Available online at www.sciencedirect.com

ScienceDirect

Automatica 44 (2008) 1697–1706

automatica

www.elsevier.com/locate/automatica

Reduction of constraints for controller synthesis based on safe Petri Nets[☆]

Abbas Dideban[a,*], Hassane Alla[b]

[a] Electrical Eng. Department, Semnan University, Semnan, Iran
[b] Gipsa Lab, ENSIEG, BP46, 38402 Saint-Martin d'Hères, France

Received 29 January 2007; received in revised form 22 September 2007; accepted 24 October 2007
Available online 2 April 2008

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 Net invariants, maximal permissive controllers are determined.

Keywords: 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 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 their capability in modeling and its mathematical properties. Very active research in the field of controller synthesis for DES emerged during the last decade (Basile, Chiacchio, & Giua, 2006; Giua & Xie, 2005; Roussel & Giua, 2005).

In Basile et al. (2006), Moody and Antsaklis (2000) and Yamalidou, Moody, Lemmon, and Antsaklis (1996), the authors use marking invariants to determine algebraically the incidence matrix of the supervisor PNs model. This method is very simple to use. 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 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 Li 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 Ghaffari, Rezg, and Xie (2003) and Uzam (2002), 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.

[☆] This paper was not presented at any IFAC meeting. This paper was recommended for publication in revised form by Associate Editor Bart De Schutter under the direction of Editor Ian Petersen.

* Corresponding author. Tel.: +98 2313320036; fax: +98 2313320036.
E-mail addresses: adideban@semnan.ac.ir (A. Dideban), hassane.alla@inpg.fr (H. Alla).

0005-1098/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.automatica.2007.10.031

– The computation time for solving the set of integer equations can be very large.

In Giua, DiCesare, and Silva (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 et al. (1996) allows the building of 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 lead to a large number of control places. In Giua 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 the P-invariants property 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 & 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 Reachability Graph (RG) as an intermediate step for calculating the controller. Although the complexity of the computation of RGs 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 Section 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 & Alla, 2005, Chap. 1–3) and the theory of supervisory control (Ramadge & 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 Support(X) of a vector $X \in \{0, 1\}^N$ is:

Support(X) = the set of marked places in X. □

The support of vector $M_0^T = [1, 0, 1, 0, 0, 1, 0]$ is:
Support (M_0) = $\{P_1 P_3 P_6\}$; or more simply:
Support (M_0) = $P_1 P_3 P_6$

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

$\mathcal{M}_R$ denotes the set of PN reachable markings. In $\mathcal{M}_R$, two subsets can be distinguished: the set of authorized states $\mathcal{M}_A$ and the set of forbidden states $\mathcal{M}_F$. The set of forbidden states correspond to two groups: (1) the set of reachable states ($\mathcal{M}_{F'}$) 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 $\mathcal{M}_{F'}$.

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

$$\mathcal{M}_A = \mathcal{M}_R \setminus \mathcal{M}_F.$$

Among the forbidden states, an important subset is constituted by the border forbidden state denoted as $\mathcal{M}_B$.

Definition 3. Let $\mathcal{M}_B$ be the set of border forbidden state:

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

where Σ_c 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 Ma_1 and Ma_2 which can work independently. The starting and the end of the tasks on these machines are respectively realized by controllable events c_1 and c_2, and by uncontrollable events f_1 and f_2. When machine Ma_1 ends its task on a part, it stays available for a new task while machine Ma_2 has to transfer its produced part into a buffer before beginning a new task (event b_2). Both machines are activated simultaneously (event *start*) but each of them can be inactivated separately (events sp_1