Reachability of Nonsynchronized Choice Petri Nets and Its Applications

Daniel Y. Chao, *Senior Member, IEEE*

***Abstract*—A new local structure called a second-order structure was proposed to generate a new class of nets called synchronized choice nets (SNC). SNC covers well-behaved free choice nets. Reachability is no longer P-Space hard problem, but can be solved with polynomial time complexity. How to extend them to non-SNC and its application to deadlock detection in flexible manufacturing systems are discussed.**

***Index Terms*—Deadlock, flexible manufacturing systems, Petri nets, reachability.**

I. Introduction

A FLEXIBLE manufacturing system (FMS) model consists of a set of working processes (WPs) competing for resources. A WP models a sequence of operations to manufacture a product. Ezpeleta *et al.* proposed a class of nets called systems of simple sequential processes with resources ($\mathrm{S^3PR}$) [1]. It is a state machine (SM) (see Def. 1) plus a set of places modeling the availability of resources. Each SM contains one idle place or state plus a number of states for the set of possible sequences of operations. The initial and the final state of a WP collapse into the idle state for cyclic models. The number of tokens at the idle state indicates the maximum number (constrained by the system resource capacity) of products that can be concurrently manufactured. Circular wait for resources can bring the system into a deadlock where some WP can never finish.

Because only one resource is used in each job stage and the processes are modeled using SMs in $\mathrm{S^3PR}$, its modeling power is limited. It cannot model iteration statements (loop) in each sequential process (SP) and the relationships of synchronization and communication among SP. At any state of a process, it cannot use multi-sets of resources. They compute all siphons (see Def. 5) that contains no traps and find the maximum number of tokens at each idle state followed by a prevention control policy of adding arcs and nodes with tokens. Most recent deadlock control approaches [2] extend this approach.

Unfortunately, the total number of siphons grows, in worst case, exponentially in the number of nodes. Thus, for large complicated systems, the prevention policy may no longer be appropriate. In this case, it is better to perform reachability analysis that explores all possible states, and hence can check various properties such as livelocks and race conditions. It is conceptually simple and relatively straightforward to automate and can be used in conjunction with model-checking procedures to check for application-specific as well as general properties. Also many control problems can be modeled by the reachability problem indicated by Ichikawa *et al.* [3].

To improve on the above sequential resource allocation system (S-RAS) [1], Ezpeleta *et al.* [4] proposed a nonsequential resource allocation system (NS-RAS), they proposed a general net model where even multiple copies of one type of resource is allowed to be used at each processing step. The modeling power is much enhanced, but the analysis becomes complicated. They, hence, proposed a deadlock avoidance approach with polynomial result by constructing reachability graph for the isolated execution of each production order, tiny compared with the size of that of the whole system, to find strongly connected components (not possible with siphon analysis).

However, prevention is preferred to avoidance because the computational effort is carried out once and off-line. Hence, it runs much faster in real-time cases compared with deadlock avoidance algorithms where much time is consumed by doing analysis online each time the system ought to change the state. Deadlock prevention control policy is essential when it is unacceptable to have deadlocks and real time response time is critical. They indicated that “the whole time to know if a system state is safe can take about two CPUs in the worst case” and “the proposed control method would be more or less permissive”. In their model, each WP is still an ordinary net as shown in [4, Fig. 2] [actually a synchronized choice net (SNC), a subclass of Petri nets (PNs) proposed in Section III].

Roszkowska [5] addressed the control problem of deadlock avoidance for compound processes. Due to the constraint of finite capacity of the resources and specific firing rules, she showed that the minimally restrictive supervisory control for assembly processes as well as compound processes with assembly and disassembly operations is NP-hard.

Jeng *et al.* [6] proposed a synthesis technique that merges resource control nets (RCN) through common transitions and transition subnets. It allows more general usage of a resource than that in [1]. Robots used at a state of a WP do not have to be released at the next state and resources can request one another with no parts involved.

The proposed algorithm holds valid only for special structures where any common transition can have at most one input operation place. Also, as shown in [6], even a single RCN could incur deadlocks while Jeng’s technique requires at least two RCN. As a result, they cannot model cases where an assembly operation is

Manuscript received May 10, 2004; revised October 3, 2004. This work was supported in part by the National Science Council under research grant NSC89-2213-E-004-001. This paper was recommended by Associate Editor M. C. Zhou.

The author is with the Department of Management and Information Science, National Cheng Chi University, Taipei, Taiwan, R.O.C. (e-mail: yaw@mis.nccu.edu.tw).

Digital Object Identifier 10.1109/TSMCB.2005.850171

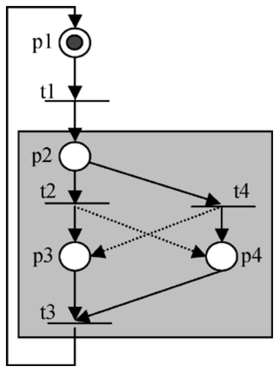

Fig. 1. Example of live and reversible SNC with no inconsistent pair.

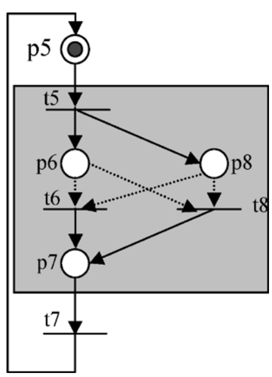

Fig. 2. Dual of the net in Fig. 1. This net is live and reversible without inconsistent pair.

performed on several different parts coming from separate preceding processes.

In [7], ERCN merged nets were extended to use local processing cycles to model degraded behavior in semiconductor manufacturing such as rework, failure, and maintenance. Each RCN is an SM with some transitions replaced by parallel blocks and has local loops.

We adopt an intermediate approach: modeling each WP with an SNC (see Def. 7) [8]–[10] and employing deadlock prevention rather than avoidance. Note that RCN is a subclass of SCN. SNC covers well-behaved (live, bounded and reversible) free choice nets (FC) (see Def. 1), yet it is not included in asymmetric choice nets (AC) (see Def. 1). An SNC allows internal choices (involving no resource) and concurrency and hence is powerful for modeling. SNC is so called because concurrent interactions between choices are synchronized. It therefore can model assembly/disassembly operations with multiple parts as well as lot split and lot merging in semiconductor manufacturing [7] and may apply to concurrent systems other than FMS such as database systems, operating systems, and parallel processing.

Any SNC (see Figs. 1–4) is bounded and its liveness conditions are simple. An integrated algorithm [8] has been presented for verification of a net being SNC and its liveness with polynomial time complexity. Designing a net in this class will suffer fewer errors than arbitrary nets and hence be more reliable. This further simplifies analysis. More important, it helps to simplify and enhance our synthesis rules in the Knitting technique [10], [11]. However, it cannot handle resource-sharing in FMS.

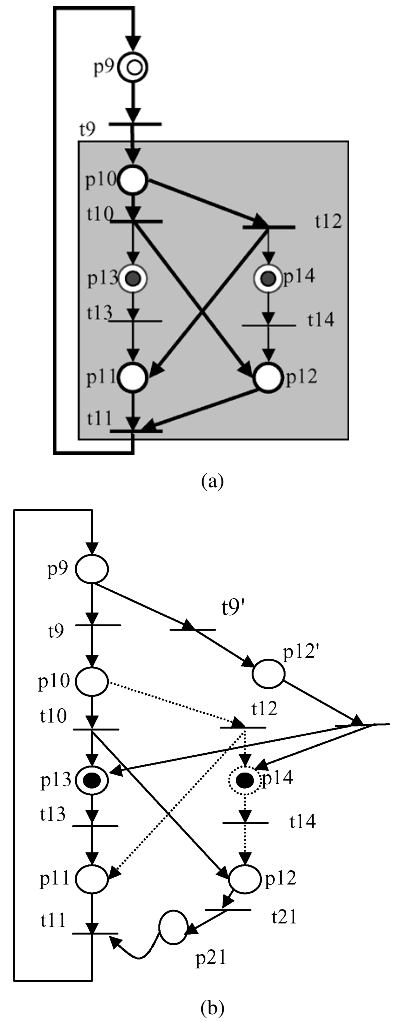

Fig. 3. (a) Irreversible SNC with PT-inconsistent pair (*p13*, *p14*). The bold part is the subnet N_x. (b) An ISNC. No longer irreversible. One $n_s^{13,21} = t12'$; (*p13*, *p14*) no longer PT-inconsistent.

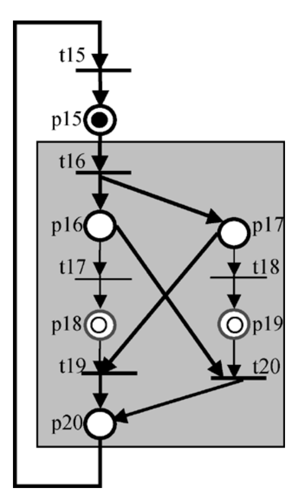

Fig. 4. Dual of the net in Fig. 3(a). The SNC is not live with TP-inconsistent pair (*p18*, *p19*).

FCs have been accepted as the largest class of PNs whose analysis can be solved in polynomial time. Lee *et al.* [12] have shown that the reachability problem of general PNs can be reduced to that of FC. However, Esparz [13] showed that the reachability problem is NP-complete even for a life and safe FC (LSFC). Bouziane [14] proposed a double exponential space algorithm for the general Petri net reachability problem. Polynomial results, however, do exist for special classes of nets such as acyclic PNs , deadlock-circuit nets, trap-circuit nets [15], extended marked graphs [16], and nets for logic controllers [17].

The reachability for homogeneous SNC (HSNC) [18], a subclass of SNC, is pretty simple because reachable markings have the property of linear additive and can be translated into a structure problem. The polynomial result can be extended to some non-SNCs (covering many FMS applications). We have proposed to simplify the reachability problem by first decomposing a net N into a number of HSNC components N_i^c and checking whether M_i^c is reachable in N_i^c for all i. However, there may be more than one way to do so. But different ways yield the same results as long as the decomposition is complete (see Def. 8).

For most FMS, it is easy to decompose it into a number of WP and resource components (RC). Each RC is an SNC and consists of a number of circuits. The decomposition is complete. If each WP is an SNC, then the reachability problem can be solved in polynomial time. There is no need to find maximum HSNC components. Otherwise, the WP must be further decomposed. If a WP contains many asymmetrical first-order structures (AFOS) (see Def. 7), then the decomposition may not be complete. This happens if it is an extended SNC (ESNC) [19] that can be transformed into a general Petri net (GPN) (see Def. 1) where the reachability problem is exponential even for a circuit.

We will apply the above concept to the detection of deadlocks or nonlive transitions and/or the derivation of the marking condition for liveness (MCL) of non-SNC FMS. Deadlocks occur when some siphons get token-free and can be detected if the marking is reachable. But the number of siphons grows exponentially with the size of the net. However, in most FMS, a number of siphons share the same controlling invariant. We only need to check that the siphon with the minimum tokens never gets token-free (the marking is not reachable). As a result, the amount of markings to verify is polynomial—we do not need to build reachability graph or to solve state equations.

Sections II and III present the basis to understand the paper. Section IV motivates the reader that Reachability Problem for non-SNC may be simplified via decomposition. Section V discusses the reachability problem for HSNC. Section VI applies the above concept to the detection of deadlocks or nonlive transitions and/or the derivation of the marking condition for liveness of non-SNC. Section VII concludes the paper.

II. Preliminaries

We follow [8] for the various terminologies of PN.

Definition 1: A Petri net is a 4-tupe where $P = \{p_1, p_2, \ldots, p_a\}$ is a set of places, $\mathrm{T} = \{t_1, t_2, \ldots, t_b\}$ a set of transitions, with $P \cup T \neq \emptyset$ and $P \cap T = \emptyset$; $F : (P \times T) \cup (T \times P) \rightarrow \{0, 1, 2, \ldots\}$ is the *flow relation* and a mapping from $(P \times T) \cup (T \times P)$ to nonnegative integers indicating the weight of directed arcs between places and transitions; $M_0 : P \rightarrow \{0, 1, 2, \ldots\}$ denotes an initial marking, then $N = (P, T, F)$ is a net. N^{-1} is the reverse net of N obtained by reversing the direction of all arcs in N. 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 (O-subnet) of N if $T_i = \bullet P_i (\mathrm{P}_i \bullet)$. M_0 denotes an initial marking whose i^{th} component, $m_0(p_i)$, represents the number of tokens in place p_i. The post-set of node x is $x\bullet = \{y \in P \cup T \mid F(x, y) > 0\}$, and its pre-set $\bullet x = \{y \in P \cup T \mid F(y, x) > 0\}$. A is the incidence matrix of a net: $A = [a_{ij}]$; a $a \times b$ matrix of integers and its typical entry is given by $a_{ij} = a_{ij}^{+} - a_{ij}^{-}$ where $a_{ij}^{+} = F(t_i, p_j)$ is the weight of the arc from transition t_i to its output place p_j, and $a_{ij}^{-} = F(p_j, t_i)$ is the weight of the arc to transition t_i from its input place p_j. t_i is firable if each place p_j *i*n $\bullet t_i$ 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., $M_0[\sigma > M'$. *General Petri nets (GPN)* are those for which $\exists j, w_j > 1$, or $\exists k, w_k > 1$. Ordinary nets are those for which $F : (P \times T) \cup (T \times P) \rightarrow \{0, 1\}$. An ordinary net is called a *state machine* if $\forall t \in T, |t \bullet| = |\bullet t| = 1$. It is a marked graph (MG) if $\forall p \in P, |p \bullet| = |\bullet p| = 1$. It is a free choice net 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 if $\forall p_1, p_2 \in P, p_1 \bullet \cap p_2\bullet \neq \phi \Rightarrow p_1\bullet \subset p_2\bullet$ or $p_1\bullet \supset p_2\bullet$.

Definition 2: Let $N = (P, T, F)$ be a net, (N, M_0) be a marked net and $R(M_0)$ 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 $\forall M \in R(M_0)$, $\forall p \in P$, $\exists k$, *a* positive integer, the marking at p, $m(p) \leq k$.

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: $\Gamma = [n_1\, n_2 \ldots n_k]$, $k \geq 1$, such that $n_i \neq n_j$ for $i \neq j$. A path is (non) *virtual* if it contains only (more than) two nodes. An elementary cycle in N is $\Gamma = [n_1\, n_2 \ldots n_k]$, $k > 1$ such that $n_i = n_j$, $1 \leq i \leq j \leq k$, implies that $i = 1$ and $j = k$.

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

Definition 4: A nonnegative, integer vector $Y(X)$ is called an S- (T-) invariant iff $Y(X) \neq 0$ and $AY = 0$ ($A^T X = 0$). The set of places p such that the component in Y, $y(p) > 0$ is called the support of the S-invariant and is defined as $||y||$. If there is a firing sequence containing all the transitions $t \in T$, such that M_0 can be recovered, the PN is said to be *consistent*. N is called *conservative iff* there exists a positive integer vector $y > 0$ such that $M^T y = M_0^T y, \forall M \in R(M_0)$.

It is well known [11] that the existence of positive S- (T-) invariant implies that N is conservative (consistent).

Definition 5: For a Petri net (N, M_0), a nonempty 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 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.

III. Handles, Bridges, First- and Second-Order Structures

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 and ends with a transition while a TP-handle starts with a transition and ends with a place. An first-order structure (FOS) consists of two handles (with no bridges inbetween) with the same end nodes. A second-order structure (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 $N = (P, T, F)$. $H_1 = [n_s n_1 n_2 \ldots n_k n_e]$ 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 $n_i \neq n'_j$ $\forall i, j$ defined above; n_s and n_e are called the start and the end nodes of H_1 and H_2. Note that n_s and n_e may be identical. An elementary directed path $B = [n_a, n_b \ldots, n_q]$ is a bridge from H_1 to H_2 if 1) $n_a \in H_1$, $n_q \in H_2$, $n_a \neq n_s$, $n_a \neq n_e$, $n_q \neq n_s$, $n_q \neq n_e$ and 2) $\forall n \in B$, if $n \neq n_a$, $n \neq n_q$, then $n \notin H_1$ and $n \notin H_2$. $p1 \leftrightarrow p2$ ($p1$ 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. The handle H *to a subnet* N' 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'.

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.

Definition 7: H_1 and H_2 are defined in Def. 6.

1) Let $\Gamma_1 = [n_s n_1 n_2 \ldots n_k p_1]$, $\Gamma_2 = [n_s n'_1 n'_2 \ldots n'_h p_2]$ and $\Gamma_1 \cap \Gamma_2$ ($\Gamma_1 \cup \Gamma_2$) denotes the intersection (union) of two graphical objects Γ_1 and Γ_2. n_s is a start node of (p_1, p_2) if $\Gamma_1 \cap \Gamma_2 = \{n_s\}$ and the *nearest start node* of (p_1, p_2); i.e., $n_s^{1,2} = n_s$ if Γ_1 does not contain other start nodes of (p_1, p_2). $n_e^{1,2}$ can be defined in a dual fashion. Let $\Gamma_3 = [p_1 n_s n_1 n_2 \ldots n_k n_e]$ and $\Gamma_4 = [p_2 n'_1 n'_2 \ldots n'_h n_e]$. n_e is an end node of (p_1, p_2) if $\Gamma_3 \cap \Gamma_4 = \{n_e\}$ and the *nearest end node* of (p_1, p_2); i.e., $n_e^{1,2} = n_e$ if it does not contain other end nodes of (p_1, p_2). $N_s^{1,2}$ ($N_e^{1,2}$) is the set of all such $n_s^{1,2}$ ($n_e^{1,2}$).
2) Let $\Upsilon = H_1 \cup H_2$ denote the union of two graphical objects H_1 and H_2. $H_1(H_2)$ is a *prime handle* to $H_2(H_1)$, if there are no bridges B between H_1 and H_2 and Υ is defined to be an FOS.
3) If $B_{12}(B_{21})$ is the only bridge from H_1 to H_2 (H_2 to H_1), then $\varphi = H_1 \cup H_2 \cup B_{12} \cup B_{21}$ is defined to be an sos (see the shaded area in Figs. 1–4).
4) (p_1, p_2) is called a TP-inconsistent pair (TPIP) of places 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 $\exists n_s^{1,2} \in P$ and $\exists n_e^{1,2} \in T$.
5) Let ω be an FOS (or SOS, handle, bridge, path), if its $n_s \in T$, $\mathrm{n}_e \in P$, 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*.
6) A strongly connected net is SNC, denoted N^c, if it satisfies the two requirements *R1* and *R2* where *R1*: (*R2*:) every PT- (TP-) handle to a certain circuit has a TP- (PT-) bridge from its complementary PT- (TP-) handle to itself.
7) An HSNC is an SNC where all n_s and n_e of a certain pair of places are of the same type (either all are transitions or all are places; otherwise it is an ISNC.

p_1 and p_2 in Def. 7.4 are inconsistent because they are concurrent (exclusive) and the tokens in them will flow to a set of mutually exclusive (concurrent) places. In Fig. 3(a), $N_s^{12,11} = \{t10, t12\}$ and $N_e^{12,11} = \{t11\}$; *p10* is not a $\mathrm{N}_s^{12,11}$ because for each path from *p10* to *p11* or *p12*, it contains other start nodes *t10* or *t12*. (*p18, p19*) in Fig. 4 is a TP-inconsistent pair 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 $p1 \leftrightarrow p2$.

Figs. 1–4 show examples of SNC 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 they 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 *t5* but they 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 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 $n_s \in P$, $n_e \in P$, it is symmetrical.

Note that any pair of places (excluding n_s and n_e) in an AFOS (asymmetrical FOS) is also inconsistent. This leads [8] to an integrated algorithm to detect SNC and liveness for an arbitrary net.

In the knitting technique by [10], 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: Each new path involved in a synthesis step must be a TT- (from transition to transition) or a PP- (from place to place) path.

Each new path added is to create new FOS or to repair the AFOS (asymmetrical FOS) created earlier to have no AFOS at the end of the step.

The addition of a TT- or PP-handle H from n_1 to n_2 is a forward (backward) generation if $n_1 \rightarrow n_2$ ($n_1 \leftarrow n_2$). A backward generation (e.g., [Put1 Fork1 Tk1] in Fig. 5) creates a new circuit and tokens, modeling the availability of resources, must be added to a place in H (Rule TT.2). If only one new (multiple)

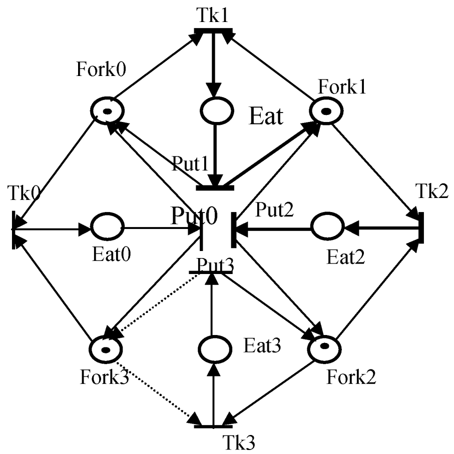

Fig. 5. PN model [14] of dining philosophers is not an SNC.

FOS is created, then it is a pure (interactive) TT- (Rule TT.1) or PP-generation (Rule PP.1). If inconsistent pair of places exist (*p1*, *p2*) where *p2* is on H and *p1* is on H_c, the complementary handle to H, create the missing bridge from H_c to H to meet *R1* or *R2* corresponding to Rule TT.4 or PP.3 respectively.

Examining the synthesis rules presented in [11], we find that each synthesized net is an SNC net. This is because both involve handles and bridges; the former are constructed by adding new handles and bridges while the latter are based on local structures of handles and bridges. This implies that we can construct *SNC componenst out of any net using the knitting technique* with polynomial time complexity [11].

IV. Reachability Analysis via Decomposition

Although HSNC does not cover all kinds of resource sharing, it can serve as the backbone of a net. We propose to find a maximum HSNC component (the solid lines in Fig. 5; all FOS symmetric and no SOS) such that the rest arcs (dashed lines in Fig. 5) or nodes, if included, would render the component no longer an HSNC. Upon this backbone guaranteed correct, we can merge resource nodes to complete the net where they are in some other HSNC components. This should save us time and space and the designed system is more reliable.

By adding a TP-path [Put3 Fork3] and a PT-path [Fork3 Tk3] to the resulting HSNC component N_1^c in Fig. 5, it forms the PN model of dining philosophers N with the same initial marking M_0. These two paths are in another HSNC component N_2^c which is an SM containing two circuits [Put3 Fork3 Tk3 Eat3 Put3] and [Put0 Fork3 Tk0 Eat0 Put0].

Their sets of reachable markings, however, are not the same. We can find $R(M_0)$ for N from $R(M_0)$ for N_1^c, by deleting the M where both Eat3 and Fork3 (also both Eat0 and Eat3) hold a token. Both Eat3 and Fork3 (also both Eat0 and Eat3) are part of a global concurrent set (Def. 10) of places in N_1^c; hence, there exists a reachable marking where they are both marked in N_1^c alone. But they are also in N_2^c. At M_0, it contains only one token, and stays so for any subsequent reachable marking. Hence, it is impossible for both Eat3 and Fork3 to hold a token in N_2^c.

The above case is a special case where there are only two HSNC components and one of them happens to be an SM. To generalize, we propose to simplify the Reachability Problem

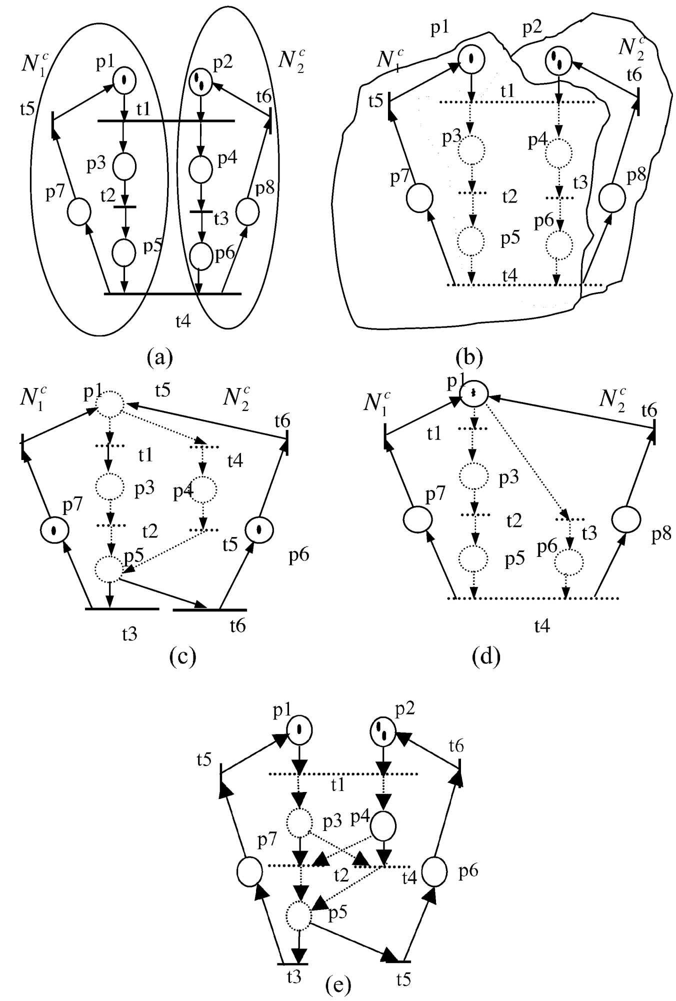

Fig. 6. RDP is not satisfied except for (b). (a) Shared module L does not contain all TT-paths from entry *t1* to exit *t4* while L in (b) does. (c) All directed paths (dashed) from entry to exit Γ are PP-paths. (d) Γ are PT-paths. (e) Γ are TP-paths.

by first decomposing N into a number of HSNC components N_1^c and checking whether each M_i^c is reachable in N_i^c. Such a property is called *reachability decomposition property* (RDP). This RDP does not hold for all decompositions.

In Fig. 6(a), $M_1 = [0\ 1\ 0\ 0] = [m(p1)\ m(p3)\ m(5)\ m(p7)]$ is reachable in N_1^c and $M_2 = [\ 0\ \ 2\ \ 0\ \ 0] = [m(p2)\ \ m(p4)\ \ m(p6)\ \ m(p8)]$ is reachable in N_2^c, but $[0\ 0\ 1\ 2\ 0\ 0\ 0\ 0]$ is not in N. Hence, RDP does not hold here. In Fig. 6(b)

$$\begin{aligned}M_1^c &= [0\ 1\ 1\ 0\ 0\ 0]\\ &= [m(p1)\ m(p3)\ m(p4)\ m(p5)\ m(p6)\ m(p7)]\end{aligned}$$

is reachable in N_1^c

$$\begin{aligned}M_2^c &= [1\ 1\ 1\ 0\ 0\ 0]\\ &= [m(p2)\ m(p3)\ m(p4)\ m(p5)\ m(p6)\ m(p8)]\end{aligned}$$

is reachable in N_2^c, and $M = [0\ 1\ 1\ 1\ 0\ 0\ 0\ 0]$ is reachable in N. Hence, RDP may hold here.

Let L be a shared module such that all its arcs and nodes are in both N^c. Then all paths from the entry to the exit must be

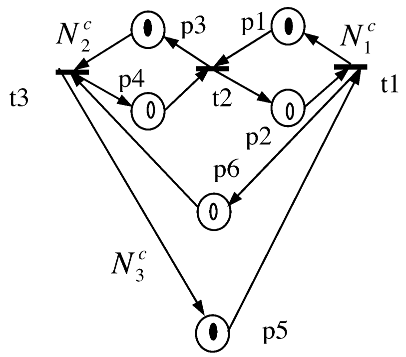

Fig. 7. Example of three interconnected MGs where the filled (empty) small circles correspond to M_0 (M_d).

TT-paths in order for RDP to hold. An entry (exit) node is a node in L with input (output) nodes not in L. Notice that the shared module L is similar to the transition subnet to be merged in [6], [7].

In Fig. 6(a), the shared module contains only the entry transition *t1* and the exit transition *t4* and it does not include all TT-paths from *t1* to *t4*. In Fig. 6(b), *t1* (*t4*) is an entry (exit) transition. And L is the subnet that contains all dashed arcs and nodes (all the TT-paths from *t1* to *t4*). M may be obtained from M_1^c and M_2^c by selecting all M_1^c and M_2^c where they have same projections from N_1^c and N_2^c onto L; i.e., $M_1^c(L) = M_2^c(L)$.

If the above TT-paths were PP-paths [Fig. 6(c)], then $M(L) = M_1^c(L) + M_2^c(L)$; hence there might not be any $M_1'^c$ such that $M_1'^c(L) = M(L)$. Similar arguments can apply to PT-paths [Fig. 6(d)]. For TP-paths [Fig. 6(e)], not all M may be obtained from M_1^c and M_2^c. This is because the token at the exit place (*p5*) goes to either N_1^c or N_2^c but not both.

Fig. 7 shows an example of three interconnected MGs where the filled (empty) small circles correspond to M_0 (M_d). $[m(p1) = 0\ m(p2) = 1]$ $[[(m(p3) = 0\ m(p4) = 1)$ and $[m(p5) = 0\ m(p6) = 1])]$ is reachable in module 1 (2 and 3) by the firing sequence $\sigma_1 = t2$ ($\sigma_2 = t3$ and $\sigma_3 = t1$). However

$$[m(p1) = 0\ m(p2) = 1\ m(p3) = 0 \\ m(p4) = 1\ m(p5) = 0\ m(p6) = 1]$$

is not reachable in N. N is an HSNC. This problem (even though all entry and exit are transitions) occurs because the three MG are connected in a circular way.

In order for $\sigma = \sigma_1\sigma_2$ to be legal, both σ_1 and σ_2 must include *t2*. This is obviously not true for $\sigma_2 = t3$ and we say N_1^c and N_2^c are *incompatible* [18]. To adjust, enlarge σ_2 to $\sigma_2 = t3\sigma_\nu$, where $\sigma_\nu = t2t3$ is the firing sequence of a T-invariant $\nu = [1\ 1]$. We say that N_1^c *turns* N_2^c *around* denoted $N_1^c \to N_2^c$. Similarly, N_2^c must turn N_3^c and N_3^c must turn $N_1^c (N_3^c \to N_1^c)$. Note that N_1^c turns N_3^c indirectly; hence $N_1^c \to N_3^c$. Thus, $N_1^c \to N_1^c$ and we say it is a *cyclic turn*; the turn goes on indefinitely without finding the firing sequence. Hence, M_d is not reachable.

However, decomposition into HSNC components without cyclic turns may not satisfy the constraint that both entry and exit be transitions. An example is shown in Fig. 8(a). We first find the maximum HSNC component N_1^c (in solid arcs and nodes). We then add TP-arc [*t2 p6*] and PT-arc [*p6 t1*] to form N. They are in another HSNC component N_2^c (in thickened arcs and nodes, and is an SM containing two circuits [*p6 t1 p2 t2 p6*] and [*p6 t3 p4 t4 p6*]). The decomposition should be such that all entry and exit transitions are transitions. Hence make $N_1'^c = N_1^c \backslash H$. $H = [t4\ p6\ t3]$. Is $M = [0\ 1\ 2\ 1\ 1\ 0]$ reachable? The shared module between $N_1'^c$ and N_2^c contains two TT-paths [*t1 p2 t2*] and [*t3 p4 t4*]. $M_1 = [0\ 1\ 2\ 1\ 1]$ ($[m(p1)\ m(p2)\ m(p3)\ m(p4)\ m(p5)]$), $M_2 = [1\ 1\ 0]$ ($[m(p2)\ m(p4)\ m(p6)]$). M_1 is reachable in $N_1'^c$, but M_2 is not reachable in N_2^c. Hence, M is not reachable. Notice that the shared module between N_1^c and N_2^c contains entry place *p6*.

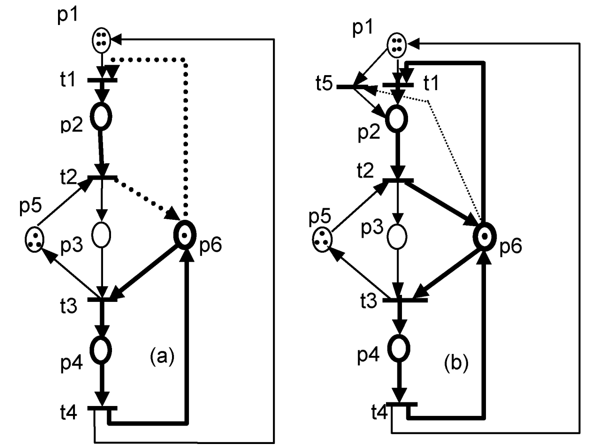

Fig. 8. (a) Example of complete decomposition [25]. (b) Without the dashed path [*p6 t1*], *p2* is an entry in L (two TT-paths [*t1 p2 t2*] and [*t3 p4 t4*]). The decomposition is not complete.

Definition 8: A decomposition is *complete* if: 1) all entry and exit nodes are transitions and 2) there are no cyclic turns.

In Fig. 8(b), without the dashed path [*p6 t5*], *p2* is an entry in L (two TT-paths [*t1 p2 t2*] and [*t3 p4 t4*]). Even though N_2^c (an SM containing two circuits [*p6 t1 p2 t2 p6*] and [*p6 t3 p4 t4 p6*]) is an HSNC, the net is unbounded. The decomposition is not complete because the entry node *p2* is not a transition. With the dashed path, we add [*t5 p2*] (*t5*) to L (the set of entry) and *p2* is no longer an entry, it is complete.

The verification of no "cyclic turns" is nontrivial since all reachable markings in each component must be checked. However, there are cases (e.g., components connected noncircularly) where checking is not necessary. The condition that components are connected in a circular way is necessary but not sufficient for being complete. Despite that $N_1'^c$ and N_2^c in Fig. 8(a) are connected circularly, it is complete—no longer so if there were places in the arcs connecting the entry (*t1*) and exit (*t2*) to the marked places *p1* and *p6*, respectively. The discussion holds true in general FMS applications. For most FMS applications, there is no need to verify the completeness of the decomposition.

Theorem 1: RDP holds if the *decomposition* is *complete*.

Proof: We prove the case of two N^c and only one common module L. The case of multiple N^c and shared modules can be proved similarly. Let the firing sequence to reach M_1^c and M_2^c be σ_1 and σ_2 respectively. If neither σ_1 nor σ_2 contains T_c (a sequence of transitions in L), then $\sigma_1\sigma_2$ is *legal* (i.e., when the firing sequence reaches a transition t in $\sigma_1\sigma_2$, t is firable). If exactly one of them, say σ_1, contains T_c, then $\sigma_2\sigma_\nu$ is also legal and contains T_c (where $\nu > 0$ is a T-invariant, σ_ν the corresponding firing sequence, and by Theorem 9 in [11], any synthesized net, an HSNC, is consistent and reversible). Hence, we need only consider the case where both σ_1 and σ_2 contain T_c. Let $\sigma_1 = T_1 T_c T_2$ and $\sigma_1 = T_1' T_c T_2'$ where T_i or T_i', $i = 1$ *or 2* stands for a sequence of transitions in N_i^c. Because all entry and exit nodes must be transitions, the firing sequence (e.g., $T_1 T_1' T_c T_2 T_2'$) is legal in N. Thus M is reachable in N. In the case of multiple N^c and shared modules, as long as there are no cyclic turns, we can perform the above operation of adding σ_ν repeatedly to make all pairs of N_i^c and N_j^c compatible in $O(l^2)$ time where l is the total number of shared modules. ∎

Note that the presence of the T-invariant ν decouples interconnected components and is the basis of decomposition. Otherwise, if ν does not exist, the RDP property may not hold. As indicated in [20], even as simple as an MG, the legal firing sequence problem is neither trivial nor intuitive. In general, each circuit can be replaced [19] by a simple HSNC where any shortest firing sequence from M_0 to M_d does not include any σ_ν.

Similar to the decomposition of markings, it seems that we can also find firing sequences via decomposition. Again, the decomposition holds only if T-invariant exists.

Now we have the following.

Algorithm I: The Reachability Algorithm for a Non-HSNC
Net N: Given M_0, is M reachable in N?
1) Break N into a number of maximum HSNC components N_i^c.
2) Verify that the decomposition is complete (normally true for FMS).
3) Break M into a number of M_i corresponding to N_i^c.
4) Verify each M_i is reachable in N_i^c. If it is "yes", then M is reachable in N. Otherwise it is not.

Remarks: Recall earlier that we can construct *an HSNC component out of any net using the knitting technique* with polynomial time complexity [11]. For most FMS applications, there is no need to verify the completeness of the decomposition. Steps 3 and 4 take polynomial computation time. Thus it takes polynomial time for the reachability problem. However, it is exponential in general. This is because some nets may not be decomposable as demonstrated earlier in Fig. 6(c)–(e). This raises two research issues: 1) How do we detect if a net is possible to be decomposed? 2) If possible, how do we do the decomposition? We start from an initially marked place and apply the knitting rules in [10] to obtain an HSNC component N_1^c where any marked place are in an elementary circuit. We then select a place, preferably initially marked, not in N_1^c and repeat the process to get another N_2^c. We continue this until all places are in some N_i^c. It is easy to check whether all entry and exit nodes are transitions.

Thus, the reachability problem for the subclass of nets that have a complete decomposition into a number of HSNC components can be solved in polynomial time.

For FMS applications, each WP (after deleting the part related to resources) forms naturally a component. The component for each resource normally consists of a number of circuits. Both are easily visualized. Both entry and exit are normally transitions. Thus, the decomposition is complete.

Alternatively, we can apply the technique to synthesize a system. We first design SNC components. Upon this backbone guaranteed correct, we can merge resource nodes to complete the net. This should save us time and space and the designed system is more reliable.

The semiconductor ERCN example in [7, Fig. 4] can be decomposed into 1) a backbone plus 2) four resource components ([7, Fig. 3]). 1) can be synthesized with steps: a) a circuit $[p_{r1}\ t1\ p1\ t2\ p2\ t3\ p3\ t4\ p4\ t5\ p5\ t6\ p10\ t10\ p_{r1}]$, b) a pure PP-path $[p1\ t15\ p13\ t16\ p_{r1}]$, c) a pure TT-path [*t2 p6 t7 p7 t8 p8 t9 p9 t6*], d) a pure backward PP-path [*p8 t13 p11 t12 p7*], e) a pure backward PP-path [*p10 t14 p12 t11 p1*]. Each resource component can be synthesized similarly except with no TT-path.

Esparza [13] indicated that the reachability for even an LSFC is NP-complete contradicting some earlier polynomial results. The LSFC in [13] modeling the 3-SAT problem is not an HSNC and not reversible; hence, it is impossible to find the firing sequence via decomposition. And therefore the problem cannot be polynomial and is NP-hard. Note that any live HSNC is always reversible and consistent [11]. If the net is initially marked to be reversible (hence the existence of T-invariant ν), then we can find firing sequences via decomposition. Thus, the reachabilty problem for both life and bounded FC (LBFC) and SNC, initially marked to be reversible, becomes polynomial instead of NP-complete! This is consistent with the result by Desel and Esparza [21] where reachability amounts to finding a nonnegative integral solution of the state equation for LBFC. In addition, our approach applies to a larger class of well-behaved SNC than LBFC.

V. Reachability of HSNC

Reachability has been determined as a P-SPACE hard problem because it is marking and behavior related. This, however, no longer holds true for SNC [19] since we can reduce the problem to a structural one. 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 of 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.

$p1|p2$, i.e., they are *exclusive* (*EX*) to each other if $\neg(p1 \leftrightarrow p2)$ and $\exists {n_s}^{1,2} \in P$.

$p1\|p2$, 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 of two places (*p1* and *p2*) in a live and safe SNC is: during the process of producing one product:

$p1 \leftrightarrow_T p2$, if *p1* is marked before *p2*;
$p1|_T p2$, if not $p1 \leftrightarrow p2$ and *p1* and *p2* never get marked at the same time;
$p1\|_T p2$, if *p1* and *p2* may get marked at the same time.

Def. 9 may be applied to transitions in a similar fashion. Thus, two transitions may be mutually concurrent, exclusive or sequential. For any pair of places in an HSNC, 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 they 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 be exactly one of the three relationships in Def. 9 and 2) structural and temporal relationships are equivalent.

Definition 10: A *GCN* or *G* is a maximal set of mutually concurrent places, i.e., $GCN = \{q|\forall\, r \in GCN, r = q\, or\, r\|q\}$. 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 $\exists p2 \in G_2$, $p1 \rightarrow p2$. G_2 is *sequentially next to* G_1, denoted as $G_1 o\!\rightarrow G_2$, if $\forall\ p1 \in G_1$, either $p1 \in G_2$ or $(p1\bullet)\bullet \in G_2$.

In Fig. 1, $G_1 = \{p2\}$, $G_2 = \{p3, p4\}$ and $G_1 o\!\rightarrow G_2$.

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

In [19], we show that for an HSNC, if $P(M_0)$ is a GCN, so is $P(M)$ for any reachable marking M and it is live. In Fig. 3(a) $\{p13, p14\}$ $(= P(M_0)$, a PT-inconsistent pair) is marked in M_0 and is not a GCN, yet the net is live and safe (but not reversible). 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 they are n_e*-concurrent* and a n_e*-GCN*; while the one in Def. 9 is n_s*-concurrent* and a n_s*-GCN*. And 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 apply directly to ISNC.* Note that the M_0 in Fig. 3(a) is not reachable from any $M \in R(M_0)$. In order for a $\mu(G)$ to be reachable, G must be n_s*-concurrent*.

The polynomial result seems to be in contradiction to the NP-complete one of even a life and safe FC in [13] and the NP-hard problem for even acyclic marked graphs with capacity constraints in [5]. However, the example in [13] involves a non-HSNC (it is not reversible) which is the sole reason for being NP-complete. More on this appeared in the paragraph after Theorem 1. Even with SNC, we have to check all possible firing sequences that satisfy the capacity constraints to see if a marking M is reachable. Thus, the time complexity becomes exponential as shown in [5].

The reachability problem for PNs can be solved by solving a set of linear state equations: $M_d = M_0 + Ax$ to decide if a marking M_d (or a state) is reachable from initial marking or state M_0 after firing each transition x_i times where A is the incidence matrix and x the firing vector. Most techniques need to solve a set of such linear equations. The solution x, however, may not be a legal firing vector (i.e., the existence of spurious solutions) except for special classes of nets such as LSFC (live and safe free choice) nets and RLBFC (reversible live and bounded choice) nets as shown in [21] where some useful reachability criteria for FC have been found.

The *firing count subnet* $N_x = (S_x, T_x, F_x)$ with respect to $x \geq 0$ [12], is defined as: $T_x = \{t_i, |t_i \in T, x_i > 0, i = 1, 2, \ldots, m\}$, $S_x = \bullet T_x \cup T_x \bullet$, F_x is the set of arcs between T_x and S_x (i.e., $F_x = (T_x \times S_s) \cup (S_x \times T_x)$). We denote the initial marking for $N_x = (S_x, T_x, F_x)$ by M_{0x} which is defined as the subvector of M_o for S_x. Similarly we define the destination marking M_x for N_x. Examples of N_x are shown in Figs. 3(a) and 4 as bold parts.

If a trap $\tau(\{p9, p10, p11, p12\})$ in Fig. 3(a) is initially marked in M_0 (where only *p9* holds a token indicated by an unfilled small circle), so will it be in M_d. Thus, if τ is empty of tokens in M_d (where only *p13* and *p14* hold a token indicated by filled small circles), then M_d is not reachable. Also if there is a siphon $(\mathrm{D} = \{p15, p16, p17, p20\}$ in Fig. 4) empty of tokens in N_x under M_0 (where only *p18* and *p19* hold a token indicated by unfilled small circles), then all output transitions of places in D cannot be fired to reach M_d (where only *p15* holds a token indicated by a filled small circle). Thus, the O-subnets of all empty D should be deleted from N. In a dual fashion, the I-subnets of all empty τ under M_d should be deleted from N. This comes from the fact that M_0 is reachable from M_d in N^{-1} *iff* M_d is reachable from M_0 in N and a siphon (trap) in N becomes a siphon (trap) in its reverse net N^{-1} [12]. There are no other deletions possible and the irreversibility occurs only when N has inconsistent pair of places. Hence, we have the following theorems from Lee *et al.* [12]:

Modified reachability theorem of NOT-net (no TP-handles to all strongly connected state machines, denoted SG—see Fig. 1): Let N be an NOT-net. For the given initial and destination markings M_0 and M_d, M_d is reachable from M_0 if and only if there exists a nonnegative integral solution x^* of the matrix state equation for (N^*, M_0^*) and M_d^*, where $N^* = (P^*, T^*, F^*)$ is a subnet of N such that I-subnets generated by all token-free traps in (N, M_d) are deleted from N, and M_0^*, M_d^* are restriction of M_0, M_d to P^* respectively.

Dual reachability theorem: Let N be an NOP-net (no PT-handles to all SG. See Fig. 2). M_d is reachable from M_0 if and only if there exists a nonnegative integral solution x^* of the matrix state equation for (N^*, M_0^*) and M_d^*, where $N^* = (P^*, T^*, F^*)$ is a subnet of N such that O-subnets generated by all token-free siphons in (N, M_0) are deleted from N, and M_0^*, M_d^* are restriction of M_0, M_d to P^* respectively.

Both TP-SOS (PT-SOS) and TP-FOS (PT-FOS) have TP-handles (PT-handles). There are neither TP-SOS (PT-SOS) nor TP-FOS (PT-FOS) in NOT (NOP)-nets. Recall that bad siphons (traps) are associated with TP-inconsistent (PT-inconsistent) pair of places in a TP-SOS (PT-SOS) of an SNC. But bad siphons are (not) associated with PT-inconsistent (TP-inconsistent) pair of places in a PT-FOS (TP-FOS). Both TP-SOS (PT-SOS) and TP-FOS (PT-FOS) may (not) appear in an NOP-net. But only TP-SOS (PT-SOS) in an NOP-net (NOT-net) will induce bad siphons (traps). Thus, in an NOT-net (NOP-net) there is no bad siphon (trap) and we do not need to delete the O-subnet (I-subnet) of all bad siphons (traps) in the above modified reachability theorem (dual reachability theorem).

Note that an SNC may be neither an NOP-net nor an NOT-net when it contains both PT- and TP-SOS. Both NOP-net and NOT-net are subclasses of FCs, but they may not be SNC since they may contain TP- or PT-FOS. Reachability problems for MGs, conflict-free nets [3], trap-circuit nets and deadlock-circuit nets [15] are subclasses [12] of NOP-nets and NOT-nets.

Note that for an NOT-net, it is impossible (possible) to reach $\mu(G_1)$ ($\mu(G_2)$) from $\mu(G_2)$ ($\mu(G_1)$), where G_1 ($= \{p13, p14\}$ in Fig. 3(a) containing PT-inconsistent pairs of places) is a N_e-*GCN* but not a N_s-*GCN* and G_2 ($= \{p10\}$) is a N_s-*GCN*. This offers a simpler way to check reachability.

For an NOP-net, if TP-inconsistent pairs of places in TP-SOS exist in n_s-*GCN*, then there are empty siphons (Fig. 4). With $M_0 = \mu(G_1)$ ($G_1 = \{p18, p19\}$ in Fig. 4), subsequent firings will eventually reach a deadlock state. Any subsequent $\mu(G_2)$ (e.g., $G_2 = \{p20\}$) is not reachable; i.e., if $N_e(G_1) \to G_2$ or $N_e(G_1) = G_2$, then $\mu(G_2)$ is not reachable where $N_e(G_1)$ is the set of n_e of all *(p1, p2)* in G_1 and hence is a n_e-*GCN* and $\forall n \in N_e(G)$, $n \in N_e(p_1, p_2)$, $p_1 \in G$, $p_2 \in G$. Other cases can also be derived easily and is a subject of further research.

Our approach posts the advantages: no need to solve for firing vector x. Also ,we do not have to: 1) verify the net is NOT (NOP) net or 2) find all empty siphons (traps) and their O-subnets (I-subnets).

Although we have assumed strongly connected nets, the same idea can apply to a net that is a nonstrongly connected subnet of an SNC. Alternatively, we may extend the method in [22] where an algorithm for reachability analysis in PNs having no transition invariants (T-invariants) is proposed.

Matsumoto [23] unified all the causes for spurious solutions by each maximal-strongly-connected siphon and trap subnet through the decomposition method. He proposed a similar method by decomposing an arbitrary Petri net (rather than our HSNC and specific non-SNC) into a number of maximal-strongly-connected and acylic subnets and determining the reachability of each such subnet.

However, most PN models for FMS are strongly connected and the time complexity is still exponential. Hence the effort for decomposition is wasted. On the other hand, our method cannot apply to nonstrongly connected nets. Thus, these two techniques complement each other and ours may simplify the analysis of each component.

Process algebra offers an alternative approach to modeling discrete event systems. Both Petri Net and process algebra have a precise mathematical definition. PN is a graphical modeling (easier to grasp) technique, while process algebra is a purely symbolic formalism and relative ease of manipulation plus rich abstraction capabilities [24]. Both can model dynamic behavior; however, the latter usually does not explicitly show system states; it proves system properties by showing the equality of behavioral descriptions.

Basten [24] proposed a method to combine PNs and process algebra into a method supporting compositional design. In this method, PNs are used to model system components. Process algebra is used to specify and verify the behavior of these components. The method is compositional in the sense that the behavior of the entire system can be derived from the behavior of its components.

Thus, PNs and process algebra are complementary. However, process algebra cannot perform the invariant analysis employed in this paper and few, if any, FMS applications adopt the approach. Just as algebraic rules of commutation and association simplify expressions or systems of equations, our knitting rules can reduce a complicated net to a simpler one for analysis. Further research is needed to study how to analyze properties via algebraic symbolic manipulations based on the knitting rules.

VI. Application

Here we apply the above concept to the detection of deadlocks or nonlive transitions and/or the derivation of MCL.

We first check the liveness of each N_i^c, which is a polynomial problem. If all live, then we merge two N_i^c and find all minimal bad siphons D_m, and apply the marking condition by Lautenbach [25] to check liveness. We then add a third N_i^c to the above merged net and follow the same to check liveness. We continue this incremental process until the last N_i^c has been added and processed.

Example [26] [Fig. 8(a)]: Upon merging N_1^c and N_2^c, $D_m = \{p4,\ p5,\ p6\}$. The support of the minimum S-invariant covering D_m is $S_m = \{p2,\ p3,\ p4,\ p5,\ p6\}$. The total number of tokens inside the support is four and is a constant independent of the reachable markings. To make D_m empty (hence a deadlock), all four tokens should retreat to the complementary set $C_m = \{p2, p3\} (= S_m \backslash D_m)$ which are not in D_m. Since only *p2* is in N_2^c, and the number of tokens in which is a constant 1, we set $m(p2) = 1$. The three remaining tokens would go to *p3* and $m(p3) = 3$. Now $M_1 = [0\ 1\ 3\ 0\ 0]$ ($[m(p1)\ m(p2)\ m(p3)\ m(p4)\ m(p5)]$ is reachable in N_1^c. Note that the net suffers a deadlock at M_1. And that if $m_0(p1) \le 3$, then it is impossible to reach a marking where $m(p2) = 1$ and $m(p3) = 3$ in N_1^c, Hence, the MCL is $m_0(p1) \le 3$ and in general, it is live if $m_0(p1) < m_0(p5) + m_0(p6)$.

Note that the above C_m is also in the S-invariant whose support $\{$*p1, p2, p3, p4*$\}$ contains *p1*. It is called a controlling invariant ν in [25]. By controlling the number of tokens in ν, we may prevent the minimal siphon D_m from being completely unloaded. The D_m is said to be *invariant-controlled*.

Another Example [27]: The net (Fig. 9) can be considered to be two components (upper *II* and lower *I*) interconnected by the regulation circuit [*t1 p14 t2 p15 t1*] after the merge along [*t12 p12 t13 p13 t14*]. In order for *t12* to fire at least once, $M_0(p1)$ must be no less than 2. For the lower subnet *I*, $D_m = \{p6, p8, p9, p10\}$ and $\mathrm{C}_m = \{p4, p5, p7\}$. The MCL: $m_0(p1) < m_0(p6) + m_0(p8) + m_0(p10) = 5$. It takes five tokens in *p1* to empty D_m. For the upper subnet II, $D_m = \{p6', p8', p9', p10', p12, p13\}$ and $C_m = \{p4', p5', p7'\}$. The MCL: $m_0(p1) < m_0(p6') + m_0(p8') + m_0(p10') = 5$. Because *t1* and *t2* fire alternatively, when *p1* has nine tokens, the D_m for I will become empty first and the net is deadlocked. Hence the net is live as long as $8 \ge M_0(p1) \ge 2$ which is the same as

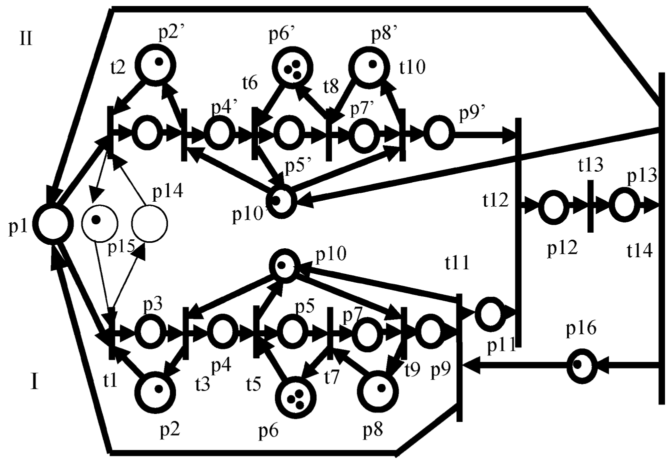

Fig. 9. Another example [26].

in [27]. This approach, however, has the disadvantages of the need to find D_m, whose number grows exponentially with the size of the net.

Alternatively, we propose to find reachable markings of Non-SNC to check liveness. We first find the maximum SNC component N_1^c (Fig. 8(a); all FOS are symmetric and no SOS). Upon the addition of the rest TP- and PT-arcs ([*t2 p6*] and [*p6 t1*]), we find N_2^c. Note that *p6* is an entry place. Hence, for the decomposition to be complete, we must delete *p6* and its incident arcs to get N_1^c. Because there is only one token in N_2^c, all reachable markings M_1 in N_1^c with both *p2* and *p4* marked must be eliminated (called *forbidden markings*). We then can search for nonlive transitions (all tokens in resource places (*p5* and *p6*) are used up) from the *next previous reachable markings* (NPRM) M_1' of M_1, i.e., $M_1'[t > M_1,\ t \in T$. We say M_1' is obtained from M_1 by rolling back t.

At M_1', the net must be not live so that the next marking M_1 is not reachable and must be eliminated. As an example, in Fig. 8(a), $M_1 = [0\ 1\ 2\ 1\ 1]$ (where both *p2* and *p4* are marked) is a marking to be eliminated (minimum among all possible forbidden cases where $m(p2) + m(p4) \leq 4$) where $m(p1) + m(p2) + m(p3) + m(p4) = m_0(p1)$ and both *p2* and *p4* are marked. One of its NPRM is $M_1' = [0\ 1\ 3\ 0\ 0]$ (by rolling back *t3*) and all transitions are not live in N (D_m is empty). Thus we do not have to search all reachable markings. We can further reduce the complexity of this search of all NPRM by noting that deadlock occurs when all resources are used up and they must be in a circuit [*p6 t3 p5 t2 p6*] since processes mutually waiting for them indefinitely. Thus, $m(p5) = 0$ which occurs by rolling back the firing of $t = t3$. This leads to M_1'.

Consider a net containing only *I* in Fig. 9. Suppose $m_0(p1) = 5$, without the TP-path: [*t5 p10*] and PT-path: [*p10 t3*], it is a maximum SNC component N_1^c. But they are also in N_2^c (an SM containing two circuits [*p10 t3 p4 t5 p10*] and [*p10 t9 p9 t11 p10*]). Hence, all reachable markings where both *p9* and *p4* are marked must be eliminated since there is only 1 token in N_2^c. Among all NPRM, we consider only those where the tokens in shared resource places are to be exhausted. *p10*, the shared resource, must be inside a D_m ({*p6, p8 p9 p10*}) since it is associated with the above TP- and PT- path. *p6* and *p8* (*p2* and *p16*), marked but not shared resource places, are in (outside) the region between *p9* and *p4* in N_2^c. Thus, they should be (not) considered and are in the circuit [*t5 p10 t9 p8 t7 p6 t5*].

$m_0(p10) + m_0(p6) + m_0(p8) = 5$ and $m_0(p1) = 5$. Thus, consider the eliminated $M = [0\ 1\ 0\ 1\ 3\ 0\ 0\ 1\ 1] = [m(p1), m(p2), \ldots, m(p9)]$ (ignoring $m(p10) - m(p13)$, $m(p16)$). Note that

$$m(p1) + m(p3) + m(p4) + m(p5) + m(p7) + m(p9) = m_0(p1)$$

and $m(p4) = m(p9) = 1$, which is impossible (possible) in N (N_1^c). Its next previous $M' = [0\ 1\ 0\ 1\ 3\ 0\ 1\ 0\ 0]$ by rolling back *t9* is reachable because M_1' and M_2' are reachable in N_1^c and N_2^c respectively and *p10, p6, p8* all have no tokens. The net is in a deadlock state at M'.

However, in general there is more than one forbidden marking to be considered. Computation time would be wasted trying each. We improve this as follows. Notice that the token at *p4* (*p2*) in Fig. 8(a) can (not) fire *t4* (*t2* since *t2* is dead) to return to *p6*. Thus, all tokens at *p6* must sit at *p2*. Similarly, all tokens at *p5* must sit at *p3*. We then check in N_1^c whether the marking is reachable. If it is, then it is not live. There is no need to find forbidden markings.

Similarly for the component *I* in Fig. 9, all three (one) tokens at *p6* (*p8*) must sit at *p5* (*p7*). The token at *p10* must sit at *p4* instead of *p9*. We then check whether the marking is reachable in N_1^c. Notice that using this technique, we can check liveness visually without resorting to tools.

For the ERCN example in [7, Fig. 4], there are only three HSNC components (for p_{r1}, p_{r2} and p_{r3} respectively) and the only possible deadlock is due to mutual waiting between p_{r2} and p_{r3} which are in circuit [p_{r2} *t8* p_{r3} *t4*] and token-free. If deadlock occurs, both *t8* and *t4* in the circuit are not live. Hence, all tokens in p_{r2} (p_{r3}) must sit at *p3* (*p7*) instead of *p4* (*p8*) where *t9* (*t5*) can fire to return tokens to p_{r2} (p_{r3}). There is a reachable marking in N_1^c where both *p7* and *p3* hold w tokens. If $w < a$ or $w < b$, then it is impossible to reach a marking where *p3* (*p7*) hold a (b) and the net is live consistent with the result in [7].

Note that we need not find D_m and C_m in the above illustration. It is quite efficient. Also if we perform the same for the dining philosopher example in Fig. 5, we would not be able to find nonlive transitions in the NPRM to the must-be-eliminated reachable markings in N_1^c where both Eat3 and Fork3 are marked.

VII. CONCLUSION

SNC covers well-behaved FC. We have extended the polynomial result for HSNC to those non-SNCs (covering many FMS applications and a larger class than HSNC). We have proposed to simplify the reachability problem by first decomposing N into a number of SNC components N^c and checking whether M_i^c is reachable in N_i^c for all i. Furthermore, we have applied the above concept to the detection of deadlocks or nonlive transitions and/or the derivation of the marking condition for liveness.

Further research should be directed to extending the polynomial results to ISNC, the subclass of nets that can be decomposed into SNC rather than merely HSNC, and nets that involve TP-FOS and PT-FOS.

ACKNOWLEDGMENT

The author would like to thank the anonymous referees for their helpful comments.

REFERENCES

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

[2] Y. Huang, "Modeling, Analysis, Deadlock Prevention and Cell Controller Implementation for Flexible Manufacturing Systems," Ph.D., National Taiwan Univ. of Sci. & Technol., Taipei, 2001.

[3] A. Ichikawa, K. Yokoyama, and S. Kurogi, "Control of event-driven systems – Reachability and control of conflict-free Petri nets," *Trans. SICE*, vol. 21, no. 4, pp. 324–330, 1985.

[4] J. Ezpeleta and L. Recale, "Deadlock avoidance approach for nonsequential resource allocation systems," *IEEE Trans. Syst,, Man, Cybern, A: Syst, Humans*, vol. 34, no. 1, pp. 93–101, Jan. 2004.

[5] E. Roszkowska, "Supervisory control for deadlock avoidance in compound processes," *IEEE Trans. Syst,. Man, Cybern. A, Syst., Humans*, vol. 34, pp. 52–64, Jan. 2004.

[6] M. D. Jeng and F. Dicesare, "Synthesis using resource control nets for modeling shared-resource systems," *IEEE Trans. Robot. Automat.*, vol. 11, no. 2, pp. 317–327, Apr. 1995.

[7] M. Jeng, X. Xie, and S. L. Chung, "ERCN* merged nets for modeling degraded behavior and parallel processes in semiconductor manufacturing systems," *IEEE Trans. Syst,. Man, Cybern. A, Syst., Humans*, vol. 34, no. 1, pp. 102–112, Jan. 2004.

[8] D. Y. Chao and J. A. Nicdao, "Liveness for synchronized choice Petri nets," *Comput. J.*, vol. 44, no. 1, pp. 124–136, 2001.

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

[10] D. Y. Chao and D. T. Wang, "Two theoretical and practical aspects of knitting technique – Invariants and a new class of Petri net," *IEEE Trans. Syst,. Man, Cybern. B, Cybern.*, vol. 27, pp. 962–977, 1997.

[11] D. Y. Chao, "A computer aided design technique for flexible manufacturing systems synthesis utilizing Petri nets," in *Computer-Aided Design, Engineering, and Manufacturing: Techniques and Applications*. Boca Raton, FL: CRC Press, 2001, vol. III, Operational Methods in Computer Aided Design, pp. 8.1–8.64.

[12] D. I. Lee, S. Kumagai, and S. Kodama, "Handles and reachability analysis of free choice nets," in *Application and Theory of Petri Nets*. New York: Springer-Verlag, 1995, vol. 935, LNCS, pp. 298–316.

[13] J. Esparza, "Reachability in live and safe free choice Petri nets is NP-complete," *Theor. Comp. Sci.*, vol. 198, no. 1–2, pp. 211–224, May 1998.

[14] Z. Bouziane, "A primitive recursive algorithm for the general Petri net reachability problem," in *Proc. 39th Annu. Symp. Foundations of Computer Science*, 1998, pp. 130–136.

[15] K. Hiraishi and A. Ichikawa, "A class of Petri nets that necessary and sufficient condition for reachability is a obtainable," *Trans. SICE, Jap.*, vol. 24, no. 6, pp. 635–640, 1988.

[16] K. Tsuji, "Useful necessary and sufficient condition for reachability of extended marked graphs," in *Proc. ISCAS '98*, vol. 3, pp. 330–333.

[17] L. Ferrarini, "On the reachability and reversibility problems in a class of Petri nets," *IEEE Trans. Syst,. Man, Cybern.*, vol. 24, pp. 1474–1482, Oct. 1994.

[18] D. Y. Chao, "Reachability and firing sequences of synchronized choice nets," *J. Inform. Sci. and Eng.*, vol. 21, pp. 129–152, 2005.

[19] ——, "Extended synchronized choice nets," *Comp. J.*, vol. 46, no. 5, pp. 505–523, 2003.

[20] J. Desel and J. Esparza, "Shortest paths in reachability graphs," in *Application and Theory of Petri Nets 1993*. Chicago, IL: Springer-Verlag, 1993, vol. 691, LNCS, pp. 224–241.

[21] ——, "Reachability in cyclic extended free choice Petri nets," *Theor. Comp. Sci.*, vol. 114, pp. 93–118, 1993.

[22] A. E. Kostin, "Reachability analysis in T-invariant-less Petri nets," *IEEE Trans. Automat. Contr.*, vol. 48, pp. 1019–1024, Jun. 2003.

[23] T. Matsumoto, Y. Miyano, and Y. Jiang, "Some useful sufficient criteria for the basic reachability problem in general Petri nets," in *Proc. 36th IEEE Conf. Decision and Control*, vol. 4, 1997, pp. 4104–4109.

[24] A. A. Basten, "In Terms of Nets: System Design with Petri Nets and Process Algebra," Ph.D., Eindhoven Univ. Technology, Eindhoven, The Netherlands, 1998.

[25] K. Lautenbach and H. Ridder, "Liveness in bounded Petri nets which are covered by T-invariants," in *Application and Theory of Petri Nets*, R. Valette, Ed. Zaragoza, Spain: Springer-Verlag, 1994, vol. 815, LNCS, pp. 358–378.

[26] M. C. Zhou and F. DiCesare, "Parallel and sequential mutual exclusions for Petri net modeling of manufacturing systems with shared resources," *IEEE Trans. Robot. Automat.*, vol. 7, no. 4, pp. 515–527, Aug. 1991.

[27] F. Chu and X. L. Xie, "Deadlock analysis of Petri nets using siphons and mathematical programming," *IEEE Trans. Robot. Automat.*, vol. 13, pp. 793–840, Dec. 1997.

Daniel Y. Chao (M'91–SM'04) received the Ph.D. degree in electrical engineering and computer science from the University of California, Berkeley, in 1987.

From 1987 to 1988, he was with Bell Laboratories. In 1988, he joined the Computer and Information Science Department, New Jersey Institute of Technology, Newark. In 1994, he joined the MIS Department, National Cheng Chi University, Taipei, Taiwan, R.O.C., as an Associate Professor. In February 1997, he was promoted to a Full Professor.

His research interests are in the application of PNs to the design and synthesis of communication protocols and the CAD implementation of a multifunction Petri net graphic tool. He has published 93 (including 24 journal) papers in the area of communication protocols, PNs , DQDB, networks, FMS, data flow graphs, and neural networks.