Optimal Siphon-based Deadlock Prevention Policy for a Class of Petri Nets in Automation

ShouGuang Wang, ChengYing Wang
College of Information & Electronic Engineering
Zhejiang Gongshang University
Hangzhou, China
wsg5000@hotmail.com, coolcake001@163.com

MengChu Zhou
Department of Electrical and Computer Engineering
New Jersey Institute of Technology
Newark, USA
zhou@njit.edu

***Abstract*—The mixed integer programming (MIP)-based deadlock detection method plays an important role in the development of deadlock prevention policies for flexible manufacturing systems (FMS). In this paper, an optimal deadlock prevention policy is proposed for a class of Petri nets called Systems of Simple Sequential Processes with Resources (S^3PR) without any ξ-resource. A ξ-resource is a one-unit resource place shared by two or more minimal siphons that do not mutually contain each other. Compared with the MIP-based deadlock prevention policies that suffer from the problem of limited behavior permissiveness and high structural complexity, the proposed one can obtain an optimal liveness-enforcing supervisor with lower structural complexity. An FMS example is used to illustrate the application of the proposed deadlock prevention policy.**

Keywords- Flexible manufacturing system; deadlock; discrete event system; Petri net; supervisory control

I. INTRODUCTION

In the past two decades, siphons played a key role in the development of deadlock prevention policies for flexible manufacturing systems (FMS) [13]. Generally, siphon-based deadlock prevention policies can be obtained in two ways: 1) compute all emptiable minimal siphons and then add a monitor to each one [5], [21], [22] or add monitors to control some minimal siphons such that all the minimal siphons can be controlled [13]; and 2) compute an emptiable minimal siphon at a time and adds a monitor to it until no such minimal siphon can be found [3], [6]-[8], [16].

The first way needs to compute all emptiable minimal siphons. It can be done by using the classical methods [5], resource circuits-based methods [11]-[13], and loop resource subsets-based methods [18]. Since the number of such siphons is in theory exponential with respect to the size of a Petri net, any deadlock prevention policy depends on complete siphon enumeration is exponential with respect to the size of a Petri net. Another shortcoming is that many redundant monitors may result [5], [21], [22]. For example, Xing *et al.*'s work [21] indicates that an optimal deadlock prevention policy can be obtained if each emptiable minimal siphon in an S^3PR without any ξ-resource is optimally controlled. However, it suffers from the problem of structural complexity.

The second way avoids complete siphon enumeration and is based on mixed integer programming (MIP). MIP-based deadlock detection method is first proposed by Chu and Xie [1] to compute a maximal emptiable siphon quickly. Since deadlock control is usually concerned with minimal siphons, researchers investigate minimal siphon extraction methods from a maximal emptiable one [7], [8], [14]. Some work on direct minimal siphon extraction using MIP is reported in [3], [4], [6], [16]. A shortcoming of an MIP-based deadlock detection method is that its solution corresponding to a maximal emptiable siphon is not always the original net's siphon. This problem is due to the fact that a marking computed by the state equations is not necessarily a reachable marking of the original Petri net.

Since ten years ago, the MIP-based deadlock detection method has been applied in iterative deadlock prevention policies that mainly consist of two strategies [7], [8], [13]: siphon control and control-induced siphon control. At each iteration, a maximal emptiable siphon is computed by the MIP-based deadlock detection method and then a minimal emptiable siphon is derived from the maximal one. Then, a monitor is added such that the minimal emptiable siphon is optimally controlled. It repeats until all emptiable minimal siphons in the original net are controlled. The second step is necessary if the resulting net after the first step contains deadlocks. At the second step, a minimal emptiable siphon that contains at least one monitor is computed by solving an MIP problem and a monitor is added such that the siphon is controlled by controlling the source transitions only. This step repeats until no emptiable siphon can be found. In [2], the number of MIP iterations is reduced by introducing the concept of basic and compound siphons. The latter is composed of the former.

An MIP-based deadlock prevention method is more efficient than those that depend on complete state or siphon enumeration [5], [20]-[22], but they can significantly restrict the behavior permissiveness. Moreover, an MIP-based siphon computation method has a shortcoming that its solution

This work is in part supported by National Natural Science Foundation of China under Grant 61034004, the Zhejiang Provincial Natural Science Foundation of China under Grant Y1090232 and Y1100830, the Science Fund for Young Scholars of Zhejiang Gongshang University, and the Zhejiang Provincial Education Department Foundation under Grant Y201018216.

corresponding to a maximal emptiable siphon is not always the original net's siphon, thereby causing a redundant monitor. Finally, it may lead to weighted monitors [7], [8], [19], which is not a desirable phenomenon as indicated in [2].

For an S^3PR net that contains no ξ-resource, Xing *et al.* indicate that an optimal liveness-enforcing supervisor can be obtained if all emptiable minimal siphons in the original net are optimally controlled [21]. However, their method suffers from the problem of structural complexity since the number of monitors grows exponentially with the net size. If the MIP-based deadlock prevention policy is used, it is hard to ensure that all emptiable minimal siphons in the original net are controlled, and thus make the supervisor more complex. This work proposes an iterative way to obtain an optimal liveness-enforcing supervisor with lower structural complexity for such nets. Its new contributions are:

1) An emptiable siphon computation algorithm is proposed, which overcomes a shortcoming of an MIP-based deadlock detection method that its solution corresponding to a maximal emptiable siphon is not always the original net's siphon;

2) An algorithm is proposed to derive an emptiable siphon that contains no monitor from a controlled Petri net, which avoids weighted monitors [7], [8], [19]; and

3) For an S^3PR net that contains no ξ-resource, an optimal deadlock prevention policy is proposed, which significantly improves the prior work [21], [22] and the MIP-based deadlock prevention methods [3], [7], [8], [10], [13] in structural complexity.

The rest of this paper is organized as follows. Section II briefly reviews preliminaries used in this paper. An optimal deadlock prevention policy is developed in Section III. Section IV presents an FMS example to demonstrate the policy. Finally, Section V concludes this paper.

II. Preliminaries

A. Petri Nets [13], [15]

A Petri net is a 3-tuple $N=(P, T, F)$, where P and T are finite, nonempty, and disjoint sets. P is a set of places, and T is a set of transitions. The set $F \subseteq (P \times T) \cup (T \times P)$ is the flow relation. Given a net $N=(P, T, F)$, and a node $x\in(P\cup T)$, ${}^\bullet x=\{y\in P\cup T \mid (y, x)\in F\}$ is the preset of x, while $x^\bullet=\{y\in P\cup T \mid (x, y)\in F\}$ is the post-set of x. $\forall X\subseteq(P\cup T)$, ${}^\bullet X=\bigcup_{x\in X}{}^\bullet x$, and $X^\bullet=\bigcup_{x\in X}x^\bullet$. $\forall x\in(P\cup T)$, ${}^{\bullet\bullet}x={}^\bullet({}^\bullet x)$, and $x^{\bullet\bullet}=(x^\bullet)^\bullet$.

The incidence matrix of N is a matrix $[N]$: $P\times T\rightarrow Z$ indexed by P and T such that $[N](p, t) = -1$ if $p\in {}^\bullet t \setminus t^\bullet$; $[N](p, t) = 1$ if $p\in t^\bullet \setminus {}^\bullet t$; otherwise $[N](p, t) = 0$ for all $p\in P$ and $t\in T$.

A P-vector is a column vector I: $P\rightarrow Z$ indexed by P and a T-vector is a column vector J: $T\rightarrow Z$ indexed by T, where Z is the set of integers. I is a P-invariant if $I\neq\mathbf{0}$ and $I^T[N]=\mathbf{0}^T$ hold. P-invariant I is a semiflow if every element of I is non-negative. $\|I\|=\{p\in P \mid I(p)\neq 0\}$ is called the support of I.

A nonempty set $S\subseteq P$ is a siphon if ${}^\bullet S\subseteq S^\bullet$. A siphon is minimal if there is no siphon contained in it as a proper subset. A minimal siphon that does not contain the support of any P-invariant is called an minimal siphon. A siphon S is said to be controlled in a net system (N, M_0) if $\forall M\in R(N, M_0)$, $M(S)>0$. S is said to be optimally controlled in a net system (N, M_0) if only the markings at which S becomes unmarked are removed.

A marking M of N is a mapping from P to $\mathbb{N}$ where $\mathbb{N}=\{0, 1, 2, ...\}$. In general, we use multi-set notation $\sum_{p\in P}M(p)p$ to denote vector M, where $M(p)$ indicates the number of tokens in p at M. For example, $M=[1, 2, 0, 0]^T$ is denoted by $M=p_1+2p_2$. p is marked by M if $M(p)>0$.

A transition t is enabled at M, denoted by $M[t>$, if $\forall p\in{}^\bullet t$, $M(p)>0$. An enabled transition t at M can fire, resulting in a new marking M', denoted by $M[t>M'$, where $M'(p)=M(p)+[N](p, t)$. A sequence of transitions $\alpha=t_1 t_2 \ldots t_k$, $t_i\in T$, $i=1, 2, \ldots, k$ is feasible from a marking M if there exist $M_i [t_i>M_{i+1}$ and $i=1, 2, \ldots, k$, where $M_1=M$. In such a case, we use $M[\alpha>M_{k+1}$ to denote that M_{k+1} is reachable from M after firing a sequence of transitions α. Let $R(N, M_0)$ denote the set of all reachable markings of N from the initial marking M_0.

A string $x_1, x_2,\ldots$, and x_n is called a path of N if $\forall i\in\{1, 2,..., n-1\}$, $x_{i+1}\in x_i^\bullet$, where $x_i\in P\cup T$. A node may appear more than once in a circuit. Two circuits are same if the sets of their nodes and arcs are the same, respectively. A circuit c can determine a unique subnet whose nodes and arcs are in c. We call this subnet a circuit too, for simplicity. Hence a circuit is a strongly connected subnet. An elementary path from x_1 to x_n is a path whose nodes are all different (except, perhaps, x_1 and x_n). A path $x_1, x_2,\ldots$, and x_n is called an elementary circuit if it is an elementary path and $x_1=x_n$.

A place without any input transition is called a source place, and one without any output transition is called a sink one.

B. S^3PR [5]

Definition 1: A System of Simple Sequential Process with Resources (S^3PR) $N=\bigcirc_{i=1}^{q}N_i=(P_A\cup P_0\cup P_R, T, F)$ is defined as the union of a set of nets $N_i=(P_A^i\cup\{p_0^i\}\cup P_R^i, T_i, F_i)$ sharing common places, where the following statements are ture:

(1) p_0^i is called the process idle place of N_i. Elements in P_A^i and P_R^i are called activity and resource places;

(2) $P_A^i\neq\emptyset$; $P_R^i\neq\emptyset$; $p_0^i\notin P_A^i$; $(P_A^i\cup\{p_0^i\})\cap P_R^i=\emptyset$;

(3) $\forall p\in P_A^i$, $\forall t\in{}^\bullet p$, $\forall t'\in p^\bullet$, $\exists r_p\in P_R^i$, ${}^\bullet t\cap P_R^i=t'^\bullet\cap P_R^i=\{r_p\}$;

(4) $\forall r\in P_R^i$, ${}^{\bullet\bullet}r\cap P_A^i=r^{\bullet\bullet}\cap P_A^i\neq\emptyset$, ${}^\bullet r\cap r^\bullet=\emptyset$;

(5) ${}^{\bullet\bullet}(p_0^i)\cap P_R^i=(p_0^i)^{\bullet\bullet}\cap P_R^i=\emptyset$;

(6) N_i' is a strongly connected state machine, where $N_i'=(P_A^i\cup\{p_0^i\}, T_i, F_i)$ is the resulting net after the places in P_R^i and related arcs are removed from N_i;

(7) Every circuit of N_i' contains place p_0^i;

(8) Any two are composable when they share a set of common places. Every shared place must be a resource one.

(9) For $r \in P_R, H(r) = ({}^{\bullet\bullet}r) \cap P_A$ is the set of activity places that use r and are called holders of r.

(10) For $p \in P_A, ({}^{\bullet\bullet}p) \cap P_R = \{r_p\}$ where resource place r_p is called the resource used by p.

Consider an S[3]PR in Fig. 1. $P_A = P_A^1 \cup P_A^2 = \{p_2, p_3, p_5, p_6, p_7\} \cup \{p_4, p_8, p_9\} = \{p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9\}$, $P_0 = \{p_1, p_{10}\}$, $P_R = \{p_{11}, p_{12}, p_{13}, p_{14}, p_{15}\}$, $H(p_{11}) = \{p_7\}$, $H(p_{12}) = \{p_3, p_9\}$, $H(p_{13}) = \{p_4, p_5\}$, and $H(p_{14}) = \{p_6, p_8\}$.

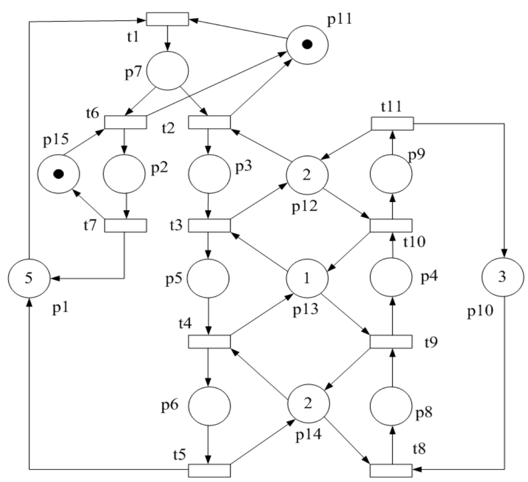

Figure 1. An S[3]PR (N, M_0)

Definition 2: Let $N = (P_A \cup P_0 \cup P_R, T, F)$ be an *S[3]PR*. An initial marking M_0 is called an acceptable one if 1) $\forall p \in P_0, M_0(p) \geq 1$; 2) $\forall p \in P_A, M_0(p) = 0$; and 3) $\forall p \in P_R, M_0(p) \geq 1$.

As shown in Fig. 1, M_0 is an acceptable initial marking if $M_0(p_1) = 5$, $M_0(p_{10}) = 3$, $M_0(p_{12}) = M_0(p_{14}) = 2$, $M_0(p_{11}) = M_0(p_{15}) = M_0(p_{13}) = 1$, and others are zero.

III. Optimal Deadlock Prevention Policy

A. Computation of an emptiable siphon

As previously stated, a solution of an MIP problem [1] is not always an emptiable siphon of the original Petri net. Therefore, we propose a new method to obtain an emptiable siphon from an acceptably initial marked S³PR (N, M_0). Let $M \in R(N, M_0)$. First, it removes all places marked by M. Second, it removes all transitions that do not have any input place and their output places. The second step is repeated until no more nodes can be removed. If there is no remaining place, we should choose another marking $M' \in R(N, M_0)$ and repeat the above steps. Otherwise, the remaining places form a siphon that is emptied at M.

Algorithm 1: Computation of an emptiable siphon.

Input: an acceptably initial marked S³PR net model (N, M_0), where $N = \mathrm{O}_{i=1}^{q} N_i = (P_A \cup P_0 \cup P_R, T, F)$.

Output: an emptiable siphon S_e.

1) Let $S_e := \emptyset$, and $N' := N$, where $N' = (P', T', F')$;

2) **for** $M \in R(N, M_0)$ **do**

3) **for** $p \in P'$ **do**

4) **if** $M(p) \geq 1$ **then**

5) $P' := P' - \{p\}$;

6) **end if**

7) **end for**

8) $F' := F \cap [(P' \times T') \cup (T' \times P')]$

9) $N' := (P', T', F')$

10) **while** there exist a source transition t in N' **do**

11) $P' := P' - t^{\bullet}$;

12) $T' := T' - \{t\}$;

13) $F' := F \cap [(P' \times T') \cup (T' \times P')]$

14) $N' := (P', T', F')$

15) **end while**

16) **if** $(P' \neq \emptyset)$ **then**

17) $S_e := P'$;

18) go to Step 21;

19) **end if**;

20) **end for**

21) Output: S_e;

22) End.

Theorem 1: Given an acceptably initial marked S³PR net (N, M_0), an emptiable siphon S_e in (N, M_0) can be obtained by Algorithm 1.

B. Deadlock prevention policy

By the previous subsection, an emptiable siphon can be obtained and a minimal one can be derived from it via the method in [7] and [14]. For a class of S[3]PR without any ξ-resource, Xing *et al*. [21], [22] prove that an optimal deadlock prevention policy can be obtained if all emptiable minimal siphons are optimally controlled. However, deadlock prevention policies based on elementary siphons [11], basic siphons [2], and essential siphons [16] indicate that controlling all emptiable minimal siphons may lead to redundant monitors. Therefore, we focus on utilizing an iterative way to obtain an optimal deadlock prevention policy with lower structural complexity.

In the remaining discussion, we assume that (N, M_0) is an S^3PR net with an acceptable initial marking.

Ezpeleta *et al*. [5] prove that if S is a minimal siphon in an S^3PR, $|S\cap P_R| > 1$. Hence, S can be represented by $S_A \cup S_R$, where $S_R = S\cap P_R$ and $S_A = S\cap P_A$.

Definition 3: Let S_i and S_j be two minimal siphon in (N, M_0) that do not mutually contain each other and $S_{iR} \cap S_{jR} \neq \emptyset$. Then every resource place r such that $r\in S_{iR} \cap S_{jR}$ with capacity 1 (i.e., unit capacity) is called a ξ-resource. S_i and S_j are said to share r. Actually, a ξ-resource is a one-unit resource one shared by two or more minimal siphons that do not mutually contain each other.

Note that the concept of ξ-resources is first proposed in [21] based on maximal perfect resource-transition (MPRT) circuits and it is proved in [21] that there exists a one-to-one correspondence between minimal siphons and MPRT-circuits. Therefore, we can use minimal siphon to define a ξ-resource.

For example, $S_1= \{p_5, p_9, p_{12}, p_{13}\}$ and $S_2= \{p_4, p_6, p_{13}, p_{14}\}$ are minimal siphons in Fig. 1 and they do not mutually contain each other. Since $S_{1R} \cap S_{2R}= \{p_{13}\}$ and $M_0(p_{13})=1$, p_{13} is ξ-resource.

Proposition 1: Let S be a minimal siphon in (N, M_0) and (N_1, M_1) be the net derived from (N, M_0) by adding a monitor p_c. S is optimally controlled if p_c is added such that 1) $\forall p\in P_A \cup P_0 \cup P_R$, $M_1(p)= M_0(p)$; 2) $M_1(p_c)= M_0(S)-1$; and 3) $I= p_x+\ldots+p_y+ p_c$ is a P-invariant of (N_1, M_1) where $\{p_x,\ldots, p_y\}=(\bigcup_{r\in S\cap P_R} H(r))\setminus S$.

Proof: Similar to the proof of Proposition 1 in [10].

Theorem 2: If an acceptably initial marked S³PR net (N, M_0) does not contain any ξ-resource, an optimal liveness-enforcing supervisor for (N, M_0) can be obtained by optimally controlling each minimal siphon in (N, M_0).

Proof: Straightforward from Theorem 7 in [21] and Theorems 3 and 4 in [22].

Lemma 1[5]: A marked S³PR net (N, M_0) is live if no minimal siphon in (N, M_0) can be emptied.

Based on Theorems 1 and 2, Proposition 1, and Lemma 1, we can propose a new deadlock prevention policy that consists of three strategies. Firstly, compute an emptiable siphon from a marked S³PR net (N, M_0) without ξ-resources and then obtain a minimal siphon from the emptiable one. Secondly, add a monitor p_c to (N, M_0) such that the emptiable minimal siphon is optimally controlled, and the controlled net is denoted as (N_c, M_{c0}). Finally, compute an emptiable siphon containing no p_c from (N_c, M_{c0}) and then obtain a minimal siphon from the emptiable one, and repeat these steps until an optimal liveness-enforcing supervisor is obtained. Note that the final step is an effective way to solve the problem of weighted monitors [7], [8], [19] in the MIP-based methods.

Algorithm 2: Deadlock prevention policy.

Input: an acceptably initial marked S³PR net model (N, M_0).

Output: (N_c, M_{c0}).* (N_c, M_{c0}) is a controlled Petri net *\

1) Apply Algorithm 1 to find an emptiable siphon S_e in (N, M_0);

2) **If** $S_e=\emptyset$ **then**

3) $(N, M_0):= (N_c, M_{c0})$;

4) go to Step 19;

5) **end if**

6) $P_C:= \emptyset$, $S^*:= S_e$;

7) Obtain an emptiable minimal siphon from S^* by the method in [7] and [14], denoted by S;

8) **while** $(S\neq\emptyset)$ do

9) Add a monitor p_c to (N, M_0) such that S is optimally controlled by Proposition 1;

10) $P_C:= P_C \cup\{p_c\}$;

11) $N_c= (P\cup P_C, T, F)$;

12) $M_{c0}:= M_0\ (P\cup P_C)$

13) Compute an emptiable siphon S^* in (N_c, M_{c0}) by Algorithm 3;

14) **if** $(S^*=\emptyset)$ **then**

15) go to Step 19;

16) **end if**

17) Obtain an emptiable minimal siphon from S^* by the method in [7] and [14], denoted by S;

18) **end while**

19) Output: (N_c, M_{c0});

20) End.

Algorithm 3: Computation of an emptiable siphon without containing any monitor

Input: $N_c= (P\cup P_C, T, F)$ with initial marking M_{c0}.

Output: an emptiable siphon S^* that contains no place in P_C.

1) Let $S^*:=\emptyset$, and $N':= N$, where $N'= (P', T', F')$;

2) **for** $M\in R\ (N_c, M_{c0})$ **do**

3) $P':=P'- P_C$;

4) **for** $p\in P'$ **do**

5) **if** $M(p)\geq 1$ **then**

6) $P':=P'-\{p\}$;

7) **end if**

8) **end for**

9) $F':= F \cap [(P' \times T') \cup (T'\times P')]$

10) $N':= (P', T', F')$

11) **while** there exist a source transition t in N' **do**

12) $P' := P' - t^{\bullet}$;

13) $T' := T' - \{t\}$;

14) $F' := F \cap [(P' \times T') \cup (T' \times P')]$

15) $N' := (P', T', F')$

16) **end while**

17) **if** ($P' \neq \emptyset$) **then**

18) $S^* := P'$;

19) go to Step 22;

20) **end if**;

21) **end for**

22) Output: S^*;

23) End.

Given a marked S^3PR net (N, M_0) without ξ-resources, an optimal liveness-enforcing supervisor can be obtained by Algorithm 2. We briefly explain Algorithm 2 as follows. First, Algorithm 1 is applied to check whether there exists an emptiable siphon in (N, M_0). If no emptiable siphon is computed, it is live by Lemma 1, as shown in Steps 1 to 5. Otherwise, the method in [7] and [14] is used to extract a minimal siphon from the obtained emptiable one. Next, a monitor is added to (N, M_0) such that the emptiable minimal siphon is optimally controlled, and the controlled net is denoted as (N_c, M_{c0}). Then, Algorithm 3 is used to compute an emptiable siphon without containing any monitor in (N_c, M_{c0}). If no emptiable siphon is found, an optimal liveness-enforcing supervisor is obtained due to Theorem 2, as shown in Steps 14 to 16. Otherwise, the method in [7] and [14] is used to extract a minimal siphon from the obtained emptiable one. If no emptiable minimal siphon is computed, an optimal liveness-enforcing supervisor is obtained due to Theorem 2 and the algorithm ends. Otherwise, add a monitor to optimally control the emptiable minimal siphon and repeat these steps until no emptiable siphon can be found.

Remark 1: Algorithm 3 is utilized to obtain an emptiable siphon without containing any monitor from a controlled Petri net (N_c, M_{c0}) and thus avoids control-induced siphons. The first step of Algorithm 3 is to remove all monitors and the remaining steps are the same as Algorithm 1. The main difference between Algorithms 1 and 3 is that Algorithm 1 is used in an acceptably initial marked S^3PR net model (N, M_0) while Algorithm 3 is used in a controlled Petri net (N_c, M_{c0}).

Theorem 3: Given an acceptably initial marked S^3PR net (N, M_0) without ξ-resources, an optimal liveness-enforcing supervisor can be obtained by Algorithm 2.

IV. FMS Example

(N, M_0) shown in Fig. 2 is an S^3PR where $P_0 = \{p_1, p_4, p_7, p_{10}, p_{13}, p_{16}, p_{19}, p_{22}\}$, $P_R = \{p_{25}\text{-}p_{29}\}$, $P_A^1 = \{p_2, p_3\}$, $P_A^2 = \{p_5, p_6\}$, $P_A^3 = \{p_8, p_9\}$, $P_A^4 = \{p_{11}, p_{12}\}$, $P_A^5 = \{p_{11}, p_{12}\}$, $P_A^6 = \{p_{11}, p_{12}\}$, $P_A^7 = \{p_{11}, p_{12}\}$, and $P_A^8 = \{p_{11}, p_{12}\}$. The reachability graph (RG) of the uncontrolled S^3PR model given in Fig. 2 has 3483 states including deadlocks. There are 162 bad states in the RG. This means that the maximally controlled system should be live and reach 3321 good states. Trivially, (N, M_0) does not contain any ξ-resource and thus an optimal liveness-enforcing supervisor for (N, M_0) can be obtained by optimally controlling each minimal siphon in (N, M_0) due to Theorem 2.

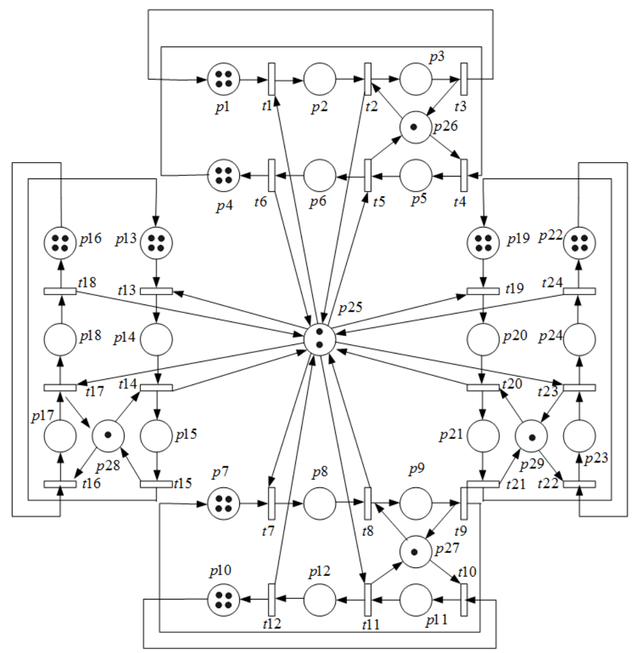

Figure 2. An S^3PR (N, M_0) without any ξ-resources

If the methods in [21] and [22] are applied, 15 monitors need to be added to obtain an optimal live net since 15 minimal siphons are emptiable in Fig. 2. In fact, five of them are redundant and applying Algorithm 2 can avoid these redundant monitors as explained as follows.

Firstly, an emptiable siphon $S^1 = \{p_3, p_6, p_8, p_9, p_{12}, p_{14}, p_{15}, p_{18}, p_{20}, p_{21}, p_{24}, p_{25}, p_{26}\ p_{27}, p_{28}, p_{29}\}$ in (N, M_0) is obtained by Algorithm 1 and an emptiable minimal siphon $S_1 = \{p_3, p_6, p_8, p_{12}, p_{14}, p_{18}, p_{20}, p_{24}, p_{25}, p_{26}\}$ in (N, M_0) is obtained from S^1 by the method in [7] and [14].

Secondly, add a monitor p_{c1} to (N, M_0) according to Proposition 1 such that S_1 is optimally controlled, where $M_{c0}(p_{c1}) = 3$, ${}^{\bullet\bullet}p_{c1} = \{t_2, t_5\}$, and $p_{c1}^{\bullet\bullet} = \{t_1, t_4\}$. The resulting net is denoted as (N_c, M_{c0}).

Next, an emptiable siphon S^2 without p_{c1} is obtained from (N_c, M_{c0}) by Algorithm 3 and an emptiable minimal siphon $S_2 = \{p_2, p_6, p_9, p_{12}, p_{14}, p_{18}, p_{20}, p_{24}, p_{25}, p_{27}\}$ is obtained from S^2. Repeat these steps until no emptiable siphon can be found. Monitors needed are shown in Table I. By INA [17], the controlled net (with respect to the net in Fig. 2 and monitors in Table I) is live and optimal.

TABLE I. MONITORS ADDED TO THE NET IN FIG.2

Monitor	Pre	Post	M_{c0}
p_{c1}	t_2, t_5	t_1, t_4	3
p_{c2}	t_8, t_{11}	t_7, t_{10}	3
p_{c3}	t_{14}, t_{17}	t_{13}, t_{16}	3
p_{c4}	t_{20}, t_{23}	t_{19}, t_{22}	3
p_{c5}	t_{14}, t_{17}, t_{20}, t_{23}	t_{13}, t_{16}, t_{19}, t_{22}	5
p_{c6}	t_8, t_{11}, t_{20}, t_{23}	t_7, t_{10}, t_{19}, t_{22}	5
p_{c7}	t_2, t_5, t_{20}, t_{23}	t_1, t_4, t_{19}, t_{22}	5
p_{c8}	t_8, t_{11}, t_{14}, t_{17}	t_7, t_{10}, t_{13}, t_{16}	5
p_{c9}	t_2, t_5, t_{14}, t_{17}	t_1, t_4, t_{13}, t_{16}	5
p_{c10}	t_2, t_5, t_8, t_{11}	t_1, t_4, t_7, t_{10}	5

V. CONCLUSION

So far, the MIP-based deadlock prevention policies are one of the most effective ways to deal with deadlocks but their behavioral permissiveness and structural complexity need improvements. This paper overcomes a shortcoming of an MIP-based deadlock detection method, i.e., its solution corresponding to a maximal emptiable siphon is not always the original net's siphon by proposing a new method to obtain an optimal liveness-enforcing supervisor with lower structural complexity for an S^3PR net that contains no ξ-resource. The main drawbacks of the proposed method are:

1) It is computationally expensive to compute the reachable markings; and

2) It can only be applied to some classes of Petri nets, i.e., those that are optimal and live if their original minimal siphons are optimally controlled.

Therefore, our future work includes lowering the computational complexity and extending the application of this method to others nets [13][23-28].

ACKNOWLEDGMENT

The authors would like to thank the anonymous referees whose comments helped us improve the quality of this paper.

REFERENCES

[1] F. Chu, X.L.Xie, "Deadlock analysis of Petri nets using siphons and mathematical programming," *IEEE Trans. Robot. Autom.*, vol. 13, no. 6, pp. 793-804, 1997.

[2] D. Y. Chao, "Technical note - reducing mip iterations for deadlock prevention of flexible manufacturing systems," *International Journal of Advanced Manufacturing Technology*, vol.41, no.3-4, pp. 343-346, 2009.

[3] D. Y. Chao, "Direct minimal empty siphon computation using MIP," International Journal of Advanced Manufacturing Technology, vol.45, no.3-4, pp.397-405, 2009.

[4] R. Cordone, L. Ferrarini, and L. Piroddi, "Enumeration algorithms for minimal siphons in Petri nets based on place constraints," *IEEE Trans. Syst.,Man, Cybern., A, Syst., Humans*, vol. 35, no. 6, pp. 844-854, Nov. 2005.

[5] J. Ezpeleta, J. M. Colom, and J.Martinez, "A Petri net based deadlock prevention policy for flexible manufacturing systems," *IEEE Trans. Robot. Autom.*, vol. 11, no. 2, pp. 173–184, Apr. 1995.

[6] A. Giua and C. Seatzu, "Modeling and supervisory control of railway networks using Petri nets," *IEEE Trans. Autom. Sci. Eng.*, vol. 5, no. 3, pp. 431-476, Jul. 2008.

[7] Y. S. Huang, M. D. Jeng, X. L. Xie, and S. L. Chung, "Deadlock prevention policy based on Petri nets and siphons," *Int. J. Prod. Res.*, vol. 39, pp. 283–305, 2001.

[8] Y. S. Huang, M. D. Jeng, X. L. Xie, and D. H. Chung, "Siphon-based deadlock prevention policy for flexible manufacturing systems," *IEEE Trans. Syst., Man, Cybern., A, Syst., Humans*, vol. 36, no. 6, pp. 1248–1256, Nov. 2006.

[9] Z. W. Li, H. S. Hu, and A. R. Wang, "Design of liveness-enforcing supervisors for flexible manufacturing systems using Petri nets," *IEEE Trans. Syst., Man, Cybern., C, Appl. Rev.*, vol.37, no.4, pp.517–526, 2007.

[10] Z. W. Li, M. C. Zhou, "Two-stage method for synthesizing liveness-enforcing supervisors for flexible manufacturing systems using Petri nets," *IEEE Transactions on Industrial Informatics*, 2(4), 313-325, 2006.

[11] Z. W. Li, M. C. Zhou, and M. D. Jeng, "A maximally permissive deadlock prevention policy for FMS based on Petri net siphon control and the theory of regions," *IEEE Trans. Autom. Sci. Eng.*, vol. 5, no. 1, pp. 182-188, Jan. 2008.

[12] Z. W. Li and A. R. Wang, "A Petri net based deadlock prevention approach for flexible manufacturing systems," Acta Automatica Sinica, vol. 29, no. 5, pp. 733-740, 2003.

[13] Z. W. Li and M. C. Zhou, "Deadlock Resolution in Automated anufacturing Systems: A Novel Petri Net Approach," *London: Springer-Verlag*, Feb. 2009.

[14] Z. W. Li and D. Liu, "A correct minimal siphons extraction algorithm from a maximal unmarked siphon of a Petri net," *Int. J. Prod. Res.*, vol. 45, no. 9, pp. 2161–2165, 2007.

[15] T. Murata, "Petri nets: Properties, analysis, and applications," *Proc. IEEE*, vol. 77, no. 4, pp. 541–580, Apr. 1989.

[16] L. Piroddi, R. Cordone, and I. Fumagalli, "Combined siphon and marking generation for deadlock prevention in Petri nets," *IEEE Trans. Syst.,Man, Cybern., A, Syst., Humans*, vol. 39, no. 3, pp. 650-661, May 2009.

[17] Starke P H. 2003. INA: Integrated Net Analyzer. http://www2.informatik.hu-berlin.de/~starke/ina.html.

[18] S. G. Wang, C. Y. Wang, and Y. P. Yu, "A method of computing strict minimal siphons in S^3PR based on resource circuits", *Proc. of the 49th IEEE Conference on Decision and Control*, pp.2785-2790, Dec. 2010.

[19] S. G, Wang, C. Y. Wang, and Y. P. Yu, "Comments on 'Siphon-based deadlock prevention policy for flexible manufacturing systems,'' *IEEE Trans. Syst., Man, Cybern., A, Syst., Humans*, to be published.

[20] M. Uzam, "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, 192–208, 2002.

[21] K. Y. Xing, M. C. Zhou, H. X. Liu, and F. Tian, "Optimal Petri-net-based polynomial-complexity deadlock-avoidance policies for Automated Manufacturing Systems," *IEEE Trans. Syst., Man, Cybern., A, Syst., Humans*, vol. 39, no. 1, pp.188-199, Jan. 2009.

[22] K. Y. Xing, M. C. Zhou, F. Wang, H. X. Liu, and F. Tian, "Resource-transition circuits and siphons for deadlock control of automated manufacturing systems," *IEEE Trans. Syst., Man, Cybern., A, Syst., Humans*, vol. 41, no. 1, pp.74-84, Jan. 2011.

[23] Xiong, P. C., Y. Fan and M. C. Zhou, "A Petri Net Approach to Analysis and Composition of Web Services," *IEEE Trans. on Systems, Man and Cybernetics, Part A,* 40(2), pp. 376 – 387, March 2010.

[24] Zhou, M. C. (Ed.), *Petri Nets in Flexible and Agile Automation*. Kluwer Academic Publishers, Boston, MA, 1995

[25] M. C. Zhou and M. P. Fanti (Ed.), *Deadlock Resolution in Computer-Integrated Systems*, Marcel Dekker, New York, January 2005.

[26] M. C. Zhou and K. Venkatesh, *Modeling, Simulation and Control of Flexible Manufacturing Systems: A Petri Net Approach*. World Scientific, Singapore, 1998.

[27] M. C. Zhou and F. DiCesare, *Petri Net Synthesis for Discrete Event Control of Manufacturing Systems*. Kluwer, Boston, MA, 1993.

[28] Zhou, M. C. and M. D. Jeng, "Modeling, Analysis, Simulation, Scheduling, and Control of Semiconductor Manufacturing Systems: A Petri Net Approach," *IEEE Transactions on Semiconductor Manufacturing*, 11(3), 333-357, 1998.