncontrollable event \( f_1 \), while it is not authorized by the specifications. This state is a forbidden state. The set of forbidden states can be determined by the algorithm established by Kumar and Holloway (1996).

Figure 3 gives the reachability Graph of the PN presented in Figure 2. The forbidden states are indicated in dark gray and the authorized states in white. The construction of the reachability graph is stopped when a forbidden state is reached.

![Fig. 3  Reachability graph](image)

From the set of forbidden states \( M_F = \{ M_5, M_6, M_7, M_8, M_9, M_{10}, M_{12}, M_{13}, M_{16}, \ldots \} \), we can construct the set of border forbidden states \( M_B = \{ M_5, M_6, M_7, M_8, M_{12}, M_{13} \} \). In a conservative and safe PN, the inequality \( m_1 + m_2 + m_3 + m_4 \leq 3 \) forbids only the state \( P_1P_2P_3P_4 \). (Giua et al. 1992). In this situation, for \( N \) forbidden states, we will need \( N \) linear constraints. The complexity of the controller model increases extremely when the number of forbidden states increases for we need one control place for each constraint (Yamalidou and Moody 1996). In this paper, we propose a method to reduce the number and the size of the linear constraints for a given set of forbidden states. We give the necessary and sufficient condition for having a maximal permissive controller in the case of non conservatives PNs. To achieve this goal, we need to introduce the important concept of “over-state”. In this paper we use a hypothesis that is presented below:

**Hypothesis 1:** All of the events are independent.
3. From forbidden states to linear constraints

Let \( M_1 = (M_1^T = [m_1, m_2, \ldots, m_n]) \) be a forbidden state\(^1\) in set \( \mathcal{M}_k \) and \( \text{Support}(M_1) = \{ P_1, P_2, \ldots, P_m \} \) the set of marked places of \( M_1 \). From a forbidden state, a linear constraint can be constructed (Giua et al. 1992).

The linear constraint deduced from the forbidden state \( M_1 \) is given below. The state \( M_1 \) does not verify this relation. Therefore, by applying this relation, \( M_1 \) will be forbidden.

\[
\sum_{k=1}^{n} m_{ik} \leq n - 1
\]

Where \( n = \text{Card}[\text{Support}(M_1)] \) is the number of marked places of \( M_1 \), and \( m_{ik} \) is the marking of place \( P_i \) of state \( M_1 \).

Let \( M = (M^T = [m_1, m_2, \ldots, m_n]) \) be a general marking and \( M_1 \) be a forbidden state. The constraint (forbidding state \( M_1 \)) is denoted by \( c_i \) and can be rewritten in the following form:

\[
M^T.M \leq \text{Card}[\text{Support}(M_1)] - 1
\]

For example if:

\[
M^T = [0, 1, 1, 0, 0, 0, 1] \Rightarrow \text{Card}[\text{Support}(M_1)] = 3
\]

Verifying Relation 1 is equivalent to forbid state \( M_1 \) when the PN model is conservative. However, in a safe PN not necessarily conservative, this equivalence is not always true. This problem will be discussed later. This equivalence is necessary to obtain the optimal supervisor.

4. Simplification by using over-state concept

4.1. Definition of an over-state concept

The concept of over-state is very important in this paper. An over-state can represent a complete state or a part of this one. In the example of the two machines, \( P_2.P_3.P_6.P_9 \) is a complete state that represents the situation of both machines and the specifications. \( P_2.P_3 \) is an over-state of this state that represents a partial state of the system. We have noted that a state can be forbidden by a linear constraint. In the same way, it is possible to forbid an over-state by its corresponding constraint.

**Definition 4:** Let \( M_1 = (P_{21} \ldots P_{2m}) \) be an accessible state, \( M_2 = (P_{11} \ldots P_{1n}) \) will be an over-state of \( M_2 \) if:

\[
M_1 \leq M_2
\]

For example \( M_1 = P_2.P_3 \) is an over-state of \( M_2 = P_1.P_2.P_3.P_6.P_9 \).

The name “over-state” is used because the constraint corresponding to an over-state holds the state’s constraint. For example, the constraint \( m_1 + m_6 \leq 1 \) that corresponds to the over-state \( M_1 = P_2.P_3 \) holds both following constraints:

\[
m_1 + m_4 + m_6 + m_9 \leq 3
\]
\[
m_2 + m_4 + m_9 \leq 3
\]

These two constraints forbid states \( M_1 = P_1.P_2.P_3.P_9 \) and \( M_2 = P_2.P_3.P_6.P_9 \). \( P_4.P_6 \) is an over-state of both states \( P_1.P_2.P_3.P_9 \) and \( P_2.P_3.P_6.P_9 \) which could be verified by \( M_1 \leq M_3 \) and \( M_2 \leq M_3 \). Thus by using only the constraint \( m_4 + m_9 \leq 1 \), both states \( M_1 \) and \( M_2 \) will be forbidden. However, this reduction is not always simple; it is possible that the simplified constraint forbids also some authorized states. We present below a method of simplification which guarantees that the constraints forbid only the forbidden states.

**Remark 2:** With each over-state \( b_i \), we associate a constraint \( c_i \) in the following way:

\[
b_i = (P_{i1}P_{i2}P_{i3} \ldots P_{i9} \ldots P_{in}) \Rightarrow c_i = (P_{i1}P_{i2}P_{i3} \ldots P_{in}, n-1)
\]

That means:

\[
m_{i1} + m_{i2} + \ldots + m_{in} \leq n-1
\]

**Remark 3:** It is possible to use an over-state without taking into account the fact that an authorized state can be forbidden. In that case, the controller would not be maximal permissive.

**Remark 4:** There are two relations of inclusion, which operate in opposite directions: a set inclusion and a marking inclusion. Let \( M_1 \leq M_2 \):

1) The set of the marked places in the over-state \( M_1 \) is included in the set of the marked places in the state \( M_2 \).

2) The set of the markings covered by \( M_1 \) contains those covered by marking \( M_2 \).

**Property 1:** Let \( M_1 \) and \( M_2 \) be two vectors of \( \{0, 1\}^N \), and \( c_1 \) and \( c_2 \) be two corresponding constraints. If \( M_1 \leq M_2 \) (\( M_1 \) is an over-state of \( M_2 \)) and \( c_1 \) is true, then \( c_2 \) is also true:

\[
M_1 \leq M_2 \text{ and } c_1, M_1^T.M \leq \text{Card}[\text{Support}(M_1)] - 1 \rightarrow c_2: M_2^T.M \leq \text{Card}[\text{Support}(M_2)] - 1
\]

**Proof:**

The PN model is safe then:

\[
(M_1^T - M_2^T).M \leq \text{Card}[\text{Support}(M_2)] - \text{Card}[\text{Support}(M_1)]
\]

And:

\[
M_1^T.M = (M_2^T - M_1^T + M_1^T).M = (M_2^T - M_1^T).M + M_1^T.M
\]

By using the constraint \( c_1 \), we have:

\[
(M_1^T - M_2^T).M + M_1^T.M \leq (\text{Card}[\text{Support}(M_2)] - \text{Card}[\text{Support}(M_1)]) + \text{Card}[\text{Support}(M_1)] - 1 \rightarrow M_2^T.M \leq \text{Card}[\text{Support}(M_2)] - 1
\]

4.2. Set of over-states

We have noted that to forbid a state, it is enough to forbid its over-state, but which over-state? This question will
be answered in the sequel. To achieve this goal, we need to construct the set of over-states for the forbidden states.

Firstly, we calculate the set of over-states for each state and then the union of all over-states gives the final set.

**Definition 5:** Let $M_i = \{P_1, P_2, P_3, ... P_m\}$ be a state of the system. The set of the over-states of $M_i$, denoted by $M_i^{\text{over}}$, is equal to the set of the subsets of $M_i$ without the empty set.

For example, the state $M_i = \{P_1, P_2, P_6, P_9\}$ gives:

$M_i^{\text{over}} = \{\{P_1\}, \{P_2\}, \{P_6\}, \{P_9\}, \{P_1, P_2\}, \{P_1, P_6\}, \{P_1, P_9\}, \{P_2, P_6\}, \{P_2, P_9\}, \{P_6, P_9\}, \{P_1, P_2, P_6, P_9\}\}$

Among, the set of forbidden states in $M_i$, only the border states have to be considered in the controller synthesis. Let $M_i$ be this set and $B_i$ be the set of over-states of $M_i$.

$$B_i = \bigcup_{i=1}^{\text{Card}(M_i)} M_i^{\text{over}}$$

### 4.3. Basic idea to build the minimal set of constraints

For a given set and a property, we can define three disjoint sub-sets:

1. The set where each element verifies this property
2. The set where each element does not verify this property, and
3. The set which is indifferent to this property.

The third set is important and will be used advantageously to improve the simplifications.

**Definition 6:** Let $E_1$ and $E_2$ be two sets included in a set $G$ and hold:

$$E_1 \cap E_2 = \emptyset$$

$$E_1 \cup E_2 = E$$

$$E \cup \bar{E} = G$$

$ar{E}$ is the complementary set of $E$ in $G$.

$V$ is an element of $G$.

A property $P$ according to $V$ is true if $V \in E_1$ and false if $V \in E_2$.

This property is not defined if $V \in \bar{E}$, it can then be said that this property is true if $V \notin E_2$.

We will use this definition on the set of states to achieve our goal. Let $E_1$ be the set of the forbidden states and $E_2$ the set of the authorized states and let $P$ be the forbidding property. The state $V$ can be forbidden if it is not in $E_2$. This means that the states which are not accessible could be forbidden. This consideration will make the constraints to be further simplified. This idea is similar to the concept of don’t care states that are used in the minimization of combinatorial and sequential logic. In logic circuits don’t care states are the states that are not reachable because of the input variables or initial states. In PN models, non reachable states are the states that are not accessible from the initial state.

In Property 1, it was shown that one over-state can cover a great number of states. Therefore, we can forbid an over-state if it does not cover any authorized state.

Our objective is to find a method to reduce the number and the bound of the constraints. For that, we build the set of all over-states of the border forbidden states. This set will be calculated by removing all authorized over-states from it. The minimal set of constraints will then be obtained. Finally, the best choice will be established.

The different steps formalizing this approach are presented in the following section.

#### 4.4. Building the reduced set of over-states

It is possible to build two sets of over-states; a set of the over-states of $A_1$, and that of the forbidden states $B_i$. It is obvious that no over-state of $A_1$ must be forbidden. Thus it is necessary to remove from set $B_i$, all over-states which are in $A_1$. This gives set $B_i$:

$$B_i = B_i \setminus A_1$$

**Remark 5:** From the implementation point of view, it is not necessary to construct $A_1$. The set $M_i$ is directly used.

**Property 2:** Let $B_i$ be the set of over-states of $M_i$ and $A_1$ be the set of over-states of $M_i$ and:

$$B_i = B_i \setminus A_1$$

The markings verifying the set of constraints $C_2$ (equivalent to $B_i$) correspond to the complete set of authorized states.

The proof of this property is obvious.

In set $B_i$, it often possible to find couple of states $M_i$ and $M_j$ such that $M_i \leq M_j$ ($M_i$ is an over-state of $M_j$). In that case, $M_i$ must be removed. It is a redundant state, and set $B_i$ is then defined formally as follows:

$$B_i = B_i - \{M_j \in B_i \; \exists \; M_i \in B_i, \; M_i \not\geq M_j\}$$

$B_i$ is the minimal set of over-states to be forbidden.

For the example, from Figure 2, the sets of border forbidden states and authorized states are:

$$M_i = \{P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9, P_{10}\}$$

$$M_i = \{P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9, P_{10}\}$$

Sets $A_1$, $B_1$, $B_2$ and $B_3$ are then calculated as follows:

$$B_1 = M_i^{\text{over}} \cup M_2^{\text{over}} \cup M_3^{\text{over}} \cup M_4^{\text{over}} \cup M_5^{\text{over}} \cup M_6^{\text{over}} = \{P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9, P_{10}\}$$

$$B_1 = M_i^{\text{over}} \cup M_2^{\text{over}} \cup M_3^{\text{over}} \cup M_4^{\text{over}} \cup M_5^{\text{over}} \cup M_6^{\text{over}} = \{P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9, P_{10}\}$$

$$B_1 = M_i^{\text{over}} \cup M_2^{\text{over}} \cup M_3^{\text{over}} \cup M_4^{\text{over}} \cup M_5^{\text{over}} \cup M_6^{\text{over}} = \{P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9, P_{10}\}$$

$$B_1 = M_i^{\text{over}} \cup M_2^{\text{over}} \cup M_3^{\text{over}} \cup M_4^{\text{over}} \cup M_5^{\text{over}} \cup M_6^{\text{over}} = \{P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9, P_{10}\}$$

$$B_1 = M_i^{\text{over}} \cup M_2^{\text{over}} \cup M_3^{\text{over}} \cup M_4^{\text{over}} \cup M_5^{\text{over}} \cup M_6^{\text{over}} = \{P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9, P_{10}\}$$

$$B_1 = M_i^{\text{over}} \cup M_2^{\text{over}} \cup M_3^{\text{over}} \cup M_4^{\text{over}} \cup M_5^{\text{over}} \cup M_6^{\text{over}} = \{P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9, P_{10}\}$$

$$B_1 = M_i^{\text{over}} \cup M_2^{\text{over}} \cup M_3^{\text{over}} \cup M_4^{\text{over}} \cup M_5^{\text{over}} \cup M_6^{\text{over}} = \{P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9, P_{10}\}$$

$$B_1 = M_i^{\text{over}} \cup M_2^{\text{over}} \cup M_3^{\text{over}} \cup M_4^{\text{over}} \cup M_5^{\text{over}} \cup M_6^{\text{over}} = \{P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9, P_{10}\}$$
In the previous section, we have determined the set $B_1$, which is the set of over-states that must be forbidden. In the two following sections, we present the necessary and sufficient conditions to design a maximal permissive controller.

With each over-state of $B_1$, we associated a constraint in the following way:

$$ b_i = (P_{i1} P_{i2} P_{i3} \ldots P_{in}) \iff c_i = (P_{i1} P_{i2} P_{i3} \ldots P_{i9}, n-1) $$

Let $C_i$ be the set of these constraints for the example:

$$ C_i = \{(P_{i1} P_{i2} 1), (P_{i1} 1), (P_{i1} P_{i2} 1), (P_{i1} P_{i2} 1), (P_{i1} P_{i2} 1)\} $$

This set $C_i$ defines the set of non-forbidden states, denoted as $M_E$. Now the objective is to compare the set of authorized states $M_A$ and $M_E$.

**Remark 6:** In reality we don’t need to construct $A_i$. It is possible calculate $B_i$ from $B_1$ and $M_E$.

5. Controller synthesis

5.1. Maximal permissive controller

In the previous section, we have determined the set $B_1$, which is the set of over-states that must be forbidden. In the two following sections, we present the necessary and sufficient conditions to design a maximal permissive controller.

With each over-state of $B_1$, we associated a constraint in the following way:

$$ b_i = (P_{i1} P_{i2} P_{i3} \ldots P_{in}) \iff c_i = (P_{i1} P_{i2} P_{i3} \ldots P_{in}, n-1) $$

Let $C_i$ be the set of these constraints for the example:

$$ C_i = \{(P_{i1} P_{i2} 1), (P_{i1} 1), (P_{i1} P_{i2} 1), (P_{i1} P_{i2} 1), (P_{i1} P_{i2} 1)\} $$

This set $C_i$ defines the set of non-forbidden states, denoted as $M_E$. Now the objective is to compare the set of authorized states $M_A$ and $M_E$.

**Remark 7:** Constraint $c_i$ and over-state $b_i$ are equivalent as shown above.

**Definition 7:** Let $B_1 = \{b_1, b_2, \ldots, b_n\}$ be the set of simplified over-states and $M_\Phi = \{M_1, M_2, \ldots, M_n\}$ be the set of border forbidden states. The relation $R: M_\Phi \times B_1 \rightarrow \{0, 1\}$ is as:

$$ R(M_i, b_j) = \begin{cases} 1 & \text{if } b_j \leq M_i \text{ (} b_j \text{ is over-state of } M_i \text{)} \\ 0 & \text{if not} \end{cases} $$

The covering of a marking is an integer number:

$$ Cv(M_i) = \sum_{j=1}^{n} R(M_i, b_j) $$

$Cv(M_i) \geq 1$ means that forbidden state $M_i$ is covered by at least one over-state.

**Property 3:** The set of non forbidden state $M_E$ is equal to the set of authorized state $M_A$, if and only if:

$$ \forall M_i \in M_E \quad Cv(M_i) \geq 1 $$

**Proof:**

**Necessary Condition:**

Assume that $M_A = M_E$, we prove that:

$$ \forall M_i \in M_E \quad Cv(M_i) \geq 1 $$

If $\exists M_i \in M_E \quad Cv(M_i) = 0 \Rightarrow R(M_i, b) = 0 \quad \forall b \in B_3$.

There is not any constraint $c_j$ deduced from $b_j$ that forbids $M_i$. Then $M_i \notin M_E$

However, $M_i$ is a forbidden state and, $M_i \notin M_A$.

Then $M_A \neq M_E$ that it is not true.

**Sufficient condition:**

Assume that $\forall M_i \in M_E \quad Cv(M_i) \geq 1$, we prove:

$$ M_A = M_E $$

$\forall M_i \in M_E \quad Cv(M_i) \geq 1 \Rightarrow \forall M_i \in M_E \exists b_j \in B_3 / R(M_i, b) = 1, (M_i$ would be forbidden by this constraint)

$$ \Rightarrow \forall M_i \in M_E, M_i \notin M_A \text{ Then: } M_A \subseteq M_E $$

In addition, according to the method used for the construction of $B_1$, $M_A \subseteq M_E$ (any authorized state is not forbidden)

Then $M_A = M_E$.

Now, let us illustrate the results established above in the example of Figure 2. Property 3 should initially be checked. For this, we construct a table (Table 1) where the first row represents the set of forbidden states $M_\Phi$ and the first column is the set of simplified over-states $B_1$. In the case of our example, these sets are:

$$ M_\Phi = \{P_1 P_2 P_3 P_4, P_2 P_3 P_4 P_5, P_3 P_4 P_5 P_6, P_4 P_5 P_6 P_7, P_5 P_6 P_7 P_8, P_6 P_7 P_8 P_9, P_7 P_8 P_9 P_{10}\} $$

$$ B_1 = \{P_5, P_4, P_3, P_2, P_1, P_6, P_7, P_8, P_9, P_{10}\} $$

$$ B_3 = \{P_4 P_5, P_5 P_6, P_6 P_7, P_7 P_8, P_8 P_9, P_9 P_{10}\} $$

<table>
<thead>
<tr>
<th>$M_A$</th>
<th>$P_1$</th>
<th>$P_2$</th>
<th>$P_3$</th>
<th>$P_4$</th>
<th>$P_5$</th>
<th>$P_6$</th>
<th>$P_7$</th>
<th>$P_8$</th>
<th>$P_9$</th>
<th>$P_{10}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_A$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

**Tab. 1. Function $R(c_j, M_i)$ and $Cv(M_i)$**

This table shows that $\forall M_i \in M_E \quad Cv(M_i) \geq 1$, and thus the set of non forbidden states $M_E$ is equal to the set of authorized states $M_A$. 

6
We will see that this is not always the case. For that, we take the example presented in (Bratosin et al. 2005). It is a system made up of two machines $M_1$ and $M_2$. The beginnings of the tasks are denoted by the controllable events $c_1$ and $c_2$ and the ends are synchronized by the uncontrollable event $f$. The specification authorizes the occurrence of the event $f$ only once. The PN model $R$ of the closed-loop operation for this system is presented in Figure 4.

The sets of the authorized and forbidden states are presented below:

$$M_0 = \{P_1, P_4, P_2, P_3\}$$

$$M_b = \{P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9, P_{10}\}$$

$$M_B = \phi$$

To choose the minimal set of constraints, denoted by $B_4$, firstly it is necessary to choose the over-state for which there exists a forbidden state that can be covered only by this over-state ($C(M_f) = 1$). If such over-states are found, we mark all the corresponding forbidden states in line $C_f(M)$. This line corresponds to the final covering.

<table>
<thead>
<tr>
<th>$M_f$</th>
<th>$C_f(M_f)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_1$</td>
<td>1</td>
</tr>
</tbody>
</table>

This corollary means that it is necessary for each forbidden state to be covered at least by one over-state. When this is verified, the maximal permissive controller is obtained.

5.3. Control places

The set of the constraints equivalent to $B_4$ is denoted by $C_4$. To calculate the control places corresponding to each linear constraint, we will use the method developed in (Yamalidou and Moody 1996). This technique based on the PN’s invariant is recalled briefly below. Let $W_R$ be the incidence matrix of the system (process and specifications). Each place of the controller will add a line to the matrix. Let $W_{RC}$ be the incidence matrix of the PN model corresponding to the controlled system. It is made up of two matrices, the original matrix of system $R$, $W_R$ and the incidence matrix of the controller, $W_C$. From the set of constraints $C_4$, matrix $L$ and constant vector $C_{bound}$ can be constructed. It is possible to calculate in an algebraic way the incidence matrix of the controller as it is presented below. $M_{RH}$ is the initial marking of system $R$ and $M_{FC}$ is the initial marking of the control places. The very simple way to calculate $W_C$ makes this approach very popular.

$$W_C = -LW_R$$

Fig. 4. Closed loop PN Model in case of non optimal supervisor

![Fig. 4. Closed loop PN Model in case of non optimal supervisor](Image 66x534 to 255x663)
\[ M_{C,i} = C_{bond} - L.M_{R,i} \]

Let us take again the example of Figure 2, the set of final constraints \( \{C_4\} \) is:

\[
\begin{bmatrix}
0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

\[ W_C = -L.W_R \]

\[
\begin{bmatrix}
0 & 0 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

\[ M_{C,i} = C_{bond} - L.M_{R,i} \]

\[ M_{c1} = 0; \quad M_{c2} = 1 \]

Yamalidou and Moody (1996) showed that if all events are controllable, the controller is maximal permissive. However, if there are uncontrollable events, the extended method presented in (Moody and Antsaklis 2000) does not generally give the optimal solution. The problem exists when a control place is synchronized with a place of the process by an uncontrollable event as indicated in figure 6.

![Fig. 6. Control place synchronized with the process by an uncontrollable event](image)

In the case presented in Figure 6, the process cannot always respect the PN firing rules. Suppose that place \( P_i \) is not marked and \( P_i \) is marked. Since \( \sigma_u \) is uncontrollable, then transition \( T_i \) is fired even if it is forbidden by the control place. It means that it is possible that the set of reachable states will be greater than the set given by the PN model. According to definition of structural controllable model in (Basile et al. 2006), the model in this case is not controllable. We prove in Property 4 that it is not a necessary condition after applying our method of controller synthesis. After using our method, when the places belong to the process are marked, the control and specifications places will be always marked.

**Definition 8:** The set of accessible states for controlled system is presented by the set \( A_{RC} \).

We are going to show that if the condition in Corollary 1 is true, the obtained controller is maximal permissive even if uncontrollable transitions exist.

**Remark 8:** A marking of the set \( A_{RC} \) differs from a marking of \( M_E \) because of the added control places. This is only a coding of these sets. To be able to compare the various sets of states, we will omit the control places for the elements of the set \( A_{RC} \).

**Property 4:** Let \( M_k \) be the set of authorized states by the constraints deduced from \( B_4 \) and let \( A_{RC} \) be the automaton that corresponds to the set of accessible state in the controlled system,

If \( M_k = M_A \), then \( A_{RC} \) is isomorphic to \( M_k \) and the controller obtained by the invariant approach is maximal permissive.

**Proof:**

By the invariant approach, we have always:

\[ M_k \subseteq A_{RC} \quad (2) \]

Now we show that:

\[ A_{RC} \subseteq M_k \quad \text{(knowing that } M_k = M_A) \]

Suppose that \( \exists \ M_i \in A_{RC} \text{ and } M_i \notin M_k \)

\[ \Rightarrow \exists \sigma_u \in \Sigma_u \text{ and } \exists M_j \in M_k, M_j \xrightarrow{\sigma_u} M_i \]

However \( M_k = M_A \Rightarrow M_i \in M_k \text{ and } M_i \notin M_A \)

\[ M_k \notin M_A \Rightarrow M_i \notin M_k \quad \text{(} M_k = M_k \setminus M_A \text{)} \]

It is obvious that: \( M_j \xrightarrow{\sigma_u} M_i \text{ then } M_i \in M_k \text{ (definition of forbidden states)} \)

\[ M_j \in M_A \text{ and } M_i \notin M_k \text{ (contradiction)}, \text{ then} \]

\[ A_{RC} \subseteq M_k \quad (3) \]

(2) and (3) \( \Rightarrow M_k = A_{RC} \Rightarrow A_{RC} = M_A \)

In the case of our example, the function \( C(M_i) \) (final covering) is equal to 1 for each \( M_i \in M_k \), therefore \( M_k = M_A \) (Corollary 1) then the controller is maximal permissive (Property 4). The PN model of the final controller is represented in Figure 7.

It should be noticed that there are some control places with uncontrollable output transitions. However, that never leads to a bad behavior, i.e. when a control place is not
marked; there is at least one non marked input place for this uncontrollable transition, which belongs to the process. Moreover, controllable events $c_1$ and $c_2$ have been removed since the control is now performed by the control places. The complete algorithm for controller synthesis is presented in Appendix I. The computation of some sets is of polynomial complexity except for the $\mathcal{M}_f$ over-states computation which is exponential. Fortunately the number of border states is often small.

Fig. 7. PN Model in closed loop with control places

5.4. From PN to SFC models

The controllers have always a deterministic behavior. A given set of inputs corresponds to a unique set of outputs. In this paper we consider an asynchronous functioning, all events are independent and the simultaneous occurring of two independent events is not possible. However in real implementation, due to cycle time in a PLC (Programmable Logic Controller), it is possible to have simultaneous occurring of events. Then, sometimes the controller obtained with our approach can be non deterministic. In that case, the conflicts must be solved for example by making a choice.

In the example of Figure 6, the model is deterministic and there is no conflict. We can transfer directly the PN model into a Sequential Function Chart (SFC) or ladder diagram language (LD).

Here, the SFC model is obtained by replacing each place of the PN model by a step. A control action is associated with each step that corresponds to the event (sensor) and belongs to the output transition. Transitions and events remain unchanged. This technique is inspired from the works presented in (Giua and DiCesare 1993; Uzam and Jones 1998). The SFC model for this example is presented in Figure 8.

Fig. 8. SFC model corresponding to the PN controller in Fig. 7

Actions $A_1$ and $A_2$ correspond to the assembly operations and action $B_2$ corresponds to the transfer operation. Sensors $f_1$, $f_2$, and $t_2$ detect the ends of operations.

6. Conclusion and future works

In this paper, we have presented a systematic method to reduce the number of linear constraints corresponding to the forbidden states for a safe PN. This is realized by using non-reachable states and by building the constraints using a systematic method. The important concept of over-state has been defined; it corresponds to a set of markings which keep the same property (forbidden or authorized). From the forbidden states, the set of over-states is calculated. The utilization of non-reachable markings allows great simplification of the constrained.

Properties which give necessary and sufficient conditions for the existence of a maximal permissive controller were established and illustrated for a manufacturing system. After the simplifications, the existence of the controller is proved. When this controller exists, the invariant approach allows the computation of the controller that can be transformed to a SFC model and be directly implemented in a PLC.

Our future work will include:

1) Developing this method of simplification to achieve more reduced results using the partial invariant idea.

2) Using this idea for simplification of conditions that are employed as predicates for controllable transitions. In this case, we can develop the idea of over-state for non-safe Petri Nets. The idea is to introduce the number of tokens as a power of the place identifiers. This can be indicated as follows: $P_1^f$, $P_2^f$, ..., place $P_i$ and place $P_j$ containing respectively 3 and 2 tokens. Thus, some of the properties presented in this paper can be generalized. Of course, some fundamental research needs to be done.

References


Appendix I: Algorithms

A) Algorithm 1: Selection of the set of final over-states
Step 1: Find the forbidden state $M_f$ for which $Cv(M_f)$ (Definition 7) is: a) non null, b) the smallest one, and c) $Cf(M_f) = 0$ ;
If $M_f$ does not exist, go to step 5;
Step 2: a) Find the set of constraints $C = \{c_1, ..., c_k, ..., c_m\}$ such that: $R(c_i, M_f) = 1$,
b) Find the constraint $c_j$ in set $C$ which covers the maximal number of states $M_r$ with $Cf(M_r) = 0$, and
c) Take the simpler $c_j$ in case of equality.
Step 3: Save $c_j$ in $B_c$;
Step 4: Mark the forbidden states which are covered by the constraint $c_j$ in the line $Cf$; Go to step 1;
Step 5: End;

B) Algorithm 2 : Complete algorithm for controller synthesis
Step 1: Compute the set of over-states $B_1$ for the set of border forbidden state $M_B$.
Step 2: Compute the set of over-states $B_2$ by deleting from $B_1$ the over states that exist in $M_B$.
Step 3: Compute $B_3$ by deleting redundant over-states from $B_2$.

Step 4: Verifying Corollary 1 for maximal permissive controller: if it is verified go to step 5 else there is no maximal permissive controller. Go to Step 8.

Step 5: Apply algorithm 1 for computing $B_c$.
Step 6: Compute the control places from set of constraints $B_4$ by Yamalidou method.
Step 7: Transforming PN model into a SFC.
Step 8 : End

Abbas Dideban received his Ph.D. in Automation control from University of Grenoble I, France in 2007. He was awarded the M.Sc. degrees in Digital Electronic from Sharif University, Iran in 1997. He joined to the University of Semnan as a lecturer from 1998. At the same time he was cooperated with Jahadaneshghahi Sharif in industrial Automation sector. Now he is an assistant professor at Semnan University. His research topics include Discrete Event Systems, Petri Nets, Industrial automation, Digital systems Design.

Hassane ALLA is Professor at the University Joseph Fourier of Grenoble. His research is mainly concerned with tools derived from Petri nets and automata used for the performance evaluation and for the control synthesis of discrete event systems. He is author or co-author of about one hundred publications. One of its main publications is a book on Continuous and Hybrid Petri nets which has been published in English and in French.