A Method Based on Depth-first Search to Compute All Minimal Siphons

Fan Ning
Hangzhou Vocational & Technical College, Hangzhou 310018, China
ningfanse@126.com

Xingxing Li, Shouguang Wang, Senior Member IEEE, Qiaoli Zhuang
School of Information & Electronic Engineering, Zhejiang Gongshang University, Hangzhou, 310018, China
lxx@mail.zjgsu.edu.cn, wsg5000@hotmail.com
School of Information Science and Technology, Zhejiang Sci-Tech University, Hangzhou 310018, China
chocolatezql@gmail.com

***Abstract*—Minimal siphons play an important role in the development of deadlock control policies for discrete event system modeled by Petri net. A new algorithm based on depth-first search of problem decomposition process is proposed to compute all minimal siphons in an ordinary Petri net. The algorithm can reduce the number of problems in the problem list. The proposed algorithm can solve the problem of high requirement for computer memory in computing all minimal siphons and decrease the memory consumption because the computer memory size is closely related to the number of problems in the problem list. Some examples are used to illustrate the superiority of the proposed algorithm.**

***Index Terms*—Petri nets, Minimal siphons, Deadlock**

I. Introduction

A Petri net is a graphical and mathematical tool that is widely used to describe and analyze the behavior of discrete event systems, including flexible manufacturing systems (FMS) [1]-[7], workflow management system, and automated guided vehicles. In recent years, more and more researchers adopt Petri net models to handle deadlock control problems, which are closely related to siphons. Siphons are a well-known structural object in a Petri net, which are closely related to some basic behavioral properties of the net, such as deadlock-free and liveness. Briefly, a siphon is a set of places such that their input transition set is included in their output transition set. It remains permanently unmarked once it loses all tokens. As a result, if a siphon is empty, their output transitions become permanently disabled, causing a partial or total system deadlock. Therefore, siphon is crucial in deadlock prevention policy of Petri net.

Manuscript received October 10, 2013; revised January 16, 2014; accepted February 8, 2014.
Supported by National Natural Science Foundation of China (61100056), Zhejiang Natural Science Foundation for Distinguished Young Scholar (LR14F020001), Zhejiang Sci.Tech. Project (2013C31111), and Zhejiang NNST Key Laboratory (2013E10012) .
Corresponding author: Shouguang Wang,

Deadlock control based on siphon [8] is first to compute minimal siphons whose efficient computation is fundamentally important and is much studied in the literature. As a result, complete or partial minimal siphon computation becomes necessity necessary. In the past two decades, researchers have proposed many methods to compute minimal siphons. In [9], Wang proposes a minimal siphons-extraction algorithm based on loop resource subsets. In his approach, Wang utilizes loop resource subsets to compute all the minimal siphons in S^3PR. Compared with the method in [10], [11], the algorithm proposed by Wang has higher computational efficiency via many generated examples [12]. Moreover, it can solve some problems in [10].

The methods mentioned above to compute minimal siphons have higher computational efficiency, but they can only be limited to a subclass of Petri nets called Systems of Simple Sequential Processes with Resources (S^3PR). Note that the INA-based method and the sign matrix one can be used to compute all minimal siphons for an ordinary Petri net. But the two methods have lower computation efficiency. An effective method is presented in [13], [14] to compute minimal siphons for ordinary Petri net and it has significant representativeness.

In [13], [14], Roberto Cordone and Luca Ferrarini have presented an interesting approach that is based on breadth-first search of problem decomposition. The approach in [13] includes two algorithms which mainly differ in the application of the partitioning procedure. The first one decomposes only the current problem, called local partitioning. The other one decomposes all problems in the unsolved problem list, called global partitioning. The local partitioning algorithm possibly generates spurious solutions, i.e., non-minimal siphons, whereas the global one finds exactly the complete set of minimal siphons. The algorithm in [13] has higher computation efficiency in computing minimal siphons for an ordinary Petri net. However, with the

doi:10.4304/jsw.9.8.2148-2153

expansion of the size of the Petri net, too many problems need to be decomposed, leading to high requirement for computer memory. So, siphon computation is expensive.

In this paper, a new algorithm based on depth-first search of problem decomposition process is proposed to compute all minimal siphons in an ordinary Petri net. We know that the computer memory size occupied by the developed program based on the proposed algorithm depends on the number of problems within the problem list. Similarly, the maximum requirement of memory occupied by the developed program depends on the maximum number of problems within the list. Therefore, the proposed algorithm can reduce the number of problems in the problem list and simplify the minimal siphon computation process, solving the problem of high requirement for computer memory in the course of computing all minimal siphons.

The rest of this paper is organized as follows. Section 2 presents the preliminaries used throughout this paper. Algorithm for finding minimal siphons is introduced in Section 3. Illustrative example and comparison with Cordone's algorithm is given in Section 4. Section 5 concludes this paper.

II. PRELLIMINARIES

A Petri net (PN) is a 3-tuple $N = (P, T, F)$, where P and T are finite, nonempty, and disjoint sets. P is the set of places, and T is the set of transitions. In a generic way, elements belonging to $P \cup T$ are called nodes. $F \subseteq (P \times T) \cup (T \times P)$ is called the flow relation or the set of directed arcs. The flux relation can be given in the form of matrices, namely the input (*PRE*), output (*PST*), and incidence ($C = PST - PRE$) matrices. Given a net $N = (P, T, F)$ and a node $x \in P \cup T$, $\bullet x = \{y \in P \cup T | (y, x) \in F\}$ is called the preset of node x, whereas $x\bullet = \{y \in P \cup T | (x, y) \in F\}$ is called the postset of node.

A nonempty set $S \subseteq P$ is a siphon iff $\bullet S \subseteq S\bullet$. S is called a trap iff $S\bullet \subseteq \bullet S$. A trap is marked if some of its places have token(s). A siphon is minimal iff there is no siphon contained in it as a proper subset.

Definition 1 [13]-*Reduction Function:* Let $G = (P, T, F)$ be a PN and $\widetilde{P} \subset P$. Then $\widetilde{G} = \text{red}(G, \widetilde{P})$ is a PN $(\widetilde{P}, \widetilde{T}, \widetilde{F})$ iff $\widetilde{T} = \{t \in T | (\bullet t \cup t\bullet) \cap \widetilde{P} \neq \phi\}$ and $\widetilde{F}(p, t) = F(p, t)$, $\widetilde{F}(t, p) = F(t, p)$ $\forall p \in \widetilde{P}, \forall t \in \widetilde{T}$.

Definition 2 [13]: Let $G = (P, T, F)$ be a PN, $P_{in} \subseteq P$, $P_{out} \subseteq P$. $\Pi = (G, P_{in}, P_{out})$ indicates the problem of finding and Σ_Π indicates the set of all siphons of G subject to the constraints: i) $\forall S \in \Sigma_\Pi$, S is a Pin-minimal siphon; ii) $\forall S \in \Sigma_\Pi$, $S \cap P_{out} = \phi$.

III. ALGORITHM FOR FINDING MINIMAL SIHONS

In this section, the global partitioning algorithm proposed by Cordone [13] is presented firstly. Then, a new algorithm based on depth-first search is proposed to compute all the minimal siphons. Subsequently, an example is shown to illustrate the proposed algorithm.

A. Cordone's Algorithm

According to the lemmas introduced in [13], an iterative search algorithm can be devised to find all siphons solving the generic problem $\Pi = (G, P_{in}, P_{out})$. This algorithm is based on suitable problem-reduction techniques (Lemmas 9 and 13) [13] and problem decomposition (Lemmas 2 and 12) [13] to explore the solution space.

In the following, lists will be employed, with the following notation. A list Λ is an ordered set of elements $\Lambda = (\lambda_1, \lambda_2, \ldots, \lambda_k)$. The pop function extracts the first element of a list: pop (Λ) returns λ_1 and the list is modified to $\Lambda = (\lambda_2, \ldots, \lambda_k)$. An empty list is denoted as $\Lambda = ()$.

Global-partitioning algorithm in [13] can be summarized as follows:

function Σ_Π = *FindAllMinimalSiphons*(G)
(Σ_Π, P_{out}) = *SinglePlaceSiphons*(G)
$\Pi = (G, \phi, P_{out})$
$\Lambda = (\Pi)$
$\Sigma_\Pi = \Sigma_\Pi \cup$ *SolveList*(Λ)

function (Σ_Π, P_{out}) = *SinglePlaceSiphons*(G)
$\Sigma_\Pi = \phi$, $\widetilde{P} = P$, $P_{out} = \phi$
while $\widetilde{P} \neq \phi$
p = Get ($\widetilde{P}$)
if $\bullet p = \phi$, then $\Sigma_\Pi = \Sigma_\Pi \cup \{p\}$, $P_{out} = P_{out} \cup \{p\}$
endif
$\widetilde{P} = \widetilde{P} - \{p\}$
endwhile

Function *SinglePlaceSiphons* is used to rule out siphons that are constituted by a single place, which is a minimal siphon.

function Σ_Π = *SolveList*(Λ)
$\Sigma_\Pi = \phi$
while $\Lambda \neq ()$
Π= *pop* (Λ)
if $\Sigma_\Pi = \phi$, then (S, Π) = *FindSiphon*(Π)
else $S = P$
endif
if $S \neq \phi$, then
if $S \neq P$in then S = *FindMinimalSiphon*(Π)
endif
$\Sigma_\Pi = \Sigma_\Pi \cup \{S\}$
$\Lambda = ((\Pi), \Lambda)$
Λ = *Partition* (Λ, S)
endif
endwhile

Function *SolveList* extracts the first problem in the list and searches for one generic siphon, subject to

the problem's place constraints, by means of the *FindSiphon* function.

function $(S,\Pi') = FindSiphon(\Pi)$
$\Pi' = \Pi$
isReducible =true
while isReducible
if $P_{in} \cap P_{out} \neq \phi$, then $S = \phi$, return
elseif $P_{in} \cup P_{out} = P$ then
if $\bullet P_{in} \subseteq P_{in}\bullet$ then $S = P_{in}$ else $S = \phi$, endif
return
endif
if $P_{out} \neq \phi$, then $G = red(G, P - P_{out})$, $\Pi' = (G, P_{in}, \phi)$
endif
(isReducible, Π') = $Reduce(\Pi')$
endwhile
$S = P$

Function *FindSiphon* iteratively tests the conditions of Lemma 8 [13] for trivial solutions of the siphon search problem and if none is found, reduces the problem according to Lemma 9 by means of the function *Reduce*.

function (isReducible, Π') = $Reduce(\Pi)$
$\Pi' = \Pi$
isReducible =true
$T = \{t \in T$, such that $\bullet t = \phi\}$, $P = T\bullet$
$\hat{T} = \{t \in \bullet Pin - Pin\bullet$, such that $|\bullet t| = 1\}$, $\hat{P} = \bullet\hat{T} \cap (P - P_{in})$
If $\overline{P} = \phi$ and $\hat{P} = \phi$ then
isReducible = false
else $\Pi' = (G, P_{in} \cup \hat{P}, P_{out} \cup \overline{P})$
endif

Function *Reduce* reduces the problem according to Lemma 9 [13] and returns a nonreducible siphon search problem and a siphon.

function $S = FindMinimalSiphon(\Pi)$
$S = P$, $\widetilde{P} = S - P_{in}$
while $\widetilde{P} \neq \phi$
$p = Get(\widetilde{P})$
if $(\bullet t \cap S) \supset \{p\}$ or $t \bullet \cap S = \phi$, $\forall t \in p\bullet$ then $S = S - \{p\}$
endif
$\widetilde{P} = \widetilde{P} - \{p\}$
endwhile
$\widetilde{P} = S - Pin$, $\widetilde{P}_{in} = P_{in}$
while $\widetilde{P} \neq \phi$
$p = Get(\widetilde{P})$, $\widetilde{G} = red(G, S - \{p\})$, $\widetilde{\Pi} = (\widetilde{G}, \widetilde{P}, \phi)$
$(\widetilde{S}, \widetilde{\Pi}) = FindSiphon(\widetilde{\Pi})$
if $\widetilde{S} \neq \phi$ then $S = \widetilde{S}$, $\widetilde{P} = S - \widetilde{P}_{in}$, $G = \widetilde{G}$
else $\widetilde{P} = \widetilde{P} - \{p\}$, $\widetilde{P}_{in} = P_{in} \cup \{p\}$ endif
endwhile

Function *FindMinimalSiphon* operates on a nonreducible siphon search problem, which admits at least one siphon, equal to the whole set of net places. Function *Get* returns an element of a set.

function $\widetilde{\Lambda} = Partition(\Lambda, S)$
$\widetilde{\Lambda} = ()$
while $\Lambda = ()$
$\Pi = pop(\Lambda)$
$\widetilde{P} = S - P_{in}$
while $\widetilde{P} \neq \phi$ and $P_{out} \cap P_{in} = \emptyset$
$p = \text{Get}(\widetilde{P})$
$\Pi = (G, P_{in}, P_{out} \cup \{p\})$
$(S, \Pi) = FindSiphon(\Pi)$
If $S \neq \phi$ then $\widetilde{\Lambda} = (\widetilde{\Lambda}, (\Pi))$
endif
$\widetilde{P} = \widetilde{P} - \{p\}$, $P_{in} = P_{in} \cup \{p\}$
endwhile
endwhile

Function *Partition* applies Lemma 12[13] to decompose the current problem in order to exclude all the siphons that contain S from the solution sets of the generated subproblems.

B. The Proposed Algorithm

The proposed algorithm is listed below:

function $\Sigma_\Pi = FindMinimalSiphon(\Pi_0)$
$\Sigma_\Pi = \phi$
$\Lambda = \phi$
$(S, \Pi) = FindSiphon(\Pi)$
If $S \neq \phi$
then $S = FindMinimalSiphon(\Pi)$
$\Sigma_\Pi = \Sigma_\Pi \cup \{S\}$
$\Lambda = (\Lambda, (\Pi, S))$
endif
while$(\Lambda \neq \phi)$
$(\Pi, S) = pop(\Lambda)$
$p = Get(S)$
$S = S - \{p\}$
if $S \neq \phi$ then
$P_{in} = P_{in} \cup \{p\}$
$\Lambda = (\Lambda, (\Pi, S))$
endif
$\Pi = (G, P_{in}, P_{out} \cup \{p\})$
$(S, \Pi) = FindSiphon(\Pi)$
If $S \neq \phi$ then
If $S \neq Pin$ then $S = FindMinimalSiphon(\Pi)$
endif
$\Sigma_\Pi = \Sigma_\Pi \cup \{S\}$
$\Lambda = (\Lambda, (\Pi, S))$
endif
endwhile

We take a simple Petri net G as an example to illustrate the proposed algorithm. Consider a simple Petri net G in Fig.1, where P= $\{p_1$-$p_6\}$ and T= $\{t_1$-$t_4\}$. G has three minimal siphons: S_1= $\{p_2, p_4, p_5, p_6\}$, S_2= $\{p_3, p_5\}$, S_3= $\{p_1, p_2\}$. The tree obtained by the proposed algorithm is shown as in Fig.2, which is the same as the tree obtained by Cordone's algorithm [13].

The problem and the number of problems can be obtained based on the proposed algorithm in the paper, which are shown in Table 1.

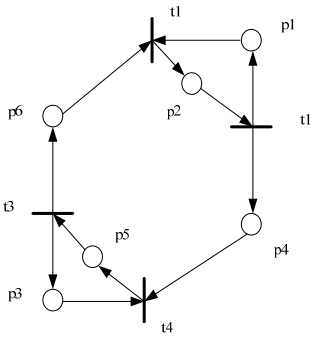


Figure 1. A Petri net G

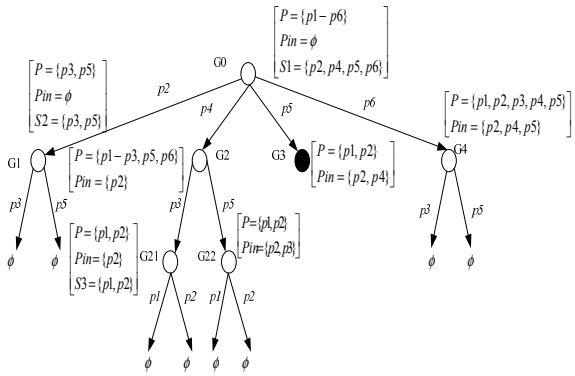


Figure 2**.** A tree obtained by the proposed algorithm

TABLE I.

PROBLEM LIST AND NUMBER OF PROBLEMS IN THE LIST

problem list	# of problem
Λ=(Π_0)	1
Λ=(Π_0, Π_1)	2
Λ=(Π_0, Π_2)	1
Λ=(Π_0, Π_2, Π_{21})	3
Λ=(Π_0, Π_2, Π_{22})	3
Λ=(Π_0, Π_4)	2
Λ=( )	2

According to the proposed algorithm, we can easily have the following results.

Theorem 1: Let G = (P, T, F) be a PN and Π_0 = (G, ϕ , ϕ) be the associated problem of finding all minimal siphons. Then, if either the proposed method or global partitioning algorithm is applied to solve Π_0, all minimal siphons of G are returned by the algorithm.

Proof: Similar to the proof in [13].

Theorem 2: Let G = (P, T, F) be a PN and Π_0 = (G, ϕ , ϕ) be the associated problem of finding all minimal siphons. Then, if either the proposed method or global partitioning algorithm is applied to solve Π_0, the algorithm will not return any non-minimal siphons of G.

Proof: Similar to the proof in [13].

Theorem 3: The maximum number of problems in the list does not exceed the number of minimal siphons.

Theorem 4: The maximum number of problems in the list is not more than the number of problem decomposition layer minus 1.

Theorem 5: The number of minimal siphons is equal to the number of problem decomposition layer minus 1.

Proof: The proof of *Theorem 3* to *Theorem 5* is obvious according to the proposed algorithm.

The global partitioning algorithm is based on breadth-first search. According to Theorem 1 and Theorem 2, if the global partitioning algorithm is applied to solve Π_0, all minimal siphons of G are returned by the algorithm. The proposed algorithm is based on depth-first search. According to Theorem 1 and Theorem 2, if the proposed algorithm is applied to solve Π_0, all minimal siphons of G are returned by the algorithm.

IV. ILLUSTRATIVE EXAMPLE AND COMPARISON WITH CORDONE'S ALGORITHM

First, a Petri net that has 36 places and 30 transitions is taken as an example[12] to be used to illustrate the difference between the two algorithms about requirement of computer memory in the course of computing minimal siphons. In Fig.3 (a), the requirement of computer memory is from 1.37 GB to 1.58 GB. In Fig.3 (b), the requirement of computer memory remains always 1.37 GB. The result indicates that the developed program based on the proposed algorithm takes up a relatively small computer memory.

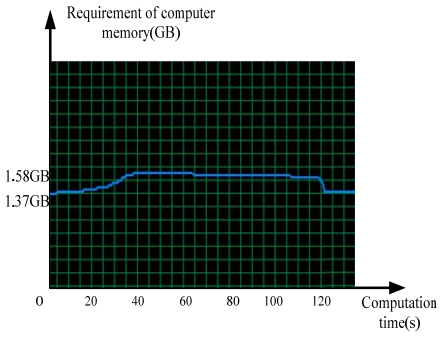
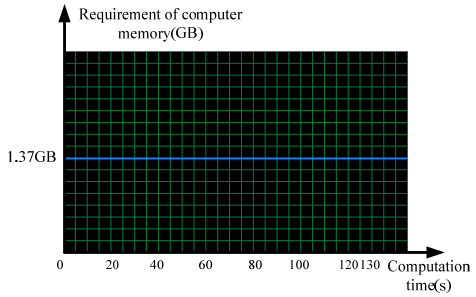


Figure 3. (a) Requirement of computer memory based on Cordone's algorithm



(b) Requirement of computer memory based on the proposed algorithm

Then consider an example [12] of a Petri net with 86 places and 70 transitions. In Fig.4 (a), the requirement of computer memory is from 1.15 GB to 2.88 GB. The computer memory is nearly full at 107th seconds and the developed program based on Cordone's algorithm stops running automatically. In Fig.4 (b), the requirements of computer memory remains always 1.15 GB and the developed program based on the proposed algorithm has been running.

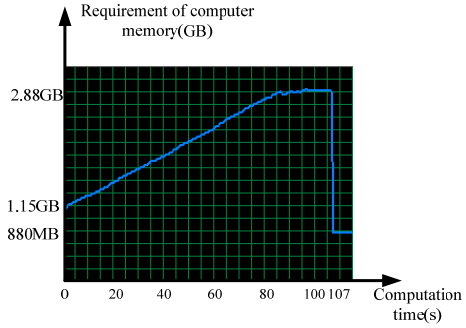
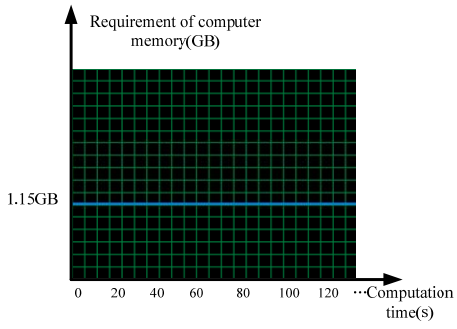


Figure 4. (a) Requirement of computer memory based on Cordone's algorithm



(b) Requirement of computer memory based on the proposed algorithm

Note that the computation above is carried out on a 2.9-GHz Pentium-III computer with 4-GB memory under Windows 7 operating system.

We know that the computer memory size which the developed program occupied depends on the number of problems that need to continue to be decomposed within the problem list. Similarly, the maximum requirement of computer memory depends on the maximum number of problems within the list.

Clearly, in Fig.4 (a), with the expansion of the size of the Petri net, the number of problems will increase exponentially and be far greater than the number of minimal siphons. Then, the more requirement of computer memory will be needed. Eventually, computer memory will be nearly full and the developed program will stop running. But, in Fig.4 (b), with the expansion of the size of the Petri net, the number of problems will be mutative within a certain range and does not exceed the number of minimal siphons. Moreover, the requirement of computer memory remains always constant. We can see that the superiority of the proposed algorithm that can greatly reduce the number of problems within the problem list is obvious. Moreover, with the expansion of the size of the Petri net, this superiority is more and more obvious.

V. CONCLUION

It is well know that deadlock problem [15]-[24] is related to minimal siphons. In the paper, a new algorithm based on depth-first search of problem decomposition process is proposed to compute all minimal siphons for an ordinary Petri net. Comparison with Cordone's algorithm, the proposed algorithm can solve the problem of excessive requirement for computer memory in the course of computing all minimal siphons and decrease requirement of computer memory. Future work includes extending this algorithm to improve computation minimal siphon efficiency for ordinary Petri net.

REFERENCES

[1] 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, November. 2006.

[2] Z. W. Li and M. C. Zhou, "Elementary siphons of Petri nets and their application to deadlock prevention in flexible manufacturing systems," IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, vol. 34, no. 1, pp. 38–51, Janury. 2004.

[3] Z.W. Li and A. R.Wang, "A Petri net based deadlock prevention approach for flexible manufacturing systems," Acta Autom. Sin., vol. 29, no. 5, pp. 733–740, 2003.

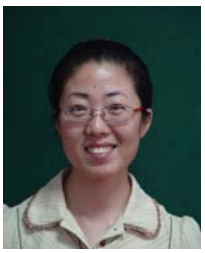
[4] Z. W. Li and M. C. Zhou, "Two-stage method for synthesizing livenessenforcing supervisors for flexible manufacturing systems using Petri nets," IEEE Trans. Ind. Inf., vol. 2, no. 4, pp. 313–325, November. 2006.

[5] 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, January. 2009.

[6] H. Hu, M.C. Zhou, and Z. Li, "Supervisor design to enforce production ratio and absence of deadlock in automated manufacturing systems," IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, vol. 41, no. 2, pp. 201–212, March. 2010.

[7] J. Park and S. A. Reveliotis, "Deadlock avoidance in sequential resource allocation systems with multiple resource acquisitions and flexible routings," IEEE Trans. Autom. Control, vol. 46, no. 10, pp. 1572–1583, Oct. 2001.

[8] J. Ezpeleta, J. M. Colom, and J.Martinez, "A Petri net based deadlock prevention policy for flexible manufacturing systems," IEEE Trans. Robot. Autom., vol. 2, no. 2, pp. 173–184, April. 1995.
[9] S.G. Wang, C.Y. Wang ,M.C. Zhou, and Z.W. Li, "A Method to Compute Strict Minimal Siphons in a Class of Petri Nets Based on Loop Resource Subsets," IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, vol. 42, no. 1, pp. 226–237, January. 2012.
[10] Z.W. Li and M. C. Wang, "On Siphon Computation for Deadlock Control in a Class of Petri Nets," IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, vol. 38, no. 3, pp. 667–679, 2008.
[11] E. R. Boer and T. Murata, "Generating basis siphons and traps of Petri nets using sign incidence matrix," IEEE Trans. Circuits Syst. I, Fundam. Theory Appl., vol. 2, no. 4, pp. 266–271, April. 1994.
[12] Computation results for SMS of S3PR, Nov. 2010. [Online]. Available: http://web.njit.edu/~zhou/SMS.rar.
[13] 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.
[14] R. Cordone, L. Ferrarini, and L. Piroddi, "Some results on the computation of minimal siphons in Petri nets," in Proc. 42nd IEEE Conf. Decision Control, Maui, HI, Dec. 9–12, 2003, vol. 4, pp. 3754–3759.
[15] S.G. Wang, D.W., Z.H. Tan, X.X. G. and C.Y. Wang, "A Fast Convergence Deadlock Control Strategy of Petri Nets Based on MIP Algorithm", IJACT: International Journal of Advancements in Computing Technology, Vol. 4, No. 17, pp. 589 –598, 2012.
[16] X.X. Guan, Y. Li, J.X. Xu, C.Y. Wang and S.G. Wang, "A Literature Review of Deadlock Prevention Policy Based on Petri Nets for Automated Manufacturing Systems", JDCTA: International Journal of Digital Content Technology and its Applications, Vol. 6, No. 21, pp. 426 – 433, 2012.
[17] M. Qin, Z. W. Li, M. C. Zhou, "Deadlock Prevention for a Class of Petri Nets With Uncontrollable and Unobservable Transitions," IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, vol. vol. 42, no. 3, pp.727–737, MAY 2012.
[18] Pan, Yen-Liang, "An Efficient Deadlock Prevention Policy for Flexible Manufacturing Systems Using the Theory of Regions and Selective Siphon Method," Advanced Science Letters, vol. 15, no. 1 pp. 342–345, 2012.
[19] R. M. Zhu, "A deadlock prevention approach for flexible manufacturing systems with uncontrollable transitions in their Petri net models," Asian Journal of Control, vol.14, no. 1 pp. 217–229, 2012.
[20] Z. W. Li, G. Y. Liu, Michael, H. M, M. C. Zhou, "Deadlock Prevention Based on Structure Reuse of Petri Net Supervisors for Flexible Manufacturing Systems," IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, vol. 42, no. 1, pp.178-191, Jan. 2012.
[21] Benjun Guo, Jianhong Gan, Jingwen Su, Yuewen Hu, "A Pervasive Computing Model of Internet of Things based on Computing Area Network", Journal of Computers, vol.7, no.7, pp.1647-1655, July 2012.
[22] Wei Qiu, Li-Chen Zhang, "Research on Real-Time Software Development Approach", Journal of software, vol.7, no.7, pp.1593-1601,July 2012.
[23] Lei Peng, Yuanzhen Wang, Guangming Dai, Zhongsheng Cao, "A Novel Differential Evolution with Uniform Design for Continuous Global Optimization", Journal of Computers, vol. 7, no. 1 (2012), 3-10, Januarury 2012.
[24] Honghua Xu, Xiaoqiang Di, Huamin Yang, and so on. "Intelligent PID Controller on Soft Computing", Journal of Computers, vol.7, n.10, pp.2417-2424, 2012.



Fan Ning received the B.S. degree in circuit systems from Hangzhou Dianzi University, HangZhou, China, in 2007. In 1999, she joined Hangzhou Vocational and Technical College, Hangzhou, China, where she is currently an Associate Professor with Department of Information Science and Electronic. Her main research interests include Petri net theory and application, supervisory control of discrete event systems.

Shouguang Wang (M'01) received the B.S. degree in computer science from Changsha University of Science and Technology, Changsha, China, in 2000 and the Ph.D. degree in electrical engineering from Zhejiang University, Hangzhou, China, in 2005 In 2005, he joined Zhejiang Gongshang University, Hanhzhou, China, where he is currently an Associate Professor with the College of Information & Electronic Engineering. He has been a visiting professor with the Electrical and Computer Engineering Department, New Jersey Institute of technology, Newark, NJ, since January 2011. His main research interests include Petri net theory and application, supervisory control of discrete event systems.

Xingxing Li received the B.S. degree in circuit systems from Hangzhou Dianzi University, HangZhou, China, in 2008. In 1999, he joined Zhejiang Gongshang University, Hangzhou, China. He main research interests include computer technology and application.

Qiaoli Zhuang Lecture in the School of Information Science and Technology, Zhe- jiang Sci-Tech University, and at the same time, she is a PhD candidate of mechanical design and theory in School of Machinery and Automatic control, Zhe- jiang Sci-Tech University. Her research interests include supervisory control of DES and Petri nets.