IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received March 6, 2017, accepted April 9, 2017, date of publication April 18, 2017, date of current version June 28, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2695558

Composition of Logical Petri Nets and Compatibility Analysis

WENJING LUAN[1], (Student Member, IEEE), LIANG QI[1], (Student Member, IEEE), AND YUYUE DU[2]

[1]College of Electronics and Information Engineering, Tongji University, Shanghai 201804, China
[2]College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China

Corresponding author: Yuyue Du (yydu001@163.com)

This work was supported by the National Natural Science Foundation of China under Grant 61170078 and Grant 61472228, in part by the Taishan Scholar Construction Project of Shandong Province, in part by the Natural Science Foundation of Shandong Province under Grant ZR2014FM009, and in part by the Promotive Research Fund for Young and Middle-Aged Scientists of Shandong Province under Grant BS2015DX010.

ABSTRACT Logical Petri nets (LPNs) can well describe and analyze batch processing functions and pass the value indeterminacy in cooperative systems. Their structure is simpler than their equivalent inhibition PNs. To analyze them, a vector matching method was given previously. We present interactive LPNs (ILPNs) in this paper. Their liveness and boundedness are analyzed for the first time. Compatibility is analyzed for a composed system and reflects the possibility of correct/proper interactions among its subsystems. To characterize different cooperative abilities in practice, compatibility is defined for ILPNs. Some relationships among compatibility, liveness, boundedness, and conservativeness are revealed. An example is presented to discuss the effectiveness of the proposed method.

INDEX TERMS Petri nets, interactive logical Petri net, Petri net composition, compatibility analysis, cooperative systems.

I. INTRODUCTION

In cooperative systems such as e-commerce systems, a merchant needs to interact, exchange messages, and thus make a deal with several different customers at a same time interval. This requires such systems to have a capacity of simultaneously processing a batch of data, called a batch processing function in the following discussion. During batch processing, both merchant and customer show indeterminacy. For example, a merchant may experience a shortage of products, while customers spend different time on each stage of the dealing process, and some even abruptly change or cancel their process at any trading progress. These are called passing value indeterminacy that appear in such interaction systems. The analysis of a batch processing function and passing value indeterminacy in such cooperative systems is very important theoretically and practically.

Petri nets (PNs) [1] are a well-founded model applied to the simulation and analysis of cooperative systems especially the concurrent ones. Based on its solid mathematical definition and graphical expression, PNs have been used to effectively model and analyze concurrent, asynchronous, distributed, parallel, and uncertain information processing systems [2]–[4] before they are put into operation, since any errors in the system may lead to a great loss of value. There are some extensions to PNs, such as time PNs [5], colored PNs [6], [7], and stochastic PNs [8]. They have been applied to many fields such as process control, communications protocols, production systems, hardware, embedded systems, and transportation systems [9]–[13]. Among these, PNs with arc weights may offer a batch processing function, but cannot address the passing value indeterminacy.

Logical Petri Nets (LPNs) are defined in [14] and [15] as a high-level PN. It can describe batch processing functions and passing value indeterminacy in cooperative systems and verify the correctness of such systems. The passing value indeterminacy is sorted into input and output indeterminacy, and described by logical input and output transitions, respectively. These logic transitions are attached with logical expressions that are defined and divided into input and output ones. A logical expression attached to a transition is a function of its input or output places. It fires while its corresponding logical expression is true, meaning that the marking of the places satisfies the requirement described by the expression. In detail, a logical input transition fires when its logical input expression is true; and a logical output fires only if its firing leads to a marking that makes its logical output expression true. In fact, the indeterminacy is implied by the logical expressions attached to the corresponding logical transitions. Due to such restriction of logical expressions labeled on transitions, LPNs can describe and analyze conveniently batch processing functions and passing value indeterminacy in cooperative systems. Obviously, LPNs enhance the expression ability of PNs and can cater to more application requirements. They have been applied to model and analyze a stock trading system [15] and e-commerce systems [16]. The equivalence between LPNs and the safe inhibition Petri nets (IPNs) [17] is proved in [14]. Although IPNs model batch processing functions and data arrival indeterminacy, they are complex. There lack any practical modeling tool and efficient analysis methods. The net structure of an LPN is much simpler than that of its equivalent IPN. A concept named Gate in a PN model named Stochastic Activity Networks [18] contain predicates and functions whose terms are markings of places. It can rule the firing of transitions. Such networks also have the batch processing functions and data arrival indeterminacy modeling function, but their single transition connect with more than one gate with several predicates and functions, thereby making the analysis very challenging. To analyze LPNs, our prior work proposed a vector matching method [19]. Some mathematical methods for analyzing an LPN are presented in [20]. Its properties such as the conservativeness, reachability and reversibility are also analyzed based on our proposed equations. However the construction complexity of a reachability tree and state equation solution is high and till now we have not revealed sufficient and necessary conditions for a live LPN. Du and Jiang [15] analyze e-commerce systems by using a subclass of LPNs in which the special structures of these systems are expressed by logical expressions; and relationships among logical transitions can be used to derive the otherwise-difficult-to-analyze fundamental properties of LPNs. In this paper, we present an interactive logical Petri nets, and their liveness and boundedness analysis are discussed. Compatibility is proposed as an important concept to decide if its subsystems own correct/proper interactions. Different cooperative abilities are thus characterized in practice. Some relationships among compatibility, liveness, and boundedness are revealed. An example is used to illustrate them.

The rest of this paper is organized as follows. Section II reviews the concepts of PNs, LPNs and IPNs. Section III gives an interactive logical Petri net, and discusses its liveness and boundedness analysis. Compatibility is defined and some relationships among compatibility, liveness, and boundedness are revealed. Section IV describes an e-commerce system and its LPN. The compatibility analysis is illustrated through a case study. Concluding remarks are made in Section V.

II. LPNs

This section first briefly reviews some notions of PNs [9]–[11], IPNs [17] and LPNs [14]–[16], [19], [20]. A simple comparison such as the net structure between an LPN and its equivalent IPN is presented. Then the capacity of an LPN in modeling batch processing functions and passing value indeterminacy in an e-commerce system is discussed. In the rest of paper, we use $\mathbb{R}_0$ to denote the real number set, $\mathbb{N}$ to denote the natural number set, i.e., $\mathbb{N} = \{0, 1, 2, \cdots\}$. Let $\mathbb{N}^+ = \mathbb{N}/\{0\}$, $\mathbb{N}_k = \{0, 1, 2, \cdots, k\}$, and $\mathrm{N}_k^+ = \{1, 2, \cdots, k\}$, $k \in \mathbb{N}^+$.

A. PNs AND LPNs

Definition 1: $N = (P, T, F)$ is a net where

1) P is a finite set of places;
2) T is a finite set of transitions with $P \cup T \neq \emptyset$ and $P \cap T = \emptyset$; and
3) $F \subseteq (P \times T) \cup (T \times P)$ is a set of directed arcs with each one from a place to a transition or from a transition to a place.

Definition 2: Given a net $N = (P, T, F)$. $x \in P \cup T$ is a node in N. ${}^\bullet x = \{y|(y, x) \in F\}$ and $x^\bullet = \{y|(x, y) \in F\}$ are its pre-set and post-set, respectively. If $X \subseteq P \cup T$, its pre-set and post-set are respectively ${}^\bullet X = \cup_{x \in \mathrm{X}} {}^\bullet x$ and $X^\bullet = \cup_{x \in \mathrm{X}} x^\bullet$.

Definition 3: $PN = (N, M, W)$ is a Petri net where

1) $N = (P, T, F)$ is a net;
2) $M: P \to \mathbb{N}$ is a marking defining the number of tokens in each place $p \in P$, indicated by $M(p)$ with M_0 being the initial marking;
3) $W: F \to \mathbb{N}^+$ is a weight function where $W(x, y) = 0$ if $(x, y) \notin F$, $x, y \in P \cup T$; and
4) It has the following transition firing rules:
   (a) A transition $t \in T$ is enabled at M if $\forall p \in {}^\bullet t$: $M(p) \geq W(p, t)$, represented by $M[t\rangle$;
   (b) If t is enabled, it can fire, generating a new marking M' from M, represented by $M[t\rangle M'$, where

$$M'(p) = \begin{cases} M(p) + W(t, p) & \text{if } p \in t^\bullet - {}^\bullet t \\ M(p) - W(p, t) & \text{if } p \in {}^\bullet t - t^\bullet \\ M(p) & \text{else} \end{cases} \quad (1)$$

   (c) If there exist a sequence of transitions $t_1, t_2, \ldots, t_k$, and markings $M_1, M_2, \ldots, M_k$ such that $M[t_1\rangle M_1[t_2\rangle M_2 \ldots M_{k-1}[t_k\rangle M_k$, then M_k is called to be reachable from M, and all markings reachable from M are denoted by $R(PN, M)$ (or $R(M)$ shortly) and $M \in R(M)$. Let $\sigma = t_1, t_2, \ldots, t_k$. We denote $M[\sigma\rangle M_k$ and $t_i \in \sigma$, $i \in \mathrm{N}_\mathrm{k}^+$.

For simplicity, a marking M is denoted as $M = \sum_{p \in P} M(p) \bullet p$. For example, $(2, 1, 0, 0, 0)^T$ can be written as $2p_1 + p_2$. In the following, $P = \{p_1, p_2, \cdots, p_n\}$ is a set of places in a PN, and we denote that $f = f(p_1, p_2, \cdots, p_n)$ is a logical expression [21], [22] on P and the operators in f only consist of logic disjunction "$\vee$", conjunction "$\wedge$"

and negation operators "¬", i.e., f is an expression with places of first-order logic as atomic expressions. ${}_{\bullet}T_{\bullet}/{}_{\bullet}F_{\bullet}$ denotes the logic value "true"/"false" with $\neg{}_{\bullet}T_{\bullet} = {}_{\bullet}F_{\bullet}$ and $\neg{}_{\bullet}F_{\bullet} = {}_{\bullet}T_{\bullet}$, where $\neg$ is a logical negation operator. When p_1 is used as a logical expression in f, the logic value of p_1 is ${}_{\bullet}T_{\bullet}$ only if $m(p_1) \geq 1$.

Definition 4 [19]: Let $PN = (P, T, F, W, M)$ be a PN. For $p_i \in P$, $i \in \mathbb{N}_n^+$, $p_i|_M$ denotes the logical value of p_i at M where

$$p_i|_M = \begin{cases} {}_{\bullet}T_{\bullet}, & \text{if } M(p_i) \geq 1 \\ {}_{\bullet}F_{\bullet}, & \text{if } M(p_i) = 0 \end{cases}$$

and $f|_M$ denotes the logical value of f at M where

$$f|_M = f(p_1, p_2, \cdots, p_n)|_M = f(p_1|_M, p_2|_M, \cdots, p_n|_M)$$

In the following discussion, we call a logical expression is satisfied at M if $f|_M = {}_{\bullet}T_{\bullet}$.

The following definition of LPNs is owing to [19].

Definition 5: $L{=}(P, T, F, M, W, I, O, \tau)$ is an LPN where

1) $P = P_D \cup P_C$ is a finite set of places with $P \neq \emptyset$ and $P_D \cap P_C = \emptyset$ where
   (a) P_D denotes a set of data places; and
   (b) P_C denotes a set of control places;
2) $T = T_D \cup T_I \cup T_O$ is a finite set of transitions with $T \neq \emptyset$, given any $i, j \in \{D, I, O\}$, if $i \neq j$, $T_i \cap T_j = \emptyset$, and $\forall t \in T_I \cup T_O$: ${}^{\bullet}t \cap t^{\bullet} = \emptyset$, where
   (a) T_D denotes a set of transitions as defined in Definition 1;
   (b) T_I is called a set of logical input transitions, where $\forall t \in T_I$, the markings of input places for t to be enabled are restricted by a logical expression $f_I({}^{\bullet}t)$;
   (c) T_O denotes a set of logical output transitions, where $\forall t \in T_O$, the markings of output places after t fires are restricted by a logical expression $f_O(t^{\bullet})$ on $t^{\bullet}$;
3) $F \subseteq (P \times T) \cup (T \times P)$ is a finite set of directed arcs;
4) $M: P \rightarrow \mathbb{N}$ is a marking;
5) $W: F \rightarrow \mathbb{N}$ is a weight function where $W(x, y) = 0$ if $(x, y) \notin F$, $x, y \in P \cup T$;
6) I is a mapping from a logical input transition to a logical input expression, i.e., $\forall t \in T_I$, we denote $I(t) = f_I({}^{\bullet}t)$;
7) O is a mapping from a logical output transition to a logical output expression, i.e., $\forall t \in T_O$, we denote $O(t) = f_O(t^{\bullet})$;
8) $\tau: T \rightarrow \mathbb{R}^+$ associates transitions with a time delay, i.e., if $t \in T$ is enabled, it fires after time $\tau(t)$; and
9) It has the following transition firing rules:
   (a) $\forall t \in T_D$, its firing rules are the same as those in PNs;
   (b) $\forall t \in T_I$, t is enabled only if $I(t)|_M = {}_{\bullet}T_{\bullet}$. $M[t\rangle M'$, where for $\forall p \in {}^{\bullet}t$, if $M(p) = 0$, $M'(p) = 0$, else, $M'(p) = M(p) - W(p, t)$; for $\forall p \in t^{\bullet}$, $M'(p) = M(p) + W(t, p)$; and $\forall p \notin {}^{\bullet}t \cup t^{\bullet}$, $M'(p) = M(p)$; and
   (c) $\forall t \in T_O$, t is enabled only if $\forall p \in {}^{\bullet}t$: $M(p) = 1$. $M[t\rangle M'$, where $\forall p \in {}^{\bullet}t$: $M'(p) = M(p) - W(p,t)$; $\forall p \notin t^{\bullet} \cup {}^{\bullet}t$: $M'(p) = M(p)$; and $\forall p \in t^{\bullet}$ must satisfy $O(t)|_{M'-M} = {}_{\bullet}T_{\bullet}$, i.e., $t^{\bullet}$ must satisfy the logical output expression $O(t)$ at M'.

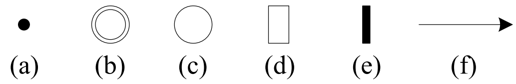

FIGURE 1. Graphical representation of elements in LPNs: (a) a token, (b) a data place, (c) a control place, (d) a logical transition, (e) an ordinary transition, and (f) a directed arc.

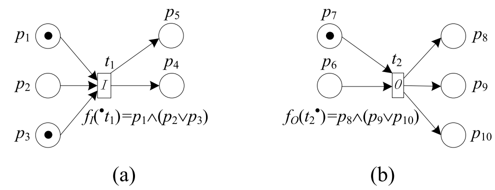

FIGURE 2. Two LPNs.

Notice that in the marking function in the above definition, we have an assumption that the maximum capacity of each control place is exactly 1 while that of data place is greater than 0.

An LPN is a high-level PN. The input/output places of a logical input/output transition are restricted by a logical input/output expression $I(t)/O(t)$, and logical input and output transitions are called logical transitions. Hence, the indeterminacy of values in input and output places is described by logical expressions. They are graphically described in Fig. 1, where (a)-(f) respectively describe a token, data place, control place, logical transition, ordinary transition, and directed arc. Two simple LPNs are presented in Fig. 2 where (a) only contains a logical input transition t_1 and (b) contains a logical output one. They are restrained by a logical input/output expression $I(t_1)/O(t_2)$ where $I(t_1) = p_1 \wedge (p_2 \vee p_3)$ and $O(t_2) = p_8 \wedge (p_9 \vee p_{10})$.

Note that each place of a logical expression has a logical value at marking M in an LPN, and by substituting the values of all places into the logical expression, the expression corresponds to a logical value. In Fig. 2, for example, if $M_0 = (1, 0, 1, 0, 0, 0, 1, 0, 0, 0)$, $I = \{p_1, p_2, p_3\}$, and $f_I({}^{\bullet}t_1) = p_1 \wedge (p_2 \vee p_3)$, then from Definition 5, we have $p_1|_{M_0} = {}_{\bullet}T_{\bullet}$, $p_2|_{M_0} = {}_{\bullet}F_{\bullet}$, $p_3|_{M_0} = {}_{\bullet}T_{\bullet}$, and $f_I({}^{\bullet}t_1)|_{M_0} = {}_{\bullet}T_{\bullet} \wedge ({}_{\bullet}F_{\bullet} \vee{}_{\bullet} T_{\bullet}) = {}_{\bullet}T_{\bullet}$.

B. EQUIVALENCE BETWEEN LPN AND IPN

Inhibition Petri nets (IPNs) [17] are an extension to PNs as defined next.

Definition 6: $IPN = (PN, H)$ is a PN with inhibitor arcs (IPN) where

1) $PN = (P, T, F, W, M)$ is a PN;
2) $H \subset P \times T$ is the inhibitor arc set, $H \cap F = \emptyset$, and $(p, t) \in H$ is represented as a non-directed arc with a small circle at t's side; and

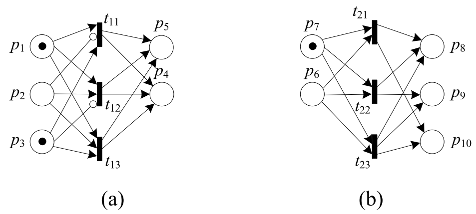

FIGURE 3. Equivalent IPNs of the LPNs in Fig. 2.

3) It has the following transition firing rules: $\forall t \in T$, t is enabled at M if $\forall p \in P$: $(p, t) \in F \Rightarrow M(p) \geq 1$, and $(p,t) \in H \Rightarrow M(p) = 0$. Firing t results in a new marking M' computed by (1) in Definition 3.

The equivalence between LPNs and IPNs has been proved in [14]. An IPN model that is equivalent to the LPN model in Fig. 2 is shown in Fig. 3.

For example, in Fig. 1, there exists a marking $M_0 = (1, 0, 1, 0, 0, 1, 0, 0, 0)$ such that the logical expression of the input transition t_1 is true at M_0. Thus t_1 can fire such that p_4 can obtain a token and the process goes on. Given other markings, t_1 may also be enabled. Though the inputs of t_1 are different, they do not affect the execution process and result that enables p_4 to obtain a token. In the corresponding IPN, no matter which transitions t_{11}, t_{12} or t_{13} fires, p_4 can obtain a token, meaning that a system could accept the inputs and proceed with its execution. In Fig. 2, if t_2 fires, the output is indeterminate, and t_{21}, t_{22}, and t_{23} are all enabled in the corresponding IPN such that the firing choice of these transitions represents the output indeterminacy. In another word, that markings enable t_{11}, t_{12} or t_{13} in Fig. 3 is equivalent to that they enable t_1 in Fig. 2, implying the input indeterminacy; the marking after choosing t_{31}, t_{32}, and t_{33} to enable in Fig. 3 is equivalent to that enabling t_1 in Fig. 2, implying the output indeterminacy. We propose LPNs to model the indeterminacy of the systems. An LPN has its corresponding equivalent IPN.

From Figs. 2 and 3, the IPN model consists of 6 transitions while its corresponding LPN has only 2 transitions. IPN requires more arcs and two inhibitor arcs. Its structure is clearly much more complex than LPN. Besides till now there is no good property analysis tool or method for IPNs.

C. LPNs OF E-COMMERCE SUB-SYSTEMS

We now discuss an e-commerce system where a seller may trade with several buyers at the same time. We use LPN to describe a batch processing function and passing value indeterminacy that exist in such a system. For an illustration purpose, we consider only a three *e-commerce subsystems* modeled by *LPNs* as shown in Fig. 4, i.e., a seller S and two buyers $B1$ and $B2$. Firstly $B1$ and $B2$ deliver orders to S if they need to buy any products from S. In one circulation, S receives the orders from $B1$ and $B2$, and check if they can satisfy the buyers' order requests such as the order quantity. At the same time it decides whether to accept or refuse the latter's orders. If it accepts an order, the corresponding buyer needs to pay for it. After receiving the payment, the S sends the product to the buyer, and the deal finishes. The tasks of a seller subsystem S include receiving order (*r_order*), checking order (*c_order*), receiving payment (*r_payment*), and sending products (*s_product*). i_S is an original place and o_S is a terminal place. Transitions *r_order*, *c_order*, *r_payment*, and *s_products* have logic expressions: $i_S \wedge (B1\_order \vee B2\_order)$, $p_{s2} \wedge (B1\_refuse \vee B1\_accept) \vee (B2\_refuse \vee B2\_accept)$, $p_{s2} \wedge ({}_\bullet T_\bullet \vee B1\_payment \vee B2\_payment)$ and $p_{s4} \wedge ({}_\bullet T_\bullet \vee B1\_product \vee B2\_product)$. The tasks of a buyer subsystem include, sending order (*s_B_order*), sending payment (*s_B_payment*), and receiving the product (*r_B_product*). i_B is an original place and o_B a terminal place. In order to ensure the correctness of the composition of the subsystems and that of the whole e-commerce system, we present some theoretical methodologies next. Note that the fundamental properties [19] of LPNs such as liveness, reachability conservativeness and reversibility, are analyzed based on a vector matching method in our previous work. This work discusses the properties of the composition of sub-nets. We suppose that each unit of resources sent or received by a buyer at each deal can be described by a token. In the model in Fig. 4, each arc except (*s_deal*, *Product*), (*Product*, *s_deal*), (*r_payment*, *Payment*), and (*Product*, *s_product*) has a weight of 1, while n_1 and n_2 are two variables of the weights and $n_1, n_2 \leq 2$.

III. COMPATIBILITY

To directly analyze LPNs, we can find their enabled transitions based on a vector matching method [19]. In the following discussion, we present an interactive logical Petri net (ILPN) which is a sub-class of LPNs, and discuss their liveness and boundedness for the first time. Compatibility is an important concept to reflect the possibility of correct/proper interaction among its subsystems in a composed system. In order to characterize different cooperative abilities in practice, we define compatibility for ILPNs and reveal the relationships among compatibility, liveness, and boundedness.

A. FIRING RULES OF LOGICAL TRANSITIONS IN AN LPN

In LPNs, passing value indeterminacies is divided into input and output indeterminacy that are respectively described via logical input and output transitions. The logic transitions are restricted by their corresponding logical input and output expressions. They fire only if their related logical expressions are true at a right marking. Normally each logical expression is transformed into a disjunctive normal form, and it is unique [22]. Each disjunctive clause in the disjunctive normal form is a conjunct consisting of all places related to a logical transition. $I(t)$ of a logic input transition t is true only if one of disjunctive clauses of $I(t)$ holds, and t is enabled. The output places of a logic output transition t depends on the disjunctive clauses of $O(t)$. If $f(x_1, x_2, \cdots, x_k)$ is logical expression in an LPN containing variables x_1-x_k, then it can be transformed into a unique disjunctive normal form denoted by $f(x_1, x_2, \cdots, x_k) = f_1 \vee f_2 \vee \ldots \vee f_m$, where $f_1 - f_m$ are the

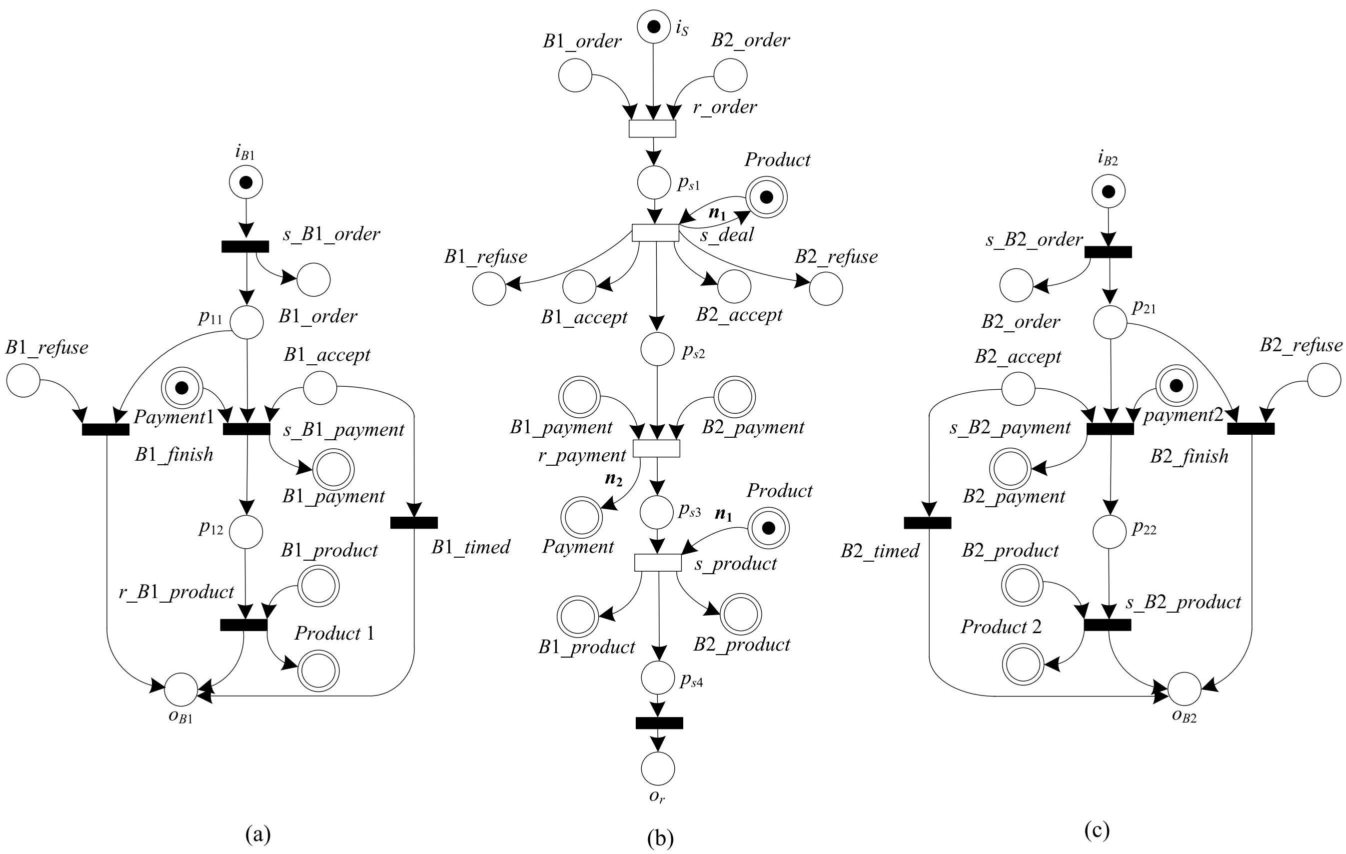

FIGURE 4. Three LPN sub-models in an e-commerce system: (a) the seller model, (b) the model of buyer 1, (c) the model of buyer 2.

corresponding conjuncts. To make sure the correct operation of a system model, these logical expressions should satisfy some restrictions.

Definition 7: Let $L = (P, T, F, M, W, I, O, \tau)$ be an LPN, $f(x_1, x_2, \cdots, x_k) = f_1 \vee f_2 \vee \ldots \vee f_m$, be a logical expression attached to a logical output transition, M be a marking, and $f|_M = {}_\bullet T_\bullet$. The output result is restricted by M such that $f_i|_M = {}_\bullet T_\bullet$ and $f_j|_M = {}_\bullet F_\bullet$ where $j = \mathbb{N}_m^+/\{i\}$.

Note that Definition 7 gives a restriction, i.e., there is only one output determined by the current marking. Some transitions may have common structures. We present their following formal definition.

Definition 8: Let x_1-$x_k \in P$ be k places in an LPN. We define $f(x_1, x_2, \cdots, x_k) = x_1 \wedge (x_2 \vee \cdots \vee x_k)$ as a template logical expression in an LPN with $x_1 \in P_C$.

Some determinacy relationship among the input or output expressions of logical transitions are defined next. For example, in an e-commerce LPN, if a logical transition t fires, then the input conditions or output results of other transitions have the corresponding relationships with those of t. There are mainly two kinds of relationships between each pair of logical transitions. Their definitions are shown as follows where $f(x_1, x_2, \cdots, x_k) = x_1 \wedge (x_2 \vee \cdots \vee x_k)$ is a template logical expression of an LPN. $\forall t_i, t_j \in T_I \cup T_O$, if $t_i \in T_I$, let $P = {}^\bullet t_i$; else $P = t_i^\bullet$ and if $t_j \in T_I$, $Q = {}^\bullet t_j$, else $Q = t_j^\bullet$.

Definition 9: Let $T_C \subseteq T_I \cup T_O$ be a set of logical transitions in an LPN, $T_{C1} \cup T_{C2} = T_C$ and $T_{C1} \cap T_{C2} = \emptyset$. $\forall t_i, t_j \in T_{C1}$ or $t_i, t_j \in T_{C2}$:

1) t_i and t_j are isomorphic and denoted by $t_i \sim t_j$ where if $M^* \in R(M_0)$, $M^*\ [t_i\rangle$ or $M_0[\sigma, t_i\rangle M^*$, then $\forall M' \in R(M^*)$, $M'\ [t_j\rangle$ or $M_0[\sigma', t_j\rangle M'$ such that $M^*_{|P} = M'_{|Q}$, where $M_{|U}$ denotes the projection of M: $P \rightarrow \mathbb{N}$ on U: $M_{|U}(u) = M(u)$ if $u \in U \cap P$ and $m_{|U}(u) = 0$ if $u \in U \backslash P$; and
2) t_i and t_j are dual as denoted by $t_i \Leftrightarrow t_j$ where if $M^* \in R(M_0)$, $M^*[t_i\rangle$ or $M_0[\sigma, t_i\rangle M^*$, then $\forall M'' \in R(M_0)$, $M''[t_j\rangle$ or $M_0[\sigma'', t_j\rangle M''$ such that $M^*_{|P}(1) = M''_{|Q}(1) = 1$, and $M^*_{|P}(x) + M''_{|Q}(x) = 1$, $k \geq x > 1$.

In this definition, M^* is denoted as the marking we choose on which a logical transition will fire or has fired. It is a marking that affects the indeterminacy conditions or results when another logical transition fires. From the above definition, we can derive the properties of these two relations.

Property 1 [19]:

1) $\sim$ is reflexive, symmetric and transitive; and
2) $\Leftrightarrow$ is symmetric;

According to the structure and relationship among logical expressions, a reachability tree can be constructed.

Definition 10: Let x_1 and x_2 be variables in a logical expression. They are called converse if only one of them can contain a token, i.e., $\forall M$: $(x_1 \vee x_2)|_M = {}_\bullet T_\bullet$, and if $x_1|_M = {}_\bullet T_\bullet$, $x_2|_M = {}_\bullet F_\bullet$; else, $x_2|_M = {}_\bullet T_\bullet$.

Let $f(x_1, x_2, \cdots, x_k) = x_1 \wedge (x_2 \vee x_3 \vee x_4 \vee \cdots \vee x_k)$ be a template logical expression of an LPN. Then we describe the template as $f(x_1, x_2, \cdots, x_k) = x_1 \wedge (x_2 \vee x_3) \vee (x_4 \vee \cdots \vee x_k)$ and $\forall M$: $f|_M = x_1 \wedge (x_4 \vee \cdots \vee x_k)|_M$.

B. INTERACTIVE LOGICAL PETRI NETS (ILPNs)

We cite the concept of a logic process as follows from [23].

Definition 11: A logic process is a net $N = (P, T, F)$ satisfying that:

1) N has two special places i and o, where $i \in P$ is called a source place, $o \in P$ is called a sink place with ${}^\bullet i = o^\bullet = \emptyset$, and
2) Let $N^* = (P, T \cup \{b\}, F \cup \{(b, i), (o, b)\})$ be the trivial extension of N and $M_0 = i$ be its initial marking, i.e., N has only one token in i. Then, N^* is strongly connected, and (N^*, M_0) is live and safe.

Following [23], we define a logic process with channels as follows.

Let $L = (P, T, F, M, W, I, O, \tau)$, $P_1 \subseteq P$, and $T_1 \subseteq T$. The subnet of L generated by P_1 and T_1 are respectively written as

$$L|_{P_1} = (P_1, {}^\bullet P_1 \cup P_1^\bullet, F', M, W, I', O', \tau) \text{ and}$$

$$L|_{T_1} = ({}^\bullet T_1 \cup T_1^\bullet, T_1, F'', M, W, I, O, \tau)$$

where

(1) $F' = F \cap \{P_1 \times ({}^\bullet P_1 \cup P_1^\bullet), ({}^\bullet P_1 \cup P_1^\bullet) \times P_1\}$;
(2) $F'' = F \cap \{T_1 \times ({}^\bullet T_1 \cup T_1^\bullet), ({}^\bullet T_1 \cup T_1^\bullet) \times T_1\}$;
(3) $I'(t) = I(t)$ and $O'(t) = O(t)$; and
(4) $\forall p \in P_1, \forall M \in R(M_0) : p|_M = {}_\bullet T_\bullet$.

Note that a logic expression attached to a logical transition only contains the variables of places in P_1. It is denoted by $f|_{P1}$ and other variables are replaced with the logical value ${}_\bullet T_\bullet$.

Definition 12: A logic process with channels (LPC) is a logical Petri net $L = (P_C \cup P_I \cup P_O, T_D \cup T_I \cup T_O, F, M, W, I, O, \tau)$ where

1) The subnet generated by $P_C \neq \emptyset$ is a logic process denoted by L^*, where P_C is the set of control places of L; and
2) P_I is a set of input channel places, P_O is a set of output channel places, $P_I \cap P_O = \emptyset$ and $P_C \cap (P_I \cup P_O) = \emptyset$.

Definition 13: Interactive Logical Petri Nets (ILPNs) are defined recursively as follows.

1) An $L = (P_{IC} \cup P_C \cup P_I \cup P_O, T_D \cup T_I \cup T_O, F, I, O)$ is an ILPN where it is an LPC with $P_{IC} = \emptyset$.
2) Let $L_i = (P_{ICi} \cup P_{Ci} \cup P_{Ii} \cup P_{Oi}, T_{Di} \cup T_{Ii} \cup T_{Oi}, F_i, D_i, I_i, O_i)$, $i \in \mathbb{N}_2^+$, be two ILPNs such that

$$(P_{IC1} \cup P_{C1}) \cap (P_{IC2} \cup P_{C2}) = \emptyset,$$
$$P_{I1} \cap P_{I2} = P_{O1} \cap P_{O2} = \emptyset,$$
$$(P_{O1} \cap P_{I2}) \cup (P_{O2} \cap P_{I1}) = P_S \neq \emptyset, \text{ and}$$
$$(T_{Di1} \cup T_{Ii1} \cup T_{Oi1}) \cap (T_{Di2} \cup T_{Ii2} \cup T_{Oi2}) = \emptyset.$$

Then, L is an ILPN where

$$\begin{aligned} P_{IC} &= P_{IC1} \cup P_{IC2}, \\ P_C &= P_{C1} \cup P_{C2} \cup P_S, \\ P_I &= (P_{I1} \cap P_{I2}) \backslash P_S, \\ P_O &= (P_{O1} \cap P_{O2}) \backslash P_S, \\ T_D &= T_{D1} \cup T_{D2}, T_I = T_{I1} \cup T_{I2}, \\ T_O &= T_{O1} \cup T_{O2}, \\ F &= F_1 \cup F_2, \\ D(t) &= D_i(t) \text{ if } t \in T_{Di} \cup T_{Ii} \cup T_{Oi}, \text{ and} \\ I(t) &= I_i(t) \text{ if } t \in T_{Ii}, O(t) = O_i(t) \text{ if } t \in T_{Oi}. \end{aligned}$$

Clearly, an ILPN is the union of m LPC L_1-L_m, $m \geq 1$, via a set of common places (i.e., P_S). To facilitate its description, an ILPN is denoted by

$$L = L_1 \oplus L_2 \oplus \ldots \oplus L_m = \oplus_{i=1}^m L_i.$$

In Definition 13, P_{IC}, P_I, P_O, $P_I \cup P_O$ are called a set of internal channel, input channel, output channel, and external channel places of L, respectively. All the channel places are data places. Input channel places can only be combined with output ones. After two LPCs are composed via a set of common channel places, these common places become internal ones and cannot take part in other combinations any more. Additionally, external channel places represent interfaces used to interact with the environment. Note that the external condition needs to be fulfilled for any meaningful design analysis. Therefore, it is assumed that all ILPNs have no external channel places. It is also assumed that every IPN is connected since, otherwise, they should be separately analyzed. Here, liveness and reversibility are used to study the compatibility. Thus a net can be live and reversible if it is closed, i.e., there is no place that has no input transition in the net. Thus the trivial extension of an ILPN is defined next.

Definition 14: Let $L = \oplus_{i=1}^m L_i$ be an ILPN. $L^* = (P_{IC} \cup P_C \cup P_I \cup P_O, T_D \cup T_I \cup T_O \cup \{b\}, \cup_{j=1}^m \{(o_j, b)(b, i_j)\} \cup F, M, W, I, O, \tau)$ is called the trivial extension of LIPN where b is a bridge transition.

C. LIVENESS AND BOUNDEDNESS ANALYSIS

Definition 15: L is an ILPN. C is called a path from node n_1 to n_k if there exists a sequence of nodes $< n_1, n_2, \ldots, n_k >$ such that $(n_i, n_{i+1}) \in F$, $i \in \mathbb{N}_{k-1}^+$. $\&(C)$ denotes the alphabet of C, i.e., $\&(C) = \{n_1, n_2, \ldots, n_k\}$. C is called a basic path where $\forall n_i, n_j \in \&(C)$, if $i \neq j$: $n_i \neq n_j$. n_j is the subsequent node of n_i, denoted by $n_i \prec_C n_j$ if $\exists n_i, n_{i+1}, \ldots, n_j \in \&(C)$ and $(n_i, n_{i+1}), (n_{i+1}, n_{i+2}), \ldots, (n_{j-1}, n_j) \in F$.

Note that a path $C = < n_1, n_2, \ldots, n_k >$ is called a circle if $< n_1, n_2, \ldots, n_{k-1} >$ is a basic path and $n_1 = n_k$.

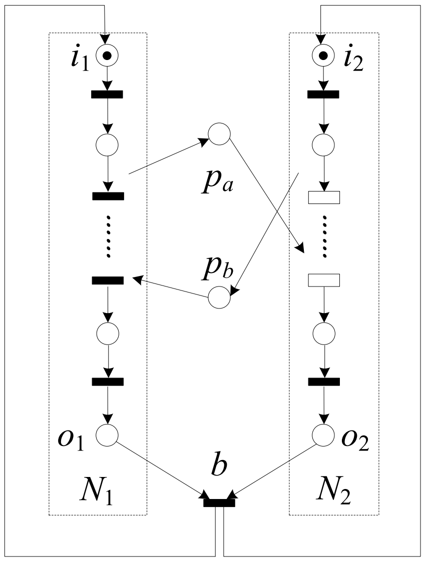

FIGURE 5. ILPN composed of two LPCs.

Let L_j, $j \in \mathbb{N}_2^+$ be two LPCs where $P_{O1} \cap P_{I2} = \{p_a\}$, and $P_{O2} \cap P_{I1} = \{p_b\}$; $\forall p'' \in \{p_a, p_b\}$: $|{}^\bullet p''| = |p''^\bullet| = 1$; and $T_{I1} \cup T_{O1} \cup T_{I2} \cup T_{O2} \subseteq \cup_{p'' \in \{p_a,p_b\}} ({}^\bullet p'' \cup p''^\bullet)$; N_j be the logic process generated by P_{Cj} where $\forall p \in P_j \backslash \{i_j, o_j\}$: $|{}^\bullet p| = |p^\bullet| = 1$, and $\forall t \in T_j$: $|{}^\bullet t| = |t^\bullet| = 1$; $L = L_1 \oplus L_2$; and L^* be the trivial extension of L, as shown in Fig. 5. Then we have the following conclusions.

Theorem 1: L^* is live and bounded iff in L, either of the following conditions hold:

1) $\cup_{p'' \in \{p_a,p_b\}} ({}^\bullet p'' \cup p''^\bullet) = T_I \cup T_O, f = p' \wedge (_\bullet T_\bullet \vee p'')$ is the template logical expression, $M^* \in R(M_0)$, $M^*(p'' = 0$, and ${}^\bullet p_a \sim p_a^\bullet \sim^\bullet p_b \sim p_b^\bullet$;
2) a) there exist no circle; and b) $\exists p'' \in \{p_a, p_b\}$: ${}^\bullet p'' \in T_D$ and $p''^\bullet \in T_I \cup T_O$ (or $p''^\bullet \in T_D$ and ${}^\bullet p'' \in T_I \cup T_O$), $M^* \in R(M_0)$: $M^*(p'') = 1$ and $f|_M^* = {}_\bullet T_\bullet$.
3) a) there is a circle; b) condition (1) is not satisfied; and c) $\exists p'' \in \{p_a, p_b\}$: ${}^\bullet p'' \cup p''^\bullet \in T_I \cup T_O$ and $M^* \in R(M_0)$: $M^*(p'') = 0$ while $p' \in \{p_a, p_b\} \backslash p''$: ${}^\bullet p' \in T_D$ and $p'^\bullet \in T_I \cup T_O$ (or $p'^\bullet \in T_D$ and ${}^\bullet p' \in T_I \cup T_O$), $M^{*\prime} \in R(M_0)$: $M^{*\prime}(p') = 1$.

From Theorem 1 we have that an ILPN composed by two LPCs is live and bound if the logical transitions respectively from the two LPCs are isomorphic; and there is no circular wait for any resources.

For the above L_j, $j \in \mathbb{N}_2^+$, let $T_{I1} = T_{O1} = \emptyset$, $T_{I2} = \{t_{I2}\}$ and $T_{O2} = \{t_{O2}\}$; $f(p', p'') = p' \wedge (_\bullet T_\bullet \vee p'')$ be the template logical expression in L_2, where $p' \in P_i$ and $p'' \in \{p_a, p_b\}$; N_j be the logic process generated by $P_{Aj} = \{i_j, p_j, o_j\}$, where $\forall p \in P_j \backslash \{i_j, o_j\}$: $|{}^\bullet p| = |p^\bullet| = 1$, and $\forall t \in T_j$: $|{}^\bullet t| = |t^\bullet| = 1$; and $L = L_1 \oplus L_2$. Suppose that ${}^\bullet p_a = \{t_{11}\}$, $p_a^\bullet = \{t_{I2}\}$, ${}^\bullet p_b = \{t_{O2}\}$, $p_b^\bullet = \{t_{12}\}$. Given this situation, we can get the result in the corollary below.

Corollary 1: L^* is live and bounded iff there exist no circle and $M^* \in R(M_0)$: $M^*(p'') = 1$ and $f|_M^* = {}_\bullet T_\bullet$.

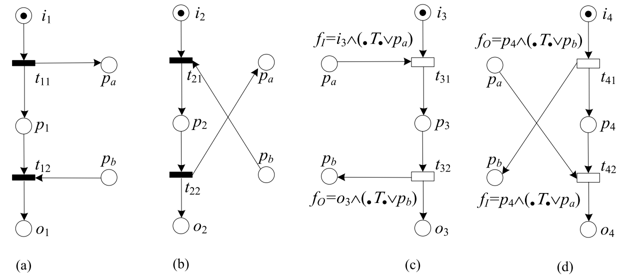

FIGURE 6. Four LPCs.

We illustrate the aforementioned corollary through some examples. There are four LPCs denoted by L_j, $j \in \mathbb{N}_4^+$ as shown in Fig. 6, where

$$
\begin{aligned}
P_{Aj} &= \{i_j, p_j, o_j\}, \quad P_{ICj} = \emptyset, \quad P_{I1} = P_{I2} = P_{O3} \\
&= P_{O4} = \{p_b\}, \text{ and } P_{O1} = P_{O2} = P_{I3} = P_{I4} = \{p_a\}, \\
T_{D1} &= \{t_{11}, t_{12}\}, T_{D2} = \{t_{21}, t_{22}\}, \\
T_{I3} &= \{t_{31}\} \textit{ with } f_I({}^\bullet t_{31}) = i_3 \wedge (_\bullet T_\bullet \vee p_a), \\
T_{O3} &= \{t_{32}\} \textit{ with } f_O(t_{32}^\bullet) = o_3 \wedge (_\bullet T_\bullet \vee p_b), \\
T_{I4} &= \{t_{42}\} \textit{ with } f_I({}^\bullet t_{42}) = p_4 \wedge (_\bullet T_\bullet \vee p_a), \\
T_{O4} &= \{t_{41}\} \textit{ with } f_O(t_{41}^\bullet) = p_4 \wedge (_\bullet T_\bullet \vee p_b), \\
T_{D3} &= T_{D4} = T_{I1} = T_{I2} = T_{O1} = T_{O2} = \emptyset, \\
F_1 &= \{(i_1, t_{11}), (t_{11}, p_a), (t_{11}, p_1), \\
&\qquad (p_1, t_{12}), (p_b, t_{12}), (t_{12}, o_1)\}, \\
F_2 &= \{(i_2, t_{21}), (p_b, t_{21}), (t_{21}, p_2), \\
&\qquad (p_2, t_{22}), (t_{22}, p_a), (t_{22}, o_1)\}, \\
F_3 &= \{(i_3, t_{31}), (p_a, t_{31}), (t_{31}, p_1), \\
&\qquad (p_3, t_{32}), (t_{32}, p_b), (t_{32}, o_3)\}, \text{ and} \\
F_4 &= \{(i_4, t_{41}), (t_{41}, p_b), (t_{41}, p_4), \\
&\qquad (p_4, t_{42}), (p_a, t_{42}), (t_{42}, o_4)\}.
\end{aligned}
$$

According to Theorem 1 and Corollary 1, the ILPN in Fig. 7 (c) is not live and bounded. The ILPNs in Figs. 7 (a), (b), and (d) are live and bounded if and only if both p_a and p_b have tokens that satisfy all the logical transitions.

D. CONSERVATIVENESS

In an e-commerce LPN, output results of a logical transition are not only determined by the transitions that are isomorphic or dual with them, but also are related with the input places which are data places. We now give the definition of its conservativeness. First we give some concepts in order to discuss the transition firing rules in LPNs.

$$
\begin{aligned}
&\forall t_i, t_j \in T_I \cup T_O, \text{if } t_i \in T_I, \text{ let } P = {}^\bullet t_i; \text{ else } P = t_i^\bullet \\
&\text{and if} \quad t_j \in T_O, Q = {}^\bullet t_j, \text{ else } Q = t_j^\bullet.
\end{aligned}
$$

Definition 16: Let L be an ILPN. If there exists an m-dimension vector $Y = [y_k]_{m\times 1}$, $y_k > 0$, $k \in \mathbb{N}_m^+$, M_0 is

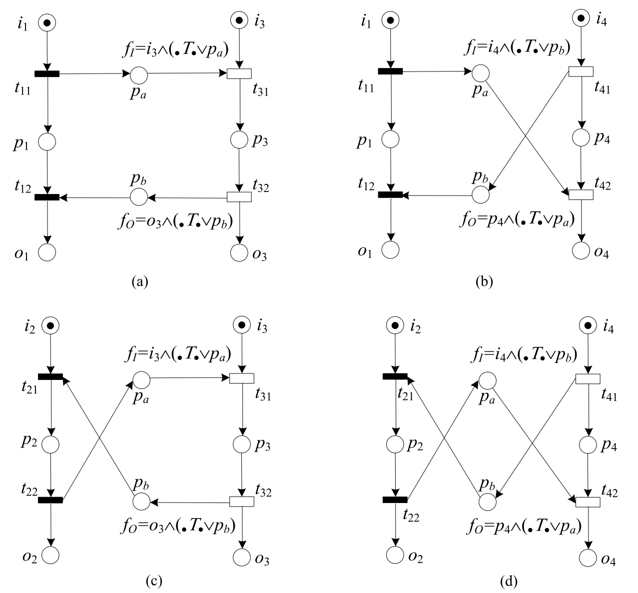

FIGURE 7. Four ILPNs composed by the LPCs in Fig. 6.

an arbitrary initial marking, and $\forall M \in R(M_0)$,

$$\sum_{k=1}^{m} M(p_k)y_k = \sum_{j=1}^{m} M_0(p_j)y_j,$$

then L is conservative. In particular, L is conservative on $P' \in P$ if $\forall M \in R(M_0)$,

$$\sum\nolimits_{p_i \in P'} M(p_i)y_i = \sum\nolimits_{p_j \in P'} M_0(p_i)y_j$$

From Definitions 5 and 13, we have $P_D \subseteq P_I \cup P_O$. In a system, the data places should preserve the resource quantity modeled by token count. Thus we have

$$\forall M \in R(L, M_0) : \sum\nolimits_{p \in P_D} M(p) = c,$$

where c is a constant describing the total number of resources.

Now we discuss the conservativeness of an LPN by using matrix calculation.

Theorem 2: Let L be an LPN. L is conservative on P_D iff

1) $\forall t \in T_D$,

$$\sum\nolimits_{p_i \in P_D} W(p_i, t) = \sum\nolimits_{p_j \in P_D} W(t, p_j);$$

2) $\forall t \in T_I$,

$$\sum\nolimits_{p_i \in P_D} W(p_i, t)\rho(p_i) = \sum\nolimits_{p_j \in P_D} W(t, p_j),$$

where if $M(p_i) > 0$, $\rho(p_i) = 1$; else, $\rho(p_i) = 0$; and

3) $\forall t \in T_O$,

$$\sum\nolimits_{p_i \in P_D} W(p_i, t) = \sum\nolimits_{p_j \in P_D} W(t, p_j)\rho(p_j),$$

where if $M[t\rangle M'$ and M-$M'(p_i) > 0$, then $\rho(p_j) = 1$, else, $\rho(p_j) = 0$.

We suppose that all data places must deposit and withdraw tokens in a system. Otherwise, the data places is meaningless.

Corollary 2: If L is conservative and $n_1 \in P_D$, then there exists a path $C =< n_1, n_2, \ldots, n_k >$ where $n_3, n_5, \ldots \in P_D$, $n_k \in P_D$ and $n_k^{\bullet} = \emptyset$.

Proof: If L has n_1 only, i.e., the net contains a place only, $C =< n_1 >$, and the result holds.

If there is a transition n_2 such that $(n_1, n_2) \in F$, because L is conservative, there must be a data place n_3 according to Theorem 2. Similarly, suppose that n_{2i-1} is a data place. If there is a transition n_{2i} such that $(n_{2i-1}, n_{2i}) \in F$, then there must be a data place n_{2i+1} according to Theorem 2. If $n_k \notin P_D$, then $n_k \in T$, it contradicts the above discussion. Thus, $n_k \in P_D$ and $n_k^{\bullet} = \emptyset$.

E. COMPATIBILITY ANALYSIS

According to the definition of liveness, boundedness, and conservativeness, we define the compatibility as follows.

Definition 17: Let $L = \oplus_{i=1}^{m} L_i$ be an ILPN composed of m LPCs L_i, $M_0 = i_1 + \ldots + i_m$, and $M_d = o_1 + \ldots + o_m$. L is compatible if:

1) *LPN* is conservative on P_D;
2) $\forall M \in R(L|_{P_{IC} \cup P_C}, M_0)$: $M_d \in R(L|_{P_{IC} \cup P_C}, M)$; and
3) $\forall t \in T, \exists M \in R(L|_{P_{IC} \cup P_C}, M_0)$: $M[t\rangle$.

In a sense, the compatibility means that a system always reaches the final state and each transition has the opportunity to fire. It also ensure that the data places preserve the resource quantity. This property is useful. When we assemble some components that are from different producers, perhaps some execution sequences cannot ensure a correct end.

Theorem 3: Let $L = \oplus_{i=1}^{m} L_i$ be an ILPN composed of m LPCs L_1-L_m, $M_0 = i_1 + \ldots + i_m$. L is compatible iff (L^*, M_0) is live and bounded and L_1-L_m are conservative on their respective data place set $P_{Dk} \subseteq P_D$, $k \in \mathbb{N}_m^+$.

Proof: We can derive conditions 2 and 3 hold iff (L^*, M_0) is live and bounded similarly with [23, Th. 1]. According to the definition of conservativeness, we can easily derive that L^* is conservative iff L_1-L_m are conservative.

IV. CASE STUDY

Now we construct the e-commerce system as shown in Fig. 8 by composing three *e-commerce sub-systems* modeled by *LPNs* as shown in Fig. 4. It is a trivial extension of LIPN where b is a bridge transition. Transitions *r_order*, *c_order*, *r_payment*, and *s_products* respectively have logic expressions:

$$i_S \wedge (B1\_order \vee B2\_order),$$
$$p_{s2} \wedge (B1\_refuse \vee B1\_accept) \vee (B2\_refuse \vee B2\_accept),$$
$$p_{s2} \wedge (B1\_payment \vee B2\_payment), \text{ and}$$
$$p_{s4} \wedge (B1\_product \vee B2\_product).$$

We suppose that in each transaction, the number of resources sent or received by a buyer is represented by a token. Note that (*s_deal*, *Product*), (*Product*, *s_deal*), and (*Product*, *s_product*) has a weight n_1, and (*r_payment*, *Payment*) has a weight n_2. We have the following results:

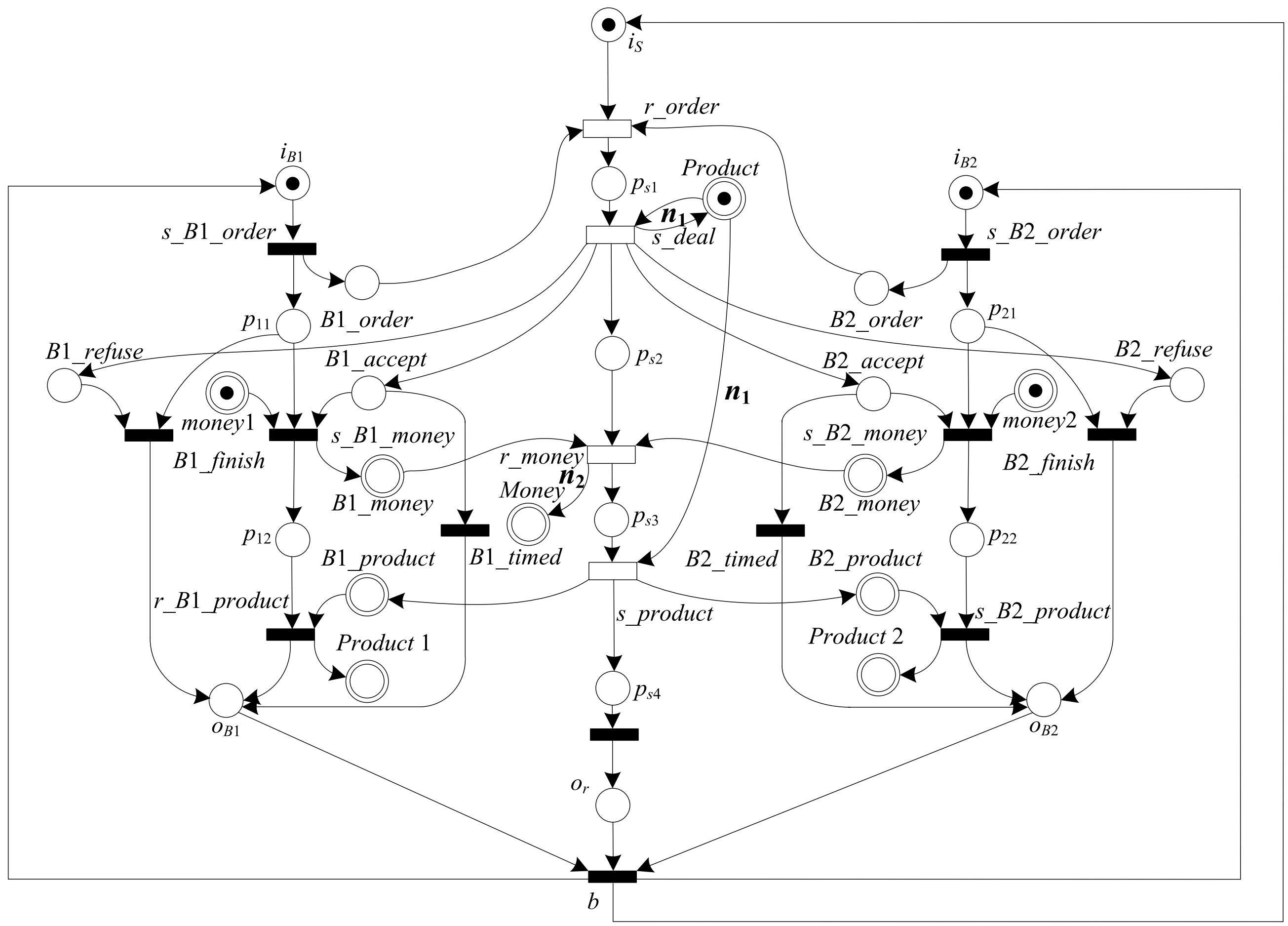

FIGURE 8. E-commerce system model.

If M(Product)≥2, $n_1 = 2$, and $M[s\_deal>M'$ such that M' (*B1_accept*)= M' (*B2_accept*)=1;

if M(Product)=1, $n_1 = 1$, and $M[s\_deal>M'$ such that M' (*B1_accept*)= M' (*B2_rufuse*)=1 or M' (*B2_accept*) = M' (*B1_rufuse*)=1;

if M(Product)=0, $n_1 = 0$, and $M[s\_deal>M'$ such that M'(*B1_refuse*)=M'(*B2_refuse*)=1. We also have *r_payment*∼*s_product*.

If M(*B1_payment*) + M(*B2_payment*)≥1, n_1=M(*B1_payment*) + M(*B2_payment*).

According to Theorems 1-3, we can conclude that the LPN model in Fig. 8 is live, bounded, conservative on data places and thus compatible.

V. CONCLUSION

As an effective formal model and extension of the traditional Petri nets, logical Petri net (LPN) has the capacity to describe and analyze batch processing functions and passing value indeterminacy in cooperative systems. In this paper we discuss the properties of composition of LPNs. We present an interactive logical Petri nets (ILPNs) and analyze their liveness, boundedness and conservativeness. In order to characterize different cooperative abilities for composed systems, compatibility is defined for ILPNs. It reflects the possibility of correct/proper interaction among subsystems. Some relationships among compatibility, liveness, boundedness, and conservativeness are revealed. There are several future works: the compatibility analysis results can be used in other cooperative systems such as those regarding sensor networks [24]; we will explore some net-structure-based or partial-order-based methods [25] to analyze the properties such as the compatibility, soundness, and deadlock of LPN.

APPENDIX

PROOF FOR THEOREM 1

Proof (Sufficiency): We discuss two cases in the following, respectively:

1) Because $\cup_{p''\in\{p_a,p_b\}} ({}^\bullet p'' \cup p''^\bullet) = T_I \cup T_O$, i.e., $({}^\bullet p_a \cup p_a^\bullet \cup^\bullet p_b \cup p_b^\bullet) \subseteq T_I \cup T_O$, the input and output transitions of the internal channel places are logical transitions. Given $M^*(p'') = 0$, if a logical transition fires, the logical expression $f = p' \wedge ({}_\bullet T_\bullet \vee p'')$ is satisfied, and $\exists p'' \in \{p_a, p_b\}$ contains no token. Given ${}^\bullet p_a \sim p_a^\bullet \sim^\bullet p_b \sim p_b^\bullet$, each logical transition will

fire under the restriction of the logical transitions $f = p' \wedge ({}_\bullet T_\bullet \vee p'')$ and $\forall p'' \in\{p_a,p_b\}$ contains no token. Because $N_j = (P_j, T_j, F_j)$ is a logic process with $\forall p \in P_j\backslash\{i_j, o_j\}$: $|{}^\bullet p| = |p^\bullet| = 1$, and $\forall t \in T_j$: $|{}^\bullet t| = |t^\bullet| = 1$, and (N_j^E, M_{j0}) is live and safe, that is $M_{j0} = i_j$ and $M_{j0}[\sigma_j\rangle M_j'$, where $M_j' = o_j$. let $\sigma_=\sigma_1\sigma_2$ be a connection of σ_1 and σ_2. We have in L^*, $M_j = i_1+i_2$, $M_j[\sigma_j\rangle M'$ with $M' = o_1 + o_2$. Also, Thus L^* is live. Similarly, we can prove it is bounded.

2) If condition 1 is not satisfied, suppose that $\exists p'' \in \{p_a,p_b\}$: ${}^\bullet p'' \in T_D$ and $p''^\bullet \in T_I \cup T_O$ (respectively, $p''^\bullet \in T_D$ and ${}^\bullet p'' \in T_I \cup T_O$), $M^* \in$R(M_0): $M^*(p'')$=1 and $f|_M^* = {}_\bullet T_\bullet$. Obviously firing $p''^\bullet$ (respectively, ${}^\bullet p''$) removes (respectively, deposits) a token from (respectively, in) p''. $\forall t \in \cup_{p''\in\{p_a,p_b\}}({}^\bullet p'' \cup p''^\bullet)$: $|{}^\bullet t| = |t^\bullet| = 1$. N_j is an logic process with $\forall p \in P_j\backslash\{i_j, o_j\}$: $|{}^\bullet p| = |p^\bullet| = 1$, and $\forall t \in T_j$: $|{}^\bullet t| = |t^\bullet| =1$. (N_j^*, M_{j0}) is live and safe, that is $M_{j0} = i_j$ and $M_{j0}[\sigma_j\rangle M$: $M({}^{\bullet\bullet}p'')$=1. Thus t is enabled.
Also, $\forall t \in T\backslash \cup_{p''\in\{p_a,p_b\}} ({}^\bullet p'' \cup p''^\bullet)$: $|{}^\bullet t| = |t^\bullet| = 1$. N_j is an logic process and (N_j^*, M_{j0}) is live and safe, and t is enabled. Thus, L^* is live and bounded.

3) If there exists a circle and condition 1 is not satisfied. There must be a transition $t \in {}^\bullet p_a \cup p_a^\bullet \cup^\bullet p_b \cup p_b^\bullet$: $t \in T_D$. We have that a transition connecting $p' \in \{p_a, p_b\}$ is a logical transition such that $M^{*\prime} \in$R(M_0): $M^{*\prime}(p') = 1$. On the other hand, in order to exclude circular wait, there exists another place $p'' \in\{p_a,p_b\}\backslash p'$ satisfying that ${}^\bullet p'' \cup p''^\bullet \in T_I \cup T_O$ and $M^* \in$ R(M_0): $M^*(p'') = 0$. Thus, the condition holds.

(Necessity) L^* is live and bounded. We verify the three conditions by dividing them into two parts: there is a circle or no circle in L^*. Suppose there exists a circle $C =< n_1, n_2, \ldots, n_k >$. we use the contradiction method. If conditions 1 and 3 do not hold, i.e., (a) no $p'' \in\{p_a,p_b\}$: ${}^\bullet p'' \cup p''^\bullet \in T_I \cup T_O$ and $M^* \in$R(M_0): $M^*(p'') = 0$, so there must be a circular wait or a token left in p'', and L^* is not live; (b) $\exists p' \in\{p_a, p_b\}$: ${}^\bullet p' \in T_D$ and $p'^\bullet \in T_I \cup T_O$ (or $p'^\bullet \in T_D$ and ${}^\bullet p' \in T_I \cup T_O$), $M^{*\prime} \in$R(M_0): $M^{*\prime}(p') = 1$. There must be a token left in p'', and L^* is not live. These contradict with the liveness of L^*. Thus, if there exists a circle, conditions 1 or 3 hold. If there exists no circle, we have $\exists p'' \in\{p_a,p_b\}$: ${}^\bullet p'' \in T_D$ and $p''^\bullet \in T_I\cup T_O$ (or $p''^\bullet \in T_D$ and ${}^\bullet p'' \in T_I\cup T_O$), $M^* \in$R(M_0): $M^*(p'') = 1$ and $f|_M^* =_\bullet T_\bullet$, or condition 2 holds. ■

PROOF FOR THEOREM 2

Proof (Necessity): According to Definition 16, L is conservative on P_D. We have $\forall M \in R(M_0)$,

$$\sum\nolimits_{p_i\in P_D} M(p_i) = \sum\nolimits_{p_j\in P_D} M_0(p_j)$$

We now discuss how to preserve the above formula after a transition t fires, i.e., $M[t > M'$. Tokens in input and output places change while those in others remain unchanged.

(1) If $t \in T_D$: let $P_D' = ({}^\bullet t \cup t^\bullet) \cap P_D$, and we have

$$\sum\nolimits_{p_k\in p_D'} M'(p_k) = \sum\nolimits_{p_i\in P_D'} M(p_k)$$

According to the firing rules in Definition 5,

$$\sum\nolimits_{p_k\in p_D'} M'(p_k) = \sum\nolimits_{p_k\in P_D'} M(p_k) - \sum\nolimits_{p_i\in P_D'\cap^\bullet t} W(p_i, t) + \sum\nolimits_{p_j\in P_D'\cap t^\bullet} W(t, p_j)$$

Thus,

$$\sum\nolimits_{p_i\in P_D'\cap^\bullet t} W(p_i, t) = \sum\nolimits_{p_j\in P_D'\cap t^\bullet} W(t, p_j)$$

Because $W(p, t) = 0$ if $p \notin {}^\bullet t$ and $W(t, p) = 0$ if $p \notin t^\bullet$, we have

$$\sum\nolimits_{p_i\in P_D} W(p_i, t) = \sum\nolimits_{p_j\in P_D} W(t, p_j)$$

(2) If $t \in T_I$:
if $p_i \in P_D$, $p_i \in {}^\bullet t$, and $M(p_i) > 0$, according to the firing rule of a logical input transition as shown in Definition 5,

$$M'(p_i) = M(p_i) - W(p_i, t);$$

if $M(p_i)$ =0 and $M'(p_i) = 0$; and
if $p_j \in P_D$ and $p_j \in t^\bullet$,

$$M'(p_j) = M(p_j) + W(t, p_j)$$

Hence we have

$$\sum\nolimits_{p_k\in p_D} M'(p_k) = \sum\nolimits_{p_k\in P_D,M(p_k)>0} M(p_k) - \sum\nolimits_{p_i\in P_D\cap^\bullet t,M(p_k)>0} W(p_i, t) + \sum\nolimits_{p_j\in P_D\cap t^\bullet} W(t, p_j)$$

Thus,

$$\sum\nolimits_{p_i\in P_D} W(p_i, t)\rho(p_i) = \sum\nolimits_{p_j\in P_D} W(t, p_j),$$

where if $M(p_i) > 0$, $\rho(p_i) = 1$, else, $\rho(p_i) = 0$.

(3) If $t \in T_O$:
if $p_i \in P_D$, $p_i \in {}^\bullet t$, according to the firing rule of a logical input transition as shown in Definition 5,

$$M'(p_i) = M(p_i) - W(p_i, t);$$

if $p_j \in P_D$ and $p_j \in t^\bullet$, because $\forall p \in t^\bullet$ satisfy $O(t)|_{M'-M} = {}_\bullet T_\bullet$, we have that if $M > M'(p_j)$, then $M'(p_j) = M(p_j)$-$W(t,p_j)$
Then,

$$\sum\nolimits_{p_k\in p_D} M'(p_k) = \sum\nolimits_{p_k\in P_D} M(p_k) - \sum\nolimits_{p_i\in P_D\cap^\bullet t} W(p_i, t) + \sum\nolimits_{p_j\in P_D\cap t^\bullet,M'(p_j)-M(p_j)>0} W(t, p_j)$$

Thus,

$$\sum\nolimits_{p_i\in P_D} W(p_i, t) = \sum\nolimits_{p_j\in P_D} W(t, p_j)\rho(p_j),$$

where if $M[t\rangle M'$ and $M > M'(p_i)$, then $\rho(p_j) = 1$; else, $\rho(p_j) = 0$.

Sufficiency proof can be obtained by following the transition firing rules. For simplicity, if each transition satisfies conditions 1-3, their firing preserves the token counts in input and output data places. As a result, the number of tokens in data places remains unchanged in the whole net, i.e., L is conservative.

REFERENCES

[1] C. A. Petri, *Kommunikation Mit Automaten*. Bonn, Germany: Institute fur Instrumentelle Mathematik, 1962.

[2] Y. Y. Du, C. J. Jiang, and M. C. Zhou, "A Petri-net-based correctness analysis of Internet stock trading systems," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 38, no. 1, pp. 93–99, Jan. 2008.

[3] H. Hu, M. Zhou, and Z. Li, "Liveness enforcing supervision of video streaming systems using nonsequential Petri nets," *IEEE Trans. Multimedia*, vol. 11, no. 8, pp. 1457–1465, Dec. 2009.

[4] L. Xu, H. Liu, S. Wang, and K. Wang, "Modeling and analysis techniques for cross-organizational workflow systems," *Syst. Res. Behavioral Sci.*, vol. 26, no. 3, pp. 367–389, 2009.

[5] J. C. Wang, Y. Deng, and G. Xu, "Reachability analysis of real-time systems using time Petri nets," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 30, no. 5, pp. 725–736, May 2000.

[6] R. Robidoux, H. Xu, L. Xing, and M. C. Zhou, "Automated modeling of dynamic reliability block diagrams using colored Petri nets," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 40, no. 2, pp. 337–351, Feb. 2010.

[7] K. Jensen and L. M. Kristensen, *Coloured Petri Nets: Modeling and Validation of Concurrent Systems*. New York, NY, USA: Springer, 2009.

[8] W. Van der Aalst, "Loosely coupled interorganizational workflows: Modeling and analyzing workflows crossing organizational boundaries," *Inf. Manage.*, vol. 37, no. 1, pp. 67–75, 2000.

[9] Z. W. Li and M. C. Zhou, *Deadlock Resolution in Automated Manufacturing Systems: A Novel Petri Net Approach*. New York, NY, USA: Springer, 2009.

[10] B. Hruz and M. C. Zhou, *Modeling and Control of Discrete Event Dynamic Systems*, London, U.K.: Springer, 2007.

[11] M. C. Zhou and M. P. Fanti, Ed., *Deadlock Resolution in Computer-Integrated Systems*, New York, NY, USA: Marcel Dekker, 2005.

[12] A. L. Feller, T. Wu, D. L. Shunk, and J. W. Fowler, "Petri net translation patterns for the analysis of eBusiness collaboration messaging protocols," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 39, no. 5, pp. 1022–1024, May 2009.

[13] L. Qi, W. J. Luan, and M. C. Zhou, "Emergency traffic-light control system design for intersections subject to accidents," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 1, pp. 170–183, Sep. 2016.

[14] Y. Y. Du and B. Q. Guo, "Logic Petri nets and equivalency," *Inf. Technol. J.*, vol. 8, no. 1, pp. 95–100, 2009.

[15] Y. Y. Du and C. J. Jiang, *Formal Representation and Analysis of Batch Stock Trading Systems by Logical Petri Net Workflows*, (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2002, pp. 221–225.

[16] Y. Du, C. Jiang, and M. Zhou, "A Petri net-based model for verification of obligations and accountability in cooperative systems," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 39, no. 2, pp. 299–308, Mar. 2009.

[17] N. Busi, "Analysis issues in Petri nets with inhibitor arcs," *Theor. Comput. Sci.*, vol. 275, nos. 1–2, pp. 127–177, 2002.

[18] W. H. Sanders and J. F. Meyer, "Stochastic activity networks: Formal definitions and concepts," *Lectures on Formal Methods and Performance Analysis* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2001, pp. 315–343.

[19] Y. Y. Du, L. Qi, and M. C. Zhou, "Analysis and application of logical Petri nets to E-commerce systems," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 44, no. 4, pp. 468–481, Apr. 2014.

[20] Y. Y. Du, L. Qi, and M. C. Zhou, "A vector matching method for analysing logic Petri nets," *Enterprise Inf. Syst.*, vol. 5, no. 4, pp. 449–468, 2011.

[21] K. Wang and L. Y. Yuan, "First-order logic characterization of program properties," *IEEE Trans. Knowl. Data Eng.*, vol. 6, no. 4, pp. 518–533, Aug. 1994.

[22] S. Clifford, L. D. Robert, and B. Kenneth, *Discrete Mathematics for Computer Scientists*, Boston, MA, USA: Addison-Wesley, 2011.

[23] G. Liu, C. Jiang, M. Zhou, and P. Xiong, "Interactive Petri nets," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 43, no. 2, pp. 291–302, Mar. 2013.

[24] D. Zhang, P. Shi, W.-A. Zhang, and L. Yu, "Energy-efficient distributed filtering in sensor networks: A unified switched system approach," *IEEE Trans. Cybern.*, to be published, doi: 10.1109/TCYB.2016.2553043.

[25] G. J. Liu and C. J. Jiang, "Net-structure-based conditions to decide compatibility and weak compatibility for a class of inter-organizational workflow nets," *Sci. China Inf. Sci.*, vol. 58, no. 7, pp. 1–16, 2015.

WENJING LUAN (S'16) received the B.S. and M.S. degrees from Shandong University of Science and Technology, Qingdao, China, in 2009 and 2012, respectively. She is currently pursuing the Ph.D. degree with the Department of Computer Science and Technology, Tongji University, Shanghai, China. Her current research interests include Petri nets and its application in discrete event systems, location-based social networks, recommender systems, and intelligent transportation systems. She received the Best Student Paper Award-Finalist from the 2016 13th IEEE International Conference on Networking, Sensing and Control.

LIANG QI (S'16) received the B.S. and M.S. degrees from Shandong University of Science and Technology, Qingdao, China, in 2009 and 2012, respectively. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Technology, Tongji University, Shanghai, China. From 2015 to 2017, he was a Visiting Doctoral Student with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, USA. He has authored more than a dozen papers in journals and conference proceedings. His current research interests include Petri nets and its application in discrete event systems, and Intelligent Transportation Systems.

YUYUE DU received the B.S. degree from Shandong University, Jinan, China, in 1982, the M.S. degree from Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 1991, and the Ph.D. degree in computer application from Tongji University, Shanghai, China, in 2003. He is currently a Professor of the Shandong University of Science and Technology, Qingdao, China. He has led or participated in more than 15 projects supported by the National Natural Science Foundation, the National Key Basic Research Development Program, and other important and key projects at provincial levels. He has authored more than 170 papers in journals and conference proceedings, such as the IEEE Transactions on Systems, Man and Cybernetics, Part A, the IEEE Transactions on System, Man and Cybernetics, Part C, the *Information Science*, and the *Enterprise Information System*. His research interests are in formal engineering methods, Petri nets, real-time systems, Web services, and workflows. He is a member of the Professional Committee of Petri Nets of the China Computer Federation. He received two prizes from Science and Technology Advancement Awards at the province level and the Excellent Doctoral Dissertation Award from Shanghai, China.