

Reachability and Firing Sequences of Homogeneous Synchronized Choice Petri Nets*

DANIEL Y. CHAO

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

A new local structure called the Second Order Structure (SOS) has been proposed to generate a new class of nets called Synchronized Choice Nets (SNC). SNC covers well-behaved free-choice nets. The reachability problem for Homogeneous SNC (HSNC, a subclass of SNC) is quite simple because reachable markings are linearly additive and can be translated into a structural problem based on the S-Matrix. An algorithm for constructing the S-Matrix has been developed to record the structural relationships among places. Hence, it is no longer a P-Space hard problem, and an algorithm with polynomial time complexity has been developed. Also presented is an algorithm for deriving the shortest firing sequence from one reachable marking to another. How the algorithm can be extended to Inhomogeneous and non-SNC is discussed.

Keywords: petri nets, synchronized choice nets, reachability, liveness, synthesis, verification, inconsistent pair

1. INTRODUCTION

Petri Net (PN) theory has been applied to the modeling of distributed systems. Reachability analysis explores all possible states of a PN to discover problems such as deadlocks. Proving liveness or, equivalently, solving the reachability problem for general Petri nets (PNs) is more difficult than proving boundedness and takes exponential time and space [9], but it is decidable.

The reachability problem is important because many control problems in discrete event systems modelled by Petri nets, such as flexible manufacturing and communication systems [10], can be modeled by the reachability problem in the net model.

The reachability problem can be solved by checking for the existence of a nonnegative integer solution of the matrix equation of the net. Even if one exists, however, there is still the problem of finding an appropriate firing sequence, which also takes exponential time and space. Lee *et al.* [8] solved this problem for NOP and NOT nets, which are subsets of Free-Choice Nets (FC). An algorithm [6] to determine the firing sequence by tracing backwards from the final marking to the initial one on the places set was presented. The proposed method has been applied to the failure diagnosis of a simple type of sequence control system.

Received April 24, 2003; revised August 14, 2003 & March 8, 2004; accepted June 1, 2004.

Communicated by Hsu-Chun Yen.

* This work was supported by the National Science Council under research grant number NSC 90-2213-E-004-001.

Petri nets can be used for the design of complex control systems via reachability analysis [8]. Solving the reachability problem is equivalent to solving a set of linear equations: $M = M_0 + A\sigma$, where A is the incidence matrix, σ is the firing vector, M_0 is the initial marking or state, and M is the new marking or state after firing each transition x_i times. The solution σ of the above state equation, however, may not be a legal firing vector except for special classes of nets, such as LSFC (live and safe free choice) nets and RLBFC (reversible live and bounded free choice) nets.

We [4] have proposed a new class of nets called Synchronized Choice Nets (SNC, see Figs. 1-4). SNC covers well-behaved and various classes of FC (free-choice) and is not included in AC (asymmetric choice). An SNC allows internal choices and concurrency and, hence, is very useful for modeling. Any SNC is bounded, and its liveness conditions are simple. An integrated algorithm has been presented for verification of a net being SNC and its liveness. SNC (N^c) is interesting for the following reasons:

- (1) It covers Structure Live and Structure Bounded (**SL&SB**) FC, where the latter is a subclass of **SL&SB** N^c .
- (2) It derives its properties from local structures.
- (3) **SL&SB** can be decided based on simple local structures; hence, it has polynomial time complexity rather than P-SPACE.

Barkaoui [1] introduced a new class of nets (see Fig. 5) that also does not cover all SNCs. Further, the structures of minimal deadlocks are not correlated to handles and bridges; hence, it is not clear or intuitive what they look like in structures. Although they have polynomial algorithms for deciding liveness for only subclasses (elementary and loop-free) of ENSeC, there are no efficient algorithms as stated in [1] to verify that a net is ENSeC. Their algorithms only work for bounded nets in the new classes.

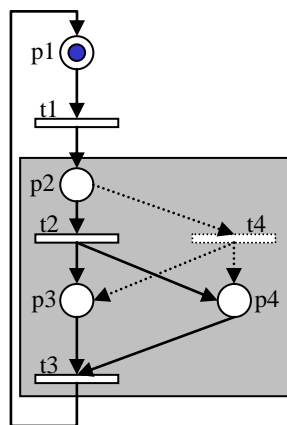


Fig. 1. An example of live and reversible SNC with no inconsistent pair.

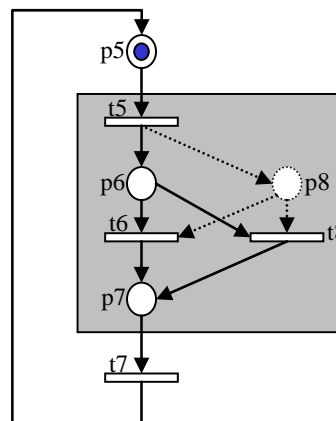


Fig. 2. Dual of the net in Fig. 1. This net is live and reversible without inconsistent.

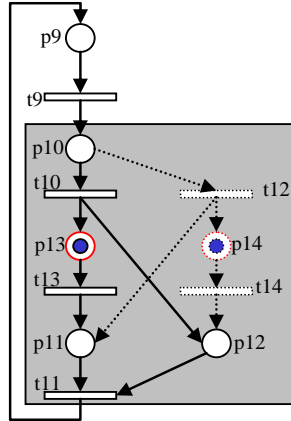


Fig. 3. (a) Irreversible SNC with PT-inconsistent pair (p13, p14).

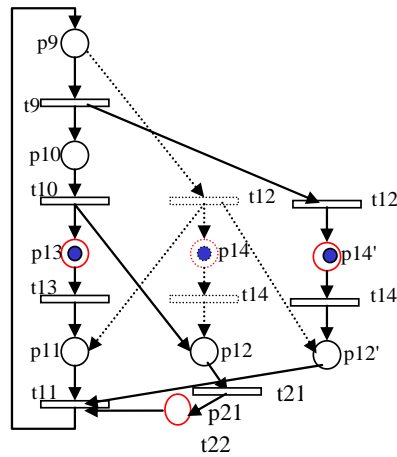


Fig. 3. (b) $H = [t10\ p12]$, $N_1 = \mathcal{N}H$; (Rule 2) before: $\neg(p10 \leftrightarrow p21)$; after: $p10 \leftrightarrow p21$. (Rule 3) before: $n_s^{13,21} = p9$; after: $n_s^{13,21} = t10$. (Rule 4) before: $n_s^{14',21} = p9$; after: $n_s^{14',21} = t9$.

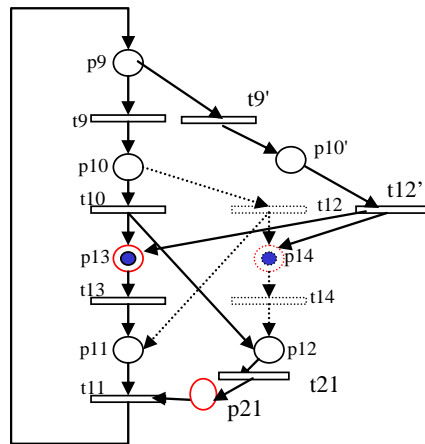


Fig. 3. (c) No longer irreversible. One $n_s^{13,21} = t12'$; (p13, p14) no longer PT-inconsistent.

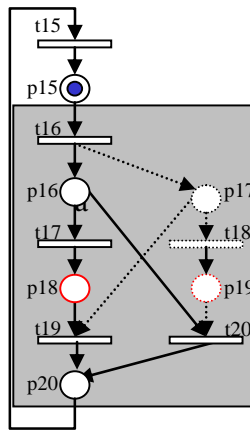


Fig. 4. Dual of the net in Fig. 3 (a). The SNC is not live with TP-inconsistent pair (p18, p19).

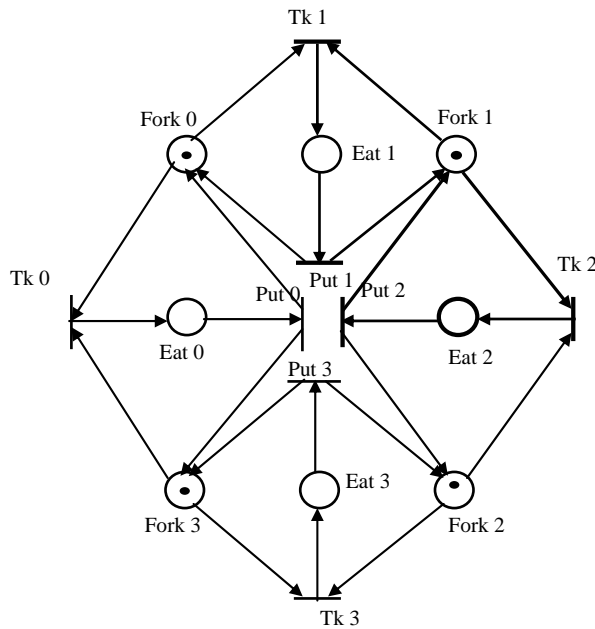


Fig. 5 [1]. The PN model of dining philosophers is not an SNC due to the AFOS with two handles: [Put1 Fork1 Tk2 Eat2 Put2 Fork2] and [Put1 Fork0 Tk0 Eat0 Put0 Fork3 Tk3 Eat3 Put3 Fork2] with no bridges across them violating R2.

Unlike other classes of nets, this new one extracts most of the nearly well-behaved nets from among all the possible nets and puts them into the SNC. As long as a net is in the new class, it is bounded (there is no need for verification). Designing a net in this class will result in fewer errors than arbitrary nets and hence is more reliable. This further simplifies the analysis. More important, it helps to simplify and enhance our synthesis rules for the Knitting technique [5].

This paper will show that the reachability problem for Homogeneous SNC (HSNC, a subclass of SNC) is quite simple because reachable markings are linearly additive and can be translated into a structural problem based on the S-Matrix. An algorithm for constructing the S-Matrix has been developed to record the structural relationships among places. Hence, it is no longer a P-Space hard problem and an algorithm with polynomial time complexity has been developed. We will also present an algorithm for deriving the shortest firing sequence from one reachable marking to another.

Section 2 presents necessary back ground for the paper. Section 3 presents the S-Matrix construction algorithm. Reachability analysis is simplified in section 4, where we also present our algorithm for the reachability problem. Section 5 presents the algorithm for the shortest legal firing sequences. Section 6 concludes the paper and discusses how the algorithm can be extended to Inhomogeneous SNC and non-SNC.

2. PRELIMINARIES

We follow the PN terminology used in [5]

Definition 1 Let $P = \{p_1, p_2, \dots, p_a\}$ be a set of places, and let $T = \{t_1, t_2, \dots, t_b\}$ be a set of transitions, with $P \cup T \neq \emptyset$ and $P \cap T = \emptyset$. F is $(P \times T) \cup (T \times P) \rightarrow \{0, 1, 2, \dots\}$; $M_0: P \rightarrow \{0, 1, 2, \dots\}$, then, $N = (P, T, F)$ is a net. M_0 denotes an initial marking whose i th component, $M_0(p_i)$, represents the number of tokens in place p_i . 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 . When this process is repeated, it reaches M' by firing a sequence of transitions. M' is said to be reachable from M_0 ; i.e., $M_0[\sigma > M'$.

Definition 2 Let $N = (P, T, F)$ be a net, let (N, M_0) be a marked net, and let $R(M_0)$ be the set of markings reachable from M_0 . A transition $t \in T$ is live under M_0 iff $\forall M \in R(M_0), \exists M' \in R(M)$, and 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 net PN is live under M_0 iff $\forall t \in T$ and t is live under M_0 . It is weakly live under M_0 if $\exists t \in T$, t is live under M_0 . It is bounded if $\forall M \in R(M_0), \forall p \in P$, and $\exists a$ positive integer, k , the marking at p , $m(p) \leq k$. A net N is structurally live (**SL**) iff $\exists M_0$ such that PN is live under M_0 . N is structurally bounded (**SB**) iff it is bounded for any finite initial marking M_0 . N is reversible iff $\forall M \in R(M_0), M_0$ is reachable from M ; i.e., $M_0 \in R(M)$. N is structurally reversible iff $\forall M_0, N$ is reversible.

Definition 3 A node x in $N = (P, T, F)$ is either a $p \in P$ or a $t \in T$. An elementary directed path Γ in N is a graphical object containing a sequence of nodes and the single arc between each two successive nodes in the sequence with the notation: $\Gamma = [n_1 n_2 \dots n_k]$, $k \geq 1$, such that $n_i \neq n_j$ for $i \neq j$. A path is (non, resp.) **virtual** if it contains only (more than, resp.) two nodes. An elementary cycle in N is $\Gamma = [n_1 n_2 \dots n_k]$, $k > 1$ such that $n_i = n_j$, $1 \leq i \leq j \leq k$, which implies that $i = 1$ and $j = k$.

In this paper, we consider only strongly connected nets in which there exist directed paths between any pair of nodes.

Definition 4 A nonnegative, integer Y (X , resp.) vector is called an S - (T -, resp.) invariant iff Y (X , resp.) $\neq 0$ and $AY = 0$ ($A^T X = 0$, resp.), where A is the incidence matrix of a net with m places and n transitions: $A = [a_{ij}]$; an $n \times m$ matrix of integers and its typical entry is given by:

$$a_{ij} = a_{ij}^+ - a_{ij}^-$$

where $a_{ij}^- = F(i, j)$ is the weight of the arc from transition i to its output place j , and $a_{ij}^+ = F(j, i)$ is the weight of the arc to transition i from its input place j . The set of places p such that the component in Y , $y(p) > 0$ for all is called the support of the S -invariant and is defined as $\|y\|$.

Definition 5 For a Petri net (N, M_0) , a non-empty subset D of places is called a siphon (trap, resp.) if $\bullet D \subseteq D \bullet$ ($D \bullet \subseteq \bullet D$); i.e., every transition having an output (input, resp.) place in D has an input (output, resp.) place in D . If $M_0(D) = \sum_{p \in D} m_0(p) = 0$, then D is called a token-free siphon at M_0 . A minimal siphon D_m does not contain a siphon as a proper subset. D_m is called a *bad siphon* if it does not contain a trap.

$\{p15, p16, p17, p20\}$ in Fig. 4 is a siphon, and $\{p9, p10, p11, p12\}$ in Fig. 3 is a trap. Both siphons and traps are a set of places. Generally, once a token leaves a siphon (enters a trap), it will never return to the siphon (leak out of the trap). If tokens continue to leave and, eventually, the siphon is empty of tokens, then no input transition of a place in the siphon can be fired again. Hence, the net is dead.

We follow the definitions of handles, bridges, AB-handles, and AB-bridges in [5] 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 and ends with a transition, while a TP-handle starts with a transition and ends with a place. A TP-bridge is a path from a transition in the handle to a place in the circuit, while a PT-bridge is a path from a place in the handle to a transition in the circuit. These bridges help ‘synchronize’ the workings of the bridge and the handle.

An FOS consists of two handles (with no bridges in between) with the same end nodes. A SOS consists of an FOS plus the two bridges between the two handles (with exactly one bridge from one handle to the other).

Definition 6 Let $N = (P, T, F)$. $H_1 = [n_s n_1 n_2 \dots n_k n_e]$ and $H_2 = [n_s n'_1 n'_2 \dots n'_h n_e]$ are elementary directed paths, $n_i, n'_j \in P \cup T$, $i = 1, 2, \dots, k, j = 1, 2, \dots, h$. Each is called a handle in N if $H_1 \cap H_2 = \{n_s, n_e\}$; n_s and n_e are called the start and the end nodes of H_1 and H_2 , respectively. Note that n_s and n_e may be identical. H_1 and H_2 are said to be *mutually complementary*. An elementary directed path $B = [n_a, \dots, n_q]$ is a bridge from H_1 to H_2 iff $B \cap H_1 = \{n_a\}$, $B \cap H_2 = \{n_q\}$, and neither n_a nor n_q are the start (n_s) or end (n_e) nodes of H_1 and H_2 . **$p1 \leftrightarrow p2$ ($p1$ and $p2$ are mutually sequential)** if $p1$ and $p2$ are

on an elementary circuit. $n_1 \rightarrow n_2$ if $n_1 \leftrightarrow n_2$ and there is an elementary directed path from n_h to n_2 via n_1 , where n_h is a reference node (initially marked), called a **home place**.

In Fig. 1, $H_1 = [p_2 t_4 p_4 t_3]$ and $H_2 = [p_2 t_2 p_3 t_3]$ $n_s = p_2$, $n_e = t_3$. $B_{12} = [t_4 p_3]$ is a bridge from H_1 to H_2 and $B_{21} = [t_2 p_4]$ a bridge from H_2 to H_1 .

A handle is an elementary directed path with only its terminal (the start and the end) nodes attached (i.e., common) to an associated subnet. A bridge is an elementary directed path with only its start node attached to one handle and the end node to another handle.

Definition 7 H_1 and H_2 are defined above in Definition 6.

- (1) Let $\Gamma_1 = [n_s n_1 n_2 \dots n_k p_1]$ and $\Gamma_2 = [n_s n'_1 n'_2 \dots n'_h p_2]$. n_s is a *start node* of (p_1, p_2) if $\Gamma_1 \cap \Gamma_2 = \{n_s\}$. Γ_1 is called a *no-middle- n_s -path* if it does not contain other start nodes of (p_1, p_2) ; it is also called a *α -path* if $\exists \theta = [n_s \dots p_1]$ contains other start nodes. n_s is the *nearest start node* of (p_1, p_2) ; i.e., $n_s^{1,2} = n_s$, if Γ_1 is a *no-middle- n_s -path*. $n_e^{1,2}$ can be defined in a dual fashion. Let $\Gamma_3 = [p_1 n_1 n_2 \dots n_k n_e]$ and $\Gamma_4 = [p_2 n'_1 n'_2 \dots n'_h n_e]$. n_e is an *end node* of (p_1, p_2) if $\Gamma_3 \cap \Gamma_4 = \{n_e\}$. Γ_3 is called a *no-middle- n_e -path* if it does not contain the other end node of (p_1, p_2) ; it is also called a *α -path* if $\exists \eta = [p_1 \dots n_e]$ contains the other end nodes. n_e is the *nearest end node* of (p_1, p_2) ; i.e., $n_e^{1,2} = n_e$ if Γ_3 is a *no-middle- n_e -path*. $N_s^{1,2}$ ($N_e^{1,2}$, resp.) is the set of all such $n_s^{1,2}$ ($n_e^{1,2}$, resp.).
- (2) Let $Y = H_1 \cup H_2$. H_1 (H_2 , resp.) is a *prime handle* to H_2 (H_1 , resp.) if there are no bridges B between H_1 and H_2 , and if Y is defined as a **first-order** structure (FOS).
- (3) If B_{12} (B_{21} , resp.) is the only bridge from H_1 to H_2 (H_2 to H_1 , resp.), then $\varphi = H_1 \cup H_2 \cup B_{12} \cup B_{21}$ is defined as a **second-order** structure (SOS) (see the shaded areas in Figs. 1-4).
- (4) (p_1, p_2) is called a TP-inconsistent pair (TPIP) of places if it is not in any FOS, and if $\exists n_s^{1,2} \in T$ and $\exists n_e^{1,2} \in P$. (p_1, p_2) is called a PT-inconsistent pair (PTIP) of places if it is not in any FOS, and if $\exists n_s^{1,2} \in P$ and $\exists n_e^{1,2} \in T$.
- (5) A strongly connected net is an SNC (Synchronized Choice net) if it satisfies the two requirements **R1** and **R2**, where **R1**: (**R2**:, resp.) every TP- (PT-, resp.) handle to a certain circuit has a PT- (TP-, resp.) bridge from its complementary TP-handle to itself.
- (6) Let ω be an FOS (or SOS, handle, bridge, path). If its $n_s \in T$ and $n_e \in P$, then ω is called a TP- ω . PT- ω , TT- ω , and PP- ω can be defined similarly. If n_s and n_e are of the same type, i.e., both are transitions or places, then ω is said to be symmetrical; otherwise, it is asymmetrical.
- (7) An Inhomogeneous SNC (ISNC) is an SNC where the N_s (or N_e) of a certain pair of places contains both transitions and places; otherwise, it is called a Homogeneous SNC (HSNC).

Note that for ISNC (see Fig. 6), there may be multiple $n_s^{1,2}$ and/or $n_e^{1,2}$. In Fig. 7 (c), $N_e^{6,8} = \{t_6, p_7\}$; p_7 is a nearest end node $n_e^{6,8}$ because there are no other end nodes n_e on $\Gamma = [p_8 t_8 p_7]$. It is also an *α -path* since the path $[p_8 t_6 p_7]$ contains another $n_e^{6,8}$. Reachability for ISNC is more complicated than that for HSNC and will be dealt with in another paper.

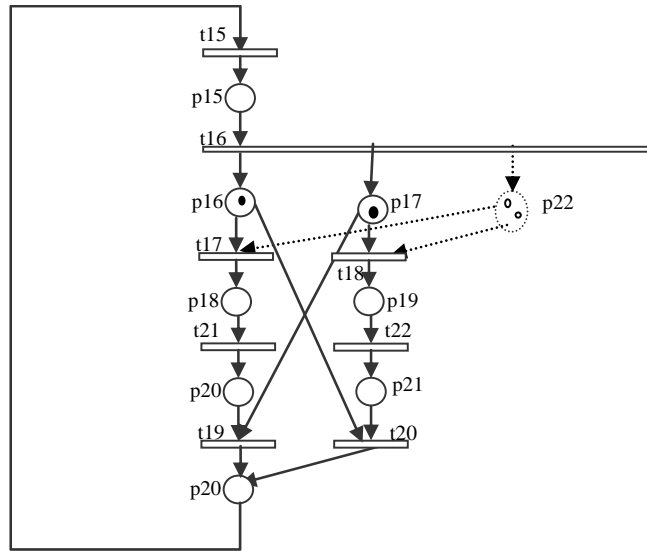
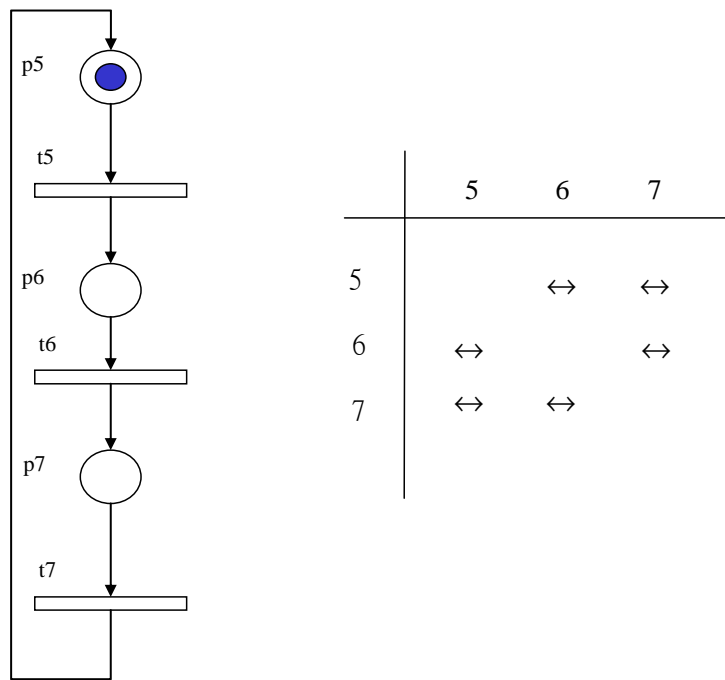
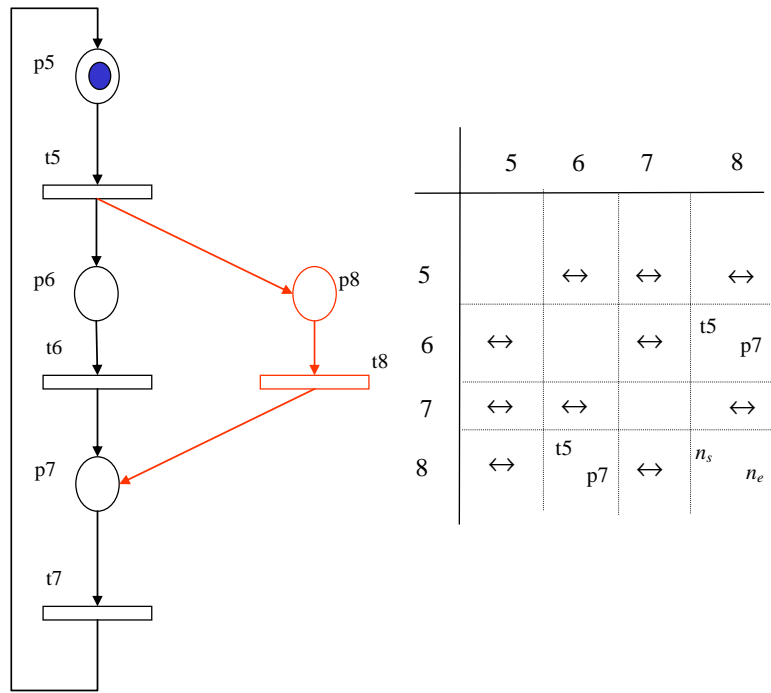


Fig. 6. The SNC is not marking monotonic and not live. But it is live if $M(p_{22}) = 1$.

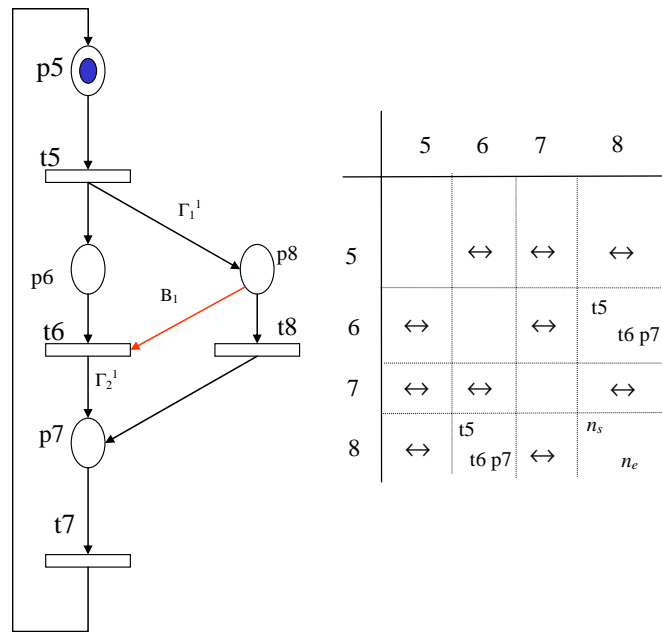


(a) A circuit and its S-Matrix.

Fig. 7. An example of constructing S-Matrix for N in Fig. 2.

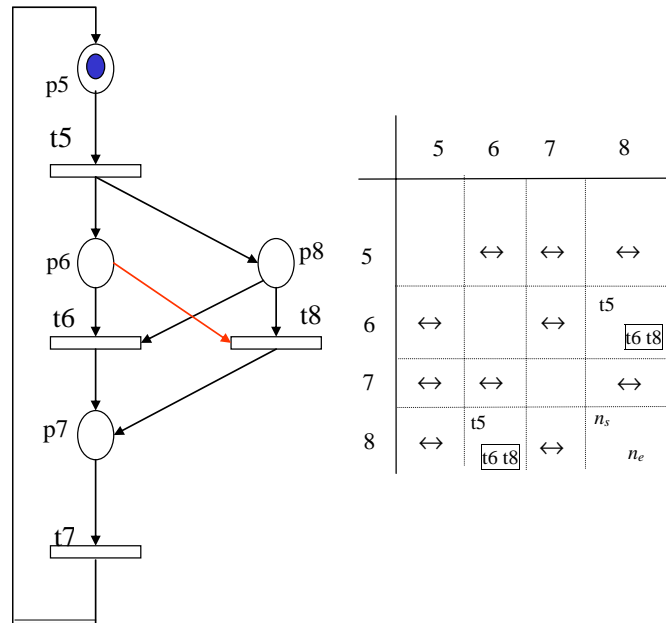


(b) Adding a new handle [t5 p8 t8 p7], and its updated S-Matrix.



(c) Adding a new handle [p8 t6], and its updated S-Matrix.

Fig. 7. (Cont'd) An example of constructing S-Matrix for N in Fig. 2.

(d) Adding another new handle $[p6 t8]$, and the final updated S-Matrix.Fig. 7. (Cont'd) An example of constructing S-Matrix for N in Fig. 2.

p_1 and p_2 in Definition 5 are inconsistent because they are concurrent (exclusive, resp.) and the tokens in them will flow to a set of mutually exclusive (concurrent, resp.) places. In Fig. 3 (a), $N_s^{12,11} = \{t10, t12\}$ and $N_e^{12,11} = \{t11\}$; $p10$ is not an $n_s^{12,11}$ because for each path from $p10$ to $p11$ or $p12$, it contains another start node, $t10$ or $t12$. ($p18, p19$) in Fig. 4 is a TP-inconsistent pair (TPIP) because $n_s^{18,19} = t16$ and $n_e^{18,19} = p20$. Note that $n_s^{1,2}$ and $n_e^{1,2}$ do not exist if $p1 \leftrightarrow p2$.

Note that the net in Fig. 3 (b) is not an Asymmetric Choice net (AC), and that for TPIP, as long as there exists a transition $n_s^{1,2}$, it is not live.

Figs. 1-4 show examples of SNCs, where the shaded areas cover the structures involving **R1** or **R2**. Note the net in Fig. 4 is neither an FC nor an EFC (Extended Free-Choice). In Fig. 1, the only two PT-handles, $H_1 = [p_2 t_4 p_4 t_3]$ and $H_2 = [p_2 t_2 p_3 t_3]$, start from the same place, p_2 , but join at a transition, t_3 . To satisfy **R1**, there is a TP-bridge $B_{12} = [t_4 p_3]$ from H_1 to H_2 and a TP-bridge $B_{21} = [t_2 p_4]$ from H_2 to H_1 . In Fig. 2, the only two TP-handles, $H_1 = [t_5 p_6 t_6 p_7]$ and $H_2 = [t_5 p_8 t_8 p_7]$, start from the same transition t_5 but join at a place p_7 . To satisfy **R2**, there is a PT-bridge $B_{12} = [p_6 t_8]$ from H_1 to H_2 and a PT-bridge $B_{21} = [p_8 t_6]$ from H_2 to H_1 .

In the dining philosopher model shown in Fig. 5, the FOS with two handles, [Put1 Fork1 Tk2 Eat2 Put2 Fork2] and [Put1 Fork0 Tk0 Eat0 Put0 Fork3 Tk3 Eat3 Put3 Fork2], have no bridges across them violating **R2**. Hence, it is not an SNC.

$[p_2 t_2 p_3]$ and $[p_2 t_4 p_3]$ in Fig. 1 are two prime handles; $n_s = p_2$ and $n_e = p_3$. Note that there are no bridges interconnecting them; hence, they together form an FOS. Since $n_s \in P$ and $n_e \in P$, it is *symmetrical*.

Note that any pair of places, p_1 and p_2 (excluding n_s and n_e) in an AFOS (asymmetrical FOS) is also inconsistent in the sense that its $n_s^{1,2}$ and $n_e^{1,2}$ are not of the same type. This leads [5] to an integrated algorithm to construct an S-Matrix (presented in the next section) to detect SNC and liveness for an arbitrary net.

3. CONSTRUCTION OF THE S-MATRIX

The S-Matrix records the n_s and n_e for all pairs of places. Thus, the S-Matrix helps us to find the structural (i.e., sequential, concurrent, and exclusive) relationship and is essential for conducting reachability analysis of SNC as described in the next section. Reachability for HSNC is quite simple because reachable markings are linearly additive and can be translated into a structural problem. The S-Matrix helps us to check whether a net is an HSNC. An algorithm for constructing the S-Matrix is presented below.

Definition 8 The S-Matrix is an $n \times n$ matrix, where $n = |P|$ is the total number of places. Entry $S_{ij} = \leftarrow \rightarrow$ if p_i and p_j are sequential; otherwise, $S_{ij} = \{N_s^{ij}, N_e^{ij}\}$.

Each entry is a pointer to a record holding information about the n_s and n_e of a certain pair composed of p_1 and p_2 . To see whether (p_1, p_2) is inconsistent, we examine the corresponding entry in the S-Matrix. If different types (places or transitions) of n_s and n_e are present in the entry, then (p_1, p_2) is inconsistent.

Initially, all records are NULL pointers. Instead of searching all the places and finding their n_s and n_e in a global fashion, we take an incremental approach. First, we find an elementary circle. All entries in the S-Matrix are " $\leftarrow \rightarrow$ ". Next, we find a handle to the circle and update the S-Matrix. Any pair of places on the pair of mutually complementary handles, respectively, has the same n_s (n_e). We continue with the process until all the places in the net have been included in the S-Matrix. Generally, the new path to be added is a handle to the subnet whose places have been included in the S-Matrix.

The above process of adding new paths is similar to the synthesis process in the knitting technique [5]. Synthesis-directed analysis offers another new structural approach. Synthesis [5] expands a net N^1 (the solid lines in Figs. 1-4), following a set of rules such that after each expansion (the dashed lines), the resulting net N^2 remains well-behaved (both live and bounded). For instance, in the knitting technique proposed by [5], a larger net can be constructed from a simple circuit by continuously adding a set of new paths of handles and bridges at each synthesis step according to TT and PP rules. That is, each new path involved in a synthesis step must be a TT- (from transition to transition) or a PP- (from place to place) path.

Depending on the structural relationship (concurrent, exclusive, or sequential) between the two end nodes of the new path, an appropriate rule should be applied, for example to forbid new, or to add more, path generations. The net will be complete when all new path generations are permitted while the net remains well-behaved. The rules involve nodes in a local fashion. They provide clues about structural constraints necessary for a net to be well-behaved and, thus, offer a new avenue for net analysis.

Examining the synthesis rules presented in [5], we find that synthesized nets and SNC nets are closely related. This is because both involve handles and bridges; the for-

mer are constructed by adding new handles and bridges, while the latter are based on local structures of handles and bridges.

While it is easy to find the structural relationship between places on the handle and places on the same cycle with n_s or n_e , it is not easy to do so for *places* far away from (having no direct connection with) the handle.

Algorithm I for constructing S-Matrix

Let $N_1 = \emptyset$, $N_2 = N$.

- A. Find an elementary circuit c in N . Execute Step C.2 and go to Step B.
- B. Find a handle H of N_1 with n_s and n_e .
- C. Enter new entries or update old entries per the following rules. n' denotes new n_s' or n_e' for a certain pair of nodes n_c and n_d .
 1. If $n_c \leftrightarrow n_d$, then never update the entry.
 2. If n_c and n_d are on an elementary circuit containing H , then $n_c \leftrightarrow n_d$.
 3. $\forall n_s \rightarrow n_c \rightarrow n_e^{s,e}, \forall \Gamma = n_s \rightarrow n_d \rightarrow n_e^{s,e}, H \subseteq \Gamma, n_s^{rc,d} = n_s, n_e^{rc,d} = n_e^{s,e}$.
 $3^r: \forall n_s^{s,e} \rightarrow n_c \rightarrow n_e, \forall \Gamma = n_e^{s,e} \rightarrow n_d \rightarrow n_e, H \subseteq \Gamma, n_s^{rc,d} = n_s^{s,e}, n_e^{rc,d} = n_e$.
 4. $\forall n_c \in N_1, \forall n_d \in H \rightarrow n_f$, if $n_s^{c,d} \rightarrow n_s^{s,c} \rightarrow n_c, n_s^{rc,d} = n_s^{s,c}$.
 $4^r: \forall n_c \in N_1, \forall n_d \in n_f \rightarrow H$, if $n_c \rightarrow n_e^{e,c} \rightarrow n_e^{c,d}, n_e^{rc,d} = n_e^{e,c}$.
- D. $N_1 = N_1 \cup H, N_2 = N \setminus H$. Repeat Step B until $N_2 = \emptyset$.
- E. check_deleteS($n_s^{c,d}$), check_deletE($n_e^{c,d}$).

We need to determine new entries for places on the handle and update old entries to either “ \leftrightarrow ” (Rule 2) or change to or add new n_s or n_e (Rules 3 & 4). There are two cases, namely (1) $n_s \leftrightarrow n_e$ and (2) $\neg(n_s \leftrightarrow n_e)$, and only (2) involves updates of old entries. There are three possibilities for new or updated entry cd : (a) “ \leftrightarrow ” (Rule 2, $p5 \leftrightarrow p8$ in Fig. 7 (b)), (b) new $n_s' = n_s$ ($n_e' = n_e$) (Rule 3, $n_s'^{6,8} = t5 = n_s$ in Fig. 7 (b)), and (c) new $n_s' = n_s^{s,c}$ ($n_e' = n_e^{e,c}$) (Rule 4, Fig. 3 (b)). Note that in Fig. 7 (b) where $n_s \leftrightarrow n_e$, the entries for pairs of places that are both in N_1 need no updates. In Fig. 7 (c), where $\neg(n_s \leftrightarrow n_e)$, $n_e^{6,8}$ is updated to obtain a new $n_e = t6$.

Each Rule i^r is formed from Rule i by applying Rule i to N^r , the reverse of N where all arcs in N are reversed. Hence, interchanging n_e with n_s and reversing the “ \rightarrow ” to obtain “ \leftarrow ” leads to the reverse rule, i^r . Each time H is obtained, new entries must be entered, and some old entries must be updated to either “ \leftrightarrow ” (Rule 2) or add or change to new n_s or n_e (Rules 3 & 4). Rule 2 finds all elementary circuits containing H and for any two nodes on such a circuit, the entry is “ \leftrightarrow ”. Rule 3 (3^r) fills entries with a new $n_s' = n_s$ ($n_e' = n_e$). Rule 4 (4^r) fills entries with a new $n_s' = n_s^{s,c}$ ($n_e' = n_e^{e,c}$), but only for nodes n_c that have not been treated in Rules 2 & 3.

Example: In Fig. 3 (b), let $H = [t10 \ p12]$, $N_1 = N \setminus H$; (Rule 2) before: $\neg(p10 \leftrightarrow p21)$; after: $p10 \leftrightarrow p21$. (Rule 3) before: $n_s^{13,21} = p9$; after: add new $n_s'^{13,21} = t10$. (Rule 4) before: $n_s^{14',21} = p9$; after: add new $n_s'^{14',21} = t9$. Fig. 7 shows an example of constructing the S-Matrix for the net shown in Fig. 2.

Note that check_deleteS($n_s^{c,d}$) (check_deletE($n_e^{c,d}$)) is performed in the last step,

and that it checks for the presence of the α -path and deletes $n_s^{c,d}$ ($n_e^{c,d}$) if it is absent based on Lemma 1.

Lemma 1 Let $B = \{n_q\}$, $n_q \in N_e^{1,2}$ (or $N_s^{1,2}$), A is the maximum set satisfying (1) $A \cup B \subseteq N_e^{1,2}$ (or $N_s^{1,2}$) and (2) $\forall n_a \in A, n_a \leftrightarrow n_q, \forall \text{pair } (n_1, n_2)$; there exists a α -path if $\exists \text{pair } (A, B), \exists n_r, n_i \rightarrow n_r \rightarrow n_q (n_q \rightarrow n_r \rightarrow n_i), \neg(n_a \leftrightarrow n_r)$, and $i = 1$ or 2 .

Proof: The path $n_i \rightarrow n_r \rightarrow n_q$ is a no-middle- n_e -path (no-middle- n_s -path). It is also a α -path due to the presence of the path $n_i \rightarrow n_a \rightarrow n_q$, where both n_a and n_q are $n_e^{1,2}$ by the definition for the α -path in Definition 7.1. \square

In Fig. 7, $(n_1, n_2) = (p6, p8)$, $B = \{p7\}$, $A = \{t6, t8\}$, and $N_e^{6,8} = A \cup B$; there does not exist a α -path. Hence, $p7$ is deleted from $N_e^{6,8}$ in the last step (E) of the algorithm.

The time complexity of finding a c and a handle is $O(|T| + |P|)$. For each handle found in Step B, it takes $O(|P|^2)$ time to check and update entries in Step C. Thus each iteration of Steps B-D takes $O(|T| + |P|) + O(|P|^2) = O(|P|^2 + |T|)$ time complexity. There are $O(|F|) = O(|P| * |T|)$ handles. Hence, the total time complexity for all iterations is $O(|P|^2 + |T|) * O(|F|) = O((|P|^2 + |T|) * |P| * |T|)$. For check_deleteS($n_s^{c,d}$) or check_deleteE($n_e^{c,d}$), we have $O(|T| + |P|)$ n_q and n_a , and $O(|P|)$ n_r in Lemma 1. Checking $n_i \rightarrow n_r \rightarrow n_q$ and $\neg(n_a \leftrightarrow n_r)$ for a single n_q takes $O(|P|)$ time and $O(|P|(|T| + |P|))$, respectively, and it takes $O(|P|(|T| + |P|))$ and $O(|P|(|T| + |P|)^2)$ time complexity, respectively, for all. Hence, the total time complexity of constructing an S-Matrix is $O((|P|^2 + |T|) * |P| * |T|) + O(|P|(|T| + |P|)^2) = O((|P|^2 + |T|) * |P| * |T|)$. Since there are $O(|P|^2)$ entries in the S-Matrix, and for each entry, there are $O(|T| + |P|)$ n_s and n_e , it follows that the total time complexity is $O(|P|^2(|T| + |P|))$. Once the S-Matrix is constructed, we can find inconsistent pairs by searching the entries in the matrix with time complexities of $O(|P|^2(|T| + |P|))$. Hence, the total time complexity for verification of an SNC and liveness is $O((|P|^2 + |T|) * |P| * |T|)$.

4. REACHABILITY

Reachability has been found to be as a P-SPACE hard problem because it is related to marking and behavior. This, however, no longer holds true for an SNC since we can reduce the problem to a structural one. The S-Matrix is helpful for finding the structural relationship between two places ($p1$ and $p2$) defined below.

We can decompose M_0 into a number of safe ones where each can be solved in a structural fashion since temporal and structural relationships are equivalent (see **Observation 1**), thus reducing the problem to a structural one.

Definition 9 In an SNC, the structural relationship between two places ($p1$ and $p2$) is one of the following:

- $p1 \leftrightarrow p2$, i.e., they are *sequential* (SQ) to each other if they are in an elementary circuit;
- $p1 \mid p2$, i.e., they are *exclusive* (EX) to each other if $\neg(p1 \leftrightarrow p2)$ and $\exists n_s^{1,2} \in P$;
- $p1 \parallel p2$, i.e., they are *concurrent* (CN) to each other if $\neg(p1 \leftrightarrow p2)$ and $\exists n_s^{1,2} \in T$.

The temporal relationship between two places ($p1$ and $p2$) in a live and safe SNC is as follows: During the process of producing one product,

- $p1 \leftrightarrow_{\tau} p2$ if $p1$ is marked before $p2$;
- $p1 \mid_{\tau} p2$ if $p1 \leftrightarrow p2$ is not true and if $p1$ and $p2$ never get marked at the same time;
- $p1 \parallel_{\tau} p2$ if $p1$ and $p2$ may get marked at the same time.

Definition 9 may be applied to transitions in a similar fashion. Thus, two transitions may be mutually concurrent, exclusive, or sequential. Note it may be the case that both $p1 \mid p2$ and $p1 \parallel p2$ hold due to the fact that there may be multiple but different types of n_s (see Fig. 3 (c)). Which one holds depends on the current marking M . In Fig. 3 (c), if only $p10$ ($p10'$) holds a token in M , then $p13 \mid p14$ ($p13 \parallel p14$) relative to $p10$ ($p10'$).

For any pair of places in an HSNC, however, they cannot be both exclusive and concurrent since all n_s must be of the same type. Note that if they are sequential, they are in an elementary circuit and can be neither exclusive nor concurrent. Hence, we have the following observation.

Observation 1: For any pair of places $p1$ and $p2$ in a safe and live HSNC, (1) they can have exactly one of the three relationships in Definition 9. (2) Structural and temporal relationships are equivalent.

(2) is a natural consequence of (1).

Definition 10 A GCN or G is a maximal set of all places that are mutually concurrent to each other, i.e., $GCN = \{q \mid \forall r \in GCN, r = q \text{ or } r \parallel q\}$. G_1 is *sequentially earlier than* G_2 , denoted as $G_1 \rightarrow G_2$, if $\forall p1 \in G_1$, either $p1 \in G_2$ or $\exists p2 \in G_2$, and $p1 \rightarrow p2$. G_2 is *sequentially next to* G_1 and denoted as $G_1 \circ \rightarrow G_2$, if $\forall p1 \in G_1$, either $p1 \in G_2$ or $(p1 \bullet) \bullet \in G_2$.

In Fig. 7 (d), $G_1 = \{p5\}$, $G_2 = \{p6, p8\}$, and $G_1 \circ \rightarrow G_2$. GCN may be defined similarly for a set of transitions.

Definition 11 $P(M)$ is the set of places with tokens under M . $\bar{P}(M_j) = G \setminus P(M_j)$ if $P(M_j) \subset G$. $\mu(\eta)$ is a marking vector, where $\eta \subset P$, $m(p) = 0$ if $p \notin \eta$ or $m(p) = 1$ if $p \in \eta$.

Definition 12 $M = (m_1, m_2, \dots, m_k)$ is a *live marking* if the HSNC is live. M is a *safe marking* if $\forall i, 1 \geq m_i$, and $\exists j, m_j > 0$. It is a *safe live marking* if it is both *safe* and a *live* marking. The residual marking M^r is obtained by removing a maximum set of safe live markings from M such that M^r is not a live marking.

We will first consider the simple case, where $M^r = 0$ (all the components of vector M^r are 0); M can be decomposed into an integer set of *safe live markings* M_i^s corresponding to a set of G 's. Each reachable marking is the sum of reachable ones of all such M_i^s as shown in Theorem 1.

Definition 13 A G is defined as being *marked* if at least one place in it is marked. It is *safely marked* if each place in it holds exactly one token. An HSNC is defined as being *safely marked* if there exists only one *safely marked* G and if no other G is marked.

Definition 14 A local concurrent set (LCN) of p_i with respect to p_k , C_{ik} , is the maximal set of all *places* which are concurrent to each other and are equal to or concurrent to p_i , but not to p_k ; i.e., $C_{ik} = \text{LCN}(p_i, p_k) = \{p_z \mid p_z = p_i \text{ or } p_z \parallel p_i, \neg(p_z \parallel p_k), \forall p_{z1}, p_{z2} \in C_{ik}, p_{z1} \parallel p_{z2}\}$. $C_{ii} = \{p_i\}$. A set of places ζ is an LCN with respect to another set if $\exists p_a, p_b$ such that $\zeta = C_{ab}$.

In Fig. 8, $C_{2,10} = \{p_2\}$ and $C_{10,2} = \{p_{10}\}$. The following lemma is useful for proving Theorem 1.

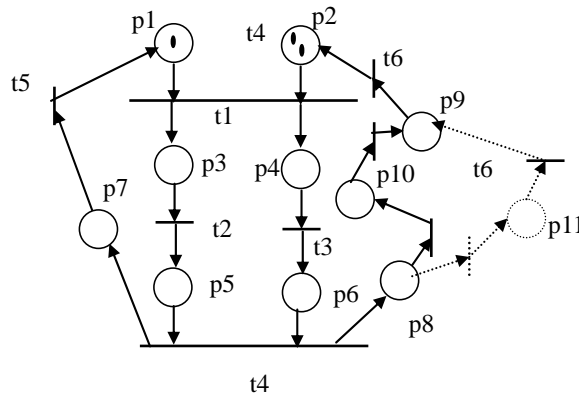


Fig. 8. An example of computing the reachable markings due to the residual (extra token) marking at p_2 .

Lemma 2 For any transition t , (1) $\bullet t$ and $t\bullet$ are mutually LCN. (2) Let G_1 and G_2 be two GCN containing $\bullet t$ and $t\bullet$, respectively; then $G_1' = G_1 \cup t\bullet \setminus \bullet t$ and $G_2' = G_2 \cup \bullet t \setminus t\bullet$ comprise each another GCN.

Proof: (1) holds by Definition 14. (2) Assume the contrary; then, $\exists p_1 \in t\bullet, p_2 \in G_1 \setminus \bullet t$ such that either (a) $p_1 \leftrightarrow p_2$ or (b) $p_1 \parallel p_2$. For (a), $\exists p_3 \in \bullet t, p_2 \leftrightarrow p_3$, but both p_2 and p_3 are in G_1 —contradiction. For (b), $\exists p_3 \in \bullet t, p_2 \parallel p_3$, but both p_2 and p_3 are in G_1 —contradiction. Hence G_1 is a GCN. The same proof applies to G_2 . \square

Theorem 1 For a safely marked live HSNC, if $P(M_0)$ is a GCN, then so is $P(M)$ for any reachable marking M .

Proof: Prove by induction. From M_0 , we fire an output transition of a $p \in P(M_0)$ to reach M_1 . $P(M_1)$ is a GCN by Lemma 2. The same reasoning applies to any reachable marking M_j instead of M_0 . The marking reached by firing one enabled transition corresponds to another GCN. \square

In Fig. 3 (b), if in M_0 , only p_9 holds a token, then M is not reachable if each of p_{13} , p_{14} , and p_{14}' (not a GCN) holds a token, but it is reachable if each of p_{11} , p_{12} , and p_{14}' (a GCN) holds a token. **No reachability analysis is needed.**

Corollary 1: Any safe and live HSNC N in M_0 stays safe for all subsequent reachable markings M .

Proof: It holds because $M = \mu(P(M)) = \mu(G)$ (by Theorem 1, $P(M) = G$) is a safe live marking. \square

But when $P(M_0)$ is not a GCN, in general, a safely marked HSNC is not live. Fig. 3 (a) shows a counter example, where $\{p_{13}, p_{14}\}$ (a PT-inconsistent pair) is marked in M_0 and is not a GCN, yet the net is live and safe. This is because their n_e is a transition and when it fires, the resulting marking corresponds to a GCN. Thus, they act like concurrent places, and we say that they are **n_e -concurrent**, while the one in Definition 9 is **n_s -concurrent**. Furthermore, when we check whether $M_1 \rightarrow M_2$, places in M_1 should be n_e -concurrent, and those in M_2 should be n_s -concurrent. *With this, our theory can be applied directly to ISNC.*

Note that M_0 in Fig. 3 (a) is not reachable from any $M \in R(M_0)$. Hence, the converse of Theorem 1 is not true if the GCN is n_e -concurrent. It is true only for n_s -concurrent GCN. In order for a $\mu(G)$ to be reachable, G must be n_s -concurrent.

We shall ignore this subtle difference in the following and assume that it is **n_s -concurrent** except in the final generalized algorithm given at the end of this section.

Theorem 2 For a safely marked live HSNC N , then $M = \mu(G_1)$ is a reachable marking, where G_1 is an arbitrary GCN.

Proof: Assume the contrary and that $\mu(G_1)$ is not reachable; then, $\exists p_1 \in G_1$ and $M(p_1) = 0$ since N is safe. This implies that $\forall t \in \bullet p_1$ and that t is not live — contradiction. \square

Lemma 3 $\text{GCN1} \cup \text{LCN2} \setminus \text{LCN1}$ is another GCN, where $\text{LCN1} \subseteq \text{GCN1}$.

Proof: Consider the subnet containing all cycles that contain one place in LCN1 and another in LCN2. The subnet N_2 is an SNC, and both LCN1 and LCN2 are GCNs in N_2 . Hence, $\exists \sigma$ such that only LCN2 is marked. And we can see that $\text{GCN1} \cup \text{LCN2} \setminus \text{LCN1}$ is marked in N . From Theorem 1, we can see that it is a GCN. \square

The following lemma shows the algebraic additive property of reachability when multiple GCN hold tokens.

Decompose M_0 into a number of $\mu(G_k)$, expressed as $M_0 = \sum_{k=0}^{k=l} \mu(G_k)$. Similarly, any reachable marking can be decomposed in the same way: $M = \sum_{k=0}^{k=q} \mu(G'_k)$. The linearly marking additive property is as follows;

Lemma 4 For an HSNC, if $M_0 = \sum_{k=0}^{k=l} \mu(G_k)$, then $M = \sum_{k=0}^{k=q} \mu(G'_k)$ and $q = l$.

Proof: The proof is based on the fact that by Theorem 1, each $\mu(G_k)$ can proceed to its reachable markings independently of other $\mu(G_j)$. The lemma holds if each $\mu(G_k)$ reaches $\mu(G'_k)$. The independence comes from the fact that the firing of enabled transitions under $\mu(G_k)$ does not need tokens from other $\mu(G_j)$ and from the fact that once they have been fired, no tokens from other $\mu(G_j)$ are removed. Independence between two G_1 and G_2 may fail when $G_1 \cap G_2 \neq \emptyset$. Let t be a firable transition under $\mu(G_1)$ and $\mu(G_2)$. After t fires, each place in $t\bullet$ ($\bullet t$, resp.) gets (loses, resp.) a token. The resulting marking is $\mu(G_1) - \mu(\bullet t) + \mu(t\bullet) + \mu(G_2)$. By Lemma 3, it is a linear sum of two $\mu(G)$: $\mu(G_1') + \mu(G_2)$, or $\mu(G_1) + \mu(G_2')$, where $\mu(G_1') = \mu(G_1) - \mu(\bullet t) + \mu(t\bullet)$ and $\mu(G_2') = \mu(G_2) - \mu(\bullet t) + \mu(t\bullet)$. Thus, the two GCN markings fire transitions independently. The same conclusion applies if more than one such transition is fired and more than two G are involved. \square

The independence assumption fails for the ISNC (but $m_0(p16) = m_0(p17) = 2$, $m_0(p22) = 4$) shown in Fig. 6, if the two $\mu(G)$ work independently and the net is live. But if the four tokens at $p22$ work together to reach M where only $p18$ and $p19$ hold tokens, then the net becomes deadlocked. The structure involved is as follows: $p18$ and $p19$ are a TPIP; i.e., $n_s^{18,19} = t1 \in T$, $n_e^{18,19} = p20 \in P$. In addition, $n_s^{18,19} = p22 \in P$, and the types of n_s of $p18$ and $p19$ include both a transition and place. Deadlock happens when the tokens at $p16$ and $p17$ flow toward the place-type $n_e^{18,19} = p20$. This situation does not happen in the case of HSNC because all n_e are of the transition type.

Note that the above place type, $n_s = p22$, forces the TPIP to be exclusive rather than concurrent. This prevents the bad siphon $\{p15, p16, p17, p20\}$ from being vacated and makes the ISNC live. However if $m(n_s) > 1$, then the TPIP becomes concurrent again and makes the ISNC not live despite the increase of marking at n_s . This is called *non-monotonic marking*.

Lemma 4 implies that the reachability problem can be translated into a structural one for an SNC. That is, a marking is reachable from M_0 if and only if the corresponding set of places holding tokens is a GCN. This nice property of reachability still holds for the case of multiple GCNs, where each place in the GCNs holds a token because of the algebraic additive nature.

Note that it is unclear how a GCN can be found when some pair of places (e.g., $p13$, $p14$ in Fig. 3 (c)) can be both concurrent and exclusive; i.e., their n_s can be both a place ($p10$) and a transition ($t12'$). Considering the fact that after firing $t12'$, both $p13$ and $p14$ will get a token, we will include both in a GCN. Note that each of $\{p10\}$ and $\{p13, p14\}$ in Fig. 3 (c) is a GCN, and that the latter can be reached from the former via $\sigma = t13t14t21t11t9't12'$.

If $M^r \neq 0$, M^r can be decomposed into an integer set of *safe* (but not live) *markings* M_j^r , then each reachable marking is the sum of the reachable ones of all such M_i^s and M_j^r . We need to find all reachable markings due to M_j^r . It is the set of all safe markings M , such that $P(M) = \chi$ and χ satisfies

$$\exists G, \chi \subset G, \chi = P(M_j^r), \text{ or } (\chi \leftrightarrow P(M_j^r), \neg(\chi \leftrightarrow \bar{P}(M_j^r))).$$

Example: In Fig. 8 (excluding the dashed arcs and nodes), $M^r = M_j^r = \mu(p_2)$; $P(M_j^r) = \{p_2\} \subset G = \{p_1, p_2\}$; $\bar{P}(M_j^r) = G \setminus P(M_j^r) = \{p_1\}$; all possible χ are: $\{p_2\}$, $\{p_8\}$, $\{p_9\}$, $\{p_{10}\}$ and $\{p_{11}\}$. When $\chi = \{p_9\}$, $G = \{p_1, p_9\}$, we have $\chi \subset G$, $\chi \leftrightarrow P(M_j^r)$, $\neg(\chi \leftrightarrow \bar{P}(M_j^r))$. Any reachable marking is of the form $\mu(G) + \mu(\chi)$. Note that we can not simply remove M_i^s from M_0 and set $M_0 = M^r$. We would obtain only $\chi = \{p_2\}$ with missing markings since $\chi = \{p_8\}$, $\chi = \{p_9\}$, $\chi = \{p_{10}\}$, and $\chi = \{p_{11}\}$ respectively. To avoid this, we should perform reverse firings, where a transition fires backwards with all the arcs in the net reversed. For instance, we could fire t_6 under $M_0 = M^r$ to remove the token in p_2 and put a token in p_9 . Thus, to obtain all the reachable markings due to M^r , we should perform all forward and reverse firings.

However, if $M_0 = M^r$, then reverse firings will lead to an incorrect $M = \mu(p_9)$ which actually is not reachable. Also, if we add a PP-path Γ (dashed) from p_8 to p_9 , the above χ (when $\chi = \{p_{10}\}$) might need to add places (p_{11}) on Γ (so that the new $\chi = \{p_{10}, p_{11}\}$). This indicates that a formal theory is required, based on the concept of LCN.

In Fig. 8, χ is any $C_{i,2} = \{p_i\}$ ($p_i = p_8, p_9, p_{10}, p_{11}$), where $C_{2,i} = \{p_2\}$. Note that when $p_i = p_4$, $C_{2,i} = \{p_1, p_2\} \neq \{p_2\}$; hence, no χ includes p_4 . Also, χ cannot be $\{p_{10}, p_{11}\}$ because $p_{10} \mid p_{11}$ and they cannot both hold tokens for $M_0 = \mu(G)$.

To generalize, we provide Lemma 5 below. Similar conclusion to Lemma 4 applies to LCNs. We can isolate a local subnet and elevate every structure that was originally local to global status. Thus, a LCN will become to a GCN relative to the local subnet. In addition, the above lemmas related to GCN can be extended to LCN.

Lemma 5 Let $C1$ be a set of places in an SNC, and let $M_1 = \mu(C1)$ such that $C1$ is an LCN with respect to another LCN $C2$; then, $\exists a \sigma$ such that each place in $C2$ holds a token and $M_2 = \mu(C2)$.

Proof: Prove by induction similar to the proof for a GCN in Lemma 4. □

Note that in a reachable marking of the net shown in Fig. 8, $P(M^r) = p_8$. $\mu(\chi)$ is any reachable marking under $P(M_0) = p_8$ and χ is any LCN with respect to p_8 ; i.e., $C_{i,8}$.

Similar to GCN, each can be decomposed into a set of LCN and a set of residual tokens. All reachable markings from M_j^r correspond to the relevant LCN, thus reducing the problem to a search for all such LCNs. In addition, they can be obtained from the S-Matrix.

We will now extend Definition 14 to two sets of mutually concurrent places, ζ_i and ζ_k , to simplify the algorithm.

Definition 15 ζ_i is a General Concurrent Set (GCS, called GCS1) with respect to ζ_k (called GCS2) if $\forall p_a \in \zeta_i, \exists p_b \in \zeta_k$ such that $C_{ab} \subseteq \zeta_i$ and $C_{ba} \subseteq \zeta_k$.

When ζ_i is a GCN (LCN), so is ζ_k . This enables us to develop a generalized algorithm instead of separate algorithms for a GCN and an LCN. Given M_0 and M , the following algorithm checks whether M is reachable from M_0 . We will ignore the case when $P(M_0)$ is a proper subset of a GCN; the net is not live and is of no interest.

Generalized Reachability Algorithm

1. Pick a marked place in M_0 and find its maximum set of mutually n_e -concurrent places η .
2. Find in M the marked GCS (n_s -concurrent) ζ of η .
3. If ζ cannot be found and $M \neq 0$ (**M is not a null vector with all components being zero**), then declare that M is not reachable and exit.
4. Otherwise, set $M_0 \rightarrow M_0 - \mu(\zeta)$ and $M \rightarrow M - \mu(\eta)$.
5. If $M_0 = M = 0$ (i.e., **both $M_0 = M$ are null vectors**), then declare that M is reachable and exit.
6. If $M_0 = M \neq 0$, then go to Step 1 and continue.
7. Only one of M_0 and M is 0; then, declare that M is not reachable and exit.

Example: Let M in Fig. 8 be $m(p5) = m(p6) = m(p10) = 1$, and let $m(p) = 0$ for all other places. Then, $\eta = \{p1, p2\}$ (Step 1), $\zeta = \{p5, p6\}$ (Step 2), $M_0 \rightarrow M_0 - \mu(\zeta) = \mu(p2)$, $M \rightarrow M - \mu(\eta) = \mu(p10)$ (Step 4), $\eta = \{p2\}$ (Step 1), $\zeta = \{p10\}$ (Step 2), $M_0 \rightarrow 0$, $M \rightarrow 0$ (Step 4), $M_0 = M = 0$, so M is reachable and exit (Step 5).

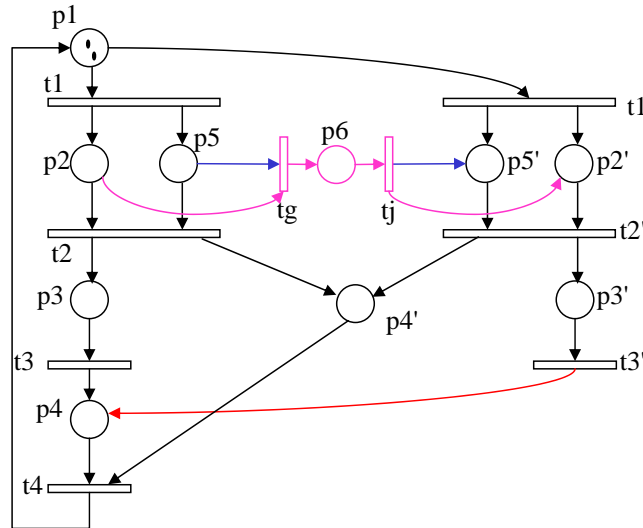


Fig. 9. Another example of reachability of homogeneous synchronized choice petri nets.

Fig. 9 shows another example. $\{p1\}$, $\{p3, p4\}$ and $\{p3', p4\}$ are GCN G_1 , G_2 and G_3 respectively. $M_0 = 2\mu(G_1)$. $\mu(G_1) + \mu(G_2)$ is reachable. Hence, it is reachable where $p3, p3', p4$ and the rest hold 1, 1, 2 and 0 tokens respectively.

Remark: Compared with the dual reachability theorems for free-choice (FC) systems [14], our approach has the following advantages: there is no need to (1) solve for the firing vector, (2) verify that the net is a NOT (NOP) net, or (3) find all empty siphons (traps) and their O-subnets (O-subnets). In addition, our approach applies to a larger class of well-behaved SNC than LBFC (life and bounded FC).

5. FIRING SEQUENCE

After determining that the target marking M_t is reachable, we can proceed to find the firing sequence σ . Similar to the decomposition of markings, it seems that we can also find firing sequences via decomposition. Hence, we will deal with the case where both $P(M_0)$ and $P(M_t)$ are GCNs.

We can tackle the problem in a gradual fashion, starting with two special and simpler SNCs: (1) an MG with multiple circuits and tokens (Fig. 8), and (2) a simple net with only one transition enabled in M_0 (Fig. 1). Multiple interconnected simple nets form a more general SNC.

We will explain how σ can be obtained in these two special cases. We will then extend them to general cases, where multiple interconnected simple nets form a more general SNC.

Definition 16 A simple net is defined as an SNC, and there exists a GCN that contains a single transition.

Let x be a T-invariant; then, $\sigma + x$ is also *legal* (i.e., when the firing sequence reaches a transition t in $\sigma + x$, t is firable), where the ‘+’ sign indicates *concatenation*. Hence, we need to find the shortest σ such that $\sigma - x$ is no longer legal (‘-’ means deleting from σ the firing sequence x).

Definition 17 σ is the shortest firing sequence if it is legal and any subsequence of it is not legal.

5.1 Mark Graph with Multiple Interconnected Circuits

Given M_0 (solid tokens) in Fig. 10 (from [7]), in order to find the σ to reach M (gray tokens), consider circuit $C_i = [p_{2i-1} \ t_i \ p_{2i} \ t_{i+1} \ p_{2i-1}]$ with T-invariant $[1 \ 1]$. t_i and t_{i+1} must both fire to return to initial marking. For the token to move from p_{2i-1} to p_{2i} , $\sigma = t_i$. For C_1 , t_1 must fire. Afterwards, tokens in p_1 and p_3 move to p_2 and p_4 (matching those in M) respectively. We say C_1 and C_2 (the rest are not) are in a *suited* position and they are compatible as defined below.

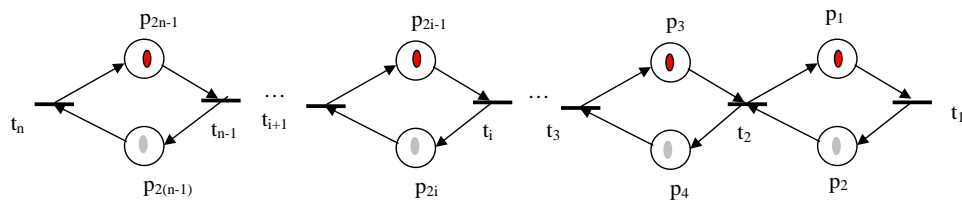


Fig. 10 [7]. An example of multiple interconnected MG. Finding a legal firing sequence is nontrivial.

Definition 18 Let N_1^c and N_2^c be two simple subnets of N with sub-firing sequences σ_1 and σ_2 , respectively; N_1^c and N_2^c are said to be incompatible on t if $\sigma_1(t) \neq \sigma_2(t)$.

If $\sigma_1(t) > \sigma_2(t)$, then additional firings of t in S_1 will drive N_2^c away from its M_t^2 . In order to return to M_t^2 , σ_2 must be lengthened by x_2 so that the new $\sigma_2'(t) = \sigma_1(t)$.

To match more, we next move a token from p_5 to p_6 by firing t_3 as shown in Fig. 10. But this moves a token away from p_4 , which is undesirable; C_2 and C_3 are incompatible. To remedy this situation, we need to fire t_2 in order for C_2 to return to its suitable position. σ_{23} remains at 1, and $\sigma_2' = \sigma_2 + t_2 t_3 = \sigma + x_2$. Continuing in this fashion, when C_2 reaches its final position, $\sigma_2' = \sigma + m * x_2$, where m is a positive integer. We need to find the exact m value. In general, $\sigma_i' = t_{i+1} + m_i * x_i$.

The key to the solution lies in the fact that for any circuit C_i , the synchronic distance $\sigma_{i+1,i} = 1$ (the synchronic distance between t_{i+1} and t_i ; i.e., t_{i+1} can fire at most once without firing t_i and vice versa). Thus, if $\sigma(t_n) = 0$, then

$$\sigma(t_{n-1}) = 1, \sigma(t_{n-2}) = 2, \dots, \sigma(t_i) = n - i, \dots, \sigma(t_3) = n - 3, \sigma(t_2) = n - 2, \sigma(t_1) = n - 1. \quad (1)$$

Note that when we fix C_3 , each C_2 and C_1 must be induced to fire once. And in general, when we fix C_i , each C_j $i \geq j$ must be induced to fire once. When we finally fix C_n , t_n fires once, and C_n is in a suitable position. After each C_j $n \geq j$ fires once, M is reached. Using the fact that $\sigma_{i+1,i} = 1$, we have Eq. (1).

Thus, even as simple as a MG, the legal firing sequence σ problem is neither trivial nor intuitive. In general, each circuit can be replaced by an SNC. A similar strategy, however, can be applied. We can first obtain σ_{j0} for each such N_j^c . We can also solve for its x_j . The final $\sigma_j = \sigma_{j0} + k * x_j$, where k , a nonnegative integer, can then be determined.

We move from M_0 toward M in a progressive manner. In each step, we consider a new N_k^c ; more places now have the same number of tokens as in M than in previous steps. If a transition involved in the new subfiring sequence is not involved in a previous step, we must add the corresponding x_k . Eventually, M is reached and we obtain σ .

In the following section, we will consider the basic case, where each of the above N_k^c is simple; i.e., there exists a GCN containing a single transition.

5.2 Simple Net

Each of $P(M_0)$ and $P(M_t)$ is a GCN (denoted G_0 and G_t , respectively), and any shortest σ corresponds to an MG subnet between $P(M_0)$ and $P(M_t)$. Each transition in the subnet appears once in σ .

Based on Observation 1, temporal and structural relationships are equivalent for a live safe SNC with $P(M_0)$ being a GCN. Hence no two transitions in σ are mutually exclusive. Thus, we have the following algorithm.

Algorithm II to find σ in a simple net

1. Set σ to NULL and $A = G_0$.
2. Select a $t \in A \bullet$. Find $G(t) \subset A \bullet$; $G(t)$ is a GCN that contains t ; $\sigma + \beta(G(t)) \rightarrow \sigma$, where $\beta(G(t))$ indicates an arbitrary serial combination of transitions in $G(t)$ (defined simi-

- larly when GCN is a set of places, and transitions in $G(t)$ are mutually n_s -concurrent); e.g., if $G(t) = \{t_1, t_2, t_3\}$, then there are six possible $\beta(G(t))$; one of them is $t_3t_1t_2$.
3. Make a new A ; $G(t) \bullet \rightarrow A$. Go to Step 2. Repeat until $A = G_t$.
 4. Output the final σ .

It takes $O(n)$ time to find one σ , where n is the number of nodes in the net.

5.3 General Cases

We first decompose the net into a number of simple nets, N_i^c , whose σ_i can be obtained using **Algorithm II**. We then adjust some σ_i to make all the N_i^c and N_j^c pairs compatible.

Algorithm III to find the shortest legal firing sequence

1. Decompose N , M_0 , and M into a number of simple nets N_i^c , M_{0i} , and M_i ($i = 1, \dots, r$), respectively, based on **Algorithm IV** (presented next).
2. If $\exists i, M_i \notin R(M_{i0})$, then M is not a reachable marking, so exit.
3. $\forall i$, find σ_i to reach M_i from M_{i0} .
4. Repeat:
 - While there exists incompatible N_i^c and N_j^c on a certain t and while $\sigma_i(t) > \sigma_j(t)$, enlarge $\sigma_j(t)$ by $(\sigma_i(t) - \sigma_j(t)) * x_j$;
 $\sigma_j'(t) = \sigma_j(t) + (\sigma_i(t) - \sigma_j(t)) * x_j$, where x_j is the j th component of the T-invariant x .
5. The final σ is obtained by summing (i.e., concatenating) all such $\sigma(t_k)$ where $t_k \in N_i^c$, $i = 1, \dots, r$, $\sigma = \sum \sigma(t_k)$.

Steps 1, 2, and 3 take $O(n^2)$, $O(n^2)$, and $O(n)$ time, respectively. It takes $O(n)$ time to find a T-invariant. Steps 4 and 5 take $O(v * n)$ time where v is the number of simple nets. In the worst case, v is $O(n)$, and the time complexity is $O(n^2)$. The total time complexity for the algorithm is, therefore, $O(n^2)$.

Algorithm IV for simple net decomposition

1. Find a GCN of places G .
2. Divide the rest of the places into a number of subsets such that the places in a subset P_i are not concurrent to places in a subset G_i of G . $P_i \cup G_i$ forms a simple net.

The time complexity is $O(n^2)$.

6. CONCLUSIONS

The S-Matrix records the structural relationship between any pair of places. It is useful for detecting whether a given net is an SNC and if it is, whether it is live and reversible. For an HSNC, we can decompose M_0 into a number of safe ones where each can be solved in a structural fashion since temporal and structural relationships are equivalent. As a result, the reachability problem for an HSNC, is equivalent to that for a GCN and an LCN. The time complexity for verifying a GCN and an LCN, hence, the reachability

problem, is polynomial. As a result, it also takes polynomial time to find legal firing sequences.

The results given below are preliminary and will be rigorously proved in a future paper.

For an ISNC, where n_s and/or n_e may be both transitions and places, we have proved that (1) its n_s and n_e cannot be both transitions and places, and that (2) at least one pair of places is TP-inconsistent (TPIP) or PT-inconsistent (PTIP).

For a safe ISNC with a PTIP, the above equivalence of GCN and LCN still holds if we find enabled transitions by considering n_s -GCN in M_0 and n_e -GCN in M_1 in order to check whether M_1 is reachable from M_0 . Once an enabled transition fires, the tokens in an n_s -GCN go to an n_e -GCN. If the n_s -GCN is not also an n_e -GCN, then it will never be marked again after firing the enabled transitions. This causes the net irreversible. If the PTIP has $n_e \in P$, then the two tokens in the TPIP may be diverted to the n_e , thus causing the net to not be safe. In this case, we can decompose M_0 and proceed as discussed above.

For a live ISNC, the above ‘linear additive’ property disappears in the presence of bad siphons. This happens in the presence of TPIP with $n_s \in P$, $m(n_s) = 1$, which can be detected using the S-Matrix. Its reachability problem can be dealt with in a similar fashion to the non-SNC approach discussed next. Note that the above n_s forces the TPIP to be exclusive rather concurrent. This prevents the bad siphon from being vacated and makes the ISNC live. However if $m(n_s) > 1$, then the TPIP becomes concurrent again and causes the ISNC to not be live despite the increase of marking at n_s . This is called *nonmonotonic marking*.

We have extended the above results to the non-SNC case for flexible manufacturing system applications. For the non-SNC case with TP- and PT- second order structures [2], the above nice property disappears, and efficient analysis may no longer be possible. However, reachable markings inside a local structure are only affected by the markings surrounding and structural characteristics inside the local structure. We can exhaust all possible reachable markings and develop rules for enumerating them from its initial marking without conducting exhaustive reachability analysis. This, combined with linear analysis for other parts of the net that conform to SNC requirement may render the analysis efficient.

We have proposed simplifying the Reachability Problem by first decomposing N into a number of SNC components, N_i^c and M_{0i} , and checking whether each such M_{0i} is reachable in N_i^c and whether all submarkings M_{0i}^s of a subnet (often involving resource sharing) of N_i^c is also reachable in another N_j^c . Further, we have applied the above concept to the detection of deadlocks, nonlive transitions, and/or derivation of the marking condition for liveness.

In a separate paper [2], we allowed TP- and PT- FOS (a first order structure); the reduced weighted net must, however, be an SNC. This class of nets, called the ESNC (the extended SNC) covers that in [4] as a special case. In future research, we will extend the proposed techniques to the ESNC.

REFERENCES

1. K. Barkaoui, J. M. Couvreur, and C. Dutheillet, “On liveness in extended non self-

- controlling nets,” in *Proceedings of 16th International Conference on Application and Theory of Petri Nets 1995*, LNCS, No. 935, Springer-Verlag, 1995, pp. 25-44.
2. D. Y. Chao, “Extended synchronized choice nets,” *The Computer Journal*, Vol. 46, 2003, pp. 505-523.
 3. D. Y. Chao, “Reachability of non-synchronized choice petri nets with resource-sharing,” in *Proceedings of IEEE International Conference on Systems, Man and Cybernetics (SMC '02)*, Vol. 5, 2002.
 4. D. Y. Chao and J. A. Nicdao, “Liveness for synchronized choice petri nets,” *The Computer Journal*, Vol. 44, 2001, pp. 124-136.
 5. D. Y. Chao and D. T. Wang, “Two theoretical and practical aspects of knitting technique – invariants and a new class of petri net,” *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 27, 1997, pp. 962-977.
 6. G. Ding and T. Sekiguchi, “A method of determining the firing sequence of petri net,” *Transactions of the Society of Instrument and Control Engineers*, Vol. 25, 1989, pp. 698-705.
 7. J. Desel and J. Esparza, “Shortest paths in reachability graphs,” in M. A. Marsan, LNCS, Vol. 691, Application and Theory of Petri Nets 1993, *Proceedings 14th International Conference*, Chicago, Illinois, pp. 224-241, Springer-Verlag, 1993. Also in *Journal of Computer and System Sciences*, Vol. 51, 1995, pp. 314-323.
 8. D. I. Lee, S. Kumagai, and S. Kodama, “Handles and reachability analysis of free choice nets,” in *Proceedings of International Conference on Application and Theory of Petri Nets*, LNCS, No. 935, Springer-Verlag, 1995, pp. 298-316.
 9. R. J. Lipton, “The reachability problem requires exponential space,” Research Report No. 62, Dept. of Computer Science, Yale University, 1976.
 10. T. Murata, N. Komoda, and K. Matsumoto, “A petri net based controller for flexible and maintainable sequence control and its applications in factory automation,” *IEEE Transactions on Industrial Electronics*, Vol. IE-33, 1986, pp. 1-8.



Daniel Y. 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 MIS 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 is now working on the exploration of propertiew of a new class of Petri nets and implementation of several CAI tools based on Visual C++. He has published 72 (including 15 journals) papers in the area of communication protocols, Petri nets, DQDB, networks, FMS, data flow graphs, and neural networks.