IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON INTELLIGENT INFORMATION SERVICES

Received January 21, 2020, accepted February 8, 2020, date of publication February 17, 2020, date of current version February 27, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2974340

Shared P-Type Logic Petri Net Composition and Property Analysis: A Vector Computational Method

LIANG QI[1], (Member, IEEE), WENJING LUAN[1], (Member, IEEE), XIAOYU SEAN LU[2], (Member, IEEE), AND XIWANG GUO[3], (Member, IEEE)

[1]College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China
[2]Department of Electrical and Computer Engineering, Stevens Institute of Technology, Hoboken, NJ 07030, USA
[3]College of Computer and Communication Engineering, Liaoning Shihua University, Fushun113001, China

Corresponding authors: Liang Qi (qiliangsdkd@163.com) and Wenjing Luan (wenjingmengjing@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61903229 and Grant 61973180, in part by the Natural Science Foundation of Shandong Province under Grant ZR2019BF004 and Grant ZR2019BF041, in part by the SDUST Research Fund under Grant 2019TDJH102, and in part by the Scientific Research Foundation of the Shandong University of Science and Technology for Recruited Talents under Grant 2019RCJJ012, in part by the Liaoning Revitalization Talents Program under Grant XLYC1907166, in part by the Liaoning Province Department of Education Foundation of China under Grant L2019027, and in part by the Liaoning Province Dr. Research Foundation of China under Grant 20170520135.

ABSTRACT Petri net (PN) is an effective modeling and analysis tool for discrete event systems. By attaching a first-order logic predicate logic formula to a transition in a PN, a high-level Petri net named Logic Petri Net (LPN) is obtained. LPN has been proved to have equivalent modeling capability with inhibition Petri nets but keeps simpler net structures than the latter. It has the advantage of modeling the cooperative systems with the function to process batch and indeterminate resources. This paper proposes a vector computational method for LPN compositional analysis. It studies the composition of LPNs with the shared P-type subnets. Each logical expression is transformed into a unique disjunctive normal one and then into a unique set of vectors. A vector computational method is proposed such that the properties of the composition of the shared P-type subnets such as liveness, boundedness, and reversibility are verified. An E-commerce system with customers, merchants, and a third-party is constructed to illustrate the method. This paper can improve the state of the art in the theory of LPNs.

INDEX TERMS Petri nets, logic petri net, P-type Petri net composition, property analysis, cooperative systems.

I. INTRODUCTION

Petri net (PN) proposed by C. A. Petri [1] is a modeling tool to describe discrete event systems and concurrent and parallel systems intuitively and effectively. In the past half-century, PN and its extensions such as colored PN (CPN) [2], [3], Timed PN (TPN) [4], and stochastic PN (SPN) [5] have been proposed and used for modeling, simulation, and analysis in a great number of application areas such as computer operating systems [6], flexible manufacturing [7], communications, business processes [8], and transportation systems [9]–[11]. Because of their well mathematical foundation and graphical expression, PN and its extensions are used to describe, analyze, and make sure the systems have desirable properties such as liveness, boundedness, and reversibility before the installment of the systems. High-level PNs such as CPN, TPN, and SPN can model complex systems with more concise net structures than the classical PNs. However, they have equivalent modeling capability with classical PNs. Thus, in order to strengthen the capability of the model, augmented PNs such as inhibition Petri nets (IPNs) [12] are proposed.

A cooperative system usually needs a sub-system to interact by exchanging messages with other sub-systems. During the intersection processes, a sub-system may process data from several other sub-systems simultaneously. Thus, it requires that the system is with a function of processing batch resources. Meanwhile, when a sub-system waits for feedback from the other sub-systems, the latter may not give a response. Another condition is that a system does not respond

The associate editor coordinating the review of this manuscript and approving it for publication was Lu Liu.

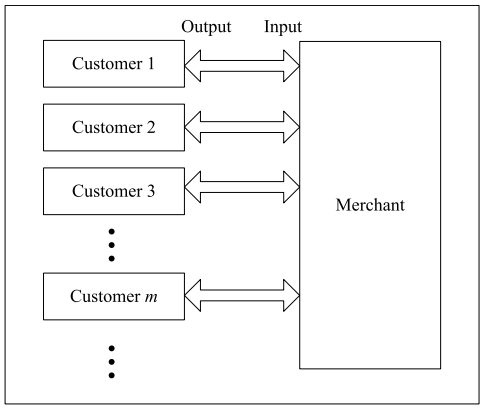

FIGURE 1. An E-commerce system.

in a certain time interval which means a delay happens. Thus, there always exist indeterminate conditions. An example of the aforementioned situations is a basic E-commerce system that realizes the trade between a merchant and several customers as shown in Fig. 1. From the figure, we can see that the merchant system needs to deal with several customers simultaneously and handles a batch of resources from the customers in a time interval. In the trading process, indeterminate resource intersection from both the merchant and customers always exists. For example, customers can order products of different quantity of products that may be ordered by different customers, shortage of products could be faced by the merchant, or a customer may cancel an order during the whole trading progress event if they have received their products. Thus, we should design some special models to describe and analyze the batch and indeterminate processing of data, which is significant from theoretical and practical aspects.

Logic Petri net (LPN) is such a model that describes and analyzes the batch and indeterminate processing of data [12]. It is obtained by attaching a first-logic predicate logic formula to a transition in a classical PN in [13]. Du and Guo [12] verify that each LPN has an equivalent inhibition Petri net (IPN) model. In an LPN, transitions are divided into two groups, i.e., normal transitions and logic transitions. Normal transitions are the same as those in classical PNs, and a logic transition has a first-order predicate logic expression with connected input or output places as the predicate. In detail, logic transitions are further divided into logic input transitions and output transitions. The former has a logic expression of their input places named logic input expression while the latter has a logic expression of their output places named as a logic output transition. A logic input expression restricts the firing of the transition by checking if the tokens in the input places satisfy the expression. A logic output expression restricts the new state such that after the transition fires the token in its output places satisfy the logic transition. Fortunately, batch and indeterminate resource processing can be modeled by the logic transitions in LPNs. Thus, LPNs have enhanced capability for modeling more complex systems. Till now, many studies have used LPNs to model and analyze E-commerce systems [13] and business processes [14]. Sanders and Meyer propose stochastic activity networks that have predicates and functions on markings of places [15]. They can describe the processing of batch and indeterminate data. However, their analysis is very complicated because each transition could have two or more predicates or functions. In [16], we present a vector matching method to analyze the properties of LPNs such as reachability, liveness, and reversibility. In [17], we study interactive logic Petri nets and design some strategies to ensure good properties. One of the important properties is compatibility which concerns if the subsystems in the cooperative system have proper interactions with each other. It characterizes cooperative abilities in practice and analyzes the relationships of compatibility with liveness and boundedness. In [18], we present a systematic LPN synthesis approach for cooperative systems by gradually refining basic design modules. It designs some control strategies to guarantee that the synthesized LPNs have desirable properties. This work is of significance in the sense that it provides industrial engineers and academic researchers a methodology of applying LPNs to modeling and analysis of cooperative systems. Till now, there is no good method to verify the composition of LPN models. Thus, this paper proposes a vector computational method for analyzing the compositional properties of LPNs. It studies the composition of LPNs with the shared P-type subnets. Each logical expression is transformed into a unique disjunctive normal one and then into a unique set of vectors. A vector computational method is proposed such that the properties of the composition of LPN models with two types such as liveness, boundedness, and reversibility are verified.

The rest of the paper is organized as follows. Section II briefly reviews the basic concepts of LPNs. Section III shows a vector computational method where each logical expression is transformed into a unique disjunctive normal one and then turned into a unique set of vectors. Section IV describes the composition of shared P-type LPN models. In Section V, an E-commerce system composed of models of customers, merchants, and a third-party is constructed to illustrate the proposed approach. Section V concludes this paper and discusses future work.

II. PRELIMINARIES

In this section, some concepts of PNs [19]–[27] and LPNs [12] are briefly presented. Firstly, we give some notations that will be used in the paper. Let $\mathbf{R}_0$ and $\mathbf{N}$ denote the real number set and the natural number set, respectively, and $\mathbf{N}^+$ be the set of positive integer numbers.

Definition 1: A six-tuple $PN = (P, T, I, O, M, M_0)$ is a Petri net, where

1) $P = \{p_1, \ldots, p_n\}$ denotes a set of places with $n \in \mathbf{N}^+$;

2) $T = \{t_1, \ldots, t_m\}$ denotes a set of transitions where $P \cup T \neq \emptyset$, $P \cap T = \emptyset$, and $m \in \mathbf{N}^+$;

3) I: $P \times T \rightarrow \mathbf{N}$ denotes a weight function on a directed arc from a place $p \in P$ to a transition $t \in T$, where

(a) $I(p, t) = 0$ if there is no arc from p to t; and

(b) $I(p, t) > 0$ if there is an arc from p to t and its weight is $I(p, t)$;

4) O: $P \times T \rightarrow \mathbf{N}$ denotes a weight function on a directed arc from a transition $t \in T$ to a place $p \in P$, where

(a) $O(p, t) = 0$ if there is no arc from t to p; and

(b) $O(p, t) > 0$ if there is an arc from t to p and its weight is $O(p, t)$; and

5) M: $P \rightarrow \mathbf{N}$ is a marking function, where $M(p)$ denotes the number of tokens in $p \in P$, and M_0 is an initial marking.

The transition firing rules include:

(a) t is enabled at M denoted by $M[t\rangle$ if for each $p \in P$: $M(p) \geq I(p, t)$; and

(b) if $M[t\rangle$, t firs generating a marking M' denoted by $M[t\rangle M'$, where for $\forall p \in P$, we have $M'(p) = M(p)$-$I(p, t) + O(p, t)$.

We say that the new marking M' is reachable from M. If there exists a transition sequence denoted by $\sigma = t_1 \ldots t_l$ and $M[t_1\rangle M_1, \ldots$, and $M_{l-1}[t_l\rangle M_l$, we say that M_l is reachable from M, which is represented by $M[\sigma\rangle M_l$. In the following content, we denote reachable marking set from M as $R(M)$ where $M \in R(M)$. ${}^\bullet x = \{y | I(y, x) > 0\}$ and $x^\bullet = \{y | O(x, y) > 0\}$ respectively denote the pre-set and post-set of $x \in P \cup T$. In the following content, we denote $M = \sum_{p \in P} M(p)\bullet p$. Some properties of PNs are then defined next.

Definition 2: Let PN be a marked Petri net. 1) PN is called k-bounded when for each place $p \in P$, $\forall M \in R(M_0)$ such that $M(p) \leq k$; 2) PN is safe, when for each place $p \in P$, $\forall M \in R(M_0)$ such that $M(p) \leq 1$; 3) PN is live if for $\forall M \in R(M_0)$, $\forall t \in T$: $M' \in R(M)$ and $M'[t\rangle$; and 4) PN is reversible if $\forall M \in R(M_0)$: $M_0 \in R(M)$.

Before presenting the definition of LPN, we give the following notations by inducing first-order predicate logic to PNs. In a first-order predicate logic expression [28], [29], there are three kinds of connectives including a logic disjunction "$\vee$", a conjunction "$\wedge$", and a negation "$\neg$". ${}_\bullet T_\bullet$ and ${}_\bullet F_\bullet$ respectively represent "true" and "false" logic values with ${}_\bullet T_\bullet = \neg {}_\bullet F_\bullet$ and ${}_\bullet F_\bullet = \neg {}_\bullet T_\bullet$. We attach first-order predicate logic to PNs by treating each place as one-predicate logic as follows.

Definition 3: Given $PN = (P, T, I, O, M, M_0)$, for $\forall p \in P$ and $\forall t \in T$, p^t represents a predicate logic of p on t, and a logic value of p^t at M is denoted and defined as

$$p^t(M) = \begin{cases} {}_\bullet T_\bullet, & \text{if } p \in {}^\bullet t \text{ and } M(p) \geq I(p, t) \\ {}_\bullet T_\bullet, & \text{if } p \in t^\bullet \text{ and } M(p) \geq O(p, t) \\ {}_\bullet F_\bullet, & \text{else} \end{cases} \quad (1)$$

We can rephrase the transition firing rule that t is enabled at M denoted by $M[t\rangle$ if for each $p \in P$: $p^t(M) = {}_\bullet T_\bullet$. Thus, for $t \in T$, $M[t\rangle$ iff $\forall p \in {}^\bullet t$: $p^t(M) = {}_\bullet T_\bullet$ and $M[t\rangle$ iff $p_1^t(M) \wedge p_2^t(M) \wedge \ldots \wedge p_k^t(M) = {}_\bullet T_\bullet$ and ${}^\bullet t = \{p_1, p_2, \ldots, p_k\}$.

Now we attach a transition $t \in T$ with a first-order predicate logic expression on its connected places. The expressions are divided into two categories, i.e., logic input expressions on the input places of t and logic output expressions on the output places of t, which are formally denoted by $f({}^\bullet t)$ and $g(t^\bullet)$, respectively. $f({}^\bullet t)$ has atomic predicates $p_{11}^t, p_{12}^t, \ldots$, and p_{1k}^t connected by connectives where ${}^\bullet t = \{p_{11}, p_{12}, \ldots, p_{1k}\}$; $g(t^\bullet)$ has atomic predicates $p_{21}^t, p_{22}^t, \ldots$, and p_{2l}^t connected by connectives where $t^\bullet = \{p_{21}, p_{22}, \ldots, p_{2l}\}$. Let $L({}^\bullet t)$ and $L(t^\bullet)$ denote all logic expressions on the input and output places of t and we have $f({}^\bullet t) \in L({}^\bullet t)$ and $g(t^\bullet) \in L(t^\bullet)$. The logic value of $f({}^\bullet t)$ and $g(t^\bullet)$ at M, denoted by $f_M({}^\bullet t)$ and $g_M(t^\bullet)$, respectively, can be obtained given the logic values of all atomic predicates at M.

Now we give the formal definition of logic Petri net (LPN) [12], [18].

Definition 4: $LPN = (P, T, I, O, M, f, g, \tau)$ is a logic Petri net (LPN) where

1) $P = P_R \cup P_C$ is place set, P_R and P_C respectively represent a set of resource and control places, where $P \neq \emptyset$ and $P_R \cap P_C = \emptyset$;

2) T is a transition set that can be divided into a set of traditional transitions, and logic transitions including a set of logic input transitions and a set of logic output transitions, denoted by T_D, T_I, and T_O, respectively. $T = T_D \cup T_I \cup T_O \neq \emptyset$, $T_D \cap T_I = T_D \cap T_O = T_I \cap T_O = \emptyset$, and $\forall t \in T_I \cup T_O$: ${}^\bullet t \cap t^\bullet = \emptyset$;

3) I: $P \times T \rightarrow \mathbf{N}$ denotes a weight function on a directed arc from a place $p \in P$ to a transition $t \in T$, where

(a) $I(p, t) = 0$ if there is no arc from p to t; and

(b) $I(p, t) > 0$ if there is an arc from p to t and its weight is $I(p, t)$;

4) O: $P \times T \rightarrow \mathbf{N}$ denotes a weight function on a directed arc from a transition $t \in T$ to a place $p \in P$, where

(a) $O(p, t) = 0$ if there is no arc from t to p; and

(b) $O(p, t) > 0$ if there is an arc from t to p and its weight is $O(p, t)$; and

5) M: $P \rightarrow \mathbf{N}$ is a marking function, where $M(p)$ denotes the number of tokens in $p \in P$, and M_0 is an initial marking.

6) f: ${}^\bullet T \rightarrow L({}^\bullet T)$ is a logic input function, where ${}^\bullet T = \{{}^\bullet t | t \in T\}$, $L({}^\bullet T) = \{L({}^\bullet t) | t \in T\}$, and if $t \in T_D \cup T_O$, $f({}^\bullet t) = p_1^t \wedge p_2^t \wedge \ldots \wedge p_m^t$ and ${}^\bullet t = \{p_1, p_2, \ldots, p_m\}$;

7) g: $T_O^\bullet \rightarrow L(T^\bullet)$ is a logic output function, where $T^\bullet = \{t^\bullet | t \in T\}$, $L(T^\bullet) = \{L(t^\bullet) | t \in T\}$, and if $t \in T_D \cup T_I$, $g(t^\bullet) = p_1^t \wedge p_2^t \wedge \ldots \wedge p_m^t$ and $t^\bullet = \{p_1, p_2, \ldots, p_m\}$; and

8) τ: $T \rightarrow \mathbf{R}^+$ attaches a time delay to each transition where an enabled transition $t \in T$ fires in time interval $\tau(t)$.

The transition firing rules is that for $\forall t \in T_I$, $M[t\rangle$ if $f_M({}^\bullet t) = {}_\bullet T_\bullet$, and after $\tau(t)$, $M[t\rangle M'$ where

(a) if $t \in T_D$, then for $\forall p \in P$, we have $M'(p) = M(p)$-$I(p, t) + O(p, t)$.

(b) if $t \in T_I$, then if $p \in {}^\bullet t$ and $p^t(M) = {}_\bullet F_\bullet$, then $M'(p) = M(p)$, else, $M'(p) = M(p) + O(p, t)$-$I(p, t)$; and

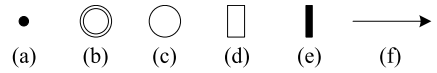

FIGURE 2. Drawing of elements in an LPN: (a) token, (b) data place, (c) control place, (d) logic transition, (e) ordinary transition, and (f) directed arc.

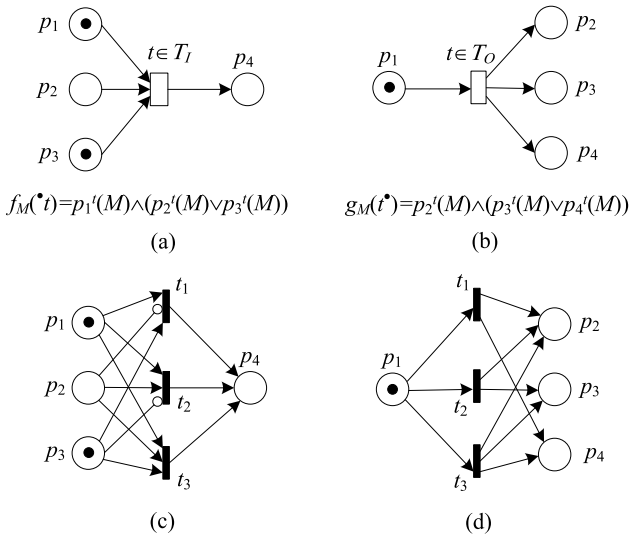

FIGURE 3. LPNs and their equivalent IPNs: (a) An LPN with an input logic transition, (b) An LPN with an output logic transition, (c) equivalent IPN of LPN in (a), and (d) equivalent IPN of LPN in (b).

(c) $g_{M'}(t^\bullet) = {}_\bullet T_\bullet$, and if $t \in T_O, p \in t^\bullet$ and $p^t(M') = {}_\bullet F_\bullet$, then $M'(p) = M(p)$, else, $M'(p) = M(p) + O(p, t)$-$I(p, t)$.

In the above definition, a transition in LPN is combined with a predicate logic. Specifically, if it is a logic input transition, it fires at a marking M only if its predicate logic of the transition's input places is true at M; if it is a logic output transition, it fires at a marking M generating a new marking M' and the predicate logic of the transition's output places is true at M' after the transition fires. Thus, logic input and output transitions endorse the batch and indeterminate resource modeling in cooperative systems. Notice that $f(^\bullet t) = p_1^t \wedge p_2^t \wedge \ldots \wedge p_m^t$ or $g(t^\bullet) = p_1^t \wedge p_2^t \wedge \ldots \wedge p_m^t$, t is not a logic input or output transition where $^\bullet t = \{p_1, p_2, \ldots, p_m\}$. Besides, time is always involved in real systems. Thus, we attach the time information to transitions. When a transition is enabled, the current state is kept for a certain time interval $\tau(t)$. After that, it fires. t is an immediate transition if $\tau(t) = 0$, and otherwise, it is a deterministic transition. In LPNs, a token, resource place, control place, logic transition, ordinary transition, and directed arc are drawn in Fig. 2, respectively. In this paper, the LPNs are restricted with I: $P \times T \rightarrow \{0, 1\}$ and O: $P \times T \rightarrow \{0,1\}$; and the capacity of a control place is 1, i.e., $\forall M \in R(M_0)$ and $\forall p \in P_C$: $M(p) \leq 1$. To ensure this, we give a capacity restriction to each place.

We give some examples of LPNs. In Fig. 3(a), the LPN has a logic input transition t with $f(^\bullet t) = p_1^t \wedge (p_2^t \vee p_3^t)$. Given $M = (1, 0, 1, 0)$, $p_1^t(M) = {}_\bullet T_\bullet$, $p_2^t(M) = {}_\bullet F_\bullet$, and $p_3^t(M) = {}_\bullet T_\bullet$. $f_M(^\bullet t) = {}_\bullet T_\bullet \wedge ({}_\bullet F_\bullet \vee {}_\bullet T_\bullet) = {}_\bullet T_\bullet$ and thus $M[t >$. After $\tau(t)$, $M[t > M^*$, where according to the firing rules $M^* = (1, 0, 1, 0)$. Similarly, given $M' = (1, 1, 1, 0)$ and $M'' = (0, 1, 1, 0)$, we have $f_{M'}(^\bullet t) = f_{M''}(^\bullet t) = {}_\bullet T_\bullet$. We can see that though the markings of input places of t are different, after t fires the generated marking is the same, i.e., only p_4 has a token. In Fig. 3(b), the LPN has an output logic transition $t \in T_O$ with $f(t^\bullet) = {}^t p_2 \wedge ({}^t p_3 \vee {}^t p_4]$ and $M[t >$ where $M = (1, 0, 0, 0)$. After $\tau(t)$, $M[t > M^*$, in order to ensure that $f_M(t^\bullet) = {}_\bullet T_\bullet$, we have that $p_2^t(M) = {}_\bullet T_\bullet$, $p_3^t(M) = {}_\bullet T_\bullet$, and $p_4^t(M) = {}_\bullet T_\bullet$; or $p_2^t(M) = {}_\bullet T_\bullet$, $p_3^t(M) = {}_\bullet F_\bullet$, and $p_4^t(M) = {}_\bullet T_\bullet$; or $p_2^t(M) = {}_\bullet T_\bullet$, $p_3^t(M) = {}_\bullet T_\bullet$, and $p_4^t(M) = {}_\bullet F_\bullet$. Thus, $M^* = (0, 1, 1, 1)$, or $M^* = (0, 1, 0, 1)$, or $M^* = (0, 1, 1, 0)$.

Inhibition Petri nets (IPNs) [12] are an augmented PN by attaching a traditional PN with inhibitor arcs. An inhibitor arc is a non-directed arc. A transition can fire only if there is no token in the places connected with the inhibitor arcs. The equivalence between an LPNs and an IPN has been proved in [12]. Given an LPN, a corresponding equivalent IPN can be obtained. For example, LPNs in Fig. 3 (a) and (b) have their equivalent IPNs respectively in Fig. 3 (c) and (d). Thus, we can see that LPN and IPN are with equal modeling capability to model input and output indeterminacy. We prefer adopting LPNs to model such indeterminacy of the systems because of the following reasons: (1) LPN has compact net structure compared with its equivalent IPN, for example, it has 2 fewer transitions and 6 fewer arcs in the LPN model in Fig. 3 (a) than the one in Fig. 3 (c); and it has 2 fewer transitions and 4 fewer arcs in the LPN model in Fig. 3 (b) than the one in Fig. 3 (d); (2) There is no good property analysis tool or method for IPNs till now; and (3) We have designed some property analysis methods in our previous work [16]–[18].

III. LPN COMPOSITION

A. VECTOR COMPUTATIONAL METHOD

A first-order logic expression can be transformed into an equivalent unique disjunctive normal logic expression. Each disjunctive item is a conjunctive expression of all atomic predicates. Each disjunctive item can be represented by a vector with the element equaling the logic value of atomic predicates. Then, a disjunctive normal logic expression can be transferred to a set of vectors. Thus, each transition is attached to a set of vectors. We now give the related definitions.

Definition 5: Let f be a disjunctive normal logic expression of atomic predicates p_1-p_m and $f_1 - f_n$ be n disjunctive items, i.e., $f = f_1 \vee f_2 \vee \ldots \vee f_n$, and $f_i = f_{i1} \wedge f_{i2} \wedge \ldots \wedge f_{im}$, $i \in \mathbf{N}_n^+$. A vector of f_i on $(p_1, p_2, \ldots, p_m)$ can be represented by $v_i = (v_{i1}, v_{i2}, \ldots, v_{im})$, where for $j \in \mathbf{N}_m^+$,

$$v_{ij} = \begin{cases} 1, & f_{ij} = p_j \\ 0, & f_{ij} = \neg p_j \end{cases} \tag{2}$$

Given $LPN = (P, T, I, O, M, f, g, \tau)$, if $t \in T_D \cup T_O$, there is only one vector of $f(^\bullet t)$ with all elements being 1, and so is $g(t^\bullet)$, $t \in T_D \cup T_I$. If $t \in T_I$ (or $t \in T_O$), the vector set

of $f(^\bullet t)$ (or $g(t^\bullet)$) is $\{v_1, v_2, \ldots, v_n\}$, denoted by denoted by $V(^\bullet t)$ (or $V(t^\bullet)$). Notice that in the following content we do not consider a 0-vector which does not make sense in real-systems.

Definition 6: *LPN*= $(P, T, I, O, M, f, g, \tau)$ is an LPN and $M \in R(M_0)$. Given $P' \in P$, the mapping of M on P' is denoted by $M|_{P'} = (M(p'_1), M(p'_2), .., M(p'_x))$, where $P' = \{p'_1, p'_2, \ldots, p'_x\}$.

We re-write transition firing rules of LPNs as that for $\forall t \in T_I$, $M[t\rangle$ if $M|_{P'} \in V(^\bullet t)$ and $P' = {}^\bullet t$. $M[t\rangle M'$ where

(a) if $t \in T_I$ and $p \in {}^\bullet t$, then $M'(p) = M(p)$-$M|_{P'}(p)$, else, $M'(p) = M(p) + O(p, t)$-$I(p, t)$; and
(b) if $t \in T_O$ and $p \in t^\bullet$, then $M'(p) = M(p) + M|_{P'}(p)$, else, $M'(p) = M(p) + O(p, t)$-$I(p, t)$.

In the following content, we only compute the vector set of the predicate logic expressions of the logic input and output transitions, denoted by $V(^\bullet t)$ and $V(t^\bullet)$, respectively.

B. LPN COMPOSITION WITH SHARED P-TYPE SUBNETS

This section will present the composition of LPNs where the components are P-type subnets. It realizes the composition of the subnets by common shared places which can be divided into resource places and control places. Given an LPN, some places are shared resource or control places while others are not. To distinguish them we call the control places which are not shared places as basic control places. In order to ensure the correctness of the composition of LPN subnets we firstly require the soundness of the subnets. We can get a subnet and an LPN on a set of places.

Definition 7: $L = (P, T, I, O, M, f, g, \tau)$ is an LPN. The subnet of L on $P' \in P$ is $L' = (P', T', I', O', M', f', g', \tau')$, where

1) $P' = P'_R \cup P'_C$, and if $p \in P_R$, then $p \in P'_R$; and if $p \in P_C$, then $p \in P'_C$;

2) $T' = T = T'_D \cup T'_I \cup T'_O$, and if $t \in T_D$, then $t \in T'_D$; if $t \in T_I$, then $t \in T'_I$; and if $t \in T_O$, then $t \in T'_O$;

3) I': $P' \times T' \to \mathbf{N}$, $I'(p, t) = I(p, t)$, $p \in P'$ and $t \in T'$;

4) O': $P' \times T' \to \mathbf{N}$, $O'(p, t) = O(p, t)$, $p \in P'$ and $t \in T'$;

5) M': $P' \to \mathbf{N}$, M'_0 is the initial marking, and $M'_0(p) = M_0(p)$ where $p \in P'$;

6) f' ($^\bullet t$-P'') is remaining of $f(^\bullet t)$ by deleting p and its connected connectives, where $p \in {}^\bullet t$-P'' and $P'' = P$-P';

7) g' ($t^\bullet$-P'') is the remaining of $g(t^\bullet)$ by deleting p and its connected connectives, where $p \in t^\bullet$-P'' and $P'' = P$-P'; and

8) $\tau'(t) = \tau(t)$.

We then give the following definition of a P-type subnet.

Definition 8: A P-type sound subnet is a logic Petri net $L = (P, T, I, O, M, f, g, \tau)$, where

1) There exist two distinct places p_{in} and p_{out}, where ${}^\bullet p_{in} = p^\bullet_{out} = t^*$, $I(p_{out}, t^*) = O(p_{in}, t^*) = 1$, and $M_0 = p_{in}$;

2) $P_S = P_I \cup P_O \in P$ denotes a set of shared places, where P_I and P_O denote a set of shared input and output places, respectively, where $P_I \cap P_O = \emptyset$.

3) *LPN* on P-P_S is live at M_0, i.e., $\forall M \in R(M_0)$ and $\forall t \in T$: $M' \in R(M)$ and $M'[t\rangle$; and

4) *LPN* on P-P_S is safe at M_0, i.e., $\forall p \in P$ and $\forall M \in R(M_0)$: $M(p) \leq 1$.

We now define the LPN composition with shared P-type subnets.

Definition 9: $L_1 = (P_1 \cup P_{I1} \cup P_{O1}, T_1, I_1, O_1, M_1, f_1, g_1, \tau_1)$ and $L_2 = (P_2 \cup P_{I2} \cup P_{O2}, T_2, I_2, O_2, M_2, f_2, g_2, \tau_2)$ are two P-type sound subnets, where $P_{I1} \cap P_{I2} = P_{O1} \cap P_{O2} = \emptyset$, $(P_{O1} \cap P_{I2}) \cup (P_{O2} \cap P_{I1}) = P_S \neq \emptyset$, and $T_1 \cap T_2 = \emptyset$. Then, the composition of L_1 and L_2 is an LPN= $(P \cup P_I \cup P_O, T, I, O, M, f, g, \tau)$, where

1) $P_I = (P_{I1} \cap P_{I2}) \backslash P_S$,

2) $P_O = (P_{O1} \cap P_{O2}) \backslash P_S$,

3) $T_D = T_{D1} \cup T_{D2}$, $T_I = T_{I1} \cup T_{I2}$, and $T_O = T_{O1} \cup T_{O2}$,

4) $\tau(t) = \tau_i(t)$ if $t \in T_{Di} \cup T_{Ii} \cup T_{Oi}$ with $i \in \{1, 2\}$, and

5) $f(t) = f_i(t)$ if $t \in T_{Ii}$, $g(t) = g_i(t)$ if $t \in T_{Oi}$ with $i \in \{1, 2\}$.

According to Definition 9, An LPN can be composed of n P-type sound subnets L_1-L_n in a pair-wise composition way through common places denoted by $L = L_1 \oplus L_2 \oplus \ldots \oplus L_m$. Only shared input places can be combined with the output ones. When two P-type sound subnets are composed based on a set of shared places, these places cannot be used for other combinations in the future. Notice that a shared place belongs to at most two subsystems, so such shared places are usually regarded as resource places in cooperative systems. This kind of shared P-type LPN composition can describe the cooperative interaction such as data interchange and order between the sub-systems.

C. PROPERTY ANALYSIS

We now discuss property analysis including liveness, reachability, and reversibility of LPN composition with P-type subnets. It is realized by the proposed vector computational method in sub-section A. Firstly we give some related notations and definitions. An LPN can be treated as a directed graph including directed arc and places and transitions as nodes. Given an *LPN*, a node n_1 is directly reached from another node n_2 if there exists a directed arc (n_1, n_2); and a node n_1 is in-directly reached from another node n_k if there exist a sequence of nodes $n_2, \ldots, n_{k-1}$ such that n_j is directly reached from $n_{j-1}, j = 2, \ldots, k$. Here, we denote $C =< n_1, n_2, \ldots, n_k >$ as a path from node n_1 to n_k. The alphabet of C is denoted by $\&(C) = \{n_1, n_2, \ldots, n_k\}$. C is a basic path if $n_i, n_j \in \&(C)$, and $i \neq j$: $n_i \neq n_j$. $n_i >_C n_j$ denotes that n_j is (directly or indirectly) reached from n_i. A path $C =< n_1, \ldots, n_k >$ is a circle when $< n_1, \ldots, n_{k-1} >$ is a basic path with $n_1 = n_k$.

Let $L_1 = (P_1, T_1, I_1, O_1, M_1, f_1, g_1, \tau_1)$ and $L_2 = (P_2, T_2, I_2, O_2, M_2, f_2, g_2, \tau_2)$ be two P-type sound subnets as shown in Fig. 4(a) and (b), respectively, where $P_S = \{p_{s11}, p_{s21}\}$, $P_{O1} \cap P_{I2} = \{p_{s11}\}$, and $P_{O2} \cap P_{I1} = \{p_{s21}\}$; $|{}^\bullet p_{s11}| = |p^\bullet_{s11}| = |{}^\bullet p_{s21}| = |p^\bullet_{s21}| = 1$; and $L = L_1 \oplus L_2$ as shown in Fig. 5. Then we discuss the properties such as liveness, safeness, and reversibility of the composed LPN of L_1 and L_2. To do so, we should analyze if the connected transitions of the shared places are traditional transitions or logic transitions.

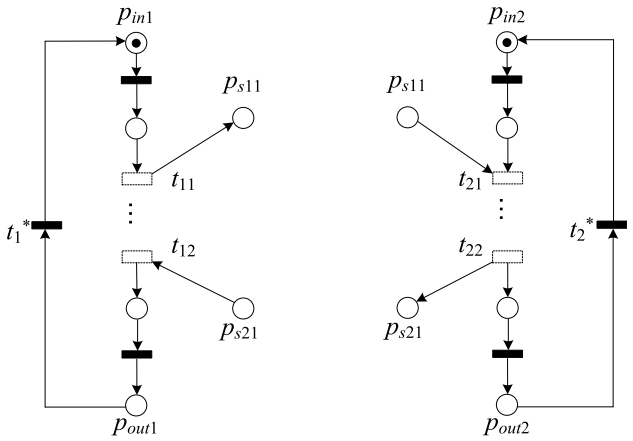

FIGURE 4. Two shared P-type subnets L_1 and L_2.

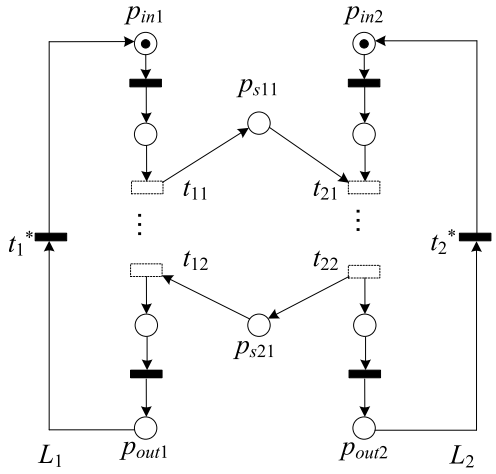

FIGURE 5. LPN composed of two shared P-type subnets.

If they are the latter ones, we should consider their vector sets of predicate logic expressions. Based on these vector sets, we have the following conclusions.

Theorem 1: Given $M_{ini} = p_{ini}$, L_i on $P_i \backslash P_S$ is reversible, where $i = 1, 2$.

Proof: Because L_1 is a P-type sound subnet, the subnet on $P_1 \backslash P_S = \{p_{s11}, p_{s21}\}$ denoted by L_1' is live and safe. Suppose L_1' at $M_{in1} = p_{in1}$ is not reversible. Because of the liveness of L_1', there exists $M_1 \in R(M_{in1})$, i.e., $M_{in1}[\sigma\rangle M_1$ such that $M_1[t_1^*\rangle$. Let $M_1[t_1^*\rangle M_2$, and because L_1' is not reversible, $M_2 \neq p_{in1}$. Suppose that $M_2(p) > 0$, and according to the safeness of L_1', $M_2(p) = 1$. Similarly, we have $M_2[\sigma t_1^*\rangle M_3$, and $M_3(p) = 2$, which contradicts the safeness of L_1'. Thus, L_1' at $M_{in1} = p_{in1}$ is reversible. Similarly, L_2 on $P_2 \backslash P_S$ is reversible $M_{in2} = p_{in2}$.

Theorem 2: Given that $M_{in} = p_{in1} + p_{in2}$, $C =< t_{11}, p_{s11}, t_{21}, \ldots, t_{22}, p_{s21}, t_{12} >$ is a basic path, $t_{11} \neq t_{12}$, and there is at most one logic transition in $T_S = \{t_{11}, t_{21}, t_{22}, t_{12}\}$. L is live, safe, and reversible iff $t \in T_S$, $p \in P_S$, and $\forall v \in V(^\bullet t)$ or $V(t^\bullet)$: $v|_p = 1$.

Proof: (Sufficiency) If $t \in T_S$, $p \in P_S$, and $\forall v \in V(^\bullet t)$ or $V(t^\bullet)$: $v|_p = 1$, it means that the logic expression ensures that: 1) if t is a logic input transition, it requires the input shared place has a token to be enabled; 2) if it is a logic output transition, it requires when it fires, its output shared place contains a token. According to Definition 8 and Theorem 1, L_i on P_i-P_S is live, safe, and reversible, where $i = 1, 2$. In order to verify that L is live, and safe, we need to consider the shared places and their connected transitions. We can easily verify that p_{s11} and p_{s21} are safe and t_{11}, t_{21}, t_{22}, and t_{12} are live. Similar to the proof of Theorem 1, we can prove that L is reversible.

(Necessity) L is live and safe and C is a basic path but not a circle. We can use a contradiction approach and suppose that t_{11} is a logic output transition. Suppose that there exists $v \in V(^\bullet t)$ or $V(t^\bullet)$ such that $v|_p = 0$. As there is at most one logic transition, $p^\bullet \in \{t_{21}, t_{12}\}$ is not live. Thus, $v|_p = 0$ and the conclusion holds. ■

Theorem 3: Given that $M_{in} = p_{in1} + p_{in2}$, $C =< t_{11}, p_{s11}, t_{21}, \ldots, t_{22}, p_{s21}, t_{12} >$ is a basic path, $t_{11} \neq t_{12}$, and there is more than one logic transition in $T_S = \{t_{11}, t_{21}, t_{22}, t_{12}\}$. L is live, safe, and reversible iff when two logic transitions $t_1, t_2 \in T_S$: $p = t_1^\bullet = {}^\bullet t_2$, we have that there exist $v_2 \in V(^\bullet t_2)$ and $v_1 \in V(t_1^\bullet)$ such that $v_1|_p = v_2|_p$.

Proof: (Sufficiency) If $t_1, t_2 \in T_S$, $p \in P_S$ is the shared place, $v_2 \in V(^\bullet t_2)$, $v_1 \in V(t_1^\bullet)$: $v_1|_p = v_2|_p$, it means that the logic input and output expressions ensure that: 1) if $v_1|_p = v_2|_p = 0$, t_1 is a logic output transition, and it fires generating no tokens in p; and 2) t_2 is a logic input transition and requires the input shared place has a token to be enabled. According to Definition 8 and Theorem 1, L_i on P_i-P_S is live, safe, and reversible, where $i = 1, 2$. In order to verify that L is live, and safe, we need to consider the shared places and their connected transitions. We can easily verify that p_{s11} and p_{s21} are safe and t_{11}, t_{21}, t_{22}, and t_{12} are live. Similar to the proof of Theorem 1, we can prove that L is reversible.

(Necessity) L is live and safe and C is a basic path but not a circle. We can use a contradiction approach and suppose that t_{11} is a logic output transition. Suppose that when two logic transitions $t_1, t_2 \in T_S$: $p = t_1^\bullet = {}^\bullet t_2$, we have $\forall v_2 \in V(^\bullet t_2)$, $\forall v_1 \in V(t_1^\bullet)$: $v_1|_p \neq v_2|_p$. 1) if $v_1|_p = 0$ and $v_2|_p = 1$, t_2 becomes dead; and 1) if $v_2|_p = 0$ and $v_1|_p = 1$, after t_1 fires, a token is generated in p which will stay in the place and make t_1 become dead. Thus, there must exist $v_2 \in V(^\bullet t_2)$ and $v_1 \in V(t_1^\bullet)$ such that $v_1|_p = v_2|_p$ and the conclusion holds. ■

Notice that, there is no circles that contain a shared place as a node. The following analysis will consider the composition of two P-type sound subnets with circles. Let two P-type sound subnets L_1 and L_2 be as shown in Fig. 6(a) and (b), respectively, where $P_S = \{p_{s11}, p_{s21}\}$, $P_{O1} \cap P_{I2} = \{p_{s11}\}$, and $P_{O2} \cap P_{I1} = \{p_{s21}\}$; $|^\bullet p_{s11}| = |p_{s11}^\bullet| = 1$; and $L = L_1 \oplus L_2$ as shown in Fig. 7. Then we discuss the properties such as liveness, safeness, and reversibility of the composed LPN of L_1 and L_2. Similarly, we analyze the properties by considering if the connected transitions of the shared places are traditional

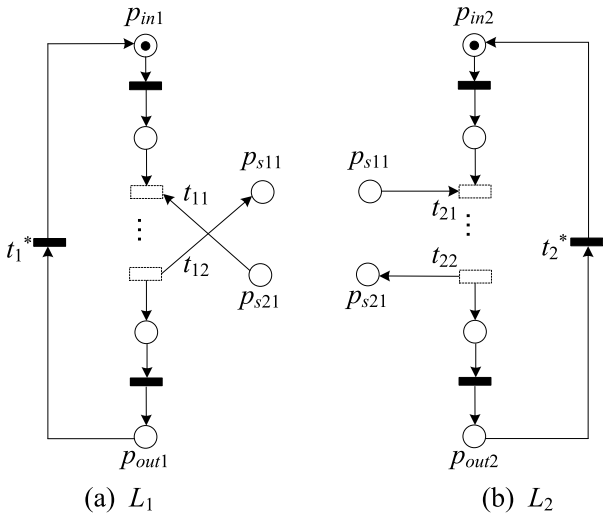

FIGURE 6. Two shared P-type subnets L_1 and L_2.

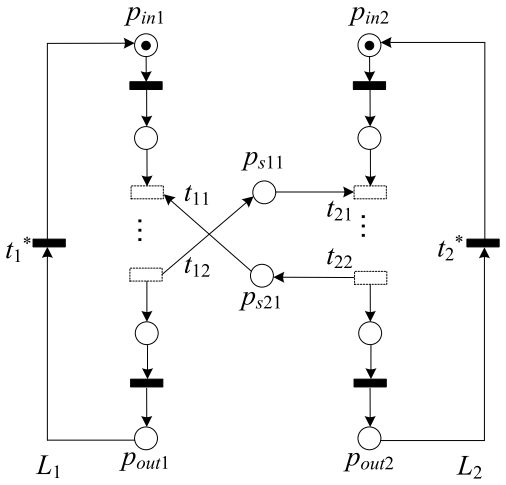

FIGURE 7. LPN composed of two shared P-type subnets.

transitions or logic transitions. We then compute the vector sets of logic expressions. According to the obtained vector sets, we have the following conclusion.

Theorem 4: Given that $M_{in} = p_{in}1 + p_{in}2$, $C =< t_{12}, p_s11, t_{21}, \ldots, t_{22}, p_s21, t_{11}, \ldots, t_{12} >$ is a circle. L is live, safe, and reversible iff 1) there is a logic input transition $t \in \{t_{11}, t_{21}\}$ and $\exists v_1 \in V(^\bullet t)$ and $\forall v_2 \in V((^{\bullet\bullet}t)^\bullet)$ such that $v_1|_p = v_2|_p = 0, p \in {}^\bullet t \cap \{p_s11, p_s21\}$; and 2) there exist $v_3 \in V(^\bullet t_2)$ and $v_4 \in V(t_1^\bullet)$ such that $v_3|_{p'} = v_4|_{p'}$, where $t_1, t_2 \in T_S$: $p = t_1^\bullet = {}^\bullet t_2$ and $p' \in \{p_s11, p_s21\} \backslash p$.

Proof: (Sufficiency) In order to verify that L is live, and safe, we need to consider the shared places and their connected transitions. Because there is a logic input transition $t \in \{t_{11}, t_{21}\}$, and $\exists v_1 \in V(^\bullet t), v_1|_p = 0$, where $p \in {}^\bullet t \cap \{p_s11, p_s21\}$, t is enabled without a token in p. Suppose that $t = t_{11}$ and $p = p_s21$. Notice that if $t = t_{21}$ and $p = p_s11$, we have the same proof process below. We can easily verify that t_{11} is enabled even if p_s21 has no tokens. According to Definition 8 and Theorem 1, L_1 on P_1-P_S is live, safe, and reversible. Thus, t_{12} can be enabled and fire. Because there exist $v_3 \in V(^\bullet t_{21})$ and $v_4 \in V(t_{12}^\bullet)$ such that $v_3|_{p'} = v_4|_{p'}$, where $p' = p_s11$, so t_{21} can be enabled and fire. Then, $\forall v_2 \in V(t_{22}^\bullet)$ such that $v_2|_p = 0$, where $p = p_s21$. Thus, t_{22} can be enabled and fire. After the firing of t_{12}, t_{21}, t_{22}, and t_{11}, successively, no tokens left in p_s21 and p_s21. Still, according to that L_i on P_i-P_S is live, safe, and reversible, where $i = 1, 2$, we can prove that L is live, safe, and reversible.

(Necessity) L is live and safe and C is a basic path but not a circle. We can use a contradiction approach to prove both 1) and 2). Firstly, suppose that $\forall t \in \{t_{11}, t_{21}\}$ and $v_1 \in V(^\bullet t)$ such that $v_1|_p = 1$. In this situation, t_{11} and t_{21} are dead because each transition's waiting for the token generated from another. Thus, there exists a transition that needs no token from the connected input shared place and 1) holds. Secondly, when two logic transitions $t_1, t_2 \in T_S$: $p = t_1^\bullet = {}^\bullet t_2$, and $p' \in \{p_s11, p_s21\} \backslash p$, we have $v_3 \in V(^\bullet t_2)$ and $v_4 \in V(t_1^\bullet)$ such that $v_3|_{p'} \neq v_4|_{p'}$. 1) if $v_3|_{p'} = 0$ and $v_4|_{p'} = 1$, t_2 becomes dead; and 1) if $v_4|_{p'} = 0$ and $v_3|_{p'} = 1$, after t_1 fires, a token is generated in p' which will stay in the place and make t_1 become dead. Thus, there must exist $v_3 \in V(^\bullet t_2)$ and $v_4 \in V(t_1^\bullet)$ such that $v_3|_{p'} = v_4|_{p'}$, and condition 2) holds.

Next, we consider an LPN that is composed of more than two P-type sound subnets. Let $L = (P, T, I, O, M, f, g, \tau)$ be the composition of L_i which is the i-th P-type sound subnet of the i-th sub-system, as shown in Fig. 7, the set of shared places between L_0 and L_i be $P_{Si} = \{p_s0i, p_s1i\}$, $P_I0 \cap P_{Oi} = \{p_s0i\}$, and $P_O0 \cap P_{Ii} = \{p_s1i\}$; $|^\bullet p_s1i| = |p_s1i^\bullet| = 1$; and $L = \oplus_{i=0}^{m} L_i$. Then, we have the following conclusion.

Corollary 1: Given that, $M_{in} = \sum_{j=0}^{i} p_{inj}$ is live, safe, and reversible iff the composition of any two models of sub-systems L_1 and L_2 is live, safe, and reversible.

The proof can be verified according to Theorems 2-4. Notice that in L, $P_S = \emptyset$, which means that there is no additional shared place after the accomplishment of the composition. For example, $L_{01} = L_0 \oplus L_1$ is live, safe, and reversible according to Theorem 4, $L_{01} \oplus L_i$ is live, safe, and reversible according to Theorems 2 and 3, and similarly, L in Fig. 8 is live, safe, and reversible.

From the above theorems and the corollary, based on the vector computational method, we can verify the properties including the liveness, safeness, and reversibility of the composition of two P-type sound subnets or the composition of more than two P-type sound subnets. We only consider the places shared by only two sub-systems, which can well describe many cooperative systems such as the E-commerce systems.

IV. AN E-COMMERCE SYSTEM MODEL

A. SYSTEM DESCRIPTION

We now use case studies to show the effectiveness of the modeling method proposed in the work. By using e-commerce systems, trades happen between merchants and buyers. As described in Fig. 1, the merchant sub-system is

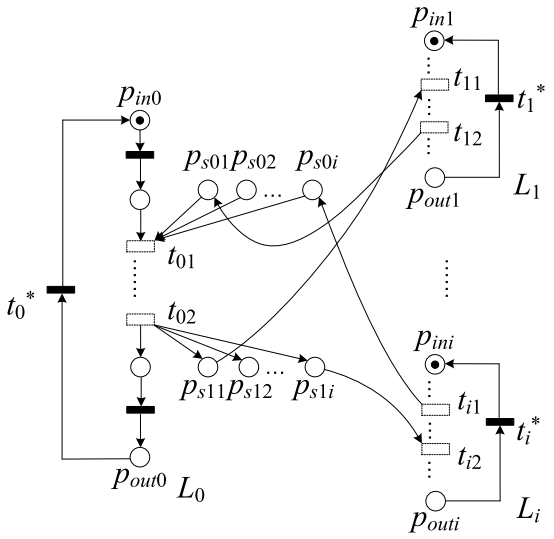

FIGURE 8. LPN composed of i shared P-type subnets.

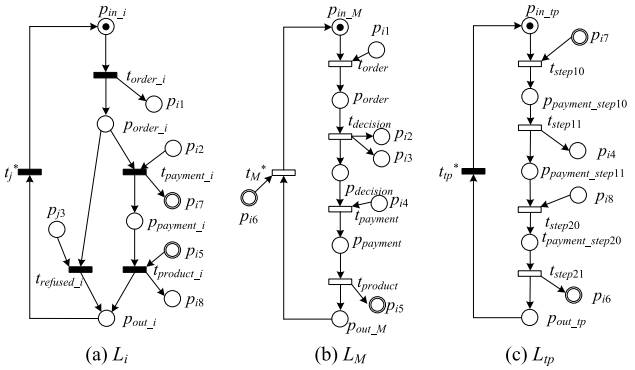

FIGURE 9. The LPN models of the sub-systems.

dealing with several buys simultaneously and should have the function of batch handling requests from buyers during a certain time interval. Simultaneously, indeterminate intersections between the two sub-systems always occur. We now design the LPN models for such cooperative systems. In detail, the models of three main sub-systems, i.e., customers, merchants, and a third-party, are designed. In order to finish trading, each system should cooperate with others correctly. Customers mainly have the main operations including order requests, payment, and getting the ordered products, sequentially; Merchants should do order processes, payment checking, mailing products, and getting paid. During the trading process, the benefits and profits of both merchants and customers are protected by a third-party, which acts as a very important organization in current E-commerce systems. Usually, the third-party will keep the payment of the customer for the merchant; after the customers ensure the correct receiving of their ordered product, payments are transferred from the third-party to the corresponding merchant. Notice that, the activities can be refined by some more-detailed processes such as checking the detailed order information by the merchant, ensuring the detailed address by the customers, etc. During any process, time acts in a very important role. For example, if customers do not trigger the finish of the payment transfer from the third-party to the merchant by ensuring the checking of the received products after getting the products in a certain time interval, the payment will automatically be transferred successfully. In the following sub-section, we will model the aforementioned cooperative system and analyze the properties of the model based on the proposed vector computational method.

B. MODELING PROCESS

Three sub-systems in the e-commerce system are modeled by LPNs as shown in Fig. 9: (a) the i-th customer sub-system L_i, (b) a merchant sub-system L_M, and (c) a third-party sub-system L_{tp}. The meanings of places and transitions are shown in Tables 1 and 2, respectively.

TABLE 1. The meanings of places in Figure 9.

P	Meanings
p_{order_i}	Customer i is waiting for the response of the merchant after delivering the order
$p_{payment\_i}$	Customer i is waiting for the receiving of the products after making the payment
p_{order}	The merchant is processing the orders
$p_{decision}$	The merchant has decided to accept or reject the order
$p_{payment}$	The merchant has got the payment and is ready to mail the products
$p_{payment\_step10}$	The third-party receives the payment from customers
$p_{payment\_step11}$	The third-party checks and keeps the payment from customers
$p_{payment\_step20}$	The third-party is ready to transfer the payment to the merchant after customers receive the products
p_{i5}	The payment
p_{i5} and p_{i5}	The products
Others	No specific meaning but for control purposes

TABLE 2. The meanings of transitions in Figure 9.

T	Meanings	Transition Type
t_{order_i}	Customer i is delivering an order	T_D
$t_{payment\_i}$	Customer i is making payment	T_D
$t_{product\_i}$	Customer i is receiving products	T_D
$t_{refused\_i}$	The order of Customer i is refused	T_D
t_{order}	The merchant is receiving the order	T_I, $p_{in\_M}\wedge(p_{i1}\vee\neg p_{i1})$
$t_{decision}$	The merchant is deciding to accept or reject the order	T_O, $p_{decision}\wedge((p_{i2}\vee\neg p_{i3})\vee(\neg p_{i2}\vee p_{i3}))$
$t_{payment}$	The third-party has get the payment	T_I, $p_{payment}\wedge(p_{i4}\vee\neg p_{i4})$
$t_{product}$	The merchant is mailing the product	T_O, $p_{out\_M}\wedge(p_{i5}\vee\neg p_{i5})$
t_{step10}	The third-party is receiving the payment from customers	T_I, $p_{payment\_step10}\wedge(p_{i4}\vee\neg p_{i4})$
t_{step11}	The third-party ensures the payment with the merchant	T_O, $p_{payment\_step11}\wedge(p_{i4}\vee\neg p_{i4})$
t_{step20}	The third-party ensures that customers receive the products	T_I, $p_{payment\_step20}\wedge(p_{i4}\vee\neg p_{i4})$
t_{step21}	The third-party transfers the payment	T_O, $p_{out\_tp}\wedge(p_{i4}\vee\neg p_{i4})$
Others	No specific meaning but for control purposes	T_D except for that t_M^* is T_I with $p_{out\_M}\wedge(p_{i6}\vee\neg p_{i6})$

In Fig. 9 (a), initially, p_{in_i} contains a token meaning that customer i is ready for purchasing a product. Then it delivers order and waits for the merchant to ensure it, which is modeled by the firing of t_{order_i}. The merchant checks the order and chooses to accept or reject it. When the order is

accepted, the customer continues to do the payment modeled by a transition $t_{payment\_i}$; or it may be rejected by the merchant for some reasons such as shortage of products and $t_{refused\_i}$ fires. If the order is accepted, customers sequentially make the payment and then can receive the ordered product, which are modeled by $t_{payment\_i}$ and $t_{product\_i}$, respectively. The models of the merchant and the third-party are shown in Fig. 9 (b) and (c), respectively. The merchant processes the order processing, payment checking, and product delivery, modeled by transitions t_{order}, $t_{payment}$, and $t_{product}$, respectively. The third-party mainly has four operations including receiving the payment of customers, and ensuring and keeping the payment, ensuring that the customer has received the products, and transferring the payment to the merchant, which are respectively denoted by t_{step10}, t_{step11}, t_{step20}, and t_{step21}. Predicate logic expressions to logic transitions in the models in Fig. 9 are designed as shown in Table 2. According to Theorems 5, we can conclude that the composition of LPN models in Fig. 9 is live, safe, and reversible.

V. CONCLUSION

This paper proposes a vector computational method for LPN composition and property analysis. It studies the composition of LPNs with the shared P-type subnets. Each logical expression can be transformed into a unique disjunctive normal one and then to a unique set of vectors. A vector computational method is proposed such that the properties of the composition of the shared P-type subnets such as liveness, boundedness, and reversibility are verified. An E-commerce system is modeled to show the proposed method. This paper can improve the state of the art in the theory of LPNs. Future work will study of composition of LPNs for other systems such as manufacturing systems [30] [31], knowledge-based systems [32], and transportation systems [33].

REFERENCES

[1] C. A. Petri, "'Bonn: Institute fur instrumentelle mathematik, schriften des IIM Nr.3, 1962. also, english translation: Communication with automata," New York, NY, USA, Griffiss Air Force Base. Tech. Rep. RADC-TR-65-377, 1966, vol. 1.
[2] R. Robidoux, H. Xu, L. Xing, and M. C. Zhou, "Automated verification of dynamic reliability block diagrams using colored Petri nets," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 40, no. 2, pp. 337–351, Mar. 2010.
[3] K. Jensen and L. M. Kristensen, *Coloured Petri Nets: Modeling and Validation of Concurrent Systems*. New York, NY, USA: Springer-Verlag, 2009.
[4] J. 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, Oct. 2000.
[5] 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.
[6] M. C. Zhou and M. P. Fanti, Eds, *Deadlock Resolution in Computer-Integrated Systems*. New York, NY, USA: Marcel Dekker, 2005.
[7] F. Yang, N. Wu, Y. Qiao, and M. Zhou, "Optimal one-wafer cyclic scheduling of hybrid multirobot cluster tools with tree topology," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 2, pp. 289–298, Feb. 2018.
[8] Y. Teng, Y. Du, L. Qi, and W. Luan, "A logic Petri net-based method for repairing process models with concurrent blocks," *IEEE Access*, vol. 7, pp. 8266–8282, 2019.
[9] L. Qi, M. Zhou, and W. Luan, "A two-level traffic light control strategy for preventing incident-based urban traffic congestion," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 1, pp. 13–24, Jan. 2018.
[10] L. Qi, M. Zhou, and W. Luan, "Impact of driving behavior on traffic delay at a congested signalized intersection," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 7, pp. 1882–1893, Jul. 2017.
[11] L. Qi, M. Zhou, and W. Luan, "Emergency traffic-light control system design for intersections subject to accidents," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 1, pp. 170–183, Jan. 2016.
[12] Y. Y. Du and B. Q. Guo, "Logic Petri nets and equivalency," *Inf. Technol. J.*, vol. 8, no. 1, pp. 95–100, Jan. 2009.
[13] 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.
[14] Y. Xu, Y. Du, W. Luan, L. Qi, and H. Sun, "Repairing process models with logical concurrent and casual relations via logical Petri nets," *IEEE Access*, vol. 6, pp. 56340–56355, 2018.
[15] W. H. Sanders and J. F. Meyer, "Stochastic activity networks: Formal definitions and concepts," in *School organized by the European Educational Forum*. Berlin, Germany: Springer, 2001, pp. 315–343.
[16] Y. Du, L. Qi, and M. 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.
[17] W. Luan, L. Qi, and Y. Du, "Composition of logical petri nets and compatibility analysis," *IEEE Access*, vol. 5, pp. 9152–9162, 2017.
[18] W. Luan, L. Qi, Z. Zhao, J. Liu, and Y. Du, "Logic Petri net synthesis for cooperative systems," *IEEE Access*, vol. 7, pp. 161937–161948, 2019.
[19] G. Liu, "Complexity of the deadlock problem for Petri nets modeling resource allocation systems," *Inf. Sci.*, vol. 363, pp. 190–197, Oct. 2016.
[20] D. Xiang, G. Liu, C. Yan, and C. Jiang, "Detecting data-flow errors based on Petri nets with data operations," *IEEE/CAA J. Autom. Sinica*, vol. 5, no. 1, pp. 251–260, Jan. 2018.
[21] S. Wang, D. You, and M. Zhou, "A necessary and sufficient condition for a resource subset to generate a strict minimal siphon in S 4PR," *IEEE Trans. Autom. Control*, vol. 62, no. 8, pp. 4173–4179, Aug. 2017.
[22] J. Sha, Y. Du, and L. Qi, "A user requirement oriented web service discovery approach based on logic and threshold petri net," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 6, pp. 1528–1542, Nov. 2019.
[23] L. Wang, Y. Y. Du, and L. Qi, "Efficient deviation detection between a process model and event logs," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 6, pp. 1352–1364, Nov. 2019.
[24] Y. Du, L. Qi, and M. Zhou, "A vector matching method for analysing logic Petri nets," *Enterprise Inf. Syst.*, vol. 5, no. 4, pp. 449–468, Nov. 2011.
[25] W. Duo, X. Jiang, O. Karoui, X. Guo, D. You, S. Wang, and Y. Ruan, "A deadlock prevention policy for a class of multithreaded software," *IEEE Access*, vol. 8, pp. 16676–16688, 2020.
[26] X. Lu, M. Zhou, A. C. Ammari, and J. Ji, "Hybrid Petri nets for modeling and analysis of microgrid systems," *IEEE/CAA J. Automatica Sinica*, vol. 3, no. 4, pp. 349–356, Oct. 2016.
[27] S. Wang, D. You, and C. Seatzu, "A novel approach for constraint transformation in Petri nets with uncontrollable transitions," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 8, pp. 1403–1410, Aug. 2018.
[28] K. Wang and L. Yan Yuan, "First-order logic characterization of program properties," *IEEE Trans. Knowl. Data Eng.*, vol. 6, no. 4, pp. 518–533, Aug. 1994.
[29] S. Clifford, L. D. Robert, and B. Kenneth, *Discrete Mathematics for Computer Scientists*. Boston, MA, USA: Addison-Wesley, 2011.
[30] X. Guo, M. Zhou, S. Liu, and L. Qi, "Lexicographic multiobjective scatter search for the optimization of sequence-dependent selective disassembly subject to multiresource constraints," *IEEE Trans. Cybern.*, to be published, doi: 10.1109/TCYB.2019.2901834.
[31] X. W. Guo, M. C. Zhou, S. X. Liu, and L. Qi, "Multi-resource constrained selective disassembly with maximal profit and minimal energy consumption," *IEEE Trans. Automat. Sci. Eng.*, to be published.
[32] Z. Zhao, C. Li, X. Zhang, F. Chiclana, and E. H. Viedma, "An incremental method to detect communities in dynamic evolving social networks," *Knowl.-Based Syst.*, vol. 163, pp. 404–415, Jan. 2019.
[33] L. Qi, M. Zhou, and W. Luan, "A dynamic road incident information delivery strategy to reduce urban traffic congestion," *IEEE/CAA J. Autom. Sinica*, vol. 5, no. 5, pp. 934–945, Sep. 2018.

LIANG QI (Member, IEEE) received the B.S. degree in information and computing science and the M.S. degree in computer software and theory from the Shandong University of Science and Technology, Qingdao, China, in 2009 and 2012, respectively, and the Ph.D. degree in Computer Software and Theory from Tongji University, Shanghai, China, in 2017. From 2015 to 2017, he was a Visiting Student with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, USA. He is currently with the Shandong University of Science and Technology, Qingdao, China. He has published more than 50 articles in journals and conference proceedings, including the IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, the IEEE/CAA JOURNAL OF AUTOMATICA SINICA, the IEEE TRANSACTIONS ON SYSTEM, MAN AND CYBERNETICS: SYSTEMS, the IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS, the IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, and the IEEE TRANSACTIONS ON CYBERNETICS. His interests include Petri nets, machine learning, optimization, and intelligent transportation systems.

WENJING LUAN (Member, IEEE) received the B.S. and M.S. degrees from the Shandong University of Science and Technology, Qingdao, China, in 2009 and 2012, respectively, and the Ph.D. degree in computer software and theory from Tongji University, Shanghai, China, in 2018. She is currently a Lecturer of computer science and technology with the Shandong University of Science and Technology, Qingdao. From May 2017 to July 2017, she was a Visiting Student with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, USA. Her current research interests include location-based social networks, data mining, recommender systems, and intelligent transportation systems. She received the Best Student Paper Award-Finalist in the 13th IEEE International Conference on Networking, Sensing and Control (ICNSC'2016) conference.

XIAOYU SEAN LU (Member, IEEE) received the B.S. degree from the Nanjing University of Technology, Nanjing, China, in 2011, and the M.S. and Ph.D. degrees from the New Jersey Institute of Technology, Newark, NJ, USA, in 2015 and 2019, respectively. He is currently a Research Scientist with the Department of Electrical and Computer Engineering, Stevens Institute of Technology, Hoboken, NJ, USA. He has published more than 20 articles in journals and conference proceedings, including the IEEE TRANSACTIONS ON SYSTEM, MAN AND CYBERNETICS: SYSTEMS, the IEEE/CAA JOURNAL OF AUTOMATICA SINICA, and the IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS. His current research interests include social media data analysis, data processing, data mining, and Petri nets and its applications in power systems.

XIWANG GUO (Member, IEEE) received the B.S. degree in computer science and technology from the Shenyang Institute of Engineering, Shenyang, China, in 2006, the M.S. degree in aeronautics and astronautics manufacturing engineering from Shenyang Aerospace University, Shenyang, in 2009, and the Ph.D. degree in System Engineering from Northeastern University, Shenyang, in 2015. He is currently an Associate Professor with the College of Computer and Communication Engineering, Liaoning Shihua University. From 2016 to 2018, he was a Visiting Scholar with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, USA. He has authored more than 40 technical articles in journals and conference proceedings, including the IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON SYSTEM, MAN AND CYBERNETICS: SYSTEMS, the IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, and the IEEE/CAA JOURNAL OF AUTOMATICA SINICA. His current research interests include Petri nets, remanufacturing, recycling and reuse of automotive, and intelligent optimization algorithm.