Int J Adv Manuf Technol (2005) 25: 735–742
DOI 10.1007/s00170-003-1907-4

ORIGINAL ARTICLE

Han Zandong · Gibum Lee

Application of Petri nets for deadlock analysis and avoidance in flexible manufacturing systems

Received: 16 May 2003 / Accepted: 07 August 2003 / Published online: 3 November 2004

Abstract Unreasonable dispatching resources to jobs in flexible manufacturing system (FMS) may result in a deadlock situation. This serious situation is studied and avoided through Petri net (PN) analysis techniques in this paper. Firstly, a production Petri net (PPN) model for a given FMS is developed. Based on a certain set of resources in PPN, the concepts of a deadlock state and a potential deadlock state are introduced. Then, we present a deadlock avoidance method that consists of two parts. One is the construction of a deadlock state equation that describes the intrinsic relationship between resources assignation and a deadlock state in PPN. This equation is a necessary and sufficient condition for the occurrence of a deadlock situation. The other is the construction of a restrictive PN controller for each deadlock state equation. This restrictive PN controller can control the resources dispatching by excluding some enabled transitions from firing, consequently avoiding the deadlock. This method is minimally restrictive and allows the maximal use of resources not only for normal FMS, but also for special FMS with cyclic deadlock structure chain (i.e., a pathological type of circular waiting structure). Finally, two applications are given to illustrate the validity of this method. The results show that this method can be efficiently implemented in practical FMS.

Keywords Deadlock state equation · FMS · PN controller · PN model · Shared resource

1 Introduction

In order to compete in today's world market and provide diversified products, the modern manufacturing systems have moved away from old style fixed-hardware sequential assembly line with flexible manufacturing system (FMS), which can be quickly reconfigured to produce new products by simply modifying the FMS controller. FMS is composed of two main parts: the physical system and the management system or decision-making system (DMS). The physical system includes resources (such as robots, machines, transportation systems and buffers, etc.), which are often shared by several jobs. By dispatching resources to jobs when they are simultaneously requested by more than one job, the DMS can make the physical system work in a reasonable way that optimizes some targets such as productivity and WIP (work-in-process) level. Failure to suitably allocate shared resources in DMS may bring serious effects on the system performance, in some extreme cases resulting in a deadlock situation.

It is important to develop a control method to prevent the FMS from entering a deadlock situation. Many researchers have achieved a lot on the problem of deadlock in recent years. Banaszak and Krogh [1] presented a deadlock avoidance algorithm (DAA) in FMS with concurrently competing process flows. This algorithm is sufficient for avoiding deadlock, but has some restriction on efficient use of resources. Hsieh and Chang [2] developed a synthesizing deadlock avoidance controller (DAC), which realized deadlock-free and high resource utilization in FMS based on an untimed Petri net formalism. Ezpeleta et al. [3] developed a deadlock prevention policy for FMS based on a PN structure called siphon. Xing et al. [4] presented a deadlock avoidance policy based on the definition of deadlock structure that is a set of certain transitions. It is known that the above deadlock avoidance schemes lack efficient computational algorithms and aren't on-line with dynamic deadlock avoidance policy when the product types or processing routes are changed. Therefore, Lewis et al. [5] formulated a necessary and sufficient condition for deadlock in reentrant flowlines and developed a deadlock avoidance policy, which can be implemented on line. Some other deadlock avoidance methods are described in [6–22].

Most of the existing deadlock avoidance schemes are necessary to avoid deadlock for a certain kind of FMS. However, there are not minimal restriction policies in some cases,

H. Zandong (✉)
Department of Mechanical Engineering,
Tsinghua University,
Beijing 100084, P.R. China
E-mail: hanzd@mail.tsinghua.edu.cn
Fax: +86-10-62788099

G. Lee
Research Institute of Industrial Science & Technology,
Pohang 790-600, Korea

especially in the PN structure called cyclic deadlock structure chain [4], which is called pathological type of circular wait structure in [5]. Most of the above literatures don't concern the FMS with cyclic deadlock structure chains. Literature [4] analyses this kind of FMS in detail and provides a deadlock avoidance method, this method is useful in most case but doesn't work in the FMS of example 5.2. Therefore, it is necessary and sufficient for a good deadlock avoidance method to prevent FMS from deadlock. This paper is trying to provide a deadlock avoidance method that is deadlock-free while allowing the maximal use of resources.

This paper studies the problem of deadlock in FMS using PN analysis techniques. First, we introduce the basic concepts of Petri net and deadlock. Then we study the conditions in which a FMS must be in a deadlock situation or will result in a deadlock situation. Finally, we design a deadlock avoidance method that can be easily realized by PN controller. This method is minimally restrictive and allows the maximal use of resources.

The remainder of this paper is arranged as follows. The definitions needed in this paper and the Petri net models of FMS are given in Sect. 2. In Sect. 3, deadlock states in production Petri net (PPN) will be studied. A deadlock avoidance method is described in Sect. 4. The application and some illustrative examples are given in Sect. 5 and also in Sects. 2 and 4. The conclusions are drawn in the last section.

2 PN models for FMS with shared resources

In this section, we only present some basic definitions and notations associated with our research. For a more detailed description, please refer to [23, 24].

Definition 2.1: A marked Petri net is a 5-tuple, $Z = (P, T, I, O, M_0)$, where $P = \{p_1, p_2, \ldots, p_n\}$ is a finite set of places, $T = \{t_1, t_2, \ldots, t_m\}$ is a finite set of transitions, where $P \cap T = \Phi$, $I \subseteq P \times T$ is the input function, $O \subseteq T \times P$ is the output function, M_0: $P \to N$ is the initial marking, where N is the set of nonnegative integers.

Definition 2.2: A Petri net is called ordinary PN iff $I(p, t) \to \{0, 1\}$ and $O(t, p) \to \{0, 1\}$, for any $p \in P$ and $t \in T$.

Definition 2.3: For a given Petri net, $^*t = \{p|(p, t) \in I\}$ and $t^* = \{p|(t, p) \in O\}$ are the input places set and output places set of t. $^*p = \{t|(t, p) \in O\}$ and $p^* = \{t|(p, t) \in I\}$ are the input transitions set and output transitions set of p. Where (x, y) represents an arc form x to y.

Definition 2.4: For a marked Petri net, a transition t is enabled under a given marking M if the following condition is satisfied: $M(p) \geq I(p, t)$, for any $p \in {}^*t$. Firing the enabled transition t under M results in a new marking M': $M'(p) = M(p) + O(t, p) - I(p, t)$, for any place p.

Definition 2.5: For a marked Petri net, let S be a firing sequence of transitions under marking M, the firing of S from M will result in M', M' is called reachable from M, this will be given as $M[S > M'$. The set of all the reachable markings is denoted by $R(Z, M)$, or simply by $R(M)$. Where $R(M) = \{M'|M'$ is reachable from $M\}$.

Before describing the deadlock avoidance method, it is important to discuss how to model a FMS by PN. The modeling methodology will be illustrated by using a simple example in the following.

Example 2.1: Let us consider a workstation that includes two kinds of multi-operation machine M_1 and M_2. The capacity of each machine is five. A robot R can pick up and deliver products between M_1 and M_2. There are two types of product produced, q_1 and q_2 . The machine requirement sequence of q_1 and q_2 is $\sigma(q_1)$ and $\sigma(q_2)$, where $\sigma(q_1) = (M_1, M_2, M_1)$ and $\sigma(q_2) = (M_1, M_2)$. Referring to the method presented in [25], a Petri net model is given in Fig. 1.

The PN is composed of two parts: one is the set of available resources, the other is two processing routes corresponding to product type q_1 and q_2, respectively. Each processing route includes several places and transitions. The firing of source transition $t_{1,1}$ ($t_{2,1}$, resp.) means a raw material enters the processing route and begins to produce q_1 (q_2, resp.). The firing of sink transition $t_{1,4}$ ($t_{2,3}$, resp.) means the processing procedure of q_1 (q_2, resp.) finishes. This model implies that the quantity of row materials and the output buffer for each product are big enough and don't form an obstacle for the processing procedure. The firing of other transitions represent that one processing step completes and next processing step begins. Each place represents a processing step (job) in which a product is operated on a certain machine. For example, a token in place $p_{1,1}$ means that a semi-product of q_1 is processing on M_1, this job is denoted as $J_{1,1}$. Firing of $t_{1,2}$ means that $J_{1,1}$ completes and $J_{1,2}$ begins.

Let us consider a FMS in general, which consists of a set of resources $R = \{r_1, r_2, \ldots\}$. The capacity of each resource r is denoted as $C_r \in N$, which indicates the total number of resource r available in the system. The system can produce a set of products $Q = \{q_1, q_2, \ldots\}$. The processing route of a product q is separated into several steps. Each step represents a job that needs only one resource (this situation is not the general case, but many FMS hold this characteristic). The resource requirement sequence of product q is specified as $\sigma(q) = (r_q(1), r_q(2), \ldots, r_q(L_q))$, where $r_q(1)$ is a resource that is re-

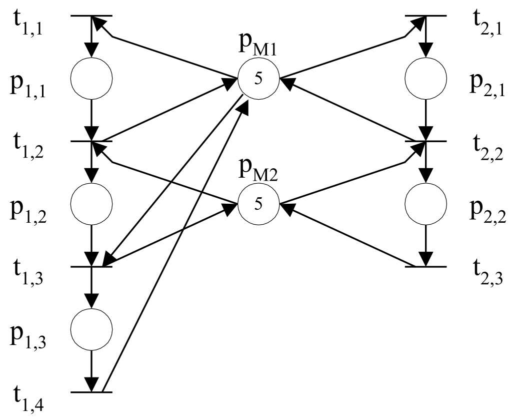

Fig. 1. Petri net model for the FMS in example 2.1

quired in the first processing step of product q, i.e., $J_{q,1}$ occupies $r_q(1)$, L_q is the length of the processing route of product q. The PN model $Z = (P, T, I, O, M_0)$ of the general FMS is an ordinary PN, which can be specified as follows:

$$P = \{p_{q,i} | q \in Q, i = 1, 2, \ldots, L_q\} \cup \{p_r | r \in R\}$$
$$T = \{t_{q,i} | q \in Q, i = 1, 2, \ldots, Lq + 1\}$$
$$I = \{(p_{q,i}, t_{q,i+1}), (p_{r_q(i)}, t_{q,i}) | q \in Q, r \in R, i = 1, 2, \ldots, L_q\}$$
$$O = \{(t_{q,i}, p_{q,i}), (t_{q,i+1}, p_{r_q(i)}) | q \in Q, r \in R, i = 1, 2, \ldots, L_q\}$$
$$M_0(p) = \begin{cases} C_r & p \in \{p_r | r \in R\} \\ 0 & \text{otherwise} \end{cases}$$

This model is called production Petri net (PPN). The initial marking represents an idle system in which all the resources are available.

Except for the source transition $t_{q,1}$ and sink transition $t_{q,Lq+1}$, the input places' set of transition t can be separated into two parts as: ${}^*t = {}^*t_r \cup {}^*t_J$, where ${}^*t_r = {}^*t \cap p_r$, and ${}^*t_J = {}^*t \cap p_q$. The output places' set of t can also be separated into two parts as: $t^* = t_r^* \cup t_J^*$, where $t_r^* = t^* \cap p_r$, and $t_J^* = t^* \cap p_q$. ${}^*t_r$ and t_r^* represent the places of resources, ${}^*t_J$ and t_J^* represent the places of jobs.

3 Deadlock states in PPN

When several jobs compete for a finite set of resources, it is possible to result in deadlock situation. A commonly used definition of deadlock in FMS is as follows. Interested readers may refer to [1].

Definition 3.1: Given sets of resources R, products Q, and PPN model of the FMS, a set of transitions $T' \subset T$, is said to be in deadlock for a marking $M \subset R(M_0)$, if (1) all transitions in T' are process enabled under marking M, and (2) no transition in T' is resource enabled for any $M' \subset R(M)$. It means that $M({}^*t_J) > 0$ and $M'({}^*t_r) = 0$ for any $t \in T'$ and $M' \subset R(M)$.

This definition is based on a certain set of transitions. It is reasonable and useful to analyze the deadlock situations. However, this definition doesn't combine the PN technique with the concept of circular wait. Therefore, the physical meaning is not easy to understand by mechanical engineers.

In a commonly used FMS, the following four restrictive conditions are almost always presented. The FMS discussed in this paper is restricted to these conditions.

1. Mutual exclusion: each job needs only one resource, and each resource can be used for only one job at a time.
2. No preemption: a resource can not be released until its job is finished.
3. Holding while waiting: a job holds resources while waiting for required resource to be available.
4. Integrated job: there are not two successive jobs using the same resource.

Under these restrictions, it is known that a necessary condition for deadlock to occur is that there is a circular wait relation among resources [5]. Nonreasonable resource dispatching in circular waiting places may result in deadlock. Therefore, a definition of deadlock is given here based on a set of certain resources.

Definition 3.2: A set of resources $R_D \in R$ is said to be in a deadlock state under marking M if it satisfies the following two conditions: (1) All the resources in R_D are occupied, i.e., $M(p_r) = 0$ for $r \in R_D$. (2) All the jobs occupying resources of R_D are waiting for other resources of R_D, i.e., ${}^*(p_q^*)_r \in \{p_r | r \in R_D\}$ for $p_q \in \{p_q | R(p_q) \in R_D,\ M(p_q) > 0\}$, where $R(p_q)$ denotes the resource occupied in processing step p_q.

Marking M is called deadlock marking, denoted as M_D. The definition means all the resources in R_D can not be released under M_D. Therefore, in order to avoid deadlock, it is significant to find the deadlock marking M_D.

Property 3.1: For a PPN and a resource $r \in R$, there must exist a P-invariant:

$$M(p_r) + \sum_{\{p_q | R(p_q) = r\}} M(p_q) = C_r$$

This property is based on the following truth: the firing of $t \in p_r^*$ will remove one token from p_r, and reside a token in $t_J^* \in \{p_q | p_q \in (p_r^*)_J^*\}$. The firing of $t \in^* p_r$ will reside a toke to p_r and remove one token from ${}^*t_J \in \{p_q | p_q \in {}^*({}^*p_r)_J\}$. Because $\{p_q | R(p_q) = r\} = \{p_q | p_q \in (p_r^*)_J^*\} = \{p_q | p_q \in {}^*({}^*p_r)_J\}$, the total tokens number in place p_r and $p_q \in \{p_q | R(p_q) = r\}$ is equal to the initial number C_r.

Definition 3.2 and proerty 3.1 establish a sufficient condition that leads to the following theorem.

Theorem 3.1: R_D is in a deadlock state under marking M, if and only if M satisfies the following equation.

$$\sum_{\{p_q | R(p_q) \in R_D, *(p_q*)_r \in p_r, r \in R_D\}} M(p_q) = \sum_{r \in R_D} C_r \qquad (1)$$

This equation is called a deadlock state equation. The set $\{p_q | R(p_q) \in R_D, {}^*(p_q^*)_r \in p_r, r \in R_D\}$ is called the critical places set for R_D.

Proof: From property 3.1, we get

$$\sum_{r \in R_D} M(p_r) + \sum_{\{p_q | R(p_q) \in R_D\}} M(p_q) = \sum_{r \in R_D} C_r \qquad (2)$$

Considering

$$\sum_{\{p_q | R(p_q) \in R_D\}} M(p_q) = \sum_{\{p_q | R(p_q) \in R_D, *(p_q*) r \in pr, r \in R_D\}} M(p_q) + \sum_{\{p_q | R(p_q) \in R_D, *(p_q*) r \notin p_r, r \in R_D\}} M(p_q)$$

Eq. 2-Eq. 1, we have

$$\sum_{r \in R_D} M(p_r) + \sum_{\{p_q | R(p_q) \in R_D, *(p_q*) r \notin pr, r \in R_D\}} M(p_q) = 0 \qquad (3)$$

Because the token number in any place can't be negative, each of the two parts in the left side of Eq. 3 must be zero.

$\sum_{r \in R_D} M(p_r) = 0$ meets condition (1) in definition 3.2.

From

$$\sum_{\{p_q|R(p_q)\in R_D,{}^*(p_q^*)r\notin p_r, r\in R_D\}} M(p_q) = 0,$$

we have ${}^*(p_q^*)_r \in \{p_r|r \in R_D\}$ for $p_q \in \{p_q|R(p_q) \in R_D, M(p_q) > 0\}$, condition (2) in definition 3.2 is also satisfied.

Applying theorem 3.1 to example 2.1, we can find for $R_D = \{M_1, M_2\}$, the deadlock state equation is:

$$M(p_{1,1}) + M(p_{1,2}) + M(p_{2,1}) = C_{M1} + C_{M2} = 10.$$

Any marking M that satisfies this equation is in deadlock.

From theorem 3.1, we can get the following deduction.

Deduction 3.1: If R_D includes only one resource, then R_D is never in deadlock under any reachable marking M while other resources are available.

Proof: Let $R_D = \{r_1\}$, assume R_D is in a deadlock state under marking M. By theorem 3.1, M has to satisfy the following deadlock state equation

$$\sum_{\{p_q|R(p_q)=r_1,*(p_q*)_r=p_{r1}\}} M(p_q) = C_{r1} \tag{4}$$

Because there are not two successive jobs using the same resource, we have $\{p_q|R(p_q) = r1, {}^*(p_q^*)r = p_{r1}\} = \Phi$, therefore, Eq. 4 is never true, i.e., R_D is never in deadlock under any marking M.

A special state sometimes occurs in FMS with cyclic deadlock structure chain [4]. We call it a potential deadlock state. Its definition is given as follows.

Definition 3.3: A set of resources $R_D \in R$ is said to be in a potential deadlock state (PDS) under marking M if it satisfies the following two conditions: (1) All the resources in R_D are occupied, i.e., $M(p_r) = 0$ for $r \in R_D$. (2) There is at least one job that can release resource r, but will result in another deadlock state in the future.

Marking M is called potential deadlock marking, also denotes as M_D. The PDS means that although the token in a certain processing step (job) can move to the next step under M_D, it can never advance to the sink transition of this processing route, i.e., the semi-product required resource of R_D will never be produced out.

It is easy to understand that theorem 3.1 can not be applied to get the PDS equation. Therefore, we develop an algorithm to construct the PDS equation based on the definition of PDS. Since PDS is based on the set of resources R_D, the PDS equation includes only the processing steps (places) that require resources of R_D.

Algorithm 3.1: Construction of the PDS equation

Step1: Let all the resources $r \notin R_D$ be available, i.e., $M(p_r) = C_r$ for $r \notin R_D$. This step makes the constructed PDS equation minimally restrictive.

Step2: By property 3.1, we have

$$\sum_{r\in R_D} M(p_r) + \sum_{\{p_q|R(p_q)\in R_D\}} M(p_q) = \sum_{r\in R_D} C_r \tag{5}$$

Step3: From the definition 3.3 (1), we have

$$\sum_{r\in R_D} M(p_r) = 0 \tag{6}$$

Substitute Eq. 6 into Eq. 5, we can get

$$\sum_{\{p_q|R(p_q)\in R_D\}} M(p_q) = \sum_{r\in R_D} C_r \tag{7}$$

Step4: If $R(p_{q,i}) \in R_D$, and for any $k > i$, $R(p_{q,k}) \notin R_D$, then let

$$M(p_{q,i}) = 0 \tag{8}$$

Substitute Eq. 8 into Eq. 7, we have

$$\sum_{\{p_{q,i}|R(p_{q,i})\in R_D, \exists k>i\ R(p_{q,k})\in R_D\}} M(p) = \sum_{r\in R_D} C_r \tag{9}$$

Step5: For all $p_{q,i}$ such that $R(p_{q,i}) \in R_D$, $R(p_{q,(i+1)}) \notin R_D$, and exist $k > i+1$, $R(p_{q,k}) \in R_D$, if the token in $p_{q,i}$ can arrive at the place $p_{q,Lq}$ by a firing sequence from any marking satisfied Eq. 9, then let

$$M(p_{q,i}) = 0 \tag{10}$$

Step6: Substitute Eq. 10 into Eq. 9, we can get the PDS equation. All the places included in PDS equation form the critical places set for R_D.

In step4, the condition $[R(p_{q,i}) \in R_D$, and for any $k > i$, $R(p_{q,k}) \notin R_D]$ means that the token in $p_{q,i}$ can arrive at the place $p_{q,Lq}$ under the restriction of Eq. 7. So, $M(p_{q,i}) > 0$ is not a deadlock state. It has to be removed from Eq. 7. For the same purpose, step5 is implemented to find out the places whose token can arrive at the input places of sink transitions under the restriction of Eq. 9. These kind of places have to be suppressed in Eq. 9.

The above algorithm can also be applied to construct a deadlock state equation. In some cases, it is difficult to determine if there exists a PDS for a set of resources in general, therefore, we apply the above algorithm to obtain a deadlock state.

4 Deadlock avoidance method

A deadlock state (to simplify, we just use deadlock state to represent both deadlock state and potential deadlock state in the following sections) is caused by unreasonable assignation of resources. It can be avoided by controlling the resources assigning. In the paper, we develop a deadlock avoidance method based on a deadlock controller, which prevents some able transitions form firing and consequently controls the assignation of resources. Its scheme is shown in Fig. 2.

The deadlock controller removes some able transitions from firing by the following steps:

1. The deadlock controller gains the set of able transitions under present marking M of PPN.

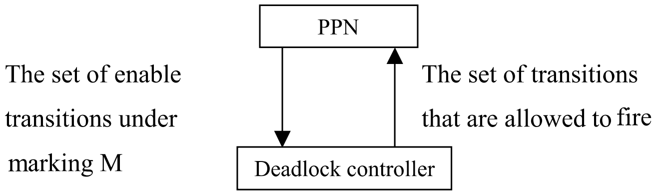

Fig. 2. Scheme of the deadlock avoidance method

2. For an abled transition t, calculate the reachable marking M' assuming t has been fired by $M'(p) = M(p) + O(t, p) - I(p, t)$.
3. If M' satisfied one of the deadlock state equations or PDS equations of PPN, t is forbidden to fire. Otherwise t can be fired freely.

The Deadlock controller can be implemented by many approaches. It will be realized by restrictive PN controller in this paper.

Let's consider example 2.1 again. The deadlock state equation $M(p_{1,1}) + M(P_{1,2}) + M(p_{2,1}) = 10$ will never be satisfied under the restrictive PN controller shown in Fig. 3a.

Because there is a self-loop between P_{c1} and $t_{1,2}$, the firing of $t_{1,2}$ does not change the validity of the corresponding deadlock state equation. Therefore, the arc $(P_{c1}, t_{1,2})$ and $(t_{1,2}, P_{c1})$ can be removed, as is shown in Fig. 3b.

For any deadlock state based on R_D, the corresponding restrictive PN controller can be constructed as follows:

1. Add a place P_c to PPN, let $M_0(P_c) = \sum_{r \in R_D} C_r - 1$.
2. Add an input arc to every input transition of critical place form P_c, and add an output arc form every output transition of critical place to P_c.
3. If there exists a self-loop between place p_c and transition t, then remove the arcs (p_c, t) and (t, p_c).

Based on the above discussion, a given FMS is deadlock-free if and only if every set of resources is deadlock-free. Therefore, in order to assure a PPN to be deadlock free, we develop a deadlock avoidance method as follows.

Algorithm 4.1: Deadlock avoid method

1. For any subset $R_D \in R$, by algorithm 3.1, construct all the deadlock state equations.
2. For each deadlock state equation, construct a restrictive PN controller.
3. In some cases, several PN controllers can be emerged into one to simplify the whole control system.

The PPN is in a deadlock state under marking M iff M satisfies a certain deadlock state equation(s). Therefore the proposed deadlock avoid method is necessary and sufficient to prevent the PPN from deadlock. The PN controller is deadlock-free while it allows the maximal use of resources.

5 Applications

In this paper, two examples will be presented to illustrate how to apply the proposed method to prevent deadlock.

Example 5.1: Let us consider a FMS that is similar to [1, 4], as is shown in Fig. 4.

There are four workstations W_1 to W_4 served by a single transport system. Two types of products q_1 and q_2 are produced in this system. Each workstation consists of one machine, an input buffer and an output buffer, each buffer has a capacity of five. Labels I_1, I_2, I_3, I_4 and O_1, O_2, O_3, O_4 denote the input and output buffers for W_1, W_2, W_3 and W_4, respectively. Then the set of resources is R={I_1, I_2, I_3, I_4 O_1, O_2, O_3, O_4}. The machine requirement sequences of q_1 and q_2 are (M_3, M_1, M_2, M_1) and (M_2, M_3, M_4), respectively. Therefore, the processing routes of q_1 and q_2 are specified as follows:

$$\sigma(q_1) = (M_3, M_1, M_2, M_1) = (I_3, O_3, I_1, O_1, I_2, O_2, I_1, O_1)$$
$$\sigma(q_2) = (M_2, M_3, M_4) = (I_2, O_2, I_3, O_3, I_4, O_4)$$

The PPN model for this FMS is given in Fig. 5.

According to [4], there is no PDS in a PPN if: (1) there is no cyclic deadlock structure chain, or (2) for any cyclic deadlock structure chain and any key resource r, the capacity of r is greater than one. Interested readers can refer to [4]. Since the capacity of each type of resources is five, there is no PDS in this PPN. Therefore we can apply theorem 3.1 to obtain the deadlock state equations for each subset $R_D \in R$.

For $R_D = \{I_1, O_1\}$, the deadlock state equation is $M(p_{1,3}) + M(p_{1,7}) = C_{I1} + C_{O1} = 10$ (in order to simplify the equation,

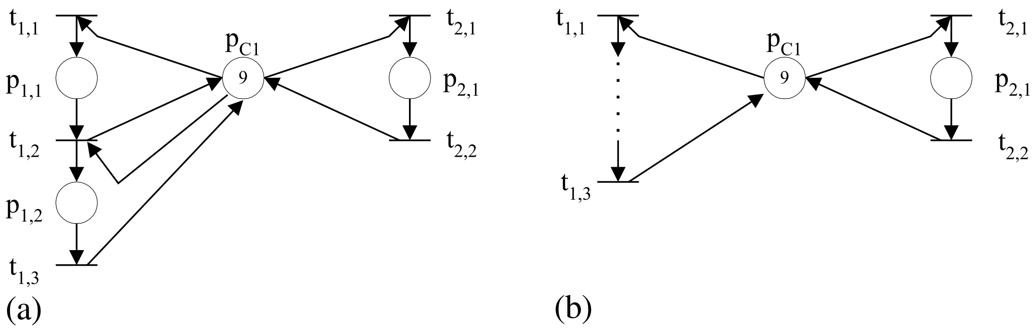

Fig. 3. Restrictive PN controller for the FMS in example 2.1

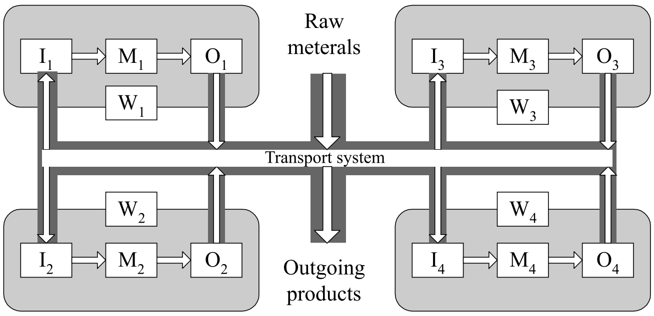

Fig. 4. A flexible manufacturing system with four workstations

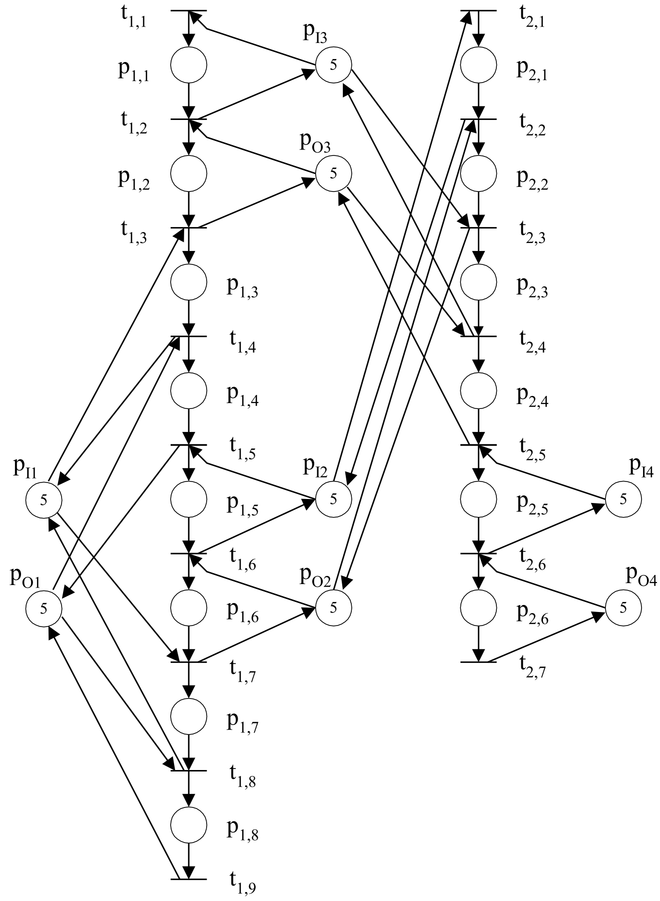

Fig. 5. Petri net model for the FMS of example 5.1

we replace $M(X)$ with X in the equation of this section, equation $M(p_{1,3}) + M(p_{1,7}) = 10$ becomes $p_{1,3} + p_{1,7} = 10$). According to property 3.1, we have $p_{1,3} + p_{1,7} + p_{I1} = 5$, hence the deadlock state equation $p_{1,3} + p_{1,7} = 10$ will never be satisfied under any marking M. Therefore, $\{I_1, O_1\}$ are never in a deadlock state.

Implementing the same process, all the possible deadlock state equations are give as follows.

For $R_D = \{I_1, O_1, I_2, O_2\}$, the deadlock state equation is

$$p_{1,3} + p_{1,4} + p_{1,5} + p_{1,6} + p_{1,7} + p_{2,1} = 20$$

For $R_D = \{I_1, O_1, I_2, O_2, I_3, O_3\}$, the deadlock state equation is

$$p_{1,1} + p_{1,2} + p_{1,3} + p_{1,4} + p_{1,5} + p_{1,6} + p_{1,7} + p_{2,1} + p_{2,2} + p_{2,3} = 30$$

For the above two deadlock state equations, we can construct two restrictive PN controllers to avoid deadlock by algorithm 4.1, as is shown in Fig. 6. Applying the two PN controllers, the FMS will never be in a deadlock state while it allows the maximal use of resources.

Example 5.2: Let us consider another example to illustrate the concept of PDS. There are four types of machines M_1, M_2, M_3, M_4 served by a single transfer robot R in the FMS. The capacity of M_1 and M_4 are both five, the capacity of M_2 and M_3 is only one. This system produces two types of products q_1 and q_2. The processing routes of q_1 and q_2 are specified as follows:

$$\sigma(q_1) = (M_1, M_2, M_3, M_4)$$
$$\sigma(q_2) = (M_4, M_3, M_2, M_1)$$

The PPN model for this FMS is given in Fig. 7.

In this PPN, there are at least three cyclic deadlock structure chains by [4]: $V_1 = \{D_1, D_2\}$, $V_2 = \{D_2, D_3\}$, $V_3 = \{D_1, D_2, D_3\}$, where $D_1 = \{t_{1,2}, t_{2,4}\}$, $D_2 = \{t_{1,3}, t_{2,3}\}$, $D_3 = \{t_{1,4}, t_{2,2}\}$. Since the available number of M_2 and M_3 is only one, there may exist

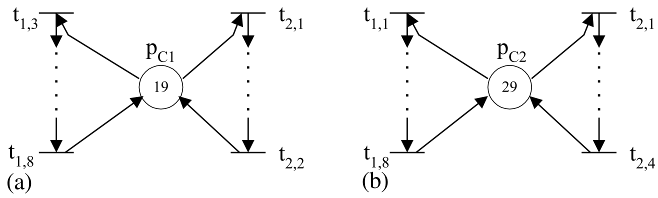

Fig. 6. PN controllers for the FMS of example 5.1

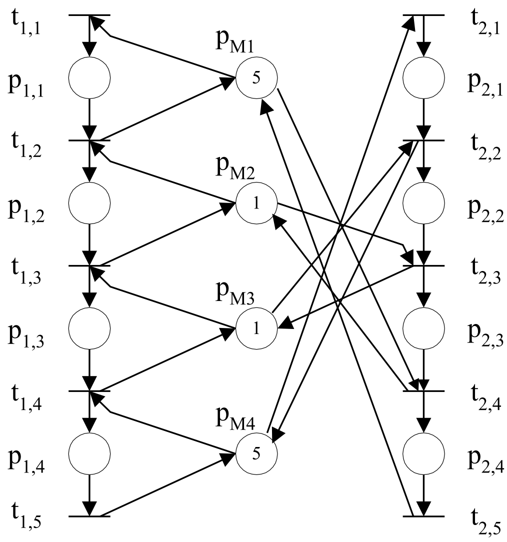

Fig. 7. Petri net model for the FMS of example 5.2

PDS in this PPN. Therefore we apply algorithm 3.1 to construct the deadlock state equations.

For $R_D = \{M_1, M_2\}$

From Eq. 7, we get

$$p_{1,1} + p_{1,2} + p_{2,3} + p_{2,4} = C_{M1} + C_{M2} = 6. \quad (11)$$

From Eq. 8, we get

$$p_{1,2} = p_{2,4} = 0. \quad (12)$$

Combine Eq. 11 and Eq. 12, we have

$$p_{1,1} + p_{2,3} = 6. \quad (13)$$

Because the tokens in $p_{1,1}$ and $p_{2,3}$ can never remove from $p_{1,1}$ and $p_{2,3}$ under the restriction of Eq. 13, $p_{1,1} + p_{2,3} = 6$ is a deadlock state equation.

All the deadlock state equations in the PPN of Fig. 7 are given in Table 1.

For $R_D = \{M_1, M_3\}$

From Eq. 7, we get

$$p_{1,1} + p_{1,3} + p_{2,2} + p_{2,4} = C_{M1} + C_{M3} = 6. \quad (14)$$

From Eq. 8, we get

$$p_{1,3} = p_{2,4} = 0. \quad (15)$$

Combine Eq. 14 and Eq. 15, we have

$$p_{1,1} + p_{2,2} = 6. \quad (16)$$

Under the restriction of Eq. 16, $t_{1,2}$ and $t_{2,3}$ are enabled. Firing $t_{1,2}$ will result in a marking satisfied deadlock state equation (22). Firing $t_{2,3}$ will result in a marking satisfied deadlock state equation (21). Therefore the tokens in $p_{1,1}$ and $p_{2,2}$ can never arrive at place $p_{1,4}$ and $p_{2,4}$. Equation 16 is a PDS equation.

All the PDS equations in the PPN of Fig. 7 are given in Table 2.

It is easy to understand that a marking M will never satisfy the deadlock state equations (24), (25) and (26) if M doesn't satisfy the deadlock state equations (21), (22) and (23). Because $p_{1,2} + p_{2,3} \leq 1$ and $p_{1,3} + p_{2,2} \leq 1$, deadlock states (34) and (35) will never occur if deadlock state (32) is avoided. Therefore, we need only construct restrictive PN controllers for deadlock state

Table 1. The deadlock state equations in the PPN of Fig. 7.

R_D	Deadlock state equation	Remark
$\{M_1, M_2\}$	$p_{1,1} + p_{2,3} = 6$	(21)
$\{M_2, M_3\}$	$p_{1,2} + p_{2,2} = 2$	(22)
$\{M_3, M_4\}$	$p_{1,3} + p_{2,1} = 6$	(23)
$\{M_1, M_2, M_3\}$	$p_{1,1} + p_{1,2} + p_{2,2} + p_{2,3} = 7$	(24)
$\{M_2, M_3, M_4\}$	$p_{1,2} + p_{1,3} + p_{2,1} + p_{2,2} = 7$	(25)
$\{M_1, M_2, M_3, M_4\}$	$p_{1,1} + p_{1,2} + p_{1,3} + p_{2,1} + p_{2,2} + p_{2,3} = 12$	(26)

Table 2. The PDS equations in the PPN of Fig. 7.

R_D	PDS equation	Remark
$\{M_1, M_3\}$	$p_{1,1} + p_{2,2} = 6$	(31)
$\{M_1, M_4\}$	$p_{1,1} + p_{2,1} = 10$	(32)
$\{M_2, M_4\}$	$p_{1,2} + p_{2,1} = 6$	(33)
$\{M_1, M_2, M_4\}$	$p_{1,1} + p_{1,2} + p_{2,1} + p_{2,3} = 11$	(34)
$\{M_1, M_3, M_4\}$	$p_{1,1} + p_{1,3} + p_{2,1} + p_{2,2} = 11$	(35)

equation (21), (22), (23), (31), (32) and (33). The figures of these PN controllers are omitted here.

By the deadlock avoidance policy ρ_2 proposed in [4], $D = \{t_{1,2}, t_{1,3}, t_{1,4}, t_{2,2}, t_{2,3}, t_{2,4}\}$ is a deadlock structure. Therefore, the FMS is deadlock free if the sum of tokens in places of ${}^*D = \{p_{1,1}, p_{1,2}, p_{1,3}, p_{2,1}, p_{2,2}, p_{2,3}\}$ is not greater than ten. This control policy is not necessary because the PPN is deadlock free under marking $p_{1,1} = 5$, $p_{2,1} = 4$, $p_{1,2} = 1$ and $p_{1,3} = 1$. It is not sufficient because the PPN is in deadlock state under marking $p_{1,1} = 5$ and $p_{2,1} = 5$.

6 Conclusion

We have presented a deadlock avoidance method for FMS with shared resources. By excluding some enabled transitions from firing, this method prevents the FMS from a deadlock situation while it allows the maximal use of resources. The most important work of the deadlock avoidance method is to construct the deadlock state equations, which are the base of restrictive PN controllers.

The construction of a deadlock state equation is carried out off-line, and the restrictive PN controllers can be easily realized in practical FMS. Two applications show that this method is effective for avoiding both deadlock state and potential deadlock state.

References

1. Banaszak ZA, Krogh BH (1990) Deadlock avoidance in flexible manufacturing systems with concurrently competing process flows. IEEE Trans Robot Automat 6(6):724–734
2. Fu-Shiung, Hsieh, Chang S-C (1994) Dispatching-driven deadlock avoidance controller synthesis for flexible manufacturing systems. IEEE Trans Robot Automat 10(2):196–208
3. Ezpeleta J, Colom JM, Martinez J (1995) A Petri net based deadlock prevention policy for flexible manufacturing systems. IEEE Trans Robot Automat 11(2):173184
4. Xing K-Y, Hu B-S, Chen H-X (1996) Deadlock avoidance policy for Petri-net modeling of flexible manufacturing systems with shared resources. IEEE Trans Automat Control 41(2):289–295
5. Lewis FL, Gurel A, Bogdan S, et al. (1998) Analysis of deadlock and circular waits using a matrix model for flexible manufacturing systems. Automatica 34(9):1083–1100
6. Fanti MP, Maione B, Turchiano B (2000) Comparing digraph and Petri net approaches to deadlock avoidance in FMS. IEEE Trans Syst Man Cybern B 30(5):783–798
7. Wu N, Zhou MC (2000) Resource-oriented Petri nets for deadlock avoidance in automated manufacturing. In: Proceedings of the 2000 IEEE international conference on robotics and automation, San Francisco, CA, pp 3377–3382

8. Wu N (1999) Necessary and sufficient conditions for deadlock-free operation in flexible manufacturing systems using a colored Petri net model. IEEE Trans Systems, Man Cybern: part C 29(2):192–204
9. Xiong HH, Zhou MC (1997) Deadlock-free scheduling of an automated manufacturing system based on Petri nets. In: Proceedings of the 1997 IEEE international conference on robotics and automation, Albuquerque, NM, pp 945–950
10. Bogdan S, Lewis FL (1997) Matrix approach to deadlock avoidance of dispatching in multi-class finite buffer reentrant flow lines. In: Proceedings of the 12th IEEE international symposium on intelligent control, Istanbul, pp 397–402
11. Dohi Y, Nomura E (1996) Petri net controller with hardware to avoid deadlocks. IEEE IECON 22nd International Conference 1:457–462
12. Barkaoui K, Abdallah IB (1994) An efficient deadlock avoidance control policy in FMS using structural analysis of Petri nets. IEEE international conference on systems, man, and cybernetics 1:525–530
13. Viswanadham N, Johnson TJ, Narahari Y (1990) Performance analysis of automated manufacturing systems with blocking and deadlock. In: Proceedings of Rensselaer's 2nd international conference on computer integrated manufacturing pp 64–68
14. Zhou MC, Dicesare F, Guo D (1990) Modeling and performance analysis of a resource-sharing manufacturing system using stochastic Petri nets. 5th IEEE international symposium on intelligent control 2: 1005–1010
15. Shatz SM, Tu S, Murata T, et al. (1996) An application of Petri net reduction for ada tasking deadlock analysis. IEEE Trans Parallel Distributed Syst 7(12):1307–1322
16. Abdallah IB, Elmaraghy HA (1998) Deadlock prevention and avoidance in FMS: A Petri net based approach. Int J Adv Manuf Technol 14(10):704–715
17. Murat U (2002) An optimal deadlock prevention policy for flexible manufacturing systems using Petri net models with resources and the theory of regions. Int J Adv Manuf Technol 19(3): 192–208
18. Valckenaers P, Brussel HV (2003) Deadlock avoidance in flexible flow shops with loops. J Intell Manuf 14:137–144
19. Ferrarini L, Piroddi L, Allegri S (1999) A comparative performance analysis of deadlock avoidance control algorithms for FMS. J Intell Manuf 14:569–585
20. Fanti MP, Maione G, Turchiano B (2002) Design of supervisors to avoid deadlock in flexile assemble systems. Int J Flex Manuf Syst 14:153–171
21. Lawley M, Reveliotis S (2001) Deadlock avoidance for sequential resource allocation systems: hard and easy cases. Int J Flex Manuf Syst 13:385–404
22. Huang HH (2001) Matrix controller design and deadlock analysis of automated manufacturing systems. Part 2: deadlock avoidance policy. Int J Adv Manuf Technol 18:490-501
23. David (1992) Petri net and grafcet. Prentice-Hall, London
24. Peterson JL (1981) Petri net theory and the modeling of systems. Prentice-Hall, London
25. Zhou MC, Zurawski R (1995) Introduction to Petri nets in flexible and agile automation. In: Zhou MC (ed) Petri nets in flexible and agile automation. Kluwer, Boston, MA, pp 1–42