An Incremental Approach to Extracting Minimal Bad Siphons.

Article *in* Journal of Information Science and Engineering · January 2007

Source: DBLP

Daniel Chao
mis, nccu

JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 23, 203-214 (2007)

An Incremental Approach to Extracting Minimal Bad Siphons

DANIEL YUH CHAO
Department of Management and Information Science
National Chengchi University
Taipei, 116 Taiwan
E-mail: yaw@mis.nccu.edu.tw

Finding all minimal bad siphons is essential for deadlock control. However, the number of siphons grows exponentially with the size of the system. Deadlock occurs due to inappropriate resource sharing. Hence most research focused on the problem of minimal siphon extraction covering a set of places representing resources — an NP-Complete problem for arbitrary Petri Nets. We develop the theory for efficient extraction of minimal bad siphons for S^3PR (*systems of simple sequential processes*) proposed by Ezpeleta *et al*. The number of minimal bad siphons that needs to be searched is linear to the number of resources. The rest can be found by adding and deleting common sets of places from existing ones significantly reducing the search time. It is very **interesting** that both nets and siphons can be synthesized by first locating a circuit followed by adding handles.

Keywords: Petri nets, siphons, traps, FMS, algorithm, liveness, deadlock

1. INTRODUCTION

Liveness in *flexible manufacturing systems* (*FMS*) modelled by *Petri nets* (*PN*) is closely related to siphons [1, 2] whose tokens can be emptied completely. A siphon (trap, respectively) is a set of places where tokens can leak out (inject in, respectively). If a siphon contains a trap, then tokens in it cannot leak out completely. A *bad siphon* (*BS*) is a siphon that does not contain a trap. Once a BS is found that can be emptied, output transitions of places in the siphon can never be fired. Hence the net is not live. In this situation, we can construct a control policy based on the total number of tokens in the BS.

The FMS model consists of a set of *working processes* (*WP*, Def. 4) competing for resources. A WP models a sequence of operations to manufacture a product. Circular wait for resources can bring the system into a deadlock where some WP can never finish.

Ezpeleta *et al.* proposed a class of nets called S^3PR [1] where each WP is a *state machine* (*SM*) plus resource places. Their idea is to compute all *minimal bad siphons* (*MBS*) based on the approaches in [3, 4] with exponential time complexity. Then it finds the maximum number of tokens at each idle state followed by a control policy of adding control arcs and nodes with tokens. Most recent deadlock control approaches [5-7] extend Ezpeleta's work. Efficient methods to compute MBS are urgently needed.

Because deadlock occurs due to inappropriate resource sharing, all deadlock pre-

Received October 28, 2004; revised February 18, 2005; accepted July 11, 2005.
Communicated by Ding-Zhu Du.

vention approaches [5-7] consider only siphons containing resource places. Watanabe *et al*. [3] showed that the *minimal siphons extraction problem* (*MSEP*) covering a set Q of places for general PN is an NP-Complete problem. They also proposed algorithms to enumerate all *minimal siphons* (*MS*) containing Q. Esparza [12] proposed a polynomial time algorithm for MSEP of *free-choice nets* (*FC*). Kemper [13] improved it with a linear time algorithm for searching minimal siphons containing a place. However, most FMS such as S^3PR cannot be modeled by FC.

References [2, 3] presented algorithms for generating all basis siphons. However, it is more efficient to employ MS (the number is usually less) to analyze liveness [1, 14]. We search only MBS. In addition, we find that they can be divided into two groups. Those in the first are called *basic siphons* (Def. 6); those in the second (called *compound siphons*) can be derived from the first in terms of formulas, thus reducing the search time greatly. For instance, in the example in [1] (Fig. 1), out of the 18 minimal bad siphons, we search only 6 and derive the rest 12. Further, these 6 are so simple that we can manually search them.

Section 2 presents the basis to understand the paper. It defines S^3PR. Section 3 shows that siphon can be synthesized by constructing handles upon a circuit. Section 4 computes all compound siphons respectively. Section 5 shows an efficient technique to compute all MBS for S^3PR. Section 6 compares ours with other approaches. Finally, section 7 concludes the paper. However, in order to make the paper as self-contained as possible, Appendix 1 is included with the definition of the main concepts related to models used in this paper. **For the sake of discussion continuity, all proofs are reported in Appendix 2.**

2. PRELIMINARIES

Definition 1 A subnet $N_i = (P_i, T_i, F_i)$ of N is generated by $X = P_i \cup T_i$, if $F_i = F \cap (X \times X)$. It is an *I–subnet* (denoted I) of N if $T_i = \bullet P_i$. An *MBS*, denoted D_m, is an MS that does not contain a trap. I_D is the I–subnet of a D_m. Note that $D = P(I_D)$; D_m is the set of places in I_D.

We follow [8] for the definitions of *handles*, *bridges*, *AB-handles*, and *AB-bridges* where A and B can be T or P. Roughly speaking, a "handle" is an alternate disjoint path between two nodes. A PT-handle starts with a place (as indicated by 'P' in 'PT-') and ends with a transition while a TP-handle starts with a transition and ends with a place.

Definition 2 Let $N = (P, T, F)$. $H_1 = [n_s n_1 n_2 \ldots n_k n_e]$ and $H_2 = [n_s n'_1 n'_2 \ldots n'_h n_e]$ are elementary directed paths, $n_i, n'_j \in P \cup T$, $i = 1, 2, \ldots, k$, $j = 1, 2, \ldots, h$. H_1 and H_2 are said to be *mutually complementary*. Each is called a handle in N if $n_i \neq n'_j$, $\forall i, j$. $P_{in}(H_1) = \{p \mid p \in H_1, p \neq n_s, p \neq n_e\}$ is the set of interior (not terminal) places of H_1. The handle *H to a subnet N*' (similar to the handle of a tea pot) is an elementary directed path from n_s in N' to another node n_e in N'; any other node in H is not in N'. H is said to be a handle *in* $N' \cup H$.

In Fig. 1, $H_1 = [p_2\, t'_2 p'_3\, t'_3\, p'_4 t'_4 p_5\, t_5]$ and $H_2 = [p_2\, t_2 p_3\, t_3\, p_4 t_4 p_5\, t_5]$, $n_s = p_2$, $n_e = t_5$. $P_{in}(H_1) = \{p'_3, p'_4, p_5\}$.

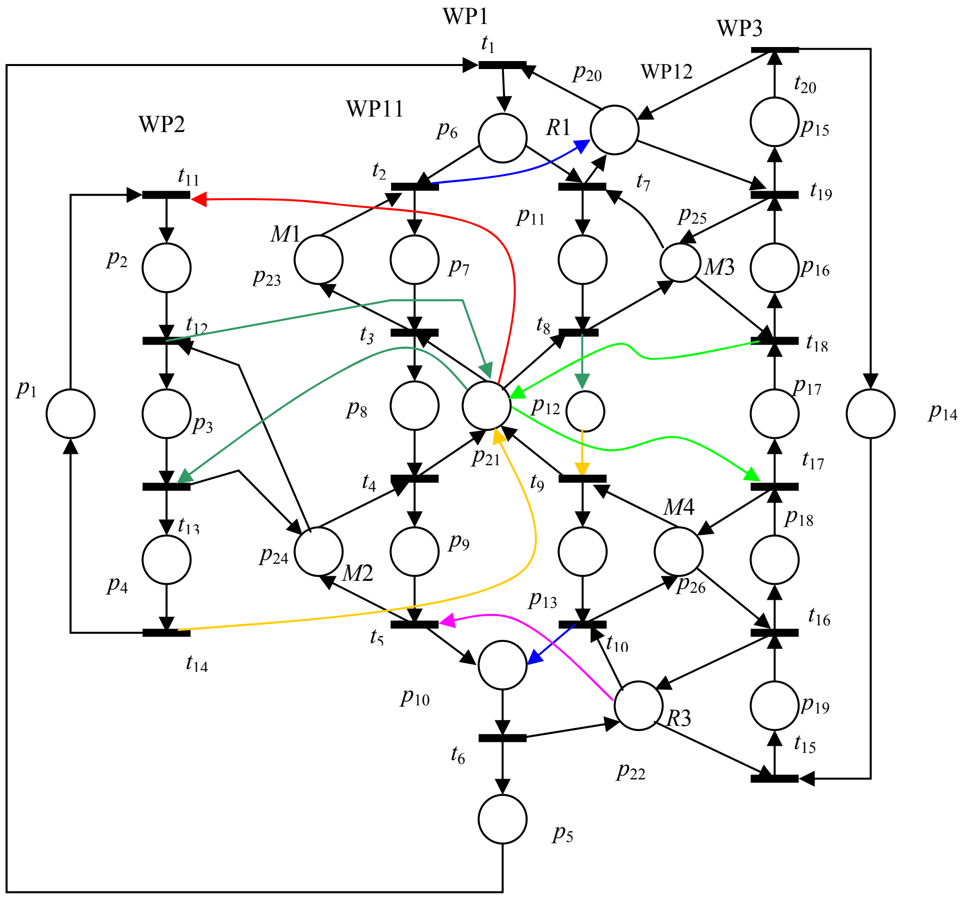

Fig. 1 [1]. An example of systems of simple sequential processes with resources (S^3PR).

Definition 3 A *Simple Sequential Process* (S^2P) is a net $N = (P \cup \{p^0\}, T, F)$ where: (1) $P \neq \emptyset$, $p^0 \notin P$ (p^0 is called the process idle or initial or final state); (2) N is strongly connected state machine; and (3) every circuit of N contains the place p^0. $\forall p \in P$, if $|p\bullet| > 1$, p is called a *choice place*.

Definition 4 A *simple sequential process with resources* (S^2PR) , also called a *working processes* (*WP*), is a $N = (P \cup \{p^0\} \cup R, T, F)$ so that (1) The subnet generated by $X = P \cup \{p^0\} \cup T$ is an S^2P; (2) $R \neq \emptyset$ and $P \cup \{p^0\} \cap R = \emptyset$; (3) $\forall p \in P$, $\forall t \in \bullet p$, $\forall t' \in p\bullet$, $\exists r_p \in R$, $\bullet t \cap R = t'\bullet \cap R = \{r_p\}$; (4) The two following statements are verified: (a) $\forall r \in R$, $\bullet\bullet r \cap P = r\bullet\bullet \cap P \neq \emptyset$; (b) $\forall r \in R$, $\bullet r \cap r\bullet = \emptyset$; (5) $\bullet\bullet(p^0) \cap R = (p^0)\bullet\bullet \cap R = \emptyset$. $\forall p \in P \cup \{p^0\}$, p is called a *state place*. $\forall r \in R$, r is called a *resource place*. $H(r) = \bullet\bullet r \cap P$ denotes the set of holders of r (states that use r). $\rho(r) = \{r\} \cup H(r)$.

The above models the constraint as follows: (3) allows only one shared resource to be used at each state; (4.a) dictates that the resource used in a state be released when moving to the next state; (4.b) shows that two adjacent states cannot use the same resource and (5) ensures that initial and final state do not use resources.

Definition 5 A *system of S^2PR* (S^3PR) is defined recursively as follows: (1) An S^2PR is defined as an S^3PR; (2) Let $N_i = (P_i \cup P_i^0 \cup R_i, T_i, F_i)$, $i \in \{1, 2\}$ be two S^{3}PR so that $(P_1 \cup P_1^0) \cap (P_2 \cup P_2^0) = \emptyset$, $R_1 \cap R_2 = P_C$ $(\neq \emptyset)$ and $T_1 \cap T_2 = \emptyset$. The net $N = (P \cup \{P^0\} \cup R, T, F)$ resulting from the composition of N_1 and N_2 via P_C defined as follows: (1) $P = P_1 \cup P_2$; (2) $P^0 = P_1^0 \cup P_2^0$; (3) $R = R_1 \cup R_2$; (4) $T = T_1 \cup T_2$ and (5) $F = F_1 \cup F_2$ is also an S^3PR. **A directed path (circuit, subnet) Γ in N is called a resource path (circuit, subnet) if $\forall p \in \Gamma, p \in R$.**

Lemma 1 [6] For an S^{3}PR, $\rho(r)$ is both a trap and a siphon.

An example of S^{3}PR is shown in Fig. 2 where $\rho(R1) = \{p_{20}, p_6, p_{15}\}$.

3. SYNTHESIS OF MBS FOR S^{3}PR

Deadlock occurs when all resources in a circuit are used up since processes mutually waiting for them indefinitely. Thus, MBS must have something to do with circuits with resources as implied by the following lemma:

Lemma 2 [14] I_D is strongly connected and has an elementary circuit.

Upon the circuit, we construct handles to form the I of an MBS based on Lemma 3.

Lemma 3 [20] (1) If subnet N' is the I-subnet of a minimal siphon, then each handle in N' is a PP- or TP- or *virtual* PT-handle (virtual means containing only two nodes), neither a TT- nor a nonvirtual PT-handles, and there are none of PP-, TP-, and virtual PT-handles to N'. (2) N' is the I-subnet of a bad siphon D, *iff* there is a nonvirtual (more than two nodes) PT-handle to N'.

Example: In Fig. 1, first find a circuit $c = [p_{21}\ t_8\ p_{25}\ t_{18}\ p_{21}]$. Second add TP-handles $[t_{18}\ p_{16}\ t_{19}\ p_{23}]$ and $[t_8\ p_{12}\ t_9\ p_{21}]$ plus PP-handles $[p_{21}\ t_{11}\ p_2\ t_{12}\ p_{21}]$, $[p_{21}\ t_{13}\ p_4\ t_{14}\ p_{21}]$, and $[p_{21}\ t_3\ p_8\ t_4\ p_{21}]$ to get $D_m^{18} = \{p_2, p_4, p_8, p_{12}, p_{16}, p_{21}, p_{23}\}$ with a nonvirtual PT-handle $[p_{21}\ t_{17}\ p_{17}\ t_{18}]$ to its c. Such a procedure to form MBS from a circuit is called *handle-construction*.

Thus, *we synthesize minimal siphons by constructing a circuit followed by handles similar to the knitting technique* [11]. This is very **interesting** since both nets and siphons can be synthesized by first locating a circuit followed by adding handles.

Based on Lemma 3, upon merging an r, we try to locate an elementary resource circuit c. If no c can be found, there is no D_m associated with r. Otherwise, we add a PP- or a TP- or a virtual PT- handle to c. Continue this process until no more such handles can be found. Deleting all TT-handles from the resulting subnet renders an I-subnet, wherein the set of places is a D_m. We then merge another nearby r and repeat the above process until all r have been merged. Thus, we merge r from top to bottom (corresponding to the direction in a WP from idle state to final state). This is more efficient than to search r in a random fashion.

The example in Fig. 1 consists of three robots ($R1$, $R2$, $R3$) and four machines ($M1$ – $M4$). We first find the backbone; each separate component is an SM. We then merge resource place $R1$; we cannot find a circuit for the first merge. For the second, we merge $M3$ with no success because it is not minimal since it contains $\rho(R1)$. We then merge $R2$ with $M3$ to find a circuit containing $R2$ and $M3$.

Note that we need to consider a second circuit $[p_{20}\ t_{19}\ p_{25}\ t_{18}\ p_{21}\ t_3\ p_{23}\ t_2 p_{20}]$ containing $M1$, which is used exclusively for the middle process (WP_1). We merge the rest in the order of $M2$, $M4$, and $R3$.

There are two types of circuits that may induce MBS: elementary (*e.g.*, $[p_{21}\ t_{13}\ p_{24}\ t_{12}\ p_{21}]$ in Fig. 1) and compound (*i.e.*, multiple interconnected elementary circuits). The corresponding siphons are called *basic siphons* and *compound* siphons (see Tables 1 and 2), denoted c_b and c_p respectively.

Table 1. Basic siphons.

Basic siphons	places	Basic circuits c_b	Between WP
D_m^1	p_{10}, p_{18},vp_{22}, p_{26}	$[p_{22}\ t_{10}\ p_{26}\ t_{16}\ p_{22}]$	WP_{12}, WP_3
D_m^4	$p_4, p_{10}, p_{17}, p_{21}, p_{22}, p_{24}, p_{26}$	$[p_{21}\ t_{17}\ p_{26}\ t_{16}\ p_{22}\ t_5\ p_{24}\ t_4\ p_{21}]$	WP_{11}, WP_3
D_m^{10}	$p_4, p_9, p_{12}, p_{17}, p_{21}, p_{24}$	$[p_{21}\ t_{13}\ p_{24}\ t_4 p_{21}]$	WP_{11}, WP_2
D_m^{16}	$p_2, p_4, p_8, p_{13}, p_{17}, p_{21}, p_{26}$	$[p_{21}\ t_{17}\ p_{26}\ t_9\ p_{21}]$	WP_{12}, WP_3
D_m^{17}	$p_2, p_4, p_8, p_{12}, p_{15}, p_{20}, p_{21}, p_{23}, p_{25}$	$[p_{21}\ t_3\ p_{23}\ t_2\ p_{20}\ t_{19}\ p_{25}\ t_{18}\ p_{21}]$	WP_{11}, WP_3
D_m^{18}	$p_2, p_4, p_8, p_{12}, p_{16}, p_{21}, p_{25}$	$[p_{21}\ t_8\ p_{25}\ t_{18} p_{21}]$	WP_{12}, WP_3

Table 2. Compound siphons.

Compound siphons	Places
D_m^2	$p_4, p_{10}, p_{15}, p_{20}, p_{21}, p_{22}, p_{23}, p_{24}, p_{25}, p_{26}$
D_m^3	$p_4, p_{10}, p_{16}, p_{21}, p_{22}, p_{24}, p_{25}, p_{26}$
D_m^5	$p_4, p_9, p_{13}, p_{15}, p_{20}, p_{21}, p_{23}, p_{24}, p_{25}, p_{26}$
D_m^6	$p_4, p_9, p_{13}, p_{16}, p_{21}, p_{24}, p_{25}, p_{26}$
D_m^7	$p_4, p_9, p_{13}, p_{17}, p_{21}, p_{24}, p_{26}$
D_m^8	$p_4, p_9, p_{12}, p_{15}, p_{20}, p_{21}, p_{23}, p_{24}, p_{25}$
D_m^9	$p_4, p_9, p_{12}, p_{16}, p_{21}, p_{24}, p_{25}$
D_m^{11}	$p_2, p_4, p_8, p_{10}, p_{15}, p_{20}, p_{21}, p_{22}, p_{23}, p_{25}, p_{26}$
D_m^{12}	$p_2, p_4, p_8, p_{13}, p_{15}, p_{20}, p_{21}, p_{23}, p_{25}, p_{26}$
D_m^{13}	$p_2, p_4, p_8, p_{10}, p_{16}, p_{21}, p_{22}, p_{25}, p_{26}$
D_m^{14}	$p_2, p_4, p_8, p_{13}, p_{16}, p_{21}, p_{25}, p_{26}$
D_m^{15}	$p_2, p_4, p_8, p_{10}, p_{17}, p_{21}, p_{22}, p_{26}$

For *compound* siphons, some PP-path to a c_b^i is another c_b^j corresponding to D_m^i and D_m^j respectively. Because $D_m^i \cup D_m^j$ is another siphon D^a, we can obtain a new D_m from it by deleting some places which are on TT-handles of $I(D^a)$. In other words, compound siphons can be derived from basic siphons. It is interesting to see that the basic and compound siphons correspond to elementary and redundant ones in [17]. The following formally defines them:

Definition 6 An elementary (compound) resource circuit (Def. 5) is called a *basic circuit*, denoted $c_b(c_p)$. The corresponding MBS obtained by the handle-construction procedure is called a *basic siphon* (*compound siphon*).

Note that c_b cannot be part of an $I(\rho(r))$. Otherwise, I_D contains only one resource place r; it cannot be bad since $\rho(r)$ is both a trap and a siphon [6]. The minimal siphon containing $M4$ is $\rho(M4) = \{p_{26}, p_{18}, p_{13}\}$, which is also a trap. The following lemma helps to locate a c_b.

Lemma 4 [20] All places in the c_b must be resource places.

Thus we need only search I_D where all places in c_b are resource places; *i.e.*, c_b is a *resource circuit* (Def. 5). Note that c_b may appear in a single WP.

Note that the set of resources shared between WP_{12} *and* WP_3 *is* $\{R1, M3, R2, M4, R3\}$ *listed in the order from top to bottom.* That between WP_{11} and WP_3 is $A = \{R1, R2, R3\}$. We say that the circuits extend between two adjacent processes. This is true (Table 1) for the S^3PR model in Fig. 1, which we will *assume in this paper*. Note that the set A is smaller than that used by WP_{12}. *There is an elementary circuit c covering each pair of two successive resources in the set.*

There is at most $k' - 1$ c_b where k' is the cardinality of the above set. This is in general true and eases the task of c_b search. Although there may be c_b that span multiple WP, they are much fewer than the aforementioned c_b in most cases. Hence the number c_b to search is $O(n)$. Since all places in a c_b are resources, we can remove all state places and their incident arcs and apply well-known algorithms [18] to search elementary directed circuits.

4. COMPUTATION OF COMPOUND SIPHONS for S^3PR

The idea is based on the following:

Lemma 5 [12] The union of two siphons $D_1 \cup D_2$ is another siphon.

Thus, new D_m may be constructed by the deletion of a set of places from the union of two basic siphons. For instance, in Table 3 we merge $M4$ to create D_m^{16} containing $R2$ and $M4$. We can then generate $D_m^5 - D_m^7$, D_m^{12}, D_m^{14} from $D_m^8 - D_m^{10}$, D_m^{17}, D_m^{18} respectively by adding the minimal siphon that contains $M4$, $\rho(M4)$ and deleting the set of places $P_4 = \{p_{12}, p_{18}\}$ — no need to find TT- and PT-paths again.

Note that all $D_m^8 - D_m^{10}$, D_m^{17}, D_m^{18} contain $R2$; hence, all I of the union of $\rho(M4)$ and $D_m^8 - D_m^{10}$, D_m^{17}, D_m^{18} share the same TT-handle $[t_8\, p_{12}\, t_9]$ and PT-handle $[p_{26}\, t_{16}\, p_{18}\, t_{17}]$.

We call $M4$ a seed (SD), D_m^8 the Companion D_m (CP), and P_4 the Common Deletion (CD). New $D_m(=\rho(SD) \cup CP \setminus CD)$ formed in this fashion are listed in Table 3.

Table 3. New D_m generated based on the formula: $D_m = D_m^c \cup \rho(SD) \setminus CD$.

Note that $D_m^{10} = \rho(R2) \cup \rho(M2) \setminus CD$, $D_m^{16} = \rho(R2) \cup \rho(M4) \setminus CD$, $D_m^1 = \rho(R3) \cup \rho(M4) \setminus CD$. No need to search these D_m; hence reducing the search time greatly.

Seed (SD)	Companion $D^c(CP)$	Common Deletions (CD)	New D_m
$M2$	D_m^{18}	$\{p_2, p_3, p_8\}$	$D_m^9 = D_m^{18} \cup \rho(M2) \setminus CD$
	D_m^{17}	$\{p_2, p_3, p_8\}$	$D_m^8 = D_m^{17} \cup \rho(M2) \setminus CD$
$M4$	D_m^{10}	$\{p_{18}, p_{12}\}$	$D_m^7 = D_m^{10} \cup \rho(M4) \setminus CD$
	D_m^{18}	$\{p_{18}, p_{12}\}$	$D_m^{14} = D_m^{18} \cup \rho(M4) \setminus CD$
	D_m^{17}	$\{p_{18}, p_{12}\}$	$D_m^{12} = D_m^{17} \cup \rho(M4) \setminus CD$
	D_m^9	$\{p_{18}, p_{12}\}$	$D_m^6 = D_m^9 \cup \rho(M4) \setminus CD$
	D_m^8	$\{p_{18}, p_{12}\}$	$D_m^5 = D_m^8 \cup \rho(M4) \setminus CD$
$R3$	D_m^{16}	$\{p_{13}, p_{19}\}$	$D_m^{15} = D_m^{16} \cup \rho(R3) \setminus CD$
	D_m^{14}	$\{p_{13}, p_{19}\}$	$D_m^{13} = D_m^{14} \cup \rho(R3) \setminus CD$
	D_m^{12}	$\{p_{13}, p_{19}\}$	$D_m^{11} = D_m^{12} \cup \rho(R3) \setminus CD$
	D_m^6	$\{p_{13}, p_{19}\}$	$D_m^3 = D_m^6 \cup \rho(R3) \setminus CD$
	D_m^5	$\{p_{13}, p_{19}\}$	$D_m^3 = D_m^5 \cup \rho(R3) \setminus CD$

The theory is briefed below. Recall that there is an elementary circuit c covering each pair of two successive resources shared between two adjacent WP. Let the resources from top to bottom be $r_1, r_2, r_3, \ldots, r_k$ corresponding to $c_b^1, c_b^2, \ldots, c_b^i, \ldots, c_b^{k-1}$ respectively. When we reach c_b^i, all compound siphons c_p (with $I_D = I_p$ and $D = D_p$) that contains c_b^{i-1} can join c_b^i to form a new c_p' (with $I_D = I_p$') and a new compound siphon D_p'. Let $I_D^i = (I(\rho(r_i) \cup I(\rho(r_{i+1})) \setminus \Gamma$ where Γ is the set of PT-handles to c_b^i. Also $D^i = \rho(r_i) \cup \rho(r_{i+1}) \setminus C_S^{r_i r_{i+1}}$ where $C_S^{r_i r_{i+1}}$ is the set of interior places (*i.e.*, no end nodes) $\boldsymbol{P_{in}}$ in Γ and called complementary siphon in [17].

Note that one PT-handle becomes a TT-handle (*e.g.*, $[t_8\ p_{12}\ t_9]$ mentioned earlier) in I_D' for D' and hence should also be deleted from $I_p \cup I(\rho(r_{i+1}))$, *i.e.*, $I_p' = (I_p \cup I(\rho(r_{i+1})) \setminus \Gamma$ and $D_p' = (D_p \cup \rho(r_{i+1})) \setminus C_S^{r_i r_{i+1}}$. Comparing this with the formula earlier, we have $CP = D_p$, $SD = r_{i+1}$, and $CD = C_S^{r_i r_{i+1}}$.

We now propose a rough algorithm to find all MBS:

Algorithm 1 D_m Computation Algorithm for S^3PR
1. Find all c_b.
2. Find all basic siphons using the handle-construction procedure.
3. Find all copound siphons using the formula: $D_m = D_m^c \cup \rho(SD) \setminus CD$.

A more detail and efficient algorithm is proposed in the next section.

5. EFFICIENT TECHNIQUE FOR S^3PR

Based on the theory presented in last two sections, we will present one efficient technique for computer implementation. It locates D_m in a local and incremental fashion. We first present a problem for the handle-construction procedure.

An example is shown in Fig. 1 to find D_m^{17}. We first locate the circuit $[p_{20}\ t_{19}\ p_{25}\ t_{18}\ p_{21}\ t_3\ p_{23}\ t_2\ p_{20}]$ followed by TP-handles $[t_3\ p_8\ t_4\ p_{21}]$, $[t_{19}\ p_{15}\ t_{20}\ p_{20}]$ and $[t_8\ p_{12}\ t_9\ p_{21}]$; PP-handles $[p_{21}\ t_{11}\ p_2\ t_{12}\ p_{21}]$ and $[p_{21}\ t_{13}\ p_4\ t_{14}\ p_{21}]$. Now if we add PP-handle $[p_{20}\ t_1\ p_6\ t_7\ p_{20}]$, we will hit virtual PT-handle $[p_6\ t_2]$ and it is not minimal since it contains $\rho(R1)$. This implies we should undo the addition of $[p_{20}\ t_1\ p_6\ t_7\ p_{20}]$ where the place $p_6 \notin D_m^{17}$. Hence, we add PP-handle $[p_{25}\ t_7\ p_{20}]$ instead. We stop here since the rest are all TT- or PT-handles (termed *case* 1).

This problem of undoing can be avoided by adding PP-handles of the form $[r\ t\ r']$ (i.e., $[p_{25}\ t_7\ p_{20}]$) prior to other kinds of PP-handles ($[p_{20}\ t_1\ p_6\ t_7\ p_{20}]$). This is performed in Step 3 of the following algorithm. The correctness of which is proved in Theorem 1.

Algorithm 2 D_m Search Algorithm for S^3PR

1. Add a new r'.
2. Find a new c_b' that contains r'. If c_b' is not found, go to step 1 and repeat.
3. Add all PP-handles of the form $[r\ t\ r']$ followed by the rest of PP-handles and all TP-handles to c_b'. Denote the resulting net I' (I-subnet) where the set of places $D_m^n = P(I')$. If D_m^n contains a $\rho(r)$, $r \in c_b$', then it is not minimal, go to step 1 and repeat.
4. Delete all TT- and nonvirtual PT-paths (except their terminal nodes) in $I^e \cup I$' where I^e is the I-subnet of any existing D_m^e that contains a place in c_b' to form a new I'.
5. Go to step 2 and repeat until all resource places have been added.

Since the I-subnet of any D_m is strongly connected, it must contain at least one circuit c_b' to be found at step 2. The rest in I-subnets are handles to c_b' as in Lemma 3. The correctness of this algorithm is established in the following theorem.

Theorem 1 Algorithm 2 computes all minimal bad siphons.

The time complexity $c(k)$ at the kth iteration step is dominated by step 4. We have $c(k) = c(k-1) + (c(k-1) + 1)$ in the worst case since $c(k-1)$ new D_m are created from $c(k-1)$ existing D_m and a new D_m is created from the new c_b'. Solving this equation, we have the total time complexity of $O(2^{n'})$, where n' is the total number of resource places. Suppose we have h working processes (WP) and each WP has f choice processes. In the worst case, each of n' resource places is shared by all processes in all WP. The total number of places in the net is around $n = fhn'$ since each state in WP uses exactly one resource place. Thus, $O(2^n) = O((2^{n'})^{fh})$ — a substantial improvement. We only need to search linear number of MBS. The rest can be computed by adding and deleting common sets of places from existing ones with search time significantly reduced.

6. COMPARISON WITH OTHER APPROACHES

Esparza [12] proposed a polynomial time algorithm for MSEP of free-choice nets. Kemper [13] improved it with a linear time algorithm for searching minimal siphons containing a place. However, most FMS such as S^3PR cannot be modeled by free choice nets.

Chue and Xie [14] proposed a linear programming approach that requires the examination of all minimal siphons. Its efficiency depends on the number of minimal siphons. Unfortunately, it is well known that the total number of minimal siphons grows quickly beyond practical limits and that, in worst case, it grows exponentially in the number of nodes. They showed that, under the assumption of structural boundness, it is possible to check deadlock-freeness or empty siphons without generating minimal siphons based on the mixed integer linear programming approach (MIP).

One way to reduce the complexity of the linear programming approach is to find efficient algorithms for generating minimal bad siphons without generating other siphons such as that proposed by Jeng *et al.* [15]. However it has to obtain a maximum siphon first.

Reference [16] employed the sign incidence matrix in [4] to compute the set of siphons containing a given resource place. However the siphons found may also be traps and time complexity is not derived. If a minimal siphon contains a marked trap, then it will never become empty of tokens. Efficient algorithms should extract MBS rather than minimal siphons.

We search only MBS. In addition, many MBS can be derived from existing ones in terms of formulas, thus reducing the search time greatly. For instance, in the example in [1], out of the 18 MBS, we search only 6 and derive the rest 12. Further, these 6 are so simple that we can manually search them.

7. CONCLUSION

We have proposed a new technique to extract MBS for S^3PR. Due to the special structure characteristics of S^3PR, we can first construct basic circuits c_b. Upon each c_b, we can add all TP- and PP-handles to form a basic siphon. From which, we can then derive the rest of minimal bad siphons. All these steps can be expressed in terms of formulas and hence, is easily subject to computer implementation in a very efficient way compared with all current techniques.

Because only one resource is used in each job stage and the processes are modeled using SM in S^3PR, its modeling power is limited. It cannot model iteration statements (loop) in each *sequential process* (*SP*) as in [8] and the relationships of synchronization and communication in SP. At any state of a process, it cannot use multi-sets of resources. If models other than SM are employed, then deadlocks may occur even within a local process. Our SNC (synchronized choice nets) model [9-11] removes these drawbacks. Finding all MBS for SNC- based FMS is a difficult problem. We attack it first for S^3PR.

Future work should extend the technique to SNC-based FMS where each local process is an SNC rather than SM in S^3PR. SNC [9, 11] covers well-behaved (live, bounded and reversible) free choice nets yet it is not included in asymmetric choice nets.

An SNC allows internal choices and concurrency. Therefore it can model not only assembly operations with multiple parts, but also parallel activity and synchronization. Hence, it is more general and powerful than S^3PR. And it [19] covers *extended resource control net merged net* (*ERCN**) [8] as a subset which cannot model cases where an assembly operation is performed on several different parts coming from separate preceding processes.

Finally, it is very interesting that both nets and siphons can be synthesized by constructing handles upon a circuit!

REFERENCES

1. J. Ezpeleta, J. M. Colom, and J. Martinez, "A Petri net based deadlock prevention policy for flexible manufacturing systems," *IEEE Transactions on Robotics and Automation*, Vol. 11, 1995, pp. 173-184.
2. K. Lautenbach, "Linear algebraic calculation of deadlocks and traps," Voss, Genrich, and Rozemberg, eds., *Concurrency and Nets*, Springer Verlag, Berlin, 1987, pp. 315-336.
3. M. Yamauchi and T. Watanabe, "Time complexity analysis of the minimal siphon extraction problem of Petri nets," *IEICE Transactions on Fundamentals*, Vol. E82-A, 1999, pp. 2558-2565.
4. E. R. Boer and T. Murata, "Generating basis siphons and traps of Petri nets using the sign incidence matrix," *IEEE Transactions on Circuits and Systems*, *I – Fundamental Theory and Applications*, Vol. 41, 1994, pp. 266-271.
5. J. Park and S. A. Reveliotis, "Deadlock avoidance in sequential resource allocation systems with multiple resource acquisitions and flexible routings," *IEEE Transactions on Automatic Control*, Vol. 46, 2001, pp. 1572-1583.
6. Y. Huang, "Modeling, analysis, deadlock prevention and cell controller implementation for flexible manufacturing systems," Ph.D. Dissertation, Dept. of Electrical Engineering, National Taiwan University of Science & Technology, Taipei, Taiwan, 2001.
7. K. Barkaoui and I. B. Adallah, "Deadlock avoidance in FMS based on structural theory of Petri nets," in *Proceedings of INRIA/IEEE Symposium on Emerging Technologies and Factories Automation*, Vol. 2, 1995, pp. 499-510.
8. M. Jeng, X. Xie, and S. L. Chung, "ERCN* merged nets for modeling degraded behavior and parallel processes in semiconductor manufacturing systems," *IEEE Transactions on Systems*, *Man and Cybernetics*, *Part A: Systems and Humans*, Vol. 34, 2004, pp. 102-112.
9. D. Y. Chao and J. A. Nicdao, "Liveness for synchronized choice Petri nets," *The Computer Journal*, Vol. 44, 2001, pp. 124-136.
10. D. Y. Chao, "Extended synchronized choice nets," *The Computer Journal*, Vol. 46, 2003, pp. 505-523.
11. D. Y. Chao and D. Wang, "Two theoretical and practical aspects of knitting technique – invariants and a new class of Petri net," *IEEE Transactions on System*, *Man and Cybernetics*, *Part B*, Vol. 27, 1997, pp. 962-977.
12. J. Esparza, "Minimal deadlocks in free choice nets," Hildesheime Informatik-

berichte, Institut fur Informatik, Universitat Hildeslieim, 1989.
13. P. Kemper, "Linear time algorithm to find a minimal deadlock in a strongly connected free-choice net," M. A. Marsan, eds., *Application and Theory of Petri Nets*, LNCS 691, Springer-Verlag, 1993, pp. 319-338.
14. F. Chu and X. L. Xie, "Deadlock analysis of Petri nets using siphons and mathematical programming," *IEEE Transactions on Robotics and Automation*, Vol. 13, 1997, pp. 793-804.
15. M. Jeng, M. Peng, and Y. Huang, "An algorithm for calculating minimal siphons and traps of Petri nets," *International Journal of Intelligent Control and Systems*, Vol. 3, 1999, pp. 263-275.
16. F. Tricas and J. Ezpeleta, "Some results on siphon computation for deadlock prevention in resource allocation systems modeled with Petri nets," in *Proceedings of the 9th IEEE International Conference on Emerging Technologies and Factory Automation*, 2003, pp. 322-329.
17. Z. Li and M. Zhou "Elementary siphons of Petri nets and their application to deadlock prevention in flexible manufacturing systems," *IEEE Transactions on System, Man, and Cybernetics, Part A: Systems and Humans*, Vol. 34, 2004, pp. 38-51.
18. D. B. Johnson, "Finding all the elementary circuits of a directed graph," *SIAM Journal on Computing*, Vol. 4, 1975, pp. 77-84.
19. D. Y. Chao, "Reachability of non-synchronized choice Petri nets and its applications," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, Vol. 35, 2005, pp. 1013-1023.
20. D. Y. Chao, "Computation of elementary siphons in Petri nets for deadlock control," *The Computer Journal*, Vol. 49, 2006, pp. 470-479.

APPENDIX 1. PETRI NET-RELATED DEFINITIONS

A Petri net (or Place/Transition net) is a 3–tuple $N = (P, T, F)$, where $P = \{p_1, p_2, \ldots, p_a\}$ is a set of places, $T = \{t_1, t_2, \ldots, t_b\}$ a set of transitions, with $P \cup T \neq \varnothing$ and $P \cap T = \varnothing$ and F a mapping from $(P \times T) \cup (T \times P)$ to nonnegative integers indicating the weight of directed arcs between places and transitions. M_0: $P \to \{0, 1, 2, \ldots\}$ denotes an initial marking whose ith component, $m_0(p_i)$, represents the number of tokens in place p_i. N is *strongly connected iff* there is a directed path from any node to any other node.

A node x in $N = (P, T, F)$ is either a $p \in P$ or a $t \in T$. The post-set of node x is $x\bullet = \{y \in P \cup T \mid F(x, y) > 0\}$, and its pre-set $\bullet x = \{y \in P \cup T \mid F(y, x) > 0\}$.

t_i is *firable* if each place p_j in $\bullet t_i$ holds no less tokens than the weight $w_j = F(p_j, t_i)$. Firing t_i under M_0 removes w_j tokens from p_j and deposits $w_k = F(t_i, p_k)$ tokens into each place p_k in $t_i\bullet$; moving the system state from M_0 to M_1. Repeating this process, it reaches M' by firing a sequence of transitions. M' is said to be reachable from M_0; *i.e.*, $M_0[\sigma > M'$.

Ordinary Petri Nets (*OPN*) are those for which F: $(P \times T) \cup (T \times P) \to \{0, 1\}$. An OPN is called a *State Machine* (*SM*) if $\forall t \in T$, $|t\bullet| = |\bullet t| = 1$. It is a *Free Choice net* (*FC*) if $\forall p_1, p_2 \in P$, $p_1\bullet \cap p_2\bullet \neq \phi \Rightarrow |p_1\bullet| = |p_2\bullet| = 1$. It is an *Asymmetric Choice net* (*AC*) if $\forall p_1\bullet \cap p_2\bullet \neq \phi \Rightarrow p_1\bullet \subseteq p_2\bullet$ or $p_1\bullet \supseteq p_2\bullet$.

$R(M_0)$ is the set of markings reachable from M_0. A transition $t \in T$ is live under M_0

iff $\forall M \in R(M_0)$, $\exists M' \in R(M)$, t is firable under M'. A transition $t \in T$ is dead under M_0 *iff* $\nexists M \in R(M_0)$ where t is firable. A PN is *live* under M_0 *iff* $\forall t \in T$, t is live under M_0. It is *bounded* if $\forall M \in R(M_0)$, $\forall p \in P$, the marking at p, $m(p) \leq k$, where k is a positive integer.

Let $\Gamma = [n_1\ n_2 \ldots n_k]$, $k \geq 1$, denote a graphical object containing a sequence of nodes and the single arc between each two successive nodes in the sequence. Γ is called an *elementary directed path* in N if $\forall(i, j)$, $1 \leq i < j \leq k$, $n_i \neq n_j$. Γ is called an *elementary circuit c* in N if $\forall(i, j)$, $1 \leq i \leq j \leq k$, $n_i = n_j$ implies that $i = 1$ and $j = k$.

For a Petri net (N, M_0), a non-empty subset $D(\tau)$ of places is called a *siphon* (*trap*) if $\bullet D \subseteq D\bullet$ ($\tau\bullet \subseteq \bullet\tau$), *i.e.*, every transition having an output (input) place in $D(\tau)$ has an input (output) place in $D(\tau)$. If $M_0(D) = \sum_{p \in D} m_0(p) = 0$, D is called a *empty siphon* at M_0. A *minimal siphon* does not contain a siphon as a proper subset. D is called a *bad siphon* (*BS*) if it does not contain a trap.

APPENDIX 2. PROOFS

Proof of Theorem 1: If the $D_m{}^n$ found in step 3 is minimal, step 4 will compute the rest of new D_m. Hence steps 3-4 compute all minimal bad siphons containing the new resource place r. Step 5 guarantees the same computation be repeated for all resource places. Hence it computes all minimal bad siphons. ❑

Daniel Yuh Chao (趙玉) received the Ph.D. degree from in Electrical Engineering and Computer Science from the University of California, Berkeley in 1987. From 1987-1988, he worked at Bell Laboratories. Since 1988, he joined the Computer and Information Science Department of New Jersey Institute. Since 1994, he joined the Management and Information Science Department of National Chengchi University as an Associate Professor. Since February, 1997, he has been promoted to a full professor. His research interest was in the application of Petri nets to the design and synthesis of communication protocols and the CAD implementation of a multi-function Petri net graphic tool. He has published 102 (including 32 journals) papers in the area of communication protocols, Petri nets, DQDB, networks, FMS, data flow graphs, and neural networks.