Controller synthesis with highly simplified linear constraints

Abbas Dideban, M. Zareiee, Hassane Alla

Controller synthesis with highly simplified linear constraints

Dideban A[*a], Zareiee M[b], Alla H[c]

[a] *Electrical Eng. Department, Semnan University, Semnan, IRAN (e-mail: adideban@semnan.ac.ir).*
[b] *Electrical Eng. Department, Semnan University, Semnan, IRAN (e-mail: mzareiee@semnan.ac.ir).*
[c] *GIPSA lab, 38402 St Martin d'Heres Cedex FRANCE, (e-mail: hassane.alla@gipsa-lab.grenoble-inp.fr).*

Phone: (+98)231-3354123

Fax: (+98)231-3354123

Email[*]: adideban@semnan.ac.ir; a_dideban@yahoo.com

Abstract. This paper deals with the problem of forbidden states in safe Petri nets to obtain a maximally permissive controller. To prevent the system from entering these states, assigning some constraints to them is possible. The constraints can be enforced on the system using control places. When the number of forbidden states is large, a large number of constraints should be assigned to them. So, a large number of control places must be added to the model of the system which in turn causes a complicated model. Some methods have been proposed to reduce the number of constraints. But they don't always give the best results. In this paper two ideas are offered to reduce the number of these constraints giving a more simplified controller.

Key words: Discrete Event System (DES), Petri Net, Supervisory control, Controller synthesis, Forbidden state

1. Introduction

Supervisory control theory which was proposed by Ramadge and Wonham, is a systematic method for controlling Discrete Event Systems (DES) [1, 2]. The basic idea behind this theory is to restrict the behavior of system according to a desired behavior. This restriction is obtained by disabling the controllable transitions under some special conditions [3]. There are a lot of methods for designing controller based on automata models. But when the number of states and transitions causing events are large, modeling based on this tool is impractical [4]. So, Petri net (PN) has been proposed for modeling DES [5]. PN is a very suitable and useful tool for the study of DES thanks to its modeling power and mathematical properties.

In a closed loop behavior, some states are called forbidden states and the controller must prevent the system from entering them. These states can be the ones that violate the specification or are deadlock states. In recent years some methods have been introduced for avoiding the forbidden states and controller synthesis [6-15].

The methods in [6, 13, 15] use conditions associated with the controllable transitions to solve the problem of forbidden states. The conditions prevent firings of the controllable transitions when the

firings lead to a forbidden state. But this method has a disadvantage since the dynamic of controller is not clear on the PN model. Theory of regions which was used in [9] is another method for solving the problem of forbidden states. This method generates some constraints where some of them verify the authorized states and the others are for preventing the system from entering the forbidden states. Solving these constraints generates some control places when adding them to the system leads to obtaining a maximally permissive behavior. The main drawbacks of the method based on regions theory are the large number of equations which must be solved to obtain the control places and the computations that take long time. Moreover, the number of control places may be large. Deadlock states in flexible manufacturing systems (FMSs) are major problems and the system must be avoided from entering them [16]. Using Siphon theory, the control places can be calculated to prevent the system from entering these states [17-21]. However, in general, some of the control places may be redundant and should be eliminated. In [22] and [23], methods are presented for eliminating the redundant control places with a major drawback, their computational complexity.

In [24], a method has been proposed for safe and conservative PN where it is possible to assign a linear constraint (called simply constraint) to each forbidden state. By using the idea in [25], these constraints can be enforced on the system using control places. But, when the number of forbidden states and consequently the number of constraints is large, a large number of control places must be added to the PN model of the system. This problem leads to a complicated model. However this number can be reduced [24]. To do that, it is possible to convert some constraints into one constraint. This simplification can be intuitively performed using the PN structural properties. In recent years, some efforts have been accomplished for reducing the number of constraints in safe PNs [26, 27]. Reducing the number of constraints in safe PN is important since these models can be easily converted in Sequential Function Chart (SFC) languages and can be used in Programmable Logic Controllers (PLCs) [28, 29].

The method in [26] uses the place invariant and partial place invariant properties to reduce the size and the number of constraints in safe and conservative nets. Another method for this purpose which eliminates the conservative limitation is proposed in [27] that uses the over-state concept. Over-states are the ones obtained from the main state (the main states are the ones that the over-states are deduced from them).Forbidding each over-state leads to forbidding the main states. Often an over-state of a specific forbidden state may be one of the over-states of some other forbidden states. So, by forbidding this over-state, avoiding two or more forbidden states is possible. Therefore it is sufficient to assign one constraint to this over-state. The over-states which are selected must be the ones such that preventing them leads to obtaining a maximally permissive behavior. However, the simplification results after using the last methods do not always correspond to the best results giving a large number of final constraints.

In [30] a method has been proposed for assigning constraints to forbidden states in non safe PNs. This method can generate a small number of constraints in the systems with shared resources by eliminating the redundant forbidden states using over-state concept. However, in the systems without shared resources and with no redundant forbidden states, the problem of large number of constraints remains. Moreover, for obtaining each constraint, an Integer Linear Programming (ILP) problem must be solved. The problem of large number of constraints may be solved by the method presented in [31] which eliminates the redundant constraints. Unfortunately, the computational complexity in this method may be high and it may not generate the least number of constraints. In [32], the authors have developed the method in [30] where a small number of constraints can be obtained by solving an ILP Problem. The drawback of this method is its computational complexity that makes it inapplicable to large scale PN models. This complexity is due to the number of variables and also to the number of constraints in ILP problem which are very large.

In this paper, two powerful ideas in safe PNs are proposed that allow performing more simplifications comparing to the previous techniques, without solving any ILP problems. The first idea presents a technique for constructing quasi partial invariant which can be used for reducing the number of constraints. Quasi Partial invariants are the inequalities verified by all the reachable markings and can be obtained from the invariant inequalities or directly from the PN structure. The second idea proposes a very efficient method for simplification of constraints. Using this idea, the reduction capacity is greater than before in safe PN. In the idea, instead of using the invariant relation or quasi partial invariant inequality, some states are checked which their unlikelihood of happening, leads to simplifying the constraints. These two new ideas are used after applying the presented method in [27] and are very efficient for simplifying the constraints. They convert a great number of constraints into a small number fulfilling the control objective. In this paper, after introducing the new method, some conditions are presented to show how the final controller is maximally permissive.

The rest of this paper is organized as follows. In Section 2, the important definitions are presented and the methods for constructing constraints from forbidden states and enforcing them on the system are explained. The methods for reducing the number of constraints which have been presented in recent years are introduced in Section 3 and an example is presented to show that these methods do not always give the least number of constraints. In Section 4, the new method for reducing the number of constraints is presented; this method is based on quasi partial invariants and semi quasi partial invariants. The conditions for having maximally permissive controller are explained in Section 5. In section 6, the new method in this paper is compared with the previous ones. Finally, the conclusion is presented in Section 7.

2. Preliminary presentation

In this section, some important definitions and basic concepts which are required for introducing the new ideas are presented.

2.1. Basics of Petri net and important definitions

A PN is represented by a quadruplet $\mathcal{R}=(P, T, W, M_0)$ where P is the set of places, $T = T_c \cup T_u$ is the set of controllable and uncontrollable transitions, W is the incidence matrix and M_0 is the initial marking.

In this paper the PN is supposed to be safe. Safe PNs are the ones that the number of tokens in each place is one or zero. So, the marking is Boolean. Here, the number of tokens in a place P_i is shown with m_i.

In a PN, the set of all reachable markings is stated with $\mathcal{M}_R$. $\mathcal{M}_R$ is divided into two subsets. The first subset is the set of authorized states $\mathcal{M}_A$ and the other one is the set of forbidden states $\mathcal{M}_F$. $\mathcal{M}_F$ is also separated into two groups as follows:

1) The states which violate the specification or are deadlock states or the states from which the system reach the deadlock states inevitably. These states are shown with $\mathcal{M}_F'$.

2) The states such that occurrence of uncontrollable events leads to the states in $\mathcal{M}_F'$.

After eliminating the set of forbidden states from the set of reachable states, the residual set is the set of authorized states.

In the set of forbidden states, there is a very important subset which is called the set of border forbidden states and is denoted as $\mathcal{M}_B$ [33]. This subset is defined below.

Definition 1: $\mathcal{M}_B$ is the set of border forbidden states and is defined as follows:

$$\mathcal{M}_B = \{M_i \in \mathcal{M}_F \mid \exists \sigma \in T_c and\ \exists M_j \in \mathcal{M}_A, M_j \xrightarrow{\sigma} M_i\}$$

where T_c is the set of controllable transitions.

❑

Preventing the reachability of the forbidden states is possible by disabling the controllable transitions when their firings lead to a border forbidden state,. Moreover, for avoiding all the forbidden states, it is sufficient to avoid the border forbidden states. So, in this paper for preventing all the forbidden states, we only forbid the border forbidden states.

Now, in Definitions 2, 3 and 4, the concepts of place invariant, partial place invariant and quasi partial place invariant are respectively introduced. These definitions are important for introducing the new ideas.

Definition 2 [34]: Let $\mathcal{R}$ be a PN, P the set of its places, $\mathcal{M}(M_0)$ the set of reachable markings from M_0 and k a constant. A subset of places $\mathcal{P}'=\{P_1,P_2,\ldots,P_r\}$ included in P with the following relation constitutes a Place invariant:

$$q_1 m_1 + q_2 m_2 + \ldots + q_r m_r = k,\ \forall\ M \in \mathcal{M}(M_0)$$

where the weights q_i for i= 1, …, r are positive integers and m_i is the number of tokens in place P_i. The set of places $\mathcal{P}'$ is a conservative component. Net $\mathcal{R}$ with the conservative component P is said to be conservative.

❑

If we remove some places from the set $\mathcal{P}'$, a partial invariant is obtained, itis defined as follows:

Definition 3: Let $\mathcal{P}' =\{P_1,P_2,\ldots,P_r\}$ be a place invariant in a PN $\mathcal{R}$, $\mathcal{P}_{i1}=\{P_1,P_2,\ldots,P_L\}$ for which $\{1,2,..L\}\subset\{1,2,..r\}$, is a partial place invariant (also called partial invariant) and it satisfies the following inequality:

$q_1 m_1 + q_2 m_2 + \ldots + q_L m_L \leq k$, $\forall M \in \mathcal{M}(M_0)$,

where q_1, q_2, …, q_n are positive integers.

❑

Sometimes this inequality cannot be deduced from the invariant relation but can be directly constructed from the PN structure. This concept is presented in Definition 4.

Definition 4: Let $\mathcal{P}=\{P_1,P_2,\ldots,P_t\}$ be a set of some places in a PN $\mathcal{R}$. $\mathcal{P}'$ is a quasi partial place invariant (also called quasi partial invariant) if:

$q_1\ m_1 + q_2\ m_2 + \ldots + q_t\ m_t \leq K$, $\forall M \in \mathcal{M}(M_0)$,

where K and q_1, q_2, …, q_t are positive integers.

❑

Remark 1: The partial invariant inequality is obtained from an invariant relation while the quasi partial invariant inequality is not necessarily obtained from the invariant relation. For the quasi partial

invariant, verifying by the authorized states is sufficient. The partial invariant is a special case of quasi partial invariant.

❑

2.2. Constructing constraints from forbidden states

The control objective addressed here is to prevent the system from entering the forbidden states. One way to achieve this goal is to build constraints. In [24], the authors have shown that in safe and conservative PNs, assigning some inequalities to forbidden states is possible (the inequalities restrict the weight sum of tokens in some places). These inequalities are called linear constraints (also called constraint) and are constructed as follows:

Suppose that in a safe and conservative PN, there is a forbidden state when the places P_{i1}, P_{i2}, …, P_{in} are marked. A constraint related to this forbidden state can be constructed as follows [24]:

$$\sum_{k=1}^{n} m_{ik} \leq n-1 \qquad (1)$$

where n is the number of marked places in this forbidden state, and m_{ik} is the number of tokens in place P_{ik}. Enforcing the constraint (1) on the system prevents it from entering the forbidden state.

For example, suppose that the state $M_i^T = [0\,1\,1\,0\,0\,1]$ is a forbidden state. This state can be rewritten in the form $P_2P_3P_6$. For this state n=3 and according to the relation (1), the constraint related to this state is as follows:

$$m_2+m_3+m_6 \leq 2.$$

For preventing the system from entering the forbidden states, some covering states of them can be prevented. So, it is possible to use the constraints related to the covering states to simplify the controller synthesis [27]. This is based on the concept of over-state which is introduced as follows:

Definition 5 [27]: let $M_2=P_{21}P_{22}\ldots P_{2m}$ be an accessible state. $M_1=P_{11}P_{12}\ldots P_{1n}$ is an over-state of M_2 if:

$$M_1 \leq M_2$$

❑

For example, the state $M_1= P_1P_5$ is an over-state of the state $M_2 =P_1P_3P_5P_7$. The concept of over-state is important since when the constraint related to an over-state is verified, the constraint related to the main state is verified too. For example the constraint related to the over-state $M_1=P_1P_5$ is $m_1+m_5 \leq 1$. This constraint verifies the constraint $m_1+m_3+m_5+m_7 \leq 3$ which is related to the state $M_2=P_1P_3P_5P_7$.

The concept of over-state must be carefully used since by forbidding the over-states, some authorized states may be suppressed. In Section 3, the idea in [27] is recalled to show how it is possible to choose the best over-states which forbidding them leads to reducing the number of constraints.

Remark 2: A constraint $m_1 + m_2 +\ldots+ m_n \leq k$ can be presented as follows:

$$\{(b_i, k),\ b_i=P_1\ldots P_n\} \ \text{ or } \ (P_1\ldots P_n, k)$$

❑

In this paper for enforcing the constraints on the system, the idea in [25] is applied. With each constraint, a control place is added to the PN model of the system. This method is explained in Section 2.3.

2.3. Control places

To calculate a control place corresponding to each linear constraint, the method in [25] is used. This method is based on the concept of invariant, and now it is briefly introduced. Consider the set of constraints as $L.M_P \leq b$ where M_P is the marking vector, L is a $n_c \times n$ matrix, b is a $n_c \times 1$ vector, n_c is the number of constraints and n is the number of places. In this method, with each constraint, a place is added to the model. Let W_p be the PN incidence matrix. For each constraint a row is added to W_p and these rows are denoted as W_c which are calculated as follows:

$$W_c = -L.W_p$$

W_c is added to W_p as the following form:

$$W = \begin{bmatrix} W_p \\ W_c \end{bmatrix}$$

If the initial marking of the model is M_{P0}, the initial marking for the added places can be calculated as follows:

$$M_{s0}=b-L.M_{P0}$$

Therefore, the initial marking of the controlled model is:

$$M_0 = \begin{bmatrix} M_{p0} \\ M_{s0} \end{bmatrix}$$

However, when the number of control places is large, the controller is complicated. This leads to the necessity for reducing the number of constraints [24]. In the next section the previous methods for reducing the number of constraints are explained.

3. Reducing the number of constraints by using the previous approaches

In this section the goal is to recall the ideas in [26, 27] for reducing the number of constraints, and an example is introduced to show that these methods don't always give the least number of constraints.

In [26] a method has been proposed where using the invariant and partial invariant properties simplifies the constraints. Using the invariant relation, the number of constraints can be reduced. For example suppose that there are three constraints as follows:

$$m_1+m_4+m_7 \leq 2,\ m_1+m_4+m_8 \leq 2\ m_1+m_4+m_9 \leq 2$$

If there is an invariant relation as ($m_7+m_8+m_9=1$), three constraints can be reduced to one constraint as follows:

$$\left.\begin{array}{l} m_1 + m_4 + m_7 \le 2 \\ m_1 + m_4 + m_8 \le 2 \\ m_1 + m_4 + m_9 \le 2 \end{array}\right\} \xrightarrow{m_7+m_8+m_9=1} m_1 + m_4 \le 1$$

Now suppose that the above invariant relation has been verified, but there are only two constraints as follows:

$$m_1+m_4+m_7 \le 2,\ m_1+m_4+m_8 \le 2$$

Then, for simplifying these constraints, the invariant relation should be changed into a partial invariant inequality as the following form:

$$m_7+ m_8 \le 1$$

By having this partial invariant inequality, the two constraints can be reduced to one constraint as follows:

$$m_1+m_4+m_7+m_8 \le 2$$

Now, we recall the idea for simplifying the constraints by using the partial invariant concept.

Proposition 1 [26]: In a safe and conservative PN, suppose that $C=\{(P_1P_{i1}\ldots P_{i(n-1)}, k), (P_2P_{i1}\ldots P_{i(n-1)}, k), \ldots, (P_rP_{i1}\ldots P_{i(n-1)}, k)\}$ is the set of constraints equivalent to forbidden states. If there is a partial invariant as follows:

$$m_1+m_2+\ldots+m_r \le 1$$

the r constraints can be reduced to one constraint as the following form:

$$(m_1+m_2+\ldots+m_r)+m_{i1}+\ldots+m_{i(n-1)} \le k$$

❑

The partial invariant inequality must be constructed from the invariant relation and it is the reason for the conservative limitation in Proposition 1.

In [27], there is an efficient method for simplification of the constraints. The method uses the over-state concept to perform simplification and the result is similar to using the invariant relations but without having these relations. This method is recalled in Algorithm 1.

Algorithm 1 [27]: Obtaining the small number of constraints using the over-state concept.

Input: The set of authorized states $\mathcal{M}_A$ and the set of border forbidden states $\mathcal{M}_B$.

Output: The small number of constraints for preventing the system from entering the forbidden states.

Step 1: Compute the set of all over-states of the border forbidden states ($\mathcal{M}_B$) which is called $\mathcal{B}_1$.

Step 2: Compute the set of all over-states of the authorized states ($\mathcal{M}_A$). This set is called $\mathcal{A}_1$.

Step 3: Remove the common over-states between $\mathcal{B}_1$ and $\mathcal{A}_1$ from the set $\mathcal{B}_1$. The new set is called $\mathcal{B}_2$.

Step 4: From $\mathcal{B}_2$, remove the states that their over-states exist in this set. The new set is called $\mathcal{B}_3$.

Step 5: Using the method in [24], assign a constraint to each over-state in the set $\mathcal{B}_3$. The set of these constraints is called C_3.

Step 6: In the set C_3, select the smallest number of constraints that avoiding them leads to avoiding all the forbidden states. The new set is called C_4, and is the set of final constraints (selecting

the final constraints is similar to final selection in Quine-McCluskey method for simplifying the logical expressions [35] and is expressed in Algorithm 2 in Appendix I. In Algorithm 2, C_1 is the set of constraints related to border forbidden states).

❑

Now, an example is introduced to explain the simplification method.

Example 1: Consider a system composed of two machines (or two process parts) and a robot. The role of the robot is discharging of the machines. The start commands are accomplished by firing the controllable transitions t_1, t_3 and the end of processes is accomplished by firing the uncontrollable transitions t_2, t_4, and t_5. Both machines can be turned on or off independently. In this example, one of the machines completes its work once while the operation of the other machine is cyclic.

The process parts of this system are illustrated in Fig. 1 where $<P_1t_1P_2t_2>$ indicates process 1 and $<P_3t_3P_4t_4P_3>$ indicates process 2. The specification model of the system is depicted in Fig. 2 and the closed loop model is illustrated in Fig. 3. Table 1 describes the places and transitions. The reachability graph of the system in Fig. 3 is presented in Fig. 4.

In the system with uncontrollable transitions, there may be a problem when the model of process and the model of specification are synchronized via an uncontrollable transition. The problem exists when the input places of the uncontrollable transition related to process are marked while the input places of this uncontrollable transition related to specification are unmarked. Due to the uncontrollability of the transition, the closed loop model cannot respect the firing rules of the PN model. For instance, in this example consider the uncontrollable transition t_4 in the closed loop model. The input places of this transition are P_4 and P_5 at which P_4 is related to process and P_5 is related to specification. Suppose that the system is in the state P_4P_6 (look at Fig. 4). In this state, according to the closed loop model, transition t_4 cannot fire but since in the process model, this transition is uncontrollable, the controller (or also the place related to specification (P_5)) cannot disable the transition to verify the specification. Therefore this state is a forbidden state [33]. So, the controller must disable the controllable transitions in special conditions (before entering the forbidden state) to prevent the system from entering the forbidden state. This concept is the same for the states $P_2P_3P_6$, $P_1P_4P_6$ and $P_2P_4P_6$. Therefore, these states are forbidden states. Moreover, state $P_2P_4P_5$ is a forbidden state since when the system is in this state, by firing the uncontrollable transition t_4, the forbidden state $P_2P_3P_6$ is obtained. In the closed loop model when the system is in these states, the states after firing the uncontrollable transitions are unknown and are indicated by ⊗.

Remark 3: It must be mentioned that the exact reachability graph of the closed loop model in Fig. 3 is similar to Fig. 4 by eliminating the sign ⊗ and the related arcs. We have shown the reachability graph in the form of Fig. 4 to express the problem of uncontrollable transitions and to say why some states are forbidden. If all the transitions are controllable, the closed loop model corresponds to the optimal controller, but when there are some uncontrollable transitions, it may not respect the specification. The whole explanation about finding the forbidden states and the problem of uncontrollable transitions are expressed in [33].

It is possible to calculate the set of border forbidden states from the reachability graph. For instance according to the border forbidden state definition, the forbidden state P_4P_6 is a border forbidden state since it is obtained by firing the controllable transition t_3 from the authorized state P_3P_6. The other border forbidden states are obtained in the same way.

In this example, the set of authorized states is:

$\mathcal{M}_A = \{P_3P_5, P_3P_6, P_4P_5, P_1P_3P_5, P_2P_3P_5, P_1P_3P_6, P_1P_4P_5\}$.

And the set of border forbidden states is:

$\mathcal{M}_B=\{P_4P_6, P_2P_4P_5, P_2P_3P_6, P_1P_4P_6\}$.

It is obvious that the model in Fig. 3 is not conservative and it is not possible to apply Proposition 1 on the example. But, Algorithm 1 can be applied on it. Now, according to this method, the sets of over-states of $\mathcal{M}_B$ and $\mathcal{M}_A$ are constructed and denoted as $\mathcal{B}_1$ and $\mathcal{A}_1$, respectively, as follows:

$\mathcal{B}_1 = \{P_1, P_2, P_3, P_4, P_5, P_6, P_2P_4, P_2P_5, P_4P_5, P_2P_3, P_2P_6, P_3P_6, P_1P_4, P_1P_6, P_4P_6, P_2P_4P_5, P_2P_3P_6, P_1P_4P_6\}$

$\mathcal{A}_1 = \{P_1, P_3, P_4, P_5, P_1P_3, P_1P_5, P_3P_5, P_1P_3P_5, P_2, P_2P_3, P_2P_5, P_3P_5, P_2P_3P_5, P_6, P_1P_6, P_3P_6, P_1P_3P_6, P_1P_4, P_4P_5, P_1P_4P_5\}$.

After this step, the states which are common between $\mathcal{B}_1$ and $\mathcal{A}_1$, are removed from $\mathcal{B}_1$ and the residual set is called $\mathcal{B}_2$:

$\mathcal{B}_2 = \{P_2P_4, P_2P_6, P_4P_6, P_2P_4P_5, P_2P_3P_6, P_1P_4P_6\}$.

Now, from $\mathcal{B}_2$, the states that their over-states are in $\mathcal{B}_2$ must be removed and the new set is called $\mathcal{B}_3$:

$\mathcal{B}_3=\{P_2P_4, P_2P_6, P_4P_6\}$.

This set contains the final over-states which preventing them leads to preventing all the forbidden states. The constraints related to the final over-states are deduced as follows:

$m_2+m_4\leq 1$, $m_2+m_6\leq 1$, $m_4+m_6\leq 1$.

In this example, the method in Algorithm 1 has reduced the number of constraints; however the reduction from 4 constraints to 3 constraints is not significant. Now, suppose that there is a relation like ($m_2+m_4+m_6 \leq 1$). This constraint doesn't forbid any of the authorized states and forbids all the border forbidden states. Then, it is a simpler solution, i.e., 1 instead of 3 constraints. In the next section, a new method is proposed to perform this simplification.

4. The new ideas for reducing the number of constraints

In the simplification method in [26] which was applicable for safe and conservative PN, it was possible to construct partial invariant inequalities from the invariant relations. For example:

$m_1+m_2+\ldots+m_n= k \quad \Rightarrow \quad m_1+m_2+\ldots+m_{n-1}\leq k.$

In a non conservative PN, we cannot construct these inequalities from the invariant property. But in this paper, we will show that using the structural properties of PN, constructing such inequalities in safe but not necessarily conservative PN is possible. These inequalities are called quasi partial invariant as it was defined in Section 2.1and may lead to reducing the number of constraints. In addition, we propose another idea for simplification of constraints which in some cases can perform more simplifications than the first idea in this paper and the previous methods. The second idea is very powerful for reducing the number of constraints.

4.1. Reducing the number of constraints by constructing quasi partial invariant

In this subsection, the objective is to present a simplification method by suppressing the conservative limitation. In Theorems 1 and 2, a new idea is proposed ensuring to construct quasi partial invariants in safe but not necessarily conservative PN.

Theorem 1: In a safe PN, suppose that m_{i1}, m_{i2} are the number of tokens in places P_{i1}, P_{i2} respectively. If $P_{i1}P_{i2}$ is not in the set of over-states of authorized states, a quasi partial invariant inequality can be constructed as follows:

$m_{i1}+m_{i2} \leq 1$.

❑

Proof. Suppose that the above constraint is not true. Then, we have:

$m_{i1} + m_{i2} > 1$.

So, for safe PN we have:

$m_{i1}+m_{i2} = 2 \rightarrow m_{i1}= m_{i2} = 1$.

This means that $P_{i1}P_{i2}$ is in the set of over-states of authorized states. But, this is not true. Then:

$m_{i1}+m_{i2} \leq 1$.

❑

Now, in Theorem 2, Theorem 1 is developed to construct another quasi partial invariant inequality with more places.

Theorem 2: In a safe PN, suppose that there is a quasi partial invariant inequality like $m_{i1}+m_{i2}+\ldots+m_{in} \leq 1$. If all the over-states $\{P_{i1}P_{i(n+1)}, P_{i2}P_{i(n+1)},\ldots, P_{in}P_{i(n+1)}\}$ are not in the set of over-states of the authorized states, we can obtain another quasi partial invariant as follows:

$m_{i1}+m_{i2}+\ldots+m_{in}+m_{i(n+1)} \leq 1$.

❑

Proof. The proof of this Theorem is similar to Proposition 1. Suppose that this relation is not true, so, we can write:
$m_{i1}+\ldots+m_{in}+m_{i(n+1)} > 1$.

Hence, we have:

$$\left.\begin{array}{l} m_{i1}+\ldots+m_{in}+m_{i(n+1)} = 2 \\ m_{i(n+1)} \leq 1 \text{ (for safe PN)} \\ m_{i1} +\ldots+ m_{in} \leq 1 \end{array}\right\} \Longrightarrow \left\{\begin{array}{l} m_{i(n+1)} = 1 \\ m_{i1} +\ldots+ m_{in} = 1 \end{array}\right.$$

$$m_{i1} +\ldots+ m_{in} = 1 \;\Rightarrow\; \exists\, m_{ik} = 1(\, k \in [1,n]).$$

Then $P_{ik}P_{i(n+1)}$ is an over-state of authorized states that is not true, then:

$m_{i1} + \ldots + m_{in} + m_{i(n+1)} \leq 1$.

❑

By using the new method (Theorems 1 and 2), conservative limitation is not necessary for constructing quasi partial invariant and this inequality is not obtained from invariant relation. So, we can reform Proposition 1 and eliminate the conservative limitation. This is performed in Theorem 3.

Theorem 3: In a safe PN**,** Let $\mathcal{M} = \{(P_1 P_{i1} \ldots P_{i(n-1)}, k), (P_2 P_{i1} \ldots P_{i(n-1)}, k), \ldots, (P_r P_{i1} \ldots P_{i(n-1)}, k)\}$ be a subset of constraints. If the authorized states verify the quasi partial invariant $m_1 + m_2 + \ldots + m_r \leq 1$, the r constraints are equivalent to one constraint as follows:

$$(m_1 + m_2 + \ldots + m_r) + m_{i1} + \mathrm{m}_{i2} \ldots + m_{i(n-1)} \leq k$$

❑

By using this method, the r constraints may be reduced to one constraint. The advantage of this method over the presented method in [26] is that the time and memory space for simplification are both small and the conservative limitation is eliminated. Moreover, the chance for having a quasi partial invariant is greater than obtaining a partial invariant deduced from invariant property (the quasi partial invariant can be constructed according to the authorized states). So, the results are often simpler than for the last method in general.

Now, to see the impact of constructing quasi partial invariant for reducing the number of constraints, we apply it on Example 1. In this example the final over-states which must be prevented are:

$\mathcal{B}_3 = \{P_2P_4, P_2P_6, P_4P_6\}$.

At first step, the new method is applied on the over-states P_2P_4 and P_2P_6 in the set $\mathcal{B}_3$. The state P_4P_6 is not in $\mathcal{A}_1$. So, according to Theorem 1, there is a quasi partial invariant inequality as:

$m_4+m_6 \leq 1$.

Looking at Theorem 3, it is possible to simplify the constraints related to these two over-states ($m_2+m_4 \leq 1$ & $m_2+m_6 \leq 1$) into one constraint as follows:

$m_2+m_4+m_6 \leq 1$.

This constraint forbids the third over-state in $\mathcal{B}_3$ (P_4P_6). Therefore, in this example, by using the new idea, the constraints related to final over-states are reduced to one constraint as follows:

$m_2+m_4+m_6 \leq 1$.

The above constraint forbids all the border forbidden states ($\mathcal{M}_B$) and verifies all the authorized states ($\mathcal{M}_A$). So, after enforcing it on the system, the obtained controller is maximally permissive.

Now, the control place for this constraint should be calculated. According to the constraint ($P_2P_4P_6$, 1), we have:

$L = [0\ 1\ 0\ 1\ 0\ 1]$.

So

$W_c = [-1\ 0\ -1\ 0\ 1]$.

The initial marking of the control place is:

$M_{s0} = m_{pc} = 1$.

The controlled PN model of the system in this example is depicted in Fig. 5.

It is obvious that the new idea (Theorems 1 and 2) helps the simplification of constraints while this was impossible using the method in [26].

The proposed method in this section and also the previous methods may not give the least number of constraints. To show this problem, we consider another system in Example 2. We will see that the number of simplified constraints by the last methods remains large. In this case, it will be shown that the constraints can be more simplified. Then a very efficient idea for simplifying the constraints will be proposed in Section 5.

Example 2: A similar example as the first one is considered. We suppose that there are three machines and two robots. The operation of each machine is cyclic. The closed loop PN model is presented in Fig. 6.

The method for calculating the border forbidden states and the authorized states is similar to Example 1. So, for this example, the result of the calculation is written. The set of authorized states is:

$\mathcal{M}_A=\{P_1P_3P_5P_7P_9,\ P_1P_3P_5P_8P_9,\ P_1P_3P_5P_7P_{10},\ P_1P_3P_5P_8P_{10},\ P_2P_3P_5P_7P_9,\ P_2P_3P_5P_8P_9,\ P_2P_3P_5P_7P_{10},\ P_1P_4P_5P_7P_9,\ P_1P_4P_5P_8P_9,\ P_1P_4P_5P_7P_{10},\ P_1P_3P_6P_7P_9,\ P_1P_3P_6P_8P_9,\ P_1P_3P_6P_7P_{10},\ P_2P_4P_5P_7P_9,\ P_2P_3P_6P_7P_9,\ P_1P_4P_6P_7P_9\}$.

And the set of border forbidden states is:

$\mathcal{M}_B=\{P_2P_4P_6P_7P_9,\ P_2P_4P_6P_8P_9,\ P_2P_4P_6P_7P_{10},\ P_2P_4P_6P_8P_{10},\ P_2P_4P_5P_8P_9,\ P_2P_4P_5P_7P_{10},\ P_2P_3P_6P_8P_9,\ P_2P_3P_6P_7P_{10},\ P_1P_4P_6P_8P_9,\ P_1P_4P_6P_7P_{10},\ P_2P_3P_5P_8P_{10},\ P_1P_4P_5P_8P_{10},\ P_1P_3P_6P_8P_{10}\}$.

By applying Algorithm 1 on this example, the simplified over-states are as follows:

$\mathcal{B}_4=\{P_2P_4P_6,\ P_2P_4P_8,\ P_2P_4P_{10},\ P_4P_6P_8,\ P_2P_6P_{10},\ P_4P_6P_{10},\ P_2P_8P_{10},\ P_6P_8P_{10},\ P_4P_8P_{10},\ P_2P_6P_8\}$.

The constraints related to the over-states in the set $\mathcal{B}_4$ are:

$C_4=\{(P_2P_4P_6,2),\ (P_2P_4P_8,2),\ (P_2P_4P_{10},2),\ (P_4P_6P_8,2),\ (P_2P_6P_{10},2),\ (P_4P_6P_{10},2),\ (P_2P_8P_{10},2),\ (P_6P_8P_{10},2),\ (P_4P_8P_{10},2),\ (P_2P_6P_8,2)\}$.

As it is obvious, by applying Algorithm 1, the number of over-states which must be forbidden is reduced to 10. These over-states are the ones that forbidding them leads to avoiding all the forbidden states and verifying all the authorized states (the states in $\mathcal{B}_4$ cover all the border forbidden states). So, the constraints related to them (C_4) can be enforced on the system. But, it is clear that the number of simplified constraints and consequently the number of control places is large. In addition, by applying the new method in this section, more simplification is not possible because there is no quasi partial invariant to simplify the constraints. In the next section, a powerful method is proposed to reduce the number of constraints more than before.

4.2. Simplification by semi quasi partial invariant

In this section the goal is to propose another new method such that reducing the number of constraints is possible more than before. Now, to show the base of this new method, we consider a simple example. Suppose that there are two constraints as F_1= ($P_1P_3P_4P_5$, 3) and F_2= ($P_2P_3P_4P_5$, 3). We want to see in which conditions these two constraints can be reduced to one constraint as F_3= ($P_1P_2P_3P_4P_5$, 3). For obtaining the constraint F_3 using the proposed method in Section 4, there must be a quasi partial invariant like $m_1+m_2 \leq 1$. But when there is not such a quasi partial invariant, obtaining the constraint F_3 by the last method is not possible (for example for the system in example 2, we cannot find a quasi partial invariant to perform more simplification). In the following, we show that it is not necessary to find a quasi partial invariant. This simplification method is called *semi quasi partial invariant*. As it is obvious, all the states which are forbidden by F_1 and F_2 are forbidden by F_3 too. So, our objective is that all the states which are authorized by F_1 and F_2 should be authorized by F_3. Now it is necessary to check the states which are authorized by F_1 and F_2, but not by F_3. These states can be achieved when the places P_1 and P_2 are marked. The states which are forbidden by F_3 but authorized by F_1 and F_2 are as follows:

$S = \{P_1P_2P_3P_4, P_1P_2P_3P_5, P_1P_2P_4P_5\}$.

It means that by enforcing the constraint F_3, the places P_1, P_2, P_3, P_4 cannot be marked at the same time and it is the same for the places P_1, P_2, P_3, P_5 and also for the places P_1, P_2, P_4, P_5. But by enforcing the constraints F_1 and F_2, these places can be marked at the same time. If we show that these states cannot be accessible in the model, we can use the constraint F_3 instead of the two constraints F_1 and F_2. This concept can be achieved when these states are not in the set of over-states of authorized states. It means that according to the model, when places P_1 and P_2 are marked at the same time ($m_1+m_2 \nleq 1$ and $m_1+m_2=2$), only one of the places P_3, P_4, P_5 can be marked ($m_3+m_4+m_5=1$). So, in this case, it is necessary that the authorized states verify the constraints $m_1+m_2+m_3+m_4 \leq 3$, $m_1+m_2+m_3+m_5 \leq 3$ and $m_1+m_2+m_4+m_5 \leq 3$. Therefore, when these constraints are verified by the authorized states, we can use the constraint F_3 instead of the two constraints F_1 and F_2 as follows:

$$\left.\begin{array}{l} m_1 + m_3 + m_4 + m_5 \leq 3 \\ m_2 + m_3 + m_4 + m_5 \leq 3 \\ m_1 + m_2 + m_3 + m_4 \leq 3 \\ m_1 + m_2 + m_3 + m_5 \leq 3 \\ m_1 + m_2 + m_4 + m_5 \leq 3 \end{array}\right\} \Rightarrow m_1 + m_2 + m_3 + m_4 + m_5 \leq 3.$$

After this simple example, we generalize and formalize this concept and propose a new idea to perform more simplification. This new method is presented in Theorem 4 and Corollary 1.

Theorem 4: In a safe PN, let $C = \{(P_{i1}P_1P_2\ldots P_n, n), (P_{i2}P_1P_2\ldots P_n, n)\}$ be a set of two constraints. Consider n over-states as follows:

$P_{i1}P_{i2}P_1P_2...P_{j-1}P_{j+1}\ldots P_n$ for $2 \leq j \leq n$-1
$P_{i1}P_{i2}P_2\ldots P_n$,
$P_{i1}P_{i2}P_1P_2\ldots P_{n-1}$

(These over-states are the ones which contain n+1 marked places from the set $\{P_{i1}P_{i2}P_1P_2\ldots P_n\}$ except the over-states $P_{i1}P_1P_2\ldots P_n$ and $P_{i2}P_1P_2\ldots P_n$).
If all of these over-states are not in the set of over-states of authorized states, then the two constraints can be reduced to one constraint as the following form:

$P_{i1}P_{i2}P_1P_2\ldots P_n$, n.

$m_{i1}+m_{i2}+m_1+m_2+\ldots+m_n \leq n$.

In this case, we have a semi quasi partial invariant for places P_{i1} and P_{i2}.

❑

Proof. The proof of this Theorem is clear. Suppose that the inequality $m_{i1}+m_{i2}+m_1+m_2+\ldots+m_n \leq n$ is not true, then $m_{i1}+m_{i2}+m_1+m_2+\ldots+m_n = n+1$, that means n+1 places from the set $\mathcal{P} = (P_{i1}, P_{i2}, P_1, P_2, \ldots, P_n)$ are marked. Therefore, at least one of the conditions in the Theorem is violated. So our supposition is not true.

❑

In Corollary 1, we extend Theorem 4 to propose a more comprehensive method for the simplification of constraints.

Corollary 1: In a safe PN, Suppose that there are the constraints $C_1 = (P_{i1}P_{i2}\ldots P_{im}P_1P_2\ldots P_n, n)$ and $C_2 = (P_{i(m+1)}P_1P_2\ \ldots P_n, n)$, that verify the authorized states. If all the over-states which contain n+1 marked places from the set $\mathcal{P} = (P_{i(m+1)}, P_{i1}, P_{i2}, \ldots, P_{im}, P_1, P_2, \ldots, P_n)$ don't exist in the set of over-states of authorized states, the two constraints C_1 and C_2 can be replaced by one constraint as follows:

$P_{i1}P_{i2}\ldots P_{im}\ P_{i(m+1)}P_1P_2\ldots P_n$, n.

❑

Proof: The proof of this Corollary is similar to the proof of Theorem 2. Suppose that the inequality $m_{i1}+m_{i2}+\ldots+m_{im}+m_{i(m+1)}+m_1+m_2+\ldots+m_n \leq n$ is not true, then $m_{i1}+m_{i2}+\ldots+m_{im}+m_{i(m+1)}+m_1+m_2+\ldots+m_n = n+1$, that means n+1 places is marked in the set $\mathcal{P} = (P_{i1}, P_{i2}, \ldots, P_{im}, P_{i(m+1),} P_1, P_2, \ldots, P_n)$. So, at least one of the conditions in the Corollary is violated.

❑

In the semi quasi partial invariant approach, a very small state space should be checked, so the chance for having an acceptable answer is high.

The obtained constraint using Theorem 4 and Corollary 1 may cover the other constraints before simplification. This concept is formalized in Theorem 5.

Theorem 5. Suppose that there are two constraints C_1 and C_2 as $C_1=\{(b_1,k_1),\ b_1 = P_{i1}\ldots P_{in}\}$ and $C_2 =\{(b_2,k_2),\ b_2 = P_{j1}\ldots P_{jm}\}$. If the two conditions $b_1 \subseteq b_2$ and $k_2 \leq k_1$ are true, then the constraint C_2 covers the constraint C_1 (It means that if C_2 is verified, C_1 is verified too). ❑

Proof: Suppose that $b_1 \subseteq b_2$, $k_2 \leq k_1$, but C_1 isn't covered by C_2. It means that there is at least one state M_1 that is forbidden by C_1 but is not forbidden by C_2. Violation of M_1 by C_1 means that $m_{i1}+ m_{i2}+\ldots + m_{in} > k_1$. By attention to $b_1 \subseteq b_2$ and $k_2 \leq k_1$, it is clear that $m_{j1}+ m_{j2}+\ldots + m_{jn} > k_2$ which means this state is forbidden by C_2 and it is a contradiction.

❑

The method in this section is formalized in Algorithm 3 in appendix I. Now, we apply this new method on Example 2 to show its effectiveness. The final constraints in section 4.1 were:

C_4={($P_2P_4P_6$,2), ($P_2P_4P_8$,2), ($P_2P_4P_{10}$,2), ($P_4P_6P_8$,2), ($P_2P_6P_{10}$,2), ($P_4P_6P_{10}$,2), ($P_2P_8P_{10}$,2), ($P_6P_8P_{10}$,2), ($P_4P_8P_{10}$,2), ($P_2P_6P_8$,2)}.

As it is obvious from this set, by using the proposed method in Section 4.1, and the previous methods [26, 27], these constraints cannot be simplified any more. Now we want to apply Theorem 4 and Corollary 1 on the set C_4 to perform more simplification. For applying Theorem 4 on this set, we select the constraints ($P_2P_4P_6$, 2) and ($P_2P_4P_8$, 2) as follows:

C_{41}= {($P_2P_4P_6$, 2), ($P_2P_4P_8$, 2)}.

The over-states $P_2P_6P_8$ and $P_4P_6P_8$ are not in the set of over-states of authorized states, so, it is possible to use Theorem 4 in the following form:

C_{41}= {($P_2P_4P_6$, 2), ($P_2P_4P_8$, 2) $\Rightarrow$ C'_{41}= {($P_2P_4P_6P_8$, 2).

Consider the third constraint in the set C_4 as follows:

C_{42}= {($P_2P_4P_{10}$, 2)}.

By applying Corollary 1 on the sets C'_{41} and C_{42}, it is clear that the over-states $P_2P_4P_6$, $P_2P_4P_8$, $P_2P_4P_{10}$, $P_2P_6P_8$, $P_2P_6P_{10}$, $P_2P_8P_{10}$, $P_4P_6P_8$, $P_4P_6P_{10}$, $P_4P_8P_{10}$, $P_6P_8P_{10}$ are not in the set of over-states of authorized states. Then these two constraints are simplified to one constraint: C'_{42}={($P_2P_4P_6P_8P_{10}$, 2)}.

C'_{42} covers all the residual constraints in the set C_4 (Theorem 5). So, the final set of simplified constraints is:
C_5 ={($P_2P_4P_6P_8P_{10}$, 2)}.

In this example, we have only one final constraint and there is no need for final selection (Algorithm 2 in Appendix I). Therefore, the final constraint that prevents the system from entering all the forbidden states is as follows:
C_6 ={($P_2P_4P_6P_8P_{10}$, 2)}.

By applying the proposed method in this section on Example 2, 10 constraints have reduced to 1 constraint. So, instead of 10 control places, only one control place needs to be added to the PN model.

The first idea in this paper needs the determination of the quasi partial invariant and allows easy constraints simplification. However the advantage of the second idea is performing simplification by checking a low state space and without having the quasi partial invariant. Moreover, in Theorem 4 and corollary 1, a small number of states should be deduced for comparing with the over-states of authorized states to reduce the number of constraints.

5. Maximally permissive controller

In this section the goal is to show that the controller is maximally permissive after enforcing the simplified constraints on the system. For this reason, it is necessary to verify two conditions:

- The final constraints must forbid all the border forbidden states.
- The authorized states should not be forbidden.

For checking the first condition, Theorem 5 could be used to show the covering property for the final constraints. Definition 6 is introduced to show how a simplified constraint (C_6) covers the constraints before this simplification (C_4).

Definition 6: Suppose that the set of constraints before simplification is:

$C_4=\{C_{11}=(b_1, k_1), C_{12}=(b_2, k_2), \ldots, C_{1n}=(b_n, k_n)\}$, $b_1 = P_{11}P_{12}...P_{1t_1}$, $\ldots, b_n = P_{n1}P_{n2}...P_{nt_n}$

and the set of simplified constraints is:

$C_6=\{C_{21}=(b_{i1}, k_{i1}), C_{22}=(b_{i2}, k_{i2}), \ldots, C_{2m}=(b_{im}, k_{im})\}$, $b_{i1} = P_{i11}P_{i12}...P_{i1q_1}$, $\ldots, b_{im} = P_{im1}P_{im2}...P_{imq_m}$

The relation $F : C_4 \times C_6 \rightarrow \{0,1\}$ is as:

$$F(C_{1j}, C_{2l}) = \begin{cases} 1 & b_j \subseteq b_{il} \ \& \ k_{il} \leq k_j \\ 0 & if \ not \end{cases}, \quad j = 1,...,n; \ l = 1,...,m$$

The term $F(C_{1j}, C_{2l}) = 1$ means that the constraint C_{1j} is covered by the constraint C_{2l}. The covering of a marking is an integer number:

$$D_v(C_{1j}) = \sum_{l=1}^{m} F(C_{1j}, C_{2l})$$

$D_v(C_{1j}) \geq 1$ means that, at least one of the constraints in C_6 covers the constraint corresponds to b_j.

❑

Now, to illustrate this part, we consider Example 2. We want to answer the question: Are all the constraints in C_4 covered by the constraints in C_6? To answer this question, Table 2 is constructed. The first row of this table represents the constraints in C_4 and the first column is the constraint in C_6.
Looking at Table 2, it is obvious that for each $C_{1j} \in C_4$, we have $D_v(C_{1j}) \geq 1$. So, all the constraints in C_4 are covered by the set C_6.

In this example, after applying the idea in [27] on the border forbidden states, 10 constraints were obtained. Enforcing these constraints (the set C_4) on the system leads to avoiding all the forbidden states and obtaining a maximally permissive controller [27]. Therefore, the constraints in the set C_4 forbid all the border forbidden states and according to Table 2, C_6 covers all the constraints in the set C_4. So, C_6 avoids all the border forbidden states.

Now in the following a theorem is introduced to show how the obtained controller can be maximally permissive.

Theorem 6: If for each $C_{1j} \in C_4$, $D_v(C_{1j}) \geq 1$, the obtained controller is maximally permissive (C_4 is the obtained constraints using the idea in [27]).

❑

Proof. The term "for each $C_{1j} \in C_4$, $D_v(C_{1j}) \geq 1$" means that all the constraints in C_4 are covered by at least one of the constraints in C_6, and all the constraints in C_6 forbid all the border forbidden states since C_4 forbids all the border forbidden states. Also, according to Theorem 4 and Corollary 1, the constraints

in C_6 don't forbid any of the authorized states. So, the states which are generated after enforcing C_6 on the system are equivalent to the set of authorized states. Therefore, the obtained controller is maximally permissive.

❑

In the case of Example 2, for each $C_{1j} \in C_4$, we have $D_v(C_{1j}) \geq 1$. So the controller is maximally permissive (Theorem 6).

Now, the control places for this example must be calculated. According to the simplified constraint ($P_2P_4P_6P_8P_{10}$, 2), we have:

L= [0 1 0 1 0 1 0 1 0 1] $\Rightarrow W_c$= [-1 0 0 -1 0 0 -1 0 0 1 1],

and the initial marking of the control place is equal to:

$M_{s0} = m_c$=2.

The controlled model is illustrated in Fig. 7. The control place and corresponding arcs are indicated in gray color and dashed lines.

For more coma deep comparison with the previous works, another example is considered below.

Example 3. Consider a system composed of two machines and a robot. The process part model related to the machines and the specification model are illustrated in Fig. 8 and Fig. 9 respectively. The closed loop model is depicted in Fig. 10. This model is similar to the model presented in [12, 36] imposing some changes (It is supposed that the model is safe and we have synchronized the specification model with the process model). In this model transitions t_5 and t_6 are controllable and transitions t_1, t_2, t_3, t_4 and t_7 are uncontrollable. Each one of the two machines constructs a product which has a redundant part and the robot must remove the redundant parts from the pieces. Then, the two products without redundant parts are assembled to construct a new product. Descriptions of places and transitions are presented in Table 3. In this example, the authorized states and the forbidden states are selected in a similar way than in Example 1.

In this system, the set of authorized states is as follows:

$\mathcal{M}_A$={P_1P_8, P_1P_9, $P_2P_3P_8$, $P_2P_3P_9$, $P_2P_5P_8$, $P_2P_7P_8$, $P_2P_7P_9$, $P_3P_4P_8$, $P_3P_6P_8$, $P_3P_6P_9$, $P_4P_7P_8$, $P_5P_6P_8$, $P_6P_7P_8$, $P_6P_7P_9$}.

And the set of border forbidden states is:

$\mathcal{M}_B$={$P_2P_5P_9$, $P_3P_4P_9$, $P_4P_5P_8$, $P_4P_5P_9$, $P_4P_7P_9$, $P_5P_6P_9$}.

By applying the method in [27], the final over-states which must be forbidden are as follows:

$\mathcal{B}$ = {P_4P_5, P_4P_9, P_5P_9}.

So, the corresponding constraints are:

C={$m_4+m_5 \leq 1$, $m_4+m_9 \leq 1$, $m_5+m_9 \leq 1$}.

But, by using the new method in this paper, these constraints can be reduced to a constraint as follows:

$m_4+m_5+m_9 \leq 1$.

According to this constraint, the control place can be calculated as the following form:

L=[0 0 0 1 1 0 0 0 1] & b=1 $\Rightarrow$ W_c=[0 0 0 0 -1 -1 1] & $M_{s0} = m_c$=1.

The controlled model of the system in this example is illustrated in Fig. 11. The control place and the corresponding arcs are depicted in grey color.

The complete Algorithm of controller synthesis for obtaining the simplified controller is presented in Algorithm 4 in Appendix I.

6. Comparing the new methods in this paper with the previous methods

The new ideas in this paper have beeb also applied on two other examples: 1) the manufacturing system presented in [26, 37], 2) the cat and mouse example presented in [25, 38].

In [26, 37] the considered example is a manufacturing system composed of two independent machines, two transfer robots of the parts and one test bench where the final products are tested. This system has 6 border forbidden states. By using the new method in this paper, the 6 constraints related to the border forbidden states are reduced into two constraints.

In cat and mouse example presented in [25, 38], there are 5 rooms at which the rooms are connected with doors. A cat and a mouse can circulate in the rooms. The problem is to control the doors so that the cat and the mouse can never be in the same room at the same time. In this model there are 5 constraints. But, by using the new ideas in this paper or using the previous methods [26, 27, 30, 32], it is not possible to reduce the number of these 5 constraints.

In Table 4, the number of final constraints after applying the new ideas in this paper is compared with the previous methods [26, 27]. As it is obvious from this table, the number of constraints after applying the new methods is smaller than or equal to the number after applying the previous methods [26, 27] which expresses the capability of the new method. However the method in [32] can generate a solution similar to our method but in this method an ILP problem with a large number of constraints and variables must be solved. For instance in Example 1, an ILP problem with 36 constraints and 27 variables, in Example 2 an ILP problem with 350 constraints and 200 variables, and in Example 3 an ILP problem with 57 constraints and 36 variables must be solved. Our method is simple and the time and memory space for reducing the number of constraints are very small. Of course it is not necessary to solve the ILP problems. The drawback of our method is its limitation which is applicable on safe PNs. In addition, generation of the reacability graph and also over-states is an exponential problem.

7. Conclusion

In this paper, we have proposed two ideas in safe PNs to reduce the number of constraints more than before. The first idea uses the concept of quasi partial invariant to reduce the number of constraints. Quasi partial invariants are the inequalities obtained from the invariants or directly deduced from the structure of PN model. Application of this idea is very simple but it does not always give the best solution. So, a second idea is proposed which does not use the quasi partial invariant concept for reducing the number of constraints, but checks the accessibility of some special states. The number of constraints can be reduced when the system cannot enter the special states. This idea uses the set of over-states of authorized states to check this condition. Finally, the conditions for obtaining the maximally permissive controller have been presented.

References

1. Ramadge, P. J. and W. M. Wonham, "Modular feedback logic for discrete event systems,". SIAM Journal of Control and Optimization., Vol. 25, No. 5, pp. 1202-1218 (1987).
2. Ramadge, P. J. and W.M. Wonham, "The control of discrete event systems,". Dynamics of discrete event systems [Special issue]. Proceedings of the IEEE., Vol. 77, No. 1, pp. 81-98 (1989).
3. Giua, A., Petri Nets as Discrete Event Models for Supervisory Control. Ph.D. Thesis (1992).
4. Moody, J.O. and P. Antsaklis, "Petri net supervisor for DES with uncontrollable and unobservable transition,". IEEE Trans. Automatic Control., Vol. 45, No. 3, pp. 462-476 (2000).
5. Krogh, B. H. and L. E. Holloway, "Synthesis of feedback control logic for discrete manufacturing systems,". Automatica., Vol. 27, No. 4, pp. 641-651 (1991).

6. Holloway, L. E., X. Guan and L. Zhang, "A generalization of state avoidance policies for controlled Petri Nets,". IEEE Trans. Autom. Control., Vol. AC-41, No. 6, pp. 804-816 (1996).
7. Park, J. and S.A. Reveliotis, "Algebraic synthesis of efficient deadlock avoidance policies for sequential resource allocation systems,". IEEE Trans Robot Automat., Vol. 16, No. 2, pp. 190−195 (2000).
8. Uzam, M., "An optimal deadlock prevention policy for flexible manufacturing systems using Petri net models with resources and the theory of regions,". Int. J. Adv. Manuf. Tech., Vol. 19, No. 3, pp. 192−208 (2002)
9. Ghaffari, A., N. Rezg and X. L. Xie, "Design of live and maximally permissive Petri Net controller using the theory of regions,". IEEE Transactions on Robotics and Automation., Vol. 19, No.1, pp. 137-142 (2003).
10. Giua, A. and X. Xie, "Control of safe ordinary Petri Nets using unfolding,". Discrete Event Dynamic Systems: Theory and Applications., Vol. 15, 349–373 (2005).
11. Reveliotis, S. A. and J. Choi, "Designing reversibility-enforcing supervisors of polynomial complexity for bounded Petri Nets through the theory of regions,". LNCS., pp. 322-341 (2006)
12. Basile, F., P. Chiacchio and A. Giua, "Suboptimal supervisory control of Petri Nets in presence of uncontrollable transitions via monitor places,". Automatica., Vol. 42, pp. 995-1004 (2006).
13. Dideban, A. and H. Alla, "Solving the problem of forbidden states by feedback control logical synthesis,". The 32nd Annual Conference of the IEEE Industrial Electronics Society, Paris, FRANCE (2006).
14. Li Z., M. Uzam and M. Zhou, "Deadlock control of concurrent manufacturing process sharing finite resources,". Int. J. Adv. Manuf. Tech., Vol. 38, pp.787-800 (2008).
15. Dideban, A. and & H. Alla, "Feedback control logic synthesis for non safe Petri nets,". The 13th IFAC symposium on information control problems in manufacturing, Moscow, Russia (2009).
16. Xing, K., M. C. Zhou., H. Liu and F. Tian, "Optimal Petri-Net-Based Polynomial-Complexity Deadlock-Avoidance Policies for Automated Manufacturing Systems,". IEEE Transaction on systems, man, and cybernetics-Part A: systems and humans., Vol. 39, pp. 188-199 (2009).
17. Li, Z. W. and M. C. Zhou, "Elementary Siphons of Petri Nets and Their Application to Deadlock Prevention in Flexible Manufacturing Systems,". IEEE Transaction on systems, man, and cybernetics-Part A: systems and humans., Vol. 34, pp. 38-51 (2004).
18. Li, Z. W. and M. C. Zhou, "Two-Stage Method for Synthesizing Liveness-Enforcing Supervisors for Flexible Manufacturing Systems Using Petri Nets,". IEEE Transaction on industrial, informatics., Vol. 2, pp. 313-325 (2006).
19. Li, Z. W., H. S. Hu and R. Wang, "Design of Liveness-Enforcing Supervisors for Flexible Manufacturing Systems Using Petri Nets,". IEEE Transaction on systems, man, and cybernetics-Part C: Applications and Reviews., Vol. 37, pp. 517-526 (2007).
20. Li, Z. W., M. C. Zhou and M. Jeng, "A Maximally Permissive Deadlock Prevention Policy for FMS Based on Petri Net Siphon Control and the theory of Regions,". IEEE Transaction on automation science and engineering., Vol. 5, pp. 182-188 (2008).
21. Li, Z. W., S. Zhu and M. C. Zhou, "A Divide-and-Conquer Strategy to Deadlock Prevention in Flexible Manufacturing Systems,". IEEE Transaction on systems, man, and cybernetics-Part C: Applications and Reviews., Vol. 39, pp. 156-169 (2009).
22. Uzam, M., Z. W. Li and M. C. Zhou, "Identification and elimination of redundant control places in petri net based liveness enforcing supervisors of FMS,". Int. J. Adv. Manuf. Tech., Vol. 35, pp. 150-168 (2007).
23. Li, Z. W. and H. S. Hu, "On systematic methods to remove redundant monitors from liveness-enforcing net supervisors,". Computers & Industrial Engineering., Vol. 56, pp. 53-62 (2009).
24. Giua, A., F. M. DiCesare and M. Silva, "Generalized mutual exclusion constraints on nets with uncontrollable transitions,". In Proc. IEEE int. conf. on systems, man, and cybernetics., pp. 974–799 (1992).
25. Yamalidou, K., J. Moody., M. Lemmon and P. Antsaklis, "Feedback control of petri nets based on place invariants,". Automatica., Vol. 32, No. 1, pp. 15–28 (1996).
26. Dideban, A. and H. Alla, "From forbidden state to linear constraints for the optimal supervisory control,". Control Engineering and applied Informatics (CEAI)., Vol. 7, No. 3, pp. 48-55 (2005).
27. Dideban, A. and H. Alla, "Reduction of constraints for controller synthesis based on safe Petri Nets,". Automatica., Vol. 44, No. 7, pp. 1697-1706 (2008).
28. Uzam, M. and A. Jones, "Discrete event control system design using automation Petri nets and their ladder diagram implementation,". Int. J. of Adv. Manuf. Tech., Vol. 14, No. 10, pp. 716–728 (1998).
29. Peng, S. and M. Zhou, "Ladder Diagram Petri-Net-based discrete event control design methods,". IEEE Transactions on systems, man, and cybernetics, part c: applications and reviews., Vol. 34, No. 4, November (2004).
30. Chen, Y., Z. W. Li., M. Khalgui and O. Mosbahi, "Design of a Maximally Permissive Liveness-Enforcing Petri Net Supervisor for Flexible Manufacturing Systems,". IEEE Transactions on automation science and engineering., Vol. 8, No. 2, pp. 374-393, Apr (2011).

31. Li, Z. W., M. M. Yan and M. C. Zhou, "Synthesis of Structurally Simple Supervisors Enforcing Generalized Mutual Exclusion Constraints in Petri Nets,". IEEE Transaction on systems, man, and cybernetics-Part C: Applications and Reviews., Vol. 40, pp. 330-340 (2010).
32. Chen, Y. and Z. W. Li, "Design of a maximally permissive liveness-enforcing supervisor with a compressed supervisory structure for flexible manufacturing systems,". Automatica., Vol. 47, No. 5, pp. 1028-1034, May (2011).
33. Kumar, R. and L. E. Holloway, "Supervisory control of deterministic Petri nets with regular specification languages,". IEEE Trans. Automatic Control., Vol. 41, No. 2, pp.245-249 (1996).
34. David, R. and H. Alla., Discrete, continuous, and hybrid Petri Nets. Springer (2005).
35. Morris Mano, M., Digital design. Prentice Hall (2001).
36. Uzam, M. "On suboptimal supervisory control of Petri nets in the presence of uncontrollable transitions via monitor places,". Int. J. Adv. Manuf. Tech., Vol. 47, pp. 567-579 (2010).
37. Kattan, B., Synthèse structurelle d'un contrôleur basée sur le Grafce, thesis, UJF, France (2004).
38. Wonham, W. M. and P. J. Ramadge, "On the supremal controllable sublanguage of a given language,". SIAM. J. control optim., Vol. 25, pp. 637-659 (1987).

APPENDIX I: ALGORITHMS

Algorithm 2. Selecting the final constraints

Input: C_1 and C_2 where $C_2=(C_{j1}, C_{j2}, \ldots, C_{jn})$ is the set of calculated constraints (the final constraints must be selected from this set) and $C_1=(C_{i1}, C_{i2}, \ldots, C_{ik})$ is the set of constraints that must be covered by the constraints in C_2.

Output: The set C_F which is the set of final constraints selected from the set C_2.

m is the number of constraints in C_1, $C_F=\phi$;

Step 1:

In the set C_2, select all the constraints in the following form:

If in the set C_1 there is a constraint C_{il} (for l=1, …, k) which is only covered by one of the constraints in C_2 (C_{jh} : h=1, …, n), *then* select the constraint C_{jh} and put it in the set C_F.

End if

Step 2:

if the number of constraints in C_F is greater than zero *then* in the set C_1, eliminate all the constraints which are covered by the selected constraints in C_F; and also in the set C_2, eliminate the selected constraints

End if

Step 3:

If $m > 0$

If in the set C_1 there is a constraint C_{il} (for l=1, …, k) which is covered by more than one of the constraints in C_2 (C_{jh}: h=1, …, n), *then* select the constraint from the set C_2 which covers the most number of constrains in the set C_1 (in the case of equality, the simplest constraint should be selected).

Put the selected constraints in the set C_F.

In the set C_1, eliminate all the constraints which are covered by the selected constraint.

In the set C_2, eliminate the selected constraint.

End if

Else

go to step 5

End if

Step 4:

go to step 3

Step 5:

C_F is the set of final constraint

Step6:

End;

Algorithm 3. More simplification by the second new idea

Input: The set of forbidden states M_B and the set of authorized states M_A.

Output: C_f which is the set of final constraints (the small number of constraints).

Step 1:

Apply the method in [27] on the set M_B and obtain the set of simplified constraints as $C_f = \{(P_{11}P_{12}...P_{1t_1}, k), \ldots, (P_{m1}P_{m2}...P_{mt_m}, k)\}$.

i=1, j=2, c_r=r^{th} component of C_f, n= the number of components of C_f.

Step 2:

if $i \geq n$

go to step 4

End if

Step 3:

Consider the two constraints c_i and c_j from the set C_f.

If all the over-states which contain k+1 places from the set ($P_{i1} \cup P_{i2} \cup ... \cup P_{it_i} \cup P_{j1} \cup P_{j2} \cup ... \cup P_{jt_j}$) are not in the set of over-states of authorized states, we can replace these two constraints by a constraint as ($P_{i1}P_{i2}...P_{it_i}P_{j1}P_{j2}...P_{jt_j}, k$), *then*, in the set C_f, replace c_i by this new constraint and remove c_j and go to step 2.

Else

if $j < n$, *then* $j=j+1$ and go to step 3.

Else $i=i+1$ and $j=i+2$ and go to step 2.

End if

End if

Step 4:

C_f is the set of final constraints.

Step 5:

End;

Algorithm 4: Complete algorithm for controller synthesis

Input: The set of forbidden states M_B and the set of authorized states M_A.

Output: final controller.

Step 1:

Compute the set of over-states B_1 for the set of border forbidden state M_B and the set of over-states A_1 for the set of authorized states M_A.

Step 2:

Compute the set of over-states B_2 by deleting the common over states between A_1 and B_1 from B_1.

Step 3:

Compute B_3 by deleting redundant over-states from B_2 (deleting the over-states that there over-states are exist).

Step 4:

if Corollary 1 in [27] is verified, *then* go to step 5

Else there is no maximally permissive controller and go to Step 10.

End if.

Step 5:

Apply Algorithm 2 for the first necessary selection in order to computing C_4.

Step 6:

Apply Algorithm 3 for more simplification (computing C_5).

Step 7:

Apply Algorithm2 for final selection for computing C_6.

Step 8:

Compute the control places from the set of constraints C_6 by the method in [25].

Step 9:

Transform PN model into a SFC language.

Step 10:

End;

Tables

Table 1. The role of places and transitions

Place	Place description	Transition	Transition description
P_1	machine 1 is in stand by state	t_1	the start command of machine 1
P_2	machine 1 is working	t_2	the end of process of machine 1
P_3	machine 2 is in stand by state	t_3	the start command of machine 2
P_4	machine 2 is working	t_4	the end of process of machine 2
P_5	the robot is idle	t_5	the end of process of robot
P_6	the robot is busy		

Table 2. The relation between the sets C_6 and C_4 in example 2

$C_4 \rightarrow$ $C_6 \downarrow$	$P_2P_4P_6$, 2	$P_2P_4P_8$, 2	$P_2P_4P_{10}$, 2	$P_4P_6P_{10}$, 2	$P_2P_6P_{10}$, 2	$P_4P_6P_8$, 2	$P_2P_8P_{10}$, 2	$P_4P_8P_{10}$, 2	$P_6P_8P_{10}$, 2
$P_2P_4P_6P_8P_{10}$,2	1	1	1	1	1	1	1	1	1
$D_v(C_{1j})$	1	1	1	1	1	1	1	1	1

Table 3. Description of places and transitions

Place	Place description	Transition	Transition description
P_9	The robot is busy	t_7	The command for releasing the robot
P_8	The robot is free	t_6	The start command of machine 2 for constructing a piece
P_7	Machine 2 is ready to start its task	t_5	The start command of machine 1 for constructing a piece
P_6	Machine 1 is ready to start its task	t_4	The command for removing the redundant part from piece 2
P_5	machine 2 is working	t_3	The command for removing the redundant part from piece 1
P_4	machine 1 is working	t_2	The command for assembling the two parts constructed by the two machines (without redundant parts)
P_3	The redundant part is removed by the robot from the constructed piece by machine 2	t_1	The end of process
P_2	The redundant part is removed by the robot from the constructed piece by machine 1		
P_1	the two pieces (without redundant parts) constructed by the machines are assembled to compose a new piece		

Table 4. Comparison of the method in this paper with the previous methods

The method	The number of border forbidden states	The number of final constraints by using the method in [26]	The number of final constraints by using the method in [27]	The number of final constraints by using the new method in this paper
Example 1 presented in this paper	4	4	3	1
Example 2 presented in this paper	13	9	10	1
Example 3 presented in this paper	6	3	3	1
The manufacturing system presented in [26, 37]	6	2	2	2
The cat and mouse example presented in [25, 38]	5	5	5	5
Sum	34	23	23	10