SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

July 2015, Vol. 58 072103:1–072103:16
doi: 10.1007/s11432-014-5259-5

Net-structure-based conditions to decide compatibility and weak compatibility for a class of inter-organizational workflow nets

LIU GuanJun[1,2*] & JIANG ChangJun[1,2]

[1]*Department of Computer Science, Tongji University, Shanghai* 201804*, China;*
[2]*Key Laboratory of the Ministry of Education for Embedded System and Service Computing, Tongji University, Shanghai* 201804*, China*

Received June 17, 2014; accepted November 10, 2014; published online January 30, 2015

Abstract Inter-organizational workflow nets (IWF-nets) can well model the interactions among multiple processes by sending/receiving messages. Compatibility and weak compatibility are crucial properties for IWF-nets. The latter guarantees that a system is deadlock-free and livelock-free while the former also guarantees that it has no dead tasks. Our previous work proved that the (weak) compatibility problem is PSPACE-complete for safe IWF-nets. This paper defines a class of IWF-nets in which some simple circuits are allowed. Necessary and sufficient conditions are presented to decide compatibility and weak compatibility for this class, and they are dependent on the net structures only. Algorithms are developed based on these conditions. In addition, we show that the traditional net structures like siphon cannot be easily used to decide the (weak) compatibility of IWF-nets.

Keywords formal method, workflow nets, compatibility, business process model, web services composition

Citation Liu G J, Jiang C J. Net-structure-based conditions to decide compatibility and weak compatibility for a class of inter-organizational workflow nets. Sci China Inf Sci, 2015, 58: 072103(16), doi: 10.1007/s11432-014-5259-5

1 Introduction

Petri nets are widely used to model concurrent/distributed systems because they can well characterize the processes of these systems and their relationships (e.g., sharing common resources or synchronous/asynchronous communication). For example, in flexible manufacturing systems [1–6], every product corresponds to one or several manufacturing processes. Every process uses a set of resources (like machines or robots) in a fixed order. These processes share common resources but not interact with each other. Another important application of Petri nets is to model and analyze such concurrent systems as web services, in which multiple parallel processes interact/collaborate via sending/receiving messages. Inter-organizational workflow nets (IWF-nets) [7–11], as a class of Petri nets, are used to model these concurrent systems. They can well characterize the system features, especially on synchronous and/or asynchronous communication. This paper focuses on IWF-nets.

* Corresponding author (email: liuguanjun@tongji.edu.cn)

https://engine.scichina.com/doi/10.1007/s11432-014-5259-5

© Science China Press and Springer-Verlag Berlin Heidelberg 2015

Compatibility [12,13] is a crucial property for IWF-nets. It guarantees that the target state can always be reached, no deadlock or livelock takes place, and each task has a (potential) right to execute. Weak compatibility [13] looses the requirements of compatibility, i.e., it only requires that a system can always terminate correctly for any run, but does not require each task to have a potential chance to execute. This property is permitted in web services composition, i.e., web services come from different owners and there is no need for some tasks/activities in some web services to take part in the whole run. In fact, the compatibility (resp. weak compatibility) of IWF-nets is equal to the soundness (resp. weak soundness) of WF-nets (i.e., workflow nets) [14]. In Section 2, it will be seen that WF-nets and IWF-nets may be viewed as two equivalent concepts.

Aalst et al. [9,13,15] proved that the (weak) soundness problem is decidable for general WF-nets. We also proved that the (weak) soundness problem is PSPACE-complete for bounded WF-nets [16]. Fortunately, Aalst et al. [13,17] gave a polynomial-time algorithm to solve the soundness problem for free-choice WF-nets. The algorithm is based on the rank theory proposed for free-choice nets by Desel and Esparza [18]. We also proved that weak soundness is equal to soundness for free-choice WF-nets [19]. Free-choice WF-nets can well model some basic structures of business processes such as AND-split and AND-join [13,17,20]. However, some concurrent systems [7,21–25] like web services composite must consider the interaction among different components/processes via sending/receiving messages, which makes the related models more and more complex so that free-choice WF-nets cannot model them. The general method to decide soundness/compatibility for these complex models is based on the reachability graph. There has not been too many other novel methods such as structure-based ones (except for free-choice WF-nets and well-structured ones proposed by Aalst et al. [17]).

Notice that for the bounded WF- and IWF-nets there must also exist the state space explosion problem. If the number of reachable states is polynomial in the size of WF- or IWF-nets, then the soundness/compatibility problem can be solved in polynomial time. This is impossible since PSPACE $\neq$ P. Fahland et al. [26] represented soundness as a temporal logic CTL and verified it by LoLA. LoLA [27] is a Petri net model checker that utilizes the state space to check properties (e.g., reachability) of a given Petri net. Fahland et al. [26] used LoLA to verify the soundness for lots of industrial business processes and their results showed that these business processes can be checked in a few milliseconds. The reason why these models can be checked in a short time is that all of the WF-nets modeling them are free-choice [26]. Therefore, it is important and interesting to look for other methods to decide soundness.

This paper defines a class of IWF-nets called *SCIWF-nets*. Necessary and sufficient conditions are proposed to decide compatibility and weak compatibility. These conditions are based on the net structures only. Furthermore, algorithms are developed on the basis of these conditions. SCIWF-nets can model many cases of interactions. To the best of our knowledge, we are the first to propose the net-structure-based conditions to solve the (weak) compatibility problem for a class of IWF-nets. Especially, it is shown in Section 6 that the traditional net-structure-based methods, such as siphon- or rank-theory-based ones [18,28–30], are hardly applicable to this problem. Note, we do not exclude the possibility of existing such methods, but we have not seen others give such a method so far. Therefore, the method proposed in this paper opens up a new possible way to explore the net-structure-based conditions of deciding the (weak) compatibility for more complex classes of IWF-nets.

The remainder of this paper is organized as follows. Section 2 introduces some basic terminologies. Section 3 defines SCIWF-nets. Section 4 proposes structure-based conditions to decide (weak) compatibility. Section 5 develops the related algorithms. Section 6 reviews some related research. Section 7 concludes this paper.

2 Petri nets and IWF-nets

Petri nets and IWF-nets are recalled in this section. For more details, one may refer to [13,31,32]. Denote $\mathbb{N} = \{0, 1, 2, \ldots\}$. Given $m \in \mathbb{N}$ and $m > 0$, denote $\mathbb{N}_m = \{1, 2, \ldots, m\}$.

https://engine.scichina.com/doi/10.1007/s11432-014-5259-5

Definition 1 (Net)**.** A *net* is a 3-tuple $N = (P, T, F)$, where P is a finite set of *places*, T a finite set of *transitions*, $F \subseteq (P \times T) \cup (T \times P)$ a set of *arcs*, and $P \cap T = \emptyset$.

A net may be seen as a directed bipartite graph. Generally, a transition is represented by a rectangle and a place by a circle in a net graph. A *path* of a net is a nonempty sequence $x_1x_2\cdots x_n$ of nodes such that $\forall j \in \mathbb{N}_{n-1}$: $(x_n, x_{n+1}) \in F$. A path $x_1x_2\cdots x_n$ is *elementary* if for any two nodes x_j and x_k of the path: $j \neq k \Rightarrow x_j \neq x_k$. An elementary path $x_1x_2\cdots x_n$ is a *circuit* if $(x_n, x_1) \in F$. A net is *acyclic* if it has no circuits. A net is *strongly connected* if for any two nodes x and y there is a path from x to y. $N' = (P', T', F')$ is a *subnet* of $N = (P, T, F)$ if $P' \subseteq P$, $T' \subseteq T$, and $F' = F \cap ((T' \times P') \cup (P' \times T'))$. Sometimes, we say that N *contains* N' if the latter is a subnet of the former.

A transition t is an *input transition* of a place p and p is an *output place* of t if $(t, p) \in F$. *Input place* and *output transition* can be defined accordingly. Given a net $N = (P, T, F)$ and a node $x \in P \cup T$, ${}^\bullet x = \{y \in P \cup T | (y, x) \in F\}$ and $x^\bullet = \{y \in P \cup T | (x, y) \in F\}$ are the *pre-set* and *post-set* of x, respectively. Two different transitions t_1 and t_2 are *in conflict* if ${}^\bullet t_1 \cap {}^\bullet t_2 \neq \emptyset$.

A *marking* of $N = (P, T, F)$ is a mapping M: $P \to \mathbb{N}$. A place $p \in P$ is *marked* at M if $M(p) > 0$. Notice, in this paper a marking is denoted as a multi-set of places. For example, the marking M such that place p_1 has one token, place p_3 has 6 tokens and other places have no tokens, is written as $M = p_1 + 6p_3$ or $M = \{p_1, 6p_3\}$. Transition t is *enabled* at M if $\forall p \in {}^\bullet t$: $M(p) > 0$, which is denoted as $M[t\rangle$. *Firing* an enabled transition t yields a new marking M', which is denoted as $M[t\rangle M'$, such that $M'(p) = M(p) - 1$ if $p \in {}^\bullet t \setminus t^\bullet$; $M'(p) = M(p) + 1$ if $p \in t^\bullet \setminus {}^\bullet t$; and $M'(p) = M(p)$ otherwise. A marking M_k is *reachable* from a marking M if there exists a transition sequence $\sigma = t_1t_2\cdots t_k$ such that $M[t_1\rangle M_1[t_2\rangle \cdots \rangle M_{k-1}[t_k\rangle M_k$. $M[\sigma\rangle M_k$ represents that M reaches M_k after firing sequence σ. The set of all markings reachable from M in a net N is denoted as $R(N, M)$. A net N with an *initial marking* M_0 is a *Petri net* or *net system* and denoted as (N, M_0).

A Petri net $(N, M_0) = (P, T, F, M_0)$ is *bounded* if $\exists k \in \mathbb{N}$, $\forall p \in P$, $\forall M \in R(N, M_0)$: $M(p) \leqslant k$. A Petri net is *safe* if each place has at most one token at each reachable marking. A net N is *structurally bounded* if (N, M_0) is bounded for any initial marking M_0. A transition t is *dead* at a marking M if $\forall M' \in R(N, M)$: $\neg M'[t\rangle$. A transition t is *live* at a marking M if for each $M' \in R(N, M)$, t is not dead at M'. (N, M_0) is *live* if each transition is live at M_0. A nonempty set S (resp. Q) of places is a *siphon* (resp. *trap*) if ${}^\bullet S \subseteq S^\bullet$ (resp. ${}^\bullet Q \supseteq Q^\bullet$). A net satisfies *ST-property* if each siphon contains a trap.

Given a net $N = (P, T, F)$, it is a *marked graph* if $\forall p \in P$: $|{}^\bullet p| = |p^\bullet| = 1$; it is a *free-choice net* if $\forall p_1, p_2 \in P$: $(p_1^\bullet \cap p_2^\bullet \neq \emptyset \wedge p_1 \neq p_2) \Rightarrow |p_1^\bullet| = |p_2^\bullet| = 1$; it is an *asymmetric-choice net* if $\forall p_1, p_2 \in P$: $p_1^\bullet \cap p_2^\bullet \neq \emptyset \Rightarrow (p_1^\bullet \subseteq p_2^\bullet \vee p_1^\bullet \supseteq p_2^\bullet)$.

Definition 2 (WF-net)**.** A net $N = (P, T, F)$ is a *WF-net* if

1. N has two special places i and o, where $i \in P$ is *source place* if ${}^\bullet i = \emptyset$ and $o \in P$ is *sink place* if $o^\bullet = \emptyset$; and
2. the *trivial extension* $N^E = (P, T \cup \{b\}, F \cup \{(b, i), (o, b)\})$ of N is strongly connected where $b \notin T$.

Definition 3 (Soundness of WF-net)**.** Let $N = (P, T, F)$ be a WF-net, $M_0 = i$, and $M_d = o$. N is *sound* if

1. $\forall M \in R(N, M_0)$: $M_d \in R(N, M)$;
2. $\forall M \in R(N, M_0)$: $M \geqslant M_d \Rightarrow M = M_d$; and
3. $\forall t \in T$, $\exists M \in R(N, M_0)$: $M[t\rangle$.

This definition was given in the early work of Aalst [12], and later he showed that the second requirement is implied by the first one [13]. The first two requirements mean that a system can always terminate correctly and the third one means that each task has a potential chance to be executed.

Generally, $M_0 = i$ and $M_d = o$ is called as the *initial* and *target* markings of a WF-net, respectively. Additionally, a safe (resp. bounded) WF-net means that the WF-net is safe (resp. bounded) at its initial marking. If the third requirement is removed from Definition 3, i.e., not each transition has a potential chance to enable, then *weak soundness* is defined.

Definition 4 (Weak soundness of WF-net)**.** Let $N = (P, T, F)$ be a WF-net, $M_0 = i$, and $M_d = o$. N is *weakly sound* if

1. $\forall M \in R(N, M_0)$: $M_d \in R(N, M)$; and
2. $\forall M \in R(N, M_0)$: $M \geqslant M_d \Rightarrow M = M_d$.

https://engine.scichina.com/doi/10.1007/s11432-014-5259-5

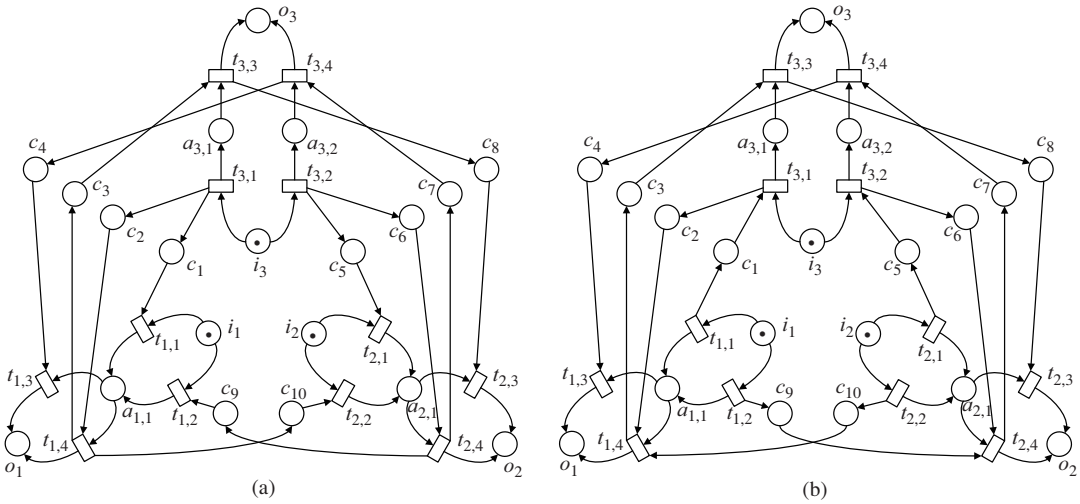

Figure 1 (a) A compatible IWF-net and (b) an incompatible IWF-net.

A class of nets called *inter-organizational workflow nets* (IWF-nets) [7] are often used to model the composition of web services, inter-organizational business processes, or some other concurrent systems in which multiple processes interact via sending/receiving messages. An IWF-net describes the synchronous and/or asynchronous communication among multiple partners (each partner is modeled by a *basic* WF-net) [7,30]. The following definition considers the asynchronous communication only.

Definition 5 (IWF-net)**.** A net $N = (N_1, \ldots, N_m, P_C, F_C)$ is an *IWF-net* if

1. $N_1 = (P_1, T_1, F_1)$, $\cdots$, and $N_m = (P_m, T_m, F_m)$ are pairwise disjoint WF-nets where $m \geqslant 1$ and they are called *basic*;
2. P_C is a finite set of *channel places* such that $P_C \cap P_j = \emptyset$ for each $j \in \mathbb{N}_m$;
3. $F_C \subseteq (P_C \times \bigcup_{j=1}^{m} T_j) \cup (\bigcup_{j=1}^{m} T_j \times P_C)$ is a set of arcs by which channel places connect with the m basic WF-nets; and
4. $\forall c \in P_C, \exists j, k \in \mathbb{N}_m$: $j \neq k \wedge {}^\bullet c \subseteq T_j \wedge c^\bullet \subseteq T_k \wedge {}^\bullet c \neq \emptyset \wedge c^\bullet \neq \emptyset$.

Figure 1 (a) and (b) are two IWF-nets whose basic WF-nets (see Figure 2 (a)–(c)) are identical but interactions are distinct. From the fourth item of Definition 5 it is known that each channel place is used only by two fixed basic WF-nets. In other words, two different basic WF-nets cannot send messages into the same channel place; similarly, two different basic WF-nets cannot take messages from the same channel place either. Certainly, two different basic WF-nets may use multiple channel places to communicate. Notice that Definition 1 uses a 3-tuple to represent a net while Definition 5 uses an $(m+2)$-tuple to represent an IWF-net. If an IWF-net is also represented by a 3-tuple (P, T, F), then $P = P_1 \cup \cdots \cup P_m \cup P_C$, $T = T_1 \cup \cdots \cup T_m$, and $F = F_1 \cup \cdots \cup F_m \cup F_C$. For convenience, an IWF-net is represented by a $(m+2)$-tuple. Additionally, $\forall j \in \mathbb{N}_m$, denote i_j and o_j as the source and sink places of N_j, respectively.

Definition 6 (Compatibility of IWF-net)**.** Let $N = (N_1, \ldots, N_m, P_C, F_C)$ be an IWF-net, $M_0 = i_1 + \cdots + i_m$, and $M_d = o_1 + \cdots + o_m$. N is *compatible* if

1. $\forall M \in R(N, M_0)$: $M_d \in R(N, M)$;
2. $\forall M \in R(N, M_0)$: $M \geqslant M_d \Rightarrow M = M_d$; and
3. $\forall t \in \bigcup_{j=1}^{m} T_j, \exists M \in R(N, M_0)$: $M[t\rangle$.

Definition 7 (Weak compatibility of IWF-net)**.** Let $N = (N_1, \ldots, N_m, P_C, F_C)$ be an IWF-net, $M_0 = i_1 + \cdots + i_m$, and $M_d = o_1 + \cdots + o_m$. N is *weakly compatible* if

1. $\forall M \in R(N, M_0)$: $M_d \in R(N, M)$; and
2. $\forall M \in R(N, M_0)$: $M \geqslant M_d \Rightarrow M = M_d$.

https://engine.scichina.com/doi/10.1007/s11432-014-5259-5

For instance, Figure 1(a) is compatible, but (b) is neither compatible nor weakly compatible. Notice that, if two special places i and o and two special transitions t_i and t_o are added to an IWF-net such that

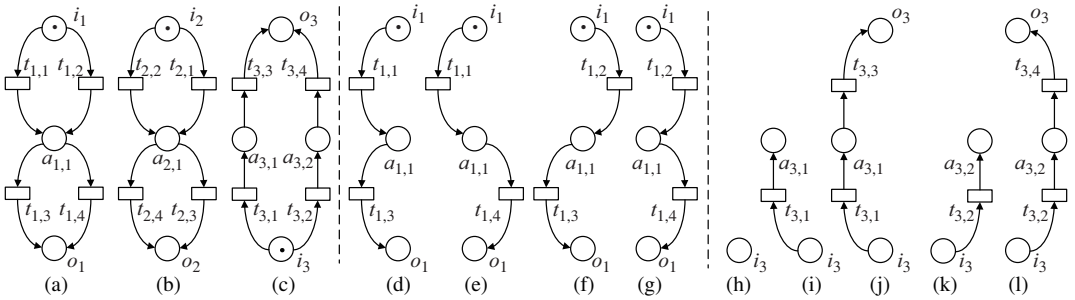

Figure 2 (a)–(c) Three basic FCWF-nets of the SCIWF-nets in Figure 1 (a) and (b); (d)–(g) four T-components of the FCWF-net in (a); (h)–(l) all caps of the FCWF-net in (c).

${}^\bullet t_i = \{i\} \wedge t_i^\bullet = \{i_1, \ldots, i_m\} \wedge {}^\bullet t_o = \{o_1, \ldots, o_m\} \wedge t_o^\bullet = \{o\}$, then the new net is a WF-net by Definition 2. Especially, the original IWF-net is compatible (resp. weakly compatible) if and only if the new WF-net is sound (resp. weakly sound). On the other hand, each IWF-net is composed of a group of basic WF-nets. Therefore, any WF-net is a special IWF-net since this IWF-nets contains only one basic WF-nets and has no channel place. Thus, the concepts of IWF-nets and WF-nets are equivalent.

For convenience, the *trivial extension* of an IWF-net is that a transition is added to the IWF-net such that the outputs of the transition are exactly all source places and the inputs of the transition exactly all sink places. M_0 and M_d in Definition 6 are called the *initial* and *target* markings of an IWF-net, respectively. If a WF-net is also a free-choice net, then it is called *free-choice WF-net* (FCWF-net for short). Similarly, *asymmetric-choice WF-net* (ACWF-net for short) can also be defined.

3 SCIWF-nets

Definition 8 (SCIWF-net)**.** $N = (N_1, \ldots, N_m, P_C, F_C)$ is a *simple circuit IWF-net* (SCIWF-net) if

1. N is an IWF-net;
2. $\forall j \in \mathbb{N}_m$: N_j is a sound acyclic FCWF-net; and
3. $\forall c \in P_C$: $|{}^\bullet c| = |c^\bullet| = 1$.

In an SCIWF-net, each basic WF-net is acyclic but the entire net may permit circuits. This is also the reason why this class is named *simple circuit IWF-nets*. In fact, Figure 1 (a) and (b) are two SCIWF-nets. The two SCIWF-nets have the same basic FCWF-nets as shown in Figure 2 (a)–(c), but their interactions are different. Each basic WF-net of an SCIWF-net is a free-choice net. FCWF-nets can not only model many basic structures of workflow such as AND-split, AND-join, OR-split, and OR-join, but also own a nice property (i.e., their soundness is decided in polynomial time [17]). Additionally, an SCIWF-net considers the simplest case of using a message channel, i.e., for a fixed channel place, a message is sent into it by firing a unique transition and the message is taken away from it by firing another unique transition. Certainly, a transition may use multiple channel places. Section 6 also shows that traditional methods such as siphon- or rank-theory-based ones are hard to be used for (weak) compatibility of SCIWF-nets. Therefore, it is valuable to explore efficient decision methods for (weak) compatibility of SCIWF-nets. We have given a novel one in the next section.

4 Conditions for compatibility and weak compatibility of SCIWF-nets

Some concepts related to the net structures are first defined.

Definition 9 (T-component of FCWF-net)**.** Let $N = (P, T, F)$ be an acyclic FCWF-net. Denote $i \in P$ and $o \in P$ as the source and sink places of N, respectively. Then $N' = (P', T', F')$ is a *T-component of* N if N' is a subnet of N such that:

https://engine.scichina.com/doi/10.1007/s11432-014-5259-5

1. $P' = {}^\bullet T' \cup T'^\bullet$, i.e., $\forall t \in T'$, its pre-set and post-set in N' are the same as its pre-set and post-set in N, respectively;

2. $i \in P' \wedge o \in P' \wedge |i^\bullet \cap T'| = |{}^\bullet o \cap T'| = 1$; and

3. $\forall p \in P' \setminus \{i, o\}$: $|p^\bullet \cap T'| = |{}^\bullet p \cap T'| = 1$.

Figure 2 (d)–(g) show all T-components of the FCWF-net in Figure 2(a). Notice that, due to $\forall t \in T'$: ${}^\bullet t \cup t^\bullet \subseteq P' \wedge \forall p \in P'$: $|{}^\bullet p| = |p^\bullet| = 1$, there always exists a path in N' from i to x for any node x of N'. Therefore, if a transition is added to a T-component such that its input is exactly the sink place and its output is exactly the source place, then a marked graph is produced. This means that the definition of T-component coincides with the traditional one in [18]. Therefore, a sound acyclic FCWF-net is covered by T-components according to the conclusion in [18]. Here, an FCWF-net is *covered* by T-components if for each transition there is a T-component containing it. In addition, a complete transition sequence of a sound acyclic FCWF-net corresponds to a T-component. Here, a transition sequence is *complete* if the marking $M_d = o$ reached from $M_0 = i$ by firing it. Aalst proved that a sound FCWF-net is safe [17]. Therefore, each transition of a T-component occurs only once and each place is marked only once when the corresponding complete transition sequence is fired. Notice that a T-component may correspond to multiple complete transition sequences due to the parallel structure.

Definition 10 (Cap of FCWF-net)**.** Let $N = (P, T, F)$ be an acyclic FCWF-net and i be its source place. Then $N' = (P', T', F')$ is a *cap of* N if N' is a subnet of N such that:

1. $P' = {}^\bullet T' \cup T'^\bullet$, i.e., $\forall t \in T'$, its pre-set and post-set in N' are the same as its pre-set and post-set in N, respectively;

2. $\forall p \in P'$: $|{}^\bullet p| \leqslant 1 \wedge |p^\bullet| \leqslant 1$ in N'; and

3. $\forall x \in T' \cup P'$, there is a path from i to x in N'.

Figure 2 (h)–(l) show all caps related to the T-component in Figure 2(c). Each T-component is also a cap that represents some complete transition sequences. The subnet only containing the source place is also viewed as a cap that reflects the empty transition sequence. In what follows, it is shown that for a sound acyclic FCWF-net each cap is a prefix of some T-component.

Lemma 1. A sound acyclic FCWF-net is covered by T-components and for each cap there exists a T-component such that the cap is a subnet of the T-component.

Proof. Let σ be a complete transition sequence, i.e., $i[\sigma\rangle o$. Because the FCWF-net is sound, it is safe for the initial marking $M_0 = i$ (see [17]). Because it is acyclic, each transition in σ occurs once only. Therefore, the transitions in σ and their pre-set and post-set directly form a T-component. Because the FCWF-net is sound, for each transition there exists a complete transition sequence containing it, thereby a T-component containing it. Therefore, the FCWF-net is covered by T-components.

For each cap we get an enabled transition sequence σ such that σ contains all transitions of the cap and each transition of the cap occurs in σ once and only once. Because the acyclic FCWF-net is sound and safe, there exists a complete transition sequence σ' such that σ is a prefix of σ'. Therefore, the cap is a subnet of the T-component that corresponds to σ'.

In fact, an acyclic FCWF-net is also sound if it is covered by T-components and for each cap there exists a T-component containing it. More generally, this conclusion is still true for SCIWF-nets (see Theorem 1). Next, *T-component* and *cap* are defined for SCIWF-nets, but Definitions 9 and 10 are not in haste referred to as their definitions. This is because that under such definitions a "T-component" or a "cap" may have a circuit that disables the related transition sequence (later, a detailed explanation will be given). Fortunately, they only need a simple constrain, i.e., no circuit is allowed.

Definition 11 (T-component of SCIWF-net)**.** Let $N = (N_1, \ldots, N_m, P_C, F_C)$ be an SCIWF-net. Then $N' = (N'_1, \ldots, N'_m, P'_C, F'_C)$ is a *T-component of* N if

1. $\forall j \in \mathbb{N}_m$, $N'_j = (P'_j, T'_j, F'_j)$ is a T-component of $N_j = (P_j, T_j, F_j)$ and called *basic*;

2. $P'_C = \bigcup_{j=1}^m ({}^\bullet T'_j \cup T'^\bullet_j) \cap P_C$, where ${}^\bullet T'_j$ and $T'^\bullet_j$ represent the pre-set and post-set of T'_j in N, respectively;

3. $F'_C = \bigcup_{j=1}^m ((P'_C \times T'_j) \cup (T'_j \times P'_C)) \cap F_C$;

https://engine.scichina.com/doi/10.1007/s11432-014-5259-5

4. $\forall c \in P'_C$: $|{}^\bullet c| = |c^\bullet| = 1$ in N'; and

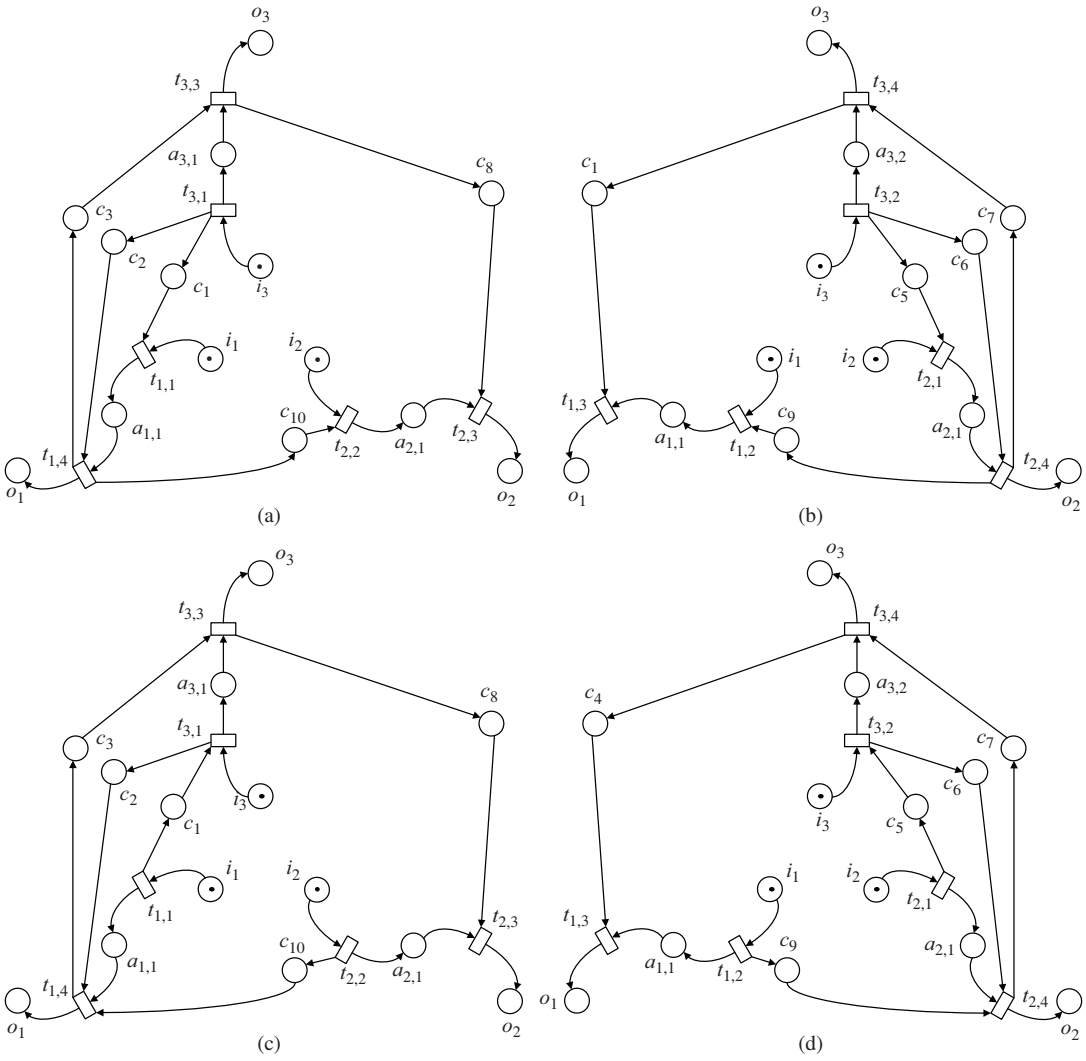

Figure 3 (a) and (b) T-components of the SIWF-net in Figure 1(a); (c) and (d) T-components of the SIWF-net in Figure 1(b).

5. N' is acyclic.

Definition 12 (Cap of SCIWF-net)**.** Let $N = (N_1, \ldots, N_m, P_C, F_C)$ be an SCIWF-net. Then $N' = (N'_1, \ldots, N'_m, P'_C, F'_C)$ is *a cap of* N if

1. $\forall j \in \mathbb{N}_m$, $N'_j = (P'_j, T'_j, F'_j)$ is cap of $N_j = (P_j, T_j, F_j)$;
2. $P'_C = \bigcup_{j=1}^{m} ({}^\bullet T'_j \cup T'^\bullet_j) \cap P_C$, where ${}^\bullet T'_j$ and $T'^\bullet_j$ represent the pre-set and post-set of T'_j in N, respectively;
3. $F'_C = \bigcup_{j=1}^{m} ((P'_C \times T'_j) \cup (T'_j \times P'_C)) \cap F_C$;
4. $\forall c \in P'_C$: $|c^\bullet| = 1 \Rightarrow |{}^\bullet c| = 1$ in N'; and
5. N' is acyclic.

The SCIWF-net in Figure 1(a) has only two T-components as shown in Figure 3 (a) and (b). The SCIWF-net in Figure 1(b) also has only two T-components as shown in Figure 3 (c) and (d). Figure 4(a) is not a cap of the SCIWF-net in Figure 1(a) since it has a circuit $c_9 t_{1,2} a_{1,1} t_{1,4} c_{10} t_{2,2} a_{2,1} t_{2,4}$. Figure 4(b) is not a cap of the SCIWF-net in Figure 1(a) since the channel place c_8 has no input in this subnet. Figure 4 (c)–(e) show the three caps of the SCIWF-net in Figure 1(b).

https://engine.scichina.com/doi/10.1007/s11432-014-5259-5

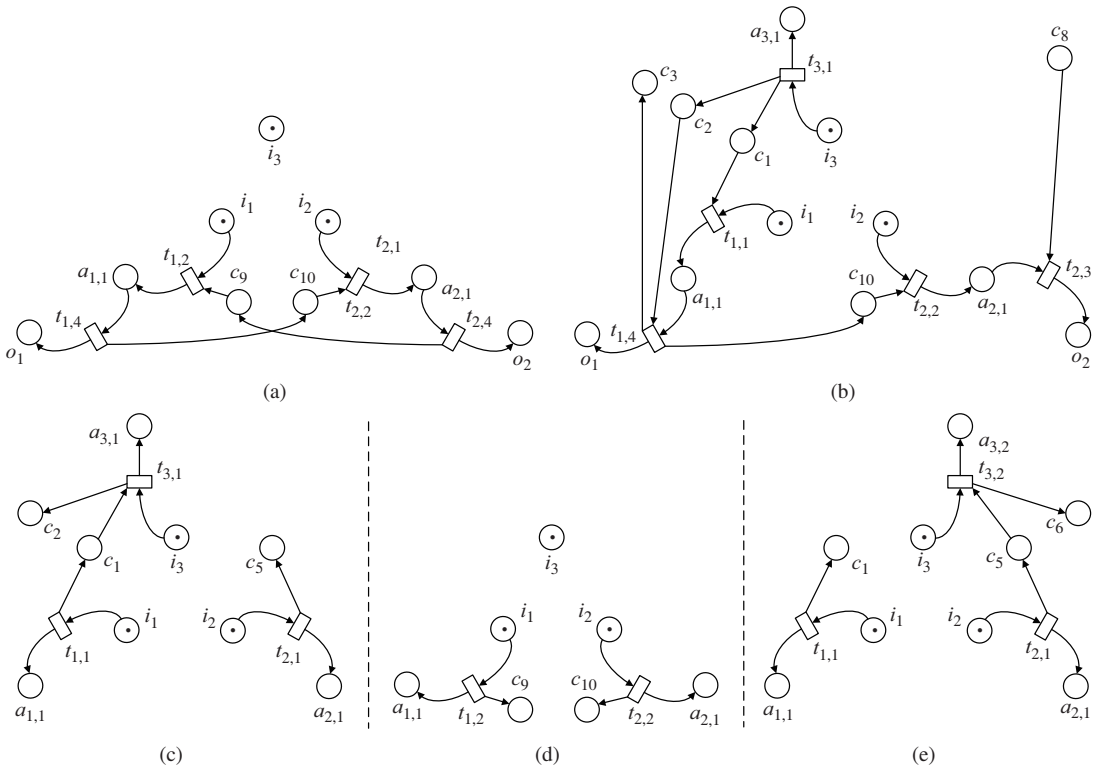

Figure 4 (a) and (b) Not caps of the SCIWF-net in Figure 1(a); (c)–(e) three caps of the SCIWF-net in Figure 1(b).

A T-component of an SCIWF-net includes only one T-component of each basic FCWF-net. Additionally, all channel places connected to those basic T-components are also in the T-component of the SCIWF-net and must satisfy the closure property (i.e., if the input transition of a channel place is in this T-component, then its output transition must also be in this T-component, and vice versa). The above examples show that not every basic T-component is in a T-component of the SCIWF-net. That is to say, some behaviors of some basic WF-nets are inhibited. For instance, the SCIWF-net in Figure 1(a) has no T-components containing the basic T-component in Figure 2(d).

If a transition is added to a T-component of an SCIWF-net such that its inputs are all sink places and its outputs are all source places, then the new net is a marked graph. This marked graph is strongly connected and each circuit contains a source place. Therefore, this marked graph is live and safe at the initial marking $M_0 = i_1 + \cdots + i_m$ (see [32]). Therefore, a T-component of an SCIWF-net corresponds to some complete transition sequences, and any complete transition sequence (i.e., it results in the target marking M_d) corresponds to a T-component. However, an incompatible SCIWF-net includes some incomplete transitions sequences (i.e., they do not lead to M_d). Therefore, caps are used to reflect all enabled transition sequences. Notice that each T-component of an SCIWF-net is also a cap of the SCIWF-net and each cap of an SCIWF-net contains only one cap of each basic FCWF-net.

Intuitively, all transitions of a circuit of an SCIWF-net cannot simultaneously participate in any run because each basic FCWF-net is acyclic and safe, which guarantees that any interaction among these FCWF-nets is not repeated. Formally, for each cap if a transition is added to it such that the input (resp. output) set of the transition is exactly those places having no outputs (resp. inputs) in the cap, then a marked graph is produced. If the cap has a circuit, then the circuit is not marked at the initial marking. Therefore, the related marked graph is not live [18,31,32]. This implies that no enabled transition sequence corresponds to the cap. Therefore, a cap/T-component must be acyclic (see Definitions 11 and 12). However, this does not mean that a compatible SCIWF-net cannot have a circuit. In fact, a compatible SCIWF-net may have circuits (Figure 1(a) shows such an example).

Next, it is proven that all caps of an SCIWF-net represent all possible runs.

Lemma 2. Let $N = (N_1, \ldots, N_m, P_C, F_C)$ be an SCIWF-net.

1. For each enabled transition sequence σ in $(N, i_1 + \cdots + i_m)$ (i.e., $(i_1 + \cdots + i_m)[\sigma\rangle$), there is a cap $N' = (N'_1, \ldots, N'_m, P'_C, F'_C)$ of N such that the cap is a subnet produced by the transitions in σ as well as their pre- and post-sets.

2. For each cap $N' = (N'_1, \ldots, N'_m, P'_C, F'_C)$ of N, there exists an enabled transition sequence σ in $(N, i_1 + \cdots + i_m)$ such that each transition in σ occurs once and only once in σ and the transitions in σ are just those in N'_j.

Proof. 1. Since σ is an enabled transition sequence in $(N, i_1 + \cdots + i_m)$, $\sigma \upharpoonright N_j$ is also an enabled transition sequence in (N_j, i_j) for each $j \in \mathbb{N}_m$, where $\sigma \upharpoonright N_j$ is the projection of σ over the transitions of N_j. By the proof of Lemma 1, we know that there is a cap N'_j of N_j corresponding to $\sigma \upharpoonright N_j$. Because each transition t in σ is fired, each channel place, which is an input of t, must be an output of some transition t' where t' is fired earlier than t in σ. Hence, the transitions of σ and their pre- and post-sets form a subnet of N and this subnet satisfies Definition 12.

2. Because the cap $N' = (N'_1, \ldots, N'_m, P'_C, F'_C)$ is acyclic and each place that has no input (i.e., the source places) is marked, there are enabled transitions in $(N', i_1 + \cdots + i_m)$. We select one of these enabled transitions to fire and then delete this fired transition as well as its input places. Then, the new Petri net is also acyclic and each place that has no inputs is also marked. By doing the above operations until the net has no transitions, we can get a transition sequence that satisfies the conditions of the lemma.

Definition 13 (Cover)**.** An SCIWF-net is *covered* by T-components if for each transition there is a T-component containing it.

Theorem 1. An SCIWF-net $N = (N_1, N_2, \ldots, N_m, P_C, F_C)$ is compatible if and only if

1. it is covered by T-components; and

2. for each cap there exists a T-component such that the cap is a subnet of the T-component.

Proof. ($\Rightarrow$) Let σ be a complete transition sequence, i.e., $(i_1 + \cdots + i_m)[\sigma\rangle(o_1 + \cdots + o_m)$. Then, $\forall j \in \mathbb{N}_m$, $\sigma \upharpoonright N_j$ is a complete transition sequence of (N_j, i_j), where $\sigma \upharpoonright N_j$ is the projection of σ over the transitions of N_j. By Lemma 1, we know that $\sigma \upharpoonright N_j$ corresponds a T-component of N_j. Additionally, all channel places have no tokens after firing σ because N is sound. Therefore, a channel place as an output of σ must be an input of σ, and vice versa. Therefore, all transitions of σ and their pre-set and post-set form a T-component of N by Definition 11. Because N is sound, for each transition there is a complete transition sequence containing it, thereby a T-component containing it. Therefore, N is covered by T-components.

By the second conclusion in Lemma 2, we know that for each cap $N' = (N'_1, \ldots, N'_m, P'_C, F'_C)$ of N, there exists an enabled transition sequence σ in $(N, i_1 + \cdots + i_m)$ such that the transitions in σ are exactly those in N'_j. Because N is sound, there is a complete transition sequence σ' such that σ is a prefix of σ'. By the first conclusion in Lemma 2, we can construct a cap N'' for σ'. Obviously, N' is a subnet of N'' and N'' is a T-component.

($\Leftarrow$) (by contradiction) We assume that N is unsound. Then, by Definition 6, we know that one of the following three cases must take place: Case (1) $\exists t \in T$, $\forall M \in R(N, i_1 + \cdots + i_m)$: $\neg M[t\rangle$; or Case (2) there is a marking $M \in R(N, i_1 + \cdots + i_m)$ that marks not only all sink places but also some channel places; or Case (3) there exists a marking $M \in R(N, i_1 + \cdots + i_m)$ at which not all sink places are marked but all transitions are dead.

Case (1) does not hold: Because N is covered by T-components, for each transition there is a T-component containing it. By Lemma 2 we know that for each T-component there is an enabled transition sequence that contains all transitions of the T-component. Therefore, $\forall t \in T$, $\exists M \in R(N, i_1 + \cdots + i_m)$: $M[t\rangle$.

Case (2) does not hold either: Let enabled transition sequence σ lead to a marking that marks all sink places as well as some channel places. By the first conclusion in Lemma 2 we know that there is a cap N' corresponding to σ. Because each sink place is marked after firing σ, N' contains one T-component of each basic FCWF-net and these marked channel places have no outputs in N'. By the given condition we know that for N' there must exist a T-component of N such that N' is a subnet of the T-component. To

https://engine.scichina.com/doi/10.1007/s11432-014-5259-5

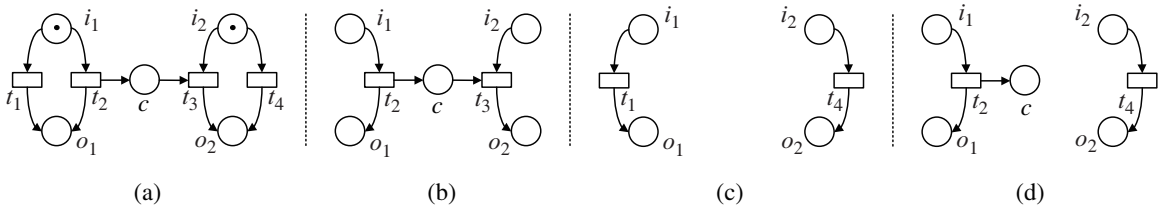

Figure 5 (a) An incompatible SCIWF-net; (b) and (c) two T-components; (d) a cap that is not a T-component.

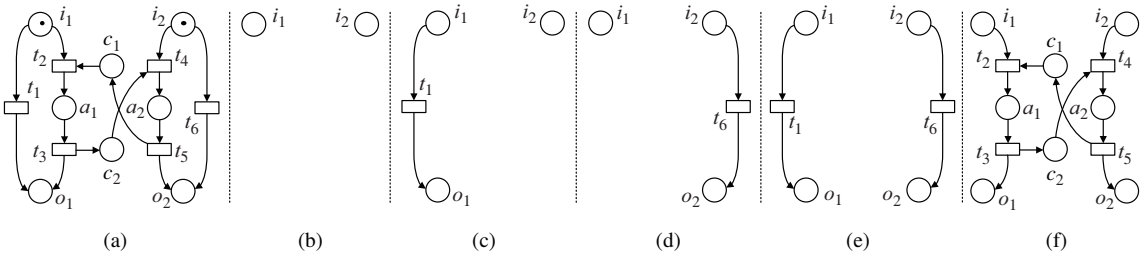

Figure 6 (a) A weakly compatible but incompatible SCIWF-net; (b)–(e) caps; (f) a subnet that is not a cap.

produce such a T-component N'', we must add some T-components of some basic FCWF-nets into N' in order to make those marked channel places to have output, which makes the required "T-component" N'' containing multiple basic T-components that are from the same basic FCWF-net. This does not satisfy the requirements of the definition of T-component, i.e., a T-component of an SCIWF-net contains one and only one basic T-component of each basic FCWF-net (see Definition 11).

Case (3) holds neither: Let σ be an enabled transition sequence leading to a marking at which some sink places are not marked but all transitions have been dead. Similar to the analysis of Case (2), we know that for the cap related to σ, there are no T-components containing it. If there is a T-component containing the cap, then there must exist an enable transition sequence σ' such that σ' corresponds to the T-component and σ is a prefix of σ'. This contradicts that all transitions are dead after firing σ.

Figure 1(a) is compatible. It is covered by T-components (see Figure 3 (a) and (b)) and each cap is a subnet of some T-component. Figure 1(b) is incompatible. Although it is covered by T-components (see Figure 3 (c) and (d)), the caps in Figure 4 (c)–(e) are not in any T-component. This example corresponds to Case (3) in the proof of Theorem 1. Figure 5(a) is incompatible either. Although it is covered by T-components (see Figure 5 (b) and (c)), there is a cap (see Figure 5(d)) that is not contained by any T-component. This example corresponds to Case (2) in the proof of Theorem 1. Figure 6(a) is incompatible since it is not covered by T-components. Although each of its caps (see Figure 6 (b)–(e)) is a subnet of its T-component (see Figure 6(e)), transitions t_2–t_5 do not belong to any T-component. This example corresponds to Case (1) in the proof of Theorem 1. Note that the net in Figure 6(f) is not a T-component of the SCIWF-net in Figure 6(a) because it has a circuit.

Definition 14 (Maximal cap of SCIWF-net)**.** A cap of an SCIWF-net is *maximal* if there are no other caps properly containing it.

Figure 4 (c)–(e) are three maximal caps of Figure 1(b). Figure 5 (b)–(d) are three maximal caps of Figure 5(a). Notice that each T-component is a maximal cap.

Corollary 1. An SCIWF-net is compatible if and only if

1. it is covered by T-components; and
2. each maximal cap is a T-component.

Proof. ($\Rightarrow$) By Theorem 1 we know that a compatible SCIWF-net is covered by T-components. If a maximal cap is not a T-component, then there are no T-components containing the maximal cap by the definition of maximal (see Definition 14), thereby making the SCIWF-net incompatible by Theorem 1. Therefore, each maximal cap is a T-component.

https://engine.scichina.com/doi/10.1007/s11432-014-5259-5

Algorithm 1 Deciding weak compatibility for SCIWF-nets

```
procedure main () {
N is a given SCIWF-net;
M_0 is the initial marking of N;
IsT-component (N, M_0);
}

procedure IsT-component (N, N') {
X := {p ∈ N' | p• = ∅ in N'};
Y := {t ∈ N | X[t⟩};
if X ≠ M_d ∧ Y ≠ ∅ then
   for each t ∈ Y do
      N'' := N' ∪ {t} ∪ t• ∪ (t × t•) ∪ (•t × t);
      IsT-component (N, N'');
   end for
else
   if X ≠ M_d ∧ Y = ∅ then
      output (N');
      exit (0);
   end if
end if
}
```

(⇐) For each cap, there is a maximal cap containing it. Therefore, for each cap there exists a T-component containing it. Therefore, the SCIWF-net is compatible by Theorem 1.

In fact, the case that for each cap there is a T-component containing it implies that the system can always terminate correctly, and the case that for each transition there is a T-component containing it means that each transition has a potential chance to enable. Therefore, a decision condition for weak compatibility is easily derived.

Theorem 2. An SCIWF-net is weakly compatible if and only if for each cap there exists a T-component such that the cap is a subnet of the T-component.

Corollary 2. An SCIWF-net is weakly compatible if and only if each maximal cap is a T-component.

Figure 6(a) is weakly compatible. It has exactly one T-component as shown in Figure 6(e), and each of its caps (see Figure 6 (b)–(e)) is a subnet of the T-component.

5 Algorithms for compatibility and weak compatibility of SCIWF-nets

5.1 Algorithm for weak compatibility

An algorithm is first given to decide weak compatibility (see Algorithm 1). Let $N = (N_1, \ldots, N_m, P_C, F_C)$ be an SCIWF-net, $M_0 = \{i_1, \ldots, i_m\}$ be the set of all source places, $M_d = \{o_1, \ldots, o_m\}$ be the set of all sink places, and N' be a cap of N. Notice, M_0 may be seen as the most basic cap.

Note that in procedure *IsT-component*, X means a reachable marking by firing all transitions in N' and Y records all transitions of N that are enabled at X.

$X \neq M_d \wedge Y = \emptyset$ represents that after firing all transitions in N' the system reaches a marking such that it is not the target marking and no transition is enabled at it. Therefore, $X \neq M_d \wedge Y = \emptyset$ means that the current cap is maximal but not a T-component, i.e., the SCIWF-net is not weakly compatible. In this case, therefore, the program outputs this counter-example and terminates early.

https://engine.scichina.com/doi/10.1007/s11432-014-5259-5

Algorithm 2 Deciding compatibility for SCIWF-nets

```
Flag := 1;
Trans := ∅;

procedure main () {
N is a given SCIWF-net;
M0 is the initial marking of N;
IsT-component-E (N, M0);
if Flag = 1 then
   if Trans = T then
      output ("N is compatible");
   else
      output ("N has a dead transition");
   end if
end if
}

procedure IsT-component-E (N, N') {
X := {p ∈ N'|p• = ∅ in N'};
Y := {t ∈ N|X[t⟩};
if X ≠ Md ∧ Y ≠ ∅ then
   for each t ∈ Y do
      Trans := Trans ∪ {t};
      N'' := N' ∪ {t} ∪ t• ∪ (t × t•) ∪ (•t × t);
      IsT-component-E (N, N'');
   end for
else
   if X ≠ Md ∧ Y = ∅ then
      Flag := 0;
      output (N');
      exit (0);
   end if
end if
}
```

$X \neq M_d \wedge Y \neq \emptyset$ means that N' is still not maximal. Hence, for each bigger cap (i.e., $N'' := N' \cup \{t\} \cup t^\bullet \cup (t \times t^\bullet) \cup ({}^\bullet t \times t)$) this procedure is recursively called to do the same decision. Note that $N' \cup \{t\} \cup t^\bullet \cup (t \times t^\bullet) \cup ({}^\bullet t \times t)$ means that the enabled transition t as well as all places and arcs related to it in N are added to N'.

$X = M_d$ means that the current procedure will end correctly and return the previous layer correctly.

Notice that if two transitions t_1 and t_2 in Y can be concurrently fired at X, then the procedure *IsT-component (N, N'')* will be called twice where N'' is the net generated by adding t_1 and t_2 as well as the related places and arcs to N'. To avoid those repeated calls, Y should be reduced, i.e., for those transitions that can be concurrently fired at X, only one is left in Y. This operation can be completed in polynomial time and is omitted here for the simplification of this algorithm.

When *IsT-component* is called in the main procedure, N' should initially be M_0 (i.e., it is the most basic cap). Therefore, (N, M_0) is weakly compatible if and only if *IsT-component (N, M0)* terminates correctly (i.e., no counter-example is outputted); (N, M_0) is not weakly compatible if and only if *IsT-*

https://engine.scichina.com/doi/10.1007/s11432-014-5259-5

component (N, M_0) outputs a counter-example.

When *IsT-component (N, M_0)* is executed, the worst case is that all maximal caps are T-components, i.e., each maximal cap is tested. The number of T-components grows exponentially for the following very special example: an SCIWF-net is composed of m basic FCWF-nets, each basic FCWF-net has k T-components, and there are no channel places. The SCIWF-net has k^m T-components. In practice, however, the number of maximal caps of an SCIWF-net does not increase like the above example. If the number n of maximal caps grows polynomially but not exponentially, then the number of recursive calls is polynomial, i.e., $O(|T| \times n)$. Next, an algorithm is proposed to decide compatibility.

5.2 Algorithm for compatibility

When *IsT-component (N, M_0)* is called in the main procedure, it can travel all caps of the SCIWF-net N (if each maximal cap is a T-component) or output a maximal cap (if there is a maximal cap that is not a T-component). However, how to decide if it is covered by T-components? In fact, we only need to add to the procedure two global variables $Flag$ and $Trans$ that are initially assigned 1 and $\emptyset$, respectively. When the current cap is checked to be maximal but not a T-component, $Flag$ is assigned 0, this maximal cap is output, and the recursive procedure terminates early. When the current cap is not maximal, we will choose an enabled transition to produce a bigger cap. Before recursively calling this procedure for this bigger cap, we add this related transition to $Trans$ (this means that there is a cap containing the transition). In the main procedure, after executing this recursive procedure, we will decide if the $Flag$ is 1 or not. If $Flag$ equals 1 (this means that each maximal cap is a T-component), then we continue to decide whether $Trans$ records all transitions. If $Trans$ exactly records all transitions, then we should output a message "N is compatible", otherwise, output "N has a dead transition" (i.e., incompatible). Algorithm 2 describes the related pseudo-code of deciding the compatibility for SCIWF-nets.

The decision conditions for (weak) compatibility of SCIWF-nets are only related to the net structures. Based on these conditions, the above recursive algorithms are developed.

6 Related work

Besides FCWF-nets, Aalst also proved that the soundness problem is solvable in polynomial time for well-structured WF-nets [17]. A WF-net is *well-structured* if and only if for any two nodes x and y of its trivial extension such that one of the nodes is a place and the other is a transition, there is always $\alpha(C_1) \cap \alpha(C_2) = \{x, y\} \Rightarrow C_1 = C_2$, where C_1 and C_2 are any two elementary paths from x to y and $\alpha(C)$ is the set of the nodes in C [17]. However, IWF-nets do not satisfy well-structured-ness because an IWF-net may have two different elementary paths that destroy the well-structured-ness (e.g., paths $i_3t_{3,2}c_6t_{2,4}$ and $i_3t_{3,1}c_2t_{1,4}c_{10}t_{2,2}a_{2,1}t_{2,4}$ in Figure 1(a), or paths $i_1t_{1,2}a_{1,1}t_{1,4}$ and $i_1t_{1,1}c_1t_{3,1}c_2t_{1,4}$ in Figure 1(a)).

Aalst utilized the results on extended non-self-controlling nets (ENSeC nets) in [33] to solve the soundness problem for well-structured WF-nets. He proved that the trivial extension of a well-structured WF-nets is elementary ENSeC and structurally bounded [17]. A net N is *elementary ENSeC* [33] if and only if for every couple (t_1, t_2) of transitions in conflict, there is no elementary path in $N\backslash{}^\bullet t_1$ leading from t_1 to t_2. However, the trivial extensions of the IWF-nets in Figures 1(a), 1(b), 5(a), and 6(a) are not elementary ENSeC. Next, it can be seen that the condition of deciding liveness for structurally bounded elementary ENSeC nets does not work for IWF-nets. Barkaoui et al. [33] proved that a structurally bounded elementary ENSeC net is structurally live if and only if each siphon is a trap. In fact, even for some compatible SCIWF-nets, there possibly exist some siphons that are not traps. Figure 1(a) shows such an example. The SCIWF-net is compatible and its trivial extension is structurally bounded, but the siphon $\{i_1, i_2, o_3, a_{1,1}, a_{2,1}, c_3, c_7, c_9, c_{10}\}$ of its trivial extension is not a trap.

Desel and Esparza [18] proposed the rank theory to decide the well-formedness for free-choice nets. Notice, a net is *well-formed* if and only if it is live and bounded at some marking [18]. Obviously, soundness is closely related to well-formedness because Aalst proved that a WF-net is sound if and only

https://engine.scichina.com/doi/10.1007/s11432-014-5259-5

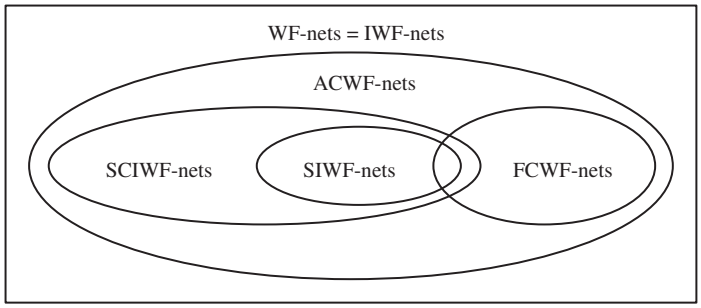

Figure 7 The relation of modeling power of SCIWF-, SIWF-, FCWF-, ACWF-, and WF-nets.

if its trivial extension is live and bounded at the initial marking [17]. Aalst also utilized the rank theory to solve the soundness problem for FCWF-nets [17]. Desel and Esparza extended the rank theory to general Petri nets and gave a sufficient condition to decide well-formedness, i.e., if a net fulfills (1) it is weakly connected, (2) it has a positive T-invariant, (3) it has a positive S-invariant, and (4) its rank plus 1 equals the number of its clusters, then the net is well-formed (see Theorem 10.16 in [18]). However, it is difficult to apply the rank theory to the compatibility of IWF-nets because the rank of an IWF-net is generally greater than or equal to the number of its clusters. For instance, the trivial extensions of IWF-nets in Figures 1(a), 1(b), and 6(a) all satisfy the first three conditions of the rank theory but do not satisfy the equation of rank and cluster.

SCIWF-nets are asymmetric-choice. For asymmetric-choice nets there are some methods to decide their liveness. For example, Best [28] proved that a marked asymmetric-choice net is live if and only if it is place-live (i.e., $\forall p \in P$, $\forall M \in R(N, M_0)$, $\exists M' \in R(N, M)$: $M'(p) > 0$), and Chu and Xie [34] proved that a marked asymmetric-choice net is live if and only if each siphon is marked at any reachable marking. These conditions need to factor in all reachable markings. Jiao et al. [35] further proved that an asymmetric-choice net such that ST-property (ST-AC-net) is well-formed if and only if it is structurally bounded, and a well-formed ST-AC-net is live and bounded for an initial marking if and only if each siphon is marked at the initial marking. If this conclusion is hoped to use for IWF-nets, the trivial extensions of these nets must satisfy ST-property. However, ST-property usually does not hold for them, even for compatible ones. For example, the SCIWF-net in Figure 1(a) is compatible but its trivial extension has a siphon $\{i_1, i_2, o_3, a_{1,1}, a_{2,1}, c_3, c_7, c_9, c_{10}\}$ that does not contain any trap.

These traditional structure-based methods are effective for the related net classes. However, the above analysis shows that it is not easy to apply them to SCIWF-nets.

7 Conclusion

This paper gives necessary and sufficient conditions to decide compatibility and weak compatibility for SCIWF-nets, and they depend on the net structures only. In [36] we defined Simple IWF-nets (SIWF-nets) that are a subclass of SCIWF-nets but do not allow circuits, and explored the condition of deciding their compatibility. Since SCIWF-nets only allow some simple cases of circuits, their modeling power is weaker compared to IWF-nets. The relation between SCIWF- and the famous FCWF-nets is that their intersection is not empty but they do not contain each other. Figure 7 shows the relationships among SCIWF-, SIWF-, FCWF-, ACWF-, and WF-nets. Aalst et al. also realized that the modeling power of FCWF-nets is limited and tried to explore the soundness for ACWF-nets [37]. However, they could not propose a universal net-structure-based condition [37]. This paper also shows that traditional net-structure-based concepts are hardly used for SCIWF- and IWF-nets. Therefore, the results of this paper add to the current knowledge in this area. Future works need to focus on some bigger classes of IWF-nets in which each basic WF-net may permit circuits.

https://engine.scichina.com/doi/10.1007/s11432-014-5259-5

Acknowledgements

This work was supported in part by Alexander von Humboldt Foundation and National Natural Science Foundation of China (Grant Nos. 61202016, 91218301). Authors would like to thank the reviewers whose comments improved the qualities of this paper.

References

1 Ezpeleta J, Colom J M, Martinez J. A Petri net based deadlock prevention policy for flexible manufacturing systems. IEEE Trans Robot Automat, 1995, 11: 173–184
2 Li Z W, Zhou M C. Elementary siphons of Petri nets and their application to deadlock prevention in flexible manufacturing systems. IEEE Trans Syst Man Cybern: Part A, 2004, 34: 38–51
3 Wang S G, Wang C, Zhou M C. Controllability conditions of resultant siphons in a class of Petri nets. IEEE Trans Syst Man Cybern: Part A, 2012, 42: 1206–1215
4 Wu N Q, Zhou M C. Deadlock resolution in automated manufacturing systems with robots. IEEE Trans Automat Sci Eng, 2007, 4: 474–480
5 Xue L, Hao Y. Autonomy-subnet based structural synthesis and liveness guarantying policy of Petri net model of flexible manufacturing system. Sci China-Ser F: Info Sci, 2004, 47: 273–286
6 Xing K Y, Zhou M C, Wang F, et al. Resource-transition circuits and siphons for deadlock control of automated manufacturing systems. IEEE Trans Syst Man Cybern: Part A, 2011, 41: 74–84
7 Aalst W M P. Interorganizational workflows: an approach based on message sequence charts and Petri nets. Syst Anal Model Simul, 1999, 34: 335–367
8 Aalst W M P, Mooij A J, Stahl C, et al. Service interaction: patterns, formalization, and analysis. Lect Notes Comput Sci, 2009, 5569: 42–88
9 van Hee K, Sidorova N, Voorhoeve M. Generalized soundness of workflow nets is decidable. Lect Notes Comput Sci, 2004, 3099: 197–216
10 Kim T H, Chang C K, Mitra S. Design of service-oriented systems using SODA. IEEE Trans Serv Comput, 2010, 3: 236–249
11 Kindler E. The ePNK: an extensible Petri net tool for PNML. Lect Notes Comput Sci, 2011, 6709: 318–327
12 Aalst W M P. The application of Petri nets to workflow management. J Circuit Syst Comp, 1998, 8: 21–66
13 Aalst W M P, van Hee K M, ter Hofstede A H M, et al. Soundness of workflow nets: classification, decidability, and analysis. Form Asp of Comput, 2011, 23: 333–363
14 Liu G J, Jiang C J, Zhou M C, et al. Interactive Petri nets. IEEE Trans Syst Man Cybern: Syst, 2013, 43: 291–302
15 Tiplea F L, Marinescu D C. Structural soundness of workflow nets is decidable. Inform Process Lett, 2005, 96: 54–58
16 Liu G J, Sun J, Liu Y, et al. Complexity of the soundness problem of workflow nets. Fundam Inform, 2014, 131: 81–101
17 Aalst W M P. Workflow verification: Fingding control-flow errors using Petri-net-based techniques. Lect Notes Comput Sci, 2000, 1806: 161–183
18 Desel J, Esparza J. Free Choice Petri Nets. Cambridge: Cambridge University Press, 1995
19 Liu G J. Some complexity results for the soundness problem of workflow nets. IEEE Trans Serv Comput, 2014, 7: 322–328
20 Weidlich M, Mendling J, Weske M. Efficient consistency measurement based on behavioral profiles of process models. IEEE Trans Softw Eng, 2011, 37: 410–429
21 Stahl C, Wolf K. Deciding service composition and substitutability using extended operating guidelines. Data Knowl Eng, 2009, 68: 819–833
22 Tan W, Fan Y S, Zhou M C. A Petri net-based method for compatibility analysis and composition of Web services in business process execution language. IEEE Trans Autom Sci Eng, 2009, 6: 94–106
23 Che X, Maag S. Testing protocols in Internet of Things by a formal passive technique. Sci China Inf Sci, 2014, 57: 032101
24 Gierds C, Mooij A J, Wolf K. Reducing adapter synthesis to controller synthesis. IEEE Trans Serv Comput, 2012, 5: 72–85
25 Martens A. On compatibility of web services. Petri Net Newslett, 2003, 65: 12–20
26 Fahland D, Favre C, Koehler J, et al. Analysis on demand: instantaneous soundness checking of industrial business process models. Data Knowl Eng, 2011, 70: 448–466
27 Wolf K. Generating Petri net state space. Lect Notes Comput Sci, 2007, 4546: 29–42
28 Best E. Structure theory of Petri nets: the free choice hiatus. Lect Notes Comput Sci, 1987, 254: 168–205
29 Wang S G, Wang C, Zhou M C, et al. A method to compute strict minimal siphons in S^3PR based on loop resource subsets. IEEE Trans Syst Man Cybern: Part A, 2012, 42: 226–237
30 Baldan P, Corradini A, Ehrig H, et al. Compositional semantics for open Petri nets based on deterministic processes.

https://engine.scichina.com/doi/10.1007/S11432-014-5259-5

Math Struct Comput Sci, 2005, 15: 1–35
31 Murata T. Petri nets: properties, analysis and applications. Proc IEEE, 1989, 77: 541–580
32 Reisig W. Understanding Petri Nets: Modeling Techniques, Analysis Methods, Case Studies. Berlin/Heidelberg: Springer-Verlag, 2013
33 Barkaoui K, Couvreur J M, Dutheillet C. On liveness in extended non self-controlling nets. Lect Notes Comput Sci, 1995, 935: 25–44
34 Chu F, Xie X L. Deadlock analysis of Petri nets using siphons and mathematical programming. IEEE Trans Robot Automat, 1997, 13: 793–840
35 Jiao L, Cheung T Y, Lu W M. On liveness and boundedness of asymmetric-choice nets. Theor Comput Sci, 2004, 311: 165–197
36 Liu G J, Chen L J. Sufficient and necessary condition to decide compatibility for a class of interorganizational workflow nets. Math Probl Eng, 2014, 2014: 392945
37 Aalst W M P, Kindler E, Desel J. Beyond asymmetric choice: a note on some extensitons. Petri Net Newslett, 1998, 55: 3–13

https://engine.scichina.com/doi/10.1007/s11432-014-5259-5