IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON INTELLIGENT INFORMATION SERVICES

Received February 5, 2020, accepted February 14, 2020, date of publication February 18, 2020, date of current version March 2, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2974883

Composition and Application of Extended Colored Logic Petri Nets to E-Commerce Systems

ZHEN WANG, WENJING LUAN, (Member, IEEE), YUYUE DU, AND LIANG QI, (Member, IEEE)

College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China

Corresponding authors: Wenjing Luan (wenjingmengjing@163.com) and Yuyue Du (yydu001@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 Key Research and Development Program of Shandong Province under Grant 2018GGX101011, in part by the Natural Science Foundation of Shandong Province under Grant ZR2019BF041, Grant ZR2018MF001, and Grant ZR2019BF004, in part by the Project funded by the China Postdoctoral Science Foundation under Grant 2019M662401, in part by the Science and Technology Innovation Project of the Shandong University of Science and Technology (SDUST) under Grant SDKDYC190224, in part by the SDUST Research Fund under Grant 2019TDJH102, and in part by the Scientific Research Foundation of SDUST for Recruited Talents under Grant 2019RCJJ012.

ABSTRACT Extended colored logic Petri nets (ECLPNs) are extensions of logic Petri nets (LPNs) and colored logic Petri nets (CLPNs). They are equivalent to LPNs and CLPNs, which can describe the batch processing and indeterminacy functions of resources in cooperative systems. The advantage of ECLPNs is that their net structures are much simpler than their equivalent CLPNs, and therefore, ECLPNs can be easily used to model and analyze cooperative systems. For systems containing several subsystems with the same function and structure, we can describe them by a single ECLPN. Then we propose a composition method of ECLPNs. We define the robustness of a system based on ECLPN which reflects the validity of the collaboration of subsystems. We define a strict conservativeness that guarantees data security. The robustness and strict conservativeness of composed ECLPNs are analyzed. An E-commerce example is presented to illustrate the modeling capacity and the advantage of ECLPNs.

INDEX TERMS Colored logic Petri net, extended colored logic Petri net, composition, e-commerce system, property analysis.

I. INTRODUCTION

E-commerce systems are becoming more and more complicated, and their compatibility analysis is a co-NP-hardness problem [1]. A good way is to decompose the E-commerce system into several subsystems, analyze properties of each subsystem, and find the inheriting conditions of the properties when the subsystems are composed. Then the properties of a complete system are obtained. Thus, an E-commerce system can be divided into three classes of subsystems: customers, sellers and a third-party. The subsystems exchange massages and perform batch processes at the same time. However, batch processing brings the problem of data indeterminacy [2].

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

Petri Nets (PNs) [3] are a class of graphs that consists of places, transitions and directed arcs between places and transitions. A siphon is a structure and closely related to properties like liveness and deadlock-free of PNs. Methods of siphon computation and interactive deadlock prevention policies have been proposed in [4]–[6]. PNs can be applied in process mining [7]–[9]. However, PNs with inhibitor arcs [10] cannot describe batch processing functions and passing value indeterminacy as clearly and concisely as logic Petri nets (LPNs) [11] do.

An LPN can describe passing value indeterminacy in cooperative systems. It is a high-level Petri net. Logic transitions in LPNs are attached with first-order predicate logic expressions. When logic expressions are attached to an input transition, they control whether the transition can fire or not. When they are attached to an output transition, they control which place tokens should flow into after the transition has fired. LPNs are applied to business process mining [12]–[14] and Web service discovery [15]. However, a logic output transition in LPNs cannot output tokens correctly according to just the attached logical output expression. LPNs cannot differentiate sources of tokens in the system, which may cause errors during the system running.

In colored Petri nets (CPNs) [16], tokens are differentiated by colors. Places of a CPN are combined with a color set, which represents the colors held in the place. Arcs have inscriptions (arc expressions), variables in an expression are assigned by a binding, and an arc expression indicates what type and how many colored tokens are needed when the corresponding transition fires. The state-of-the-art applications of CPNs are given in [17]–[20]. The extension of CPNs can help to solve the problems in daily-life systems [21]–[26]. In fact, CPNs can be simplified by logical expressions when describing passing value indeterminacy.

Like logical expressions can fold PNs with inhibitor arcs, logical expressions can fold CPNs as well. Colored logic Petri nets (CLPNs) proposed in [27] guarantee that output logic transitions can correctly output tokens by checking their colors and logical expressions. However, there are some defects in CLPNs. Many analysis methods and the firing rules of transitions in CLPNs are the same as those in LPNs. The scale of a CLPN model is nearly the same as that of an LPN model when a complicated system is modeled. Comparing with CLPN, the effectiveness of an ECLPN model can be illustrated.

ECLPNs [28] can describe multiple LPN models with the same structure by one model. Thus, the process can be further simplified by using ECLPNs. Customer subsystems share the same structure, so only one customer subsystem needs to be built, and all customers share the subsystem with distinct colored tokens. Therefore, ECLPNs can simplify net structures, and ensure the same modeling ability with CLPNs at the same time. Thus, we propose a composition method of ECLPNs to model E-commerce systems in this paper. Firstly, the basic design module is defined. The definition of the composition of ECLPNs is given. Then, the maintaining conditions of robustness and strict conservativeness of ECLPNs are analyzed.

The rest of the paper is organized as follows. Section II presents the basic concepts of LPNs, CPNs, CLPNs, and ECLPNs briefly. Section III gives the composition method of ECLPNs. Section IV describes an E-commerce system and makes a comparison between an ECLPN model and the corresponding LPN model. Section V concludes this paper and discusses the future work.

II. PRELIMINARIES

In this section, concepts of PN [29]–[38], LPN [11] and CPN [16] are reviewed briefly, and the formal definition of ECLPN is presented. More information about these models can be referred to the corresponding references.

In the rest of this paper, we use $\mathbb{B} \in \{false, true\}$ to denote the Boolean type, $\mathbb{N}^+ = \{1, 2, \ldots\}$ to denote the set of all positive integers, and $\mathbb{N}_n^+ = \{1, 2, \ldots, n\}$ to denote the first n elements in $\mathbb{N}^+$.

Definition 1 [3]: A Petri Net is described as a three-tuple $\mathrm{PN} = (P, T, F)$, where P is a finite set of places; T is a finite set of transitions; and F is a set of directed arcs.

Definition 2 [2]: Given a Petri net $\mathrm{PN} = (P, T, F)$. If $x \in P \cup T$ is a node in PN, then ${}^\bullet x = \{y|(y, x) \in F\}$ and $x^\bullet = \{y|(x, y) \in F\}$ are defined as its pre-set and post-set, respectively.

Definition 3 [11]: A Logic Petri Net is described as a six-tuple $\mathrm{LPN} = (P, T, F, I, O, M)$, where P is a finite set of places; $T = T_D \cup T_I \cup T_O$ is a finite set of transitions; F is a set of directed arcs; I is a mapping function such that $\forall t \in T_I$, $I(t)$ is a logical input expression denoted by f_I; O is a mapping function such that $\forall t \in T_O$, $O(t)$ is a logical output expression denoted by f_O; and M is a marking function, which defines the number of tokens in each place.

Definition 4 [16]: A coloured Petri net is described as a nine-tuple $\mathrm{CPN} = (\sum, P, T, A, N, C, G, E, Init)$, where $\sum$ is a finite set of colors; P is a finite set of places; T is a finite set of transitions; A is a finite set of directed arcs; N is a node function; C is a color function; G is a guard function; E is an arc expression; and $Init$ is an initialization function.

Definition 5 [27]: A coloured logic Petri net is described as an eight-tuple $\mathrm{CLPN} = (C, P, T, F, I, O, M, IC)$, where C is a finite set of consecutive prime numbers, where each prime represents a token color; P is a finite set of places, and each place can store a token; T is a finite set of transitions; F is a finite set of directed arcs; I is a mapping from a logic input transition to a logic input expression; O is a mapping from a logic output transition to a logic output expression; M is a marking function that maps P to the set $\{0, 1\}$; and IC is a color marking function.

The formal definition of ECLPNs is presented as follows.

Definition 6 [28]: An ECLPN is described as an eleven-tuple $\mathrm{ECLPN} = (\sum, P, T, A, C, G, E, Init, I, O, M)$, where

1) $\sum$ is a finite set of non-empty types, called a color set;
2) P is a finite set of places, including one initial place p_0, and one end place p_E;
3) $T = T_D \cup T_L$ is a finite set of transitions, where
   a) T_D is the set of ordinary transitions. T_D is the same with the set of transitions in Colored Petri Nets; and
   b) T_L is called a set of logic transitions, where $\forall t \in T_L$, tokens in input places for t to be enabled are restricted by a logic input expression $f_I(t)$, and tokens in output places after t fires are restricted by a logic output expression $f_O(t)$;
4) A is a finite set of arcs such that $P \cap A = T \cap A = \emptyset$;
5) C: $P \rightarrow \sum$ is a color function, which specifies the set of token colors on a place. $\forall p \in P$, $C(p) = \sum$ by default;

6) G is a guard function. It is defined from T into expressions such that:

$$\forall t \in T : Type(G(t)) = \mathbb{B} \wedge Type(Var(G(t))) \subseteq \sum,$$

where *Type* [16] means the type of an expression or a variable, *Var* [16] means the set of variables in an expression, and $G(t) = true$ by default;

7) E is an arc expression function. It is defined from A into expressions with the binding b [16] such that
   a) $\forall(p, t) \in A$, if $t \in T_L$, then $E(p, t)<b>$ is the set of all multisets of the token type. Otherwise, $E(p, t)<b>$ is the same as those in CPNs; and
   b) $\forall(t, p) \in A$, if $t \in T_L$, then $E(t, p)<b>$ is the set of all multisets of the token type. Otherwise, $E(t, p)<b>$ is the same as those in CPNs;

8) *Init* is an initialization function. It is defined from P into closed expressions, such that:

$$\forall p \in P : Type(Init(p)) = C(p)_{MS};$$

9) I is a mapping from T_L to a logic input expression, such that $\forall t \in T_L, I(t) = f_I(t)$;

10) O is a mapping from T_L to a logic output expression, such that $\forall t \in T_L, O(t) = f_O(t)$; and

11) M: $P \to \sum_{MS}$ is a marking function. $\forall p \in P$, $M(p)$ denotes the multiset of colored tokens in p. At a time point in the net, the set of multisets $\{M(p)|p \in P\}$ is called a marking, denoted by M. The initial marking is denoted by M_0, and the end marking is denoted by M_E.

Here is a detailed explanation about $E(p, t)<b>$.

For $t \in T$ and $p \in^{\bullet} t$, b is a binding, assigning colors to the tokens. The expression set on a directed arc (p, t) is $E(p, t)<b>$. Because of the passing value indeterminacy, there is more than one expression that can be accepted by the logic expression, which is attached to a logic transition. For instance, given a binding $b =<v_1 = s, v_2 = b_1, v_3 = b_2>$, a logic expression is $LE = {}_{\bullet}T_{\bullet} \vee b_1 \vee b_2$, where ${}_{\bullet}T_{\bullet}$ denotes the Boolean variable *true*. Thus any expression in the expression set $E(p, t)<b>= \{\emptyset, b_1, b_2, b_1 + b_2\}$ can be accepted by logic expression *LE*.

To derive the firing rules of a transition in ECLPNs, some calculation methods of multisets are defined as follows.

Definition 7 [28]: *MS* is a multiset that has n types of element. The type set $Elem = \{e_1, e_2, \ldots, e_n\}$ is formed by n types above. Function S maps a multiset to its type set. The coefficient of each type element composes the coefficient set $H(Elem) = \{h(e)|e \in Elem\}$, where $h(e)$ is the coefficient of e in *MS*, which represents the number of e in *MS*. $|MS| = \sum_{e \in E} h(e)$ denotes the total number of elements in *MS*. $f_{int}(MS)$ maps a multiset to an integer. Let $D = \{2, 3, 5, 7, 11, \ldots\}$ be the ascending prime number set, and $D_n = \{d_1, d_2, \ldots, d_n\}$ be the set composed by the first n elements in D. Suppose that $|\ Elem\ | = n$, then $f_{int}(MS) = \prod_{i=1}^{n} d_i^{h(ei)}$. $MS = \sum_{e \in E} h(e) \cdot e$ is called the sum form of *MS*, whereas $f_{int}(MS)= \prod_{i=1}^{n} d_i^{h(ei)}$ is called the product form of *MS*.

For example, $MS = 3\ a + 4\ b + c$, then $Elem = \{a, b, c\}$, $S(3a + 4\ b + c) = \{a, b, c\}$. $H(Elem) = \{3, 4, 1\}$, $|MS| = 3 + 4 + 1 = 8$, $f_{int}(MS)= 2^3 \times 3^4 \times 5 = 3240$. $(3a + 4\ b + c)$ is the sum form of *MS*, whereas 3240 is the product form of *MS*.

Here we define the subset of a multiset.

Definition 8: MS_1 and MS_2 are two multisets. $Elem_i$ is the type set of MS_i, and $h_i(e)$ is the coefficient of e in MS_i, $i \in \{1, 2\}$. MS_1 is the subset of MS_2 if

1) $Elem_1 \subseteq Elem_2$; and
2) $\forall e \in Elem_1$: $h_1(e) \le h_2(e)$, where if $e \notin Elem_2$, then $h_2(e) = 0$.

The firing rule of ECLPNs relies on colored tokens in pre-sets of a transition.

Definition 9 [28]: For $ECLPN = \{\sum, P, T, A, C, G, E, Init, I, O\}$, M is a marking, and b is a binding.

1) If $t \in T_D$, t is enabled if
   $\forall p \in^{\bullet} t$: $E(p, t)<b> \subseteq M(p)$; and
2) If $t \in T_L$, t is enabled if
   (a) $\forall p \in^{\bullet} t$, $\exists MS_I \in E(p, t)<b>$, such that $MS_I \subseteq M(p)$, where MS_I is a multiset; and
   (b) $\cup_{p \in^{\bullet} t} S(MS_I)$ makes $f_I(t) = true$.

After t fires, $M[t > M'$, and it is necessary to meet the following conditions.

1) If $t \in T_D$,

$$M'(p) = \begin{cases} M(p) - E(p, t) < b >, & p \in {}^{\bullet}t \\ M(p) + E(t, p) < b >, & p \in t^{\bullet} \\ M(p), & \text{else}; \end{cases}$$

2) If $t \in T_L$, MS_{Imax} is the max in number of tokens in $\{MS_I\}$,
   a) $\cup_{pa \in t^{\bullet}} S(E(t, p_a))<b>$ makes that $f_O(t) = true$;
   b) $\forall p_a \in t^{\bullet}$, $\exists MS_{out} \in E(t, p_a)<b>$ and $\exists p_b \in^{\bullet} t$ such that $MS_{out} \subseteq M(p_b)$, where MS_{outMax} is the max in number of tokens in $\{MS_{out}\}$; and
   c) $M'(p) = \begin{cases} M(p) - MS_{Imax}, & p \in {}^{\bullet}t \\ M(p) + MS_{outMax}, & p \in t^{\bullet} \\ M(p), & \text{else}. \end{cases}$

III. THE COMPOSITION OF ECLPNS

In this section, the input matrix and output matrix are defined. Basic CPN modules are given based on the matrix. The formal definition of composed ECLPNs is presented. Robustness and strict conservativeness of composed ECLPNs are analyzed. Finally, the procedure of ECLPN composition is given.

A. BASIC DESIGN MODULES

Definition 10: $CPN = \{\sum, P, T, A, N, C, G, E, I\}$, $\forall p_j \in P$, $t_i \in T$, and binding b. Its input matrix is defined as $IM^- = \{im_{ij}^-\}$, where

$$im_{ij}^- = \begin{cases} f_{int}(E(p_j, t_i) < b >), & \text{if } (p_j,\ t_i) \in A \\ 1, & \text{else}. \end{cases}$$

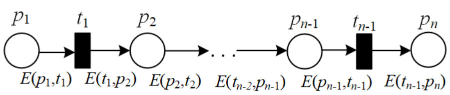


FIGURE 1. **A sequential CPN module.**

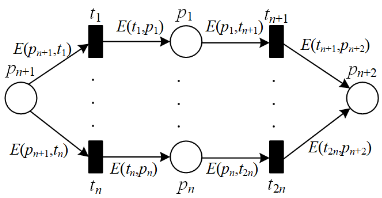


FIGURE 2. **A choice CPN module.**

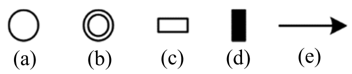


FIGURE 3. **Graphical representation of elements: (a) a normal place, (b) a channel place, (c) a logic transition, (d) an ordinary transition, and (e) a directed arc.**

The output matrix is defined as $OM^+ = \{om_{ij}^+\}$, where

$$om_{ij}^+ = \begin{cases} f_{int}(E(t_i, p_j) < b >), & \text{if } (t_i,\ p_j) \in A \\ 1, & \text{else} \end{cases}$$

A system is composed of several subsystems, and the subsystems are composed of basic design modules. Two kinds of modules are introduced in this section: sequential CPN module (Fig. 1) and choice CPN module (Fig. 2). and the legend is presented in Fig. 3.

The sequential CPN module is defined below.

Definition 11: SCM $= \{\sum, P, T, A, E, IM_S^-, OM_S^+\}$ is a sequential CPN module, and the default binding is b, where

1) $\sum$ is a finite color set;
2) $P = \{p_1, p_2, \ldots, p_n\}$ with $n \in \mathbb{N}^+$;
3) $T = \{t_1, t_2, \ldots, t_{n-1}\}$;
4) A is a set of directed arcs;
5) E is an arc expression function;
6) $IM_S^- = \begin{bmatrix} Int_{n-1} \\ 1_{n-1}^T \end{bmatrix}$, where Int_{n-1} is an $(n-1)$-dimension square matrix, elements on the principal diagonal are $f_{int}(E(p_i,\ t_i)\text{<}b\text{>})$, $i \in \mathbb{N}_{n-1}^+$, and other elements are one. 1_{n-1} is an $(n-1)$-dimension vector with each element being one; and
7) $OM_S^+ = \begin{bmatrix} 1_{n-1}^T \\ Int'_{n-1} \end{bmatrix}$, where Int'_{n-1} is an $(n-1)$-dimension square matrix, elements on the principal diagonal are $f_{int}(E(t_i,\ p_{i+1})\text{<}b\text{>})$, $i \in \mathbb{N}_{n-1}^+$, and other elements are one.

The sequential CPN module is applied in describing a series of successive operations.

For the parallel PN module in [29], $\exists P_{para} \subseteq P$, if $\forall p_i \in P_{para}$, $p_j \in P_{para}$: ${}^\bullet p_i = {}^\bullet p_j$ and $p_i^\bullet = p_j^\bullet$, where $i, j \in \mathbb{N}_n^+$ and $i \neq j$, then all places in P_{para} can be represented by a place p_c in a sequential CPN module, and $\forall p_i \in P_{para}$, $i \in \mathbb{N}_n^+$, is represented by a unique colored token in p_c.

For example, when $n = 3$, the input matrix IM_S^- and the output matrix OM_S^+ are respectively given below.

$$IM_S^- = \begin{bmatrix} E_1 & 1 \\ 1 & E_2 \\ 1 & 1 \end{bmatrix}, \quad OM_S^+ = \begin{bmatrix} 1 & 1 \\ E_3 & 1 \\ 1 & E_4 \end{bmatrix}.$$

For IM_S^- and OM_S^+, the row label is $[p_1,\ p_2,\ p_3]^T$, the column label is $[t_1,\ t_2]$, and $E_1 = f_{int}(E(p_1,\ t_1)\text{<}b\text{>})$, $E_2 = f_{int}(E(p_2,\ t_2)\text{<}b\text{>})$, $E_3 = f_{int}(E(t_1,\ p_2)\text{<}b\text{>})$, and $E_4 = f_{int}(E(t_2, p_3)\text{<}b\text{>})$.

The choice CPN module is defined below.

Definition 12: CCM $= \{\sum, P, T, A, E, IM_C^-, OM_C^+\}$ is a choice CPN module, and the default binding is b, where

1) $\sum$ is a finite color set;
2) $P = \{p_1, p_2, \ldots, p_n\}$ with $n \in \mathbb{N}^+$;
3) $T = \{t_1, t_2, \ldots, t_{n-1}\}$;
4) A is a set of directed arcs;
5) E is an arc expression function;
6) $IM_C^- = \begin{bmatrix} 1_{n\times n} & Int''_n \\ VInt_n^T & 1_n^T \\ 1_n^T & 1_n^T \end{bmatrix}$,
where $1_{n\times n}$ is an $n \times n$ matrix with each element being one. $Int''_{\ n}$ is an n-dimension square matrix, elements on the principal diagonal are $f_{int}(E(p_i,\ t_{i+n})$ $\text{<}b\text{>})$, $i \in \mathbb{N}_n^+$, and other elements are one. $VInt_n$ is an n-dimension with elements being $f_{int}(E(p_{1+n}, t_i)\text{<}b\text{>})$, $i \in \mathbb{N}_n^+$, and 1_n is an n-dimension with elements being one; and
7) $OM_C^+ = \begin{bmatrix} Int'''_n & 1_{n\times n} \\ 1_n^T & 1_n^T \\ 1_n^T & VInt'^T_n \end{bmatrix}$,
where $Int'''_{\ n}$ is an n-dimension square matrix, elements on the principal diagonal are $f_{int}(E(t_i,\ p_i)\text{<}b\text{>})$, $i \in \mathbb{N}_n^+$, and other elements are one. $VInt'_{\ n}$ is an n-dimension with elements being $f_{int}(E(t_{i+n},\ p_{n+2})$ $\text{<}b\text{>})$, $i \in \mathbb{N}_n^+$.

A choice CPN module represents multiple competing choices for a successive operation.

For example, when $n = 3$, the input matrix IM_C^- and the output matrix OM_C^+ are respectively given below.

$$IM_C^- = \begin{bmatrix} 1 & 1 & 1 & E_1 & 1 & 1 \\ 1 & 1 & 1 & 1 & E_2 & 1 \\ 1 & 1 & 1 & 1 & 1 & E_3 \\ E_4 & E_5 & E_6 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix},$$

$$OM_C^+ = \begin{bmatrix} E_7 & 1 & 1 & 1 & 1 & 1 \\ 1 & E_8 & 1 & 1 & 1 & 1 \\ 1 & 1 & E_9 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & E_{10} & E_{11} & E_{12} \end{bmatrix}.$$

In IM_C^- and OM_C^+, the row label is $[p_1,\ p_2,\ p_3,\ p_4,\ p_5]^T$, the column label is $[t_1,\ t_2,\ t_3,\ t_4,\ t_5,\ t_6]$, and $E_1 = f_{int}(E(p_1,$

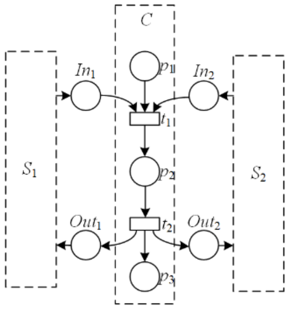


FIGURE 4. An ECLPN example.

$t_4)<b>)$, $E_2 = f_{int}(E(p_2, t_5)<b>)$, $E_3 = f_{int}(E(p_3, t_6)<b>)$, $E_4 = f_{int}(E(p_4, t_1)<b>)$, $E_5 = f_{int}(E(p_4, t_2)<b>)$, $E_6 = f_{int}(E(p_4, t_3)<b>)$, $E_7 = f_{int}(E(t_1, p_1)<b>)$, $E_8 = f_{int}(E(t_2, p_2)<b>)$, $E_9 = f_{int}(E(t_3, p_3)<b>)$, $E_{10} = f_{int}(E(t_4, p_5)<b>)$, $E_{11} = f_{int}(E(t_5, p_5)<b>)$, and $E_{12} = f_{int}(E(t_6, p_5)<b>)$.

B. THE COMPOSITION OF ECLPNS

In this section, a complicated system S is modeled by the composition of ECLPNs. The subsystem ss of S is modeled by an ECLPN, and CPN modules form ss.

A complicated system is composed of multiple subsystems. There is no guarantee that different subsystems can run at exactly the same pace, and one subsystem process cannot be delayed by another slow subsystem. To solve this problem, we use an ECLPN to describe the composition in this work. A subsystem S in Fig. 4 is an example.

S is formed by one sequential CPN module C and two CPN modules S_1 and S_2. S_1 and S_2 have different structures and are connected with C via places In_1, In_2, Out_1, and Out_2. If t_1 fires by the CPN firing rules, it requires tokens in all its pre-sets. Therefore, it is necessary to attach a logic expression to t_1, for the sake of guaranteeing the independence of S_1 and S_2. Similarly, if t_2 fires by the CPN firing rule, it outputs tokens to all its post-sets, without checking whether C has imported a token from a certain subsystem or not. Thus, it is necessary to attach a logic expression to t_2, to ensure that t_2 outputs tokens to correct places.

The binding for S is $b = \{v_1 = c, v_2 = s_1, v_3 = s_2\}$, where c denotes a token in C, s_1 denotes a token in S_1, and s_2 denotes a token in S_2. Logic expressions attached to both t_1 and t_2 are $f_I(t_1) = f_O(t_2) = c \wedge (s_1 \vee s_2)$. Expressions on directed arcs are set as follows:

$E(p_1, t_1)<b>= c$, $E(In_1, t_1)<b>= s_1$, $E(In_2, t_1)<b>= s_2$,

$E(t_1, p_2)<b>= \{c, (c, s_1), (c, s_2), (c, s_1, s_2)\}$,

$E(p_2, t_2)<b>= \{c, (c, s_1), (c, s_2), (c, s_1, s_2)\}$,

$E(t_2, Out_1)<b>= s_1$, $E(t_2, p_3)<b>= c$, $E(t_2, Out_2)<b>= s_2$.

By the firing rule in Definition 9, the ECLPN model can describe the system soundly.

The composition of ECLPNs is defined below.

Definition 13: $\text{ECLPN}_l = (\sum_l, P_l, T_l, A_l, C_l, G_l, E_l, Init_l, I_l, O_l, M_l)$, $l \in \mathbb{N}_n^+$, are n ECLPNs in a complicated system. $\text{ECLPN}_C = (\sum, P, T, A, C, G, E, Init, I, O, M)$ is the composition of ECLPN_l, $l \in \mathbb{N}_n^+$, where

1) $\sum = \cup_l \sum_l$;
2) $P = (\cup_l P_l) \cup P_C$, where P_C is the set of channel places. Different ECLPNs are connected with channel places;
3) $T = \cup_l T_l$;
4) $A = (\cup_l A_l) \cup A_C$, A_C is a set of arcs. Arcs in A_C are related to channel places;
5) $C: P \rightarrow \sum$,
$$C(p) = \begin{cases} \Sigma_l, & p \in P_l \\ \Sigma_i \cup \Sigma_j, & p \in P_c, \end{cases}$$
where $\sum_i$ and $\sum_j$ are color sets of ECLPN_i and ECLPN_j;
6) G is a guard function, and $G(t) = G_l(t)$, $t \in T_l$;
7) E is an arc expression function, the binding $b = \cup_l b_l$,
$$E(a) < b >= \begin{cases} E_l(a) < b_l >, & a \in A_l \\ E_c(a) < b_l >, & a \in A_c, \end{cases}$$
where $E_c(a)<b>$ is the expression on arcs $A \in A_c$, which is set by the actual situation. $\forall p_c \in P_c$: $E_c({}^\bullet p_c, p_c)<b>= E_c(p_c, p_c^\bullet)<b>$, because channel places change no token;
8) $Init$ is an initialization function, where
$$Init(p) = \begin{cases} Init_l, & p \in P_l \\ \emptyset, & p \in P_c; \end{cases}$$
9) I is a mapping from T_L to a logic input expression, such that $\forall t \in \cup_l T_{Ll}$, $I(t) = I_l(t)$;
10) O is a mapping from T_L to a logic output expression, such that $\forall t \in \cup_l T_{Ll}$, $O(t) = f_O$(t); and
11) $M: P \rightarrow \sum_{MS}$ is the marking function. M_0 denotes the initial marking, whereas M_E denotes the end marking.

C. THE PROPERTY ANALYSIS OF ECLPNC

In this section, we discuss the conditions for maintaining robustness and strict conservativeness of ECLPNs.

The robustness of an ECLPN shows that all the tokens in the initial place at the initial marking can flow to the end place at the end marking. It reveals that every subsystem can complete the task and there is no remaining token in the process. Let M_0 denote the initial marking, and M_E denote the end marking. The definition of robustness for an ECLPN is presented below.

Definition 14: An $\text{ECLPN} = \{\sum, P, T, A, C, G, E, Init, I, O\}$ is robust if

1) $M_0(p) = \begin{cases} Init(p), & p = p_0 \\ \emptyset, & p \in P \backslash \{p_0\}; \end{cases}$
2) $M_E(p) = \begin{cases} Init(p_0), & p = p_E \\ \emptyset, & p \in P \backslash \{p_E\}; \end{cases}$ and
3) $\forall t \in T$, t is live.

The composition of ECLPNs is an ECLPN. Conditions for the inheritance of robustness are discussed below.

If a component ECLPN is not robust, then, the composition is not robust. Provided that each system is robust before composition, the robustness of the composition depends on arc expressions which are connected to channel places. we have the following result.

Theorem 1: Let ECLPN_C be the composition of $\text{ECLPN}_l = (\sum_l, P_l, T_l, A_l, C_l, G_l, E_l, Init_l, I_l, O_l, M_l)$, $l \in \{1, 2\}$, ECLPN_C is robust if

1) ECLPN_1 and ECLPN_2 are robust; and
2) For the binding b of ECLPN_C, there is a set of channel places $P_{cI} \subseteq P_c$, $P_{cI} = \{p_{cIj} | {}^\bullet p_{cIj} \in \text{ECLPN}_1\}$, $j \in \mathbb{N}^+_{|PcI|}$ and $a_{Ij} = ({}^\bullet p_{cIj}, p_{cIj})$. Similarly, there is a channel place set $P_{cO} \subseteq P_c$, $P_{cO} = \{p_{cOk} | p^\bullet_{cOk} \in \text{ECLPN}_1\}$, $k \in \mathbb{N}^+_{|PcO|}$ and $a_{Ok} = (p_{cOk}, p^\bullet_{cOk})$, and then,
$\Sigma_{j=1}^{|P_{cI}|} E(a_{Ij})<b> \subseteq \Sigma_{k=1}^{|P_{cO}|} E(a_{Ok})<b>$, where the subset of a multiset is defined in Definition 8.

Proof: Let Sum_1 denote $\Sigma_{j=1}^{|P_{cI}|} E(a_{Ij})<b>$ and Sum_2 denote $\Sigma_{k=1}^{|P_{cO}|} E(a_{Ok})<b>$. The type sets of Sum_1 and Sum_2 are respectively $Elem_1$ and $Elem_2$. The coefficient sets of Sum_1 and Sum_2 are respectively $H_1(Elem_1) = \{h_1(e)|e \in Elem_1\}$ and $H_2(Elem_2) = \{h_2(e)|e \in Elem_2\}$. The theorem is proved by 6 situations of relation between Sum_1 and Sum_2, which can be divided into 3 classes of relation between $Elem_1$ and $Elem_2$ and whether the Condition a) holds or not. Condition a) is $\forall e \in Elem_1$: $h_1(e) \leq h_2(e)$, where if $e \notin Elem_2$, then $h_2(e) = 0$.

Relations between Sum_1 and Sum_2 are discussed by 6 situations as follows:

a): Condition a) fails and $Elem_1 \supsetneq Elem_2$;
b): Condition a) fails and $Elem_1 \subseteq Elem_2$;
c): Condition a) fails and $\exists e_1 \in Elem_1$: $e_1 \notin Elem_2$ and $\exists e_2 \in Elem_2$: $e_2 \notin Elem_1$;
d): Condition a) holds and $Elem_1 \supsetneq Elem_2$;
e): Condition a) holds and $Elem_1 \subseteq Elem_2$; and
f): Condition a) holds and $\exists e_1 \in Elem_1$: $e_1 \notin Elem_2$ and $\exists e_2 \in Elem_2$: $e_2 \notin Elem_1$.

1) If Condition a) fails, then $\exists e_1 \in Elem_1$: $h_1(e_1) > h_2(e_1)$. Let $r = h_1(e_1)\text{-}h_2(e_1)$, there is no transition in ECLPN_2 consumes r tokens colored as e_1, then $r \cdot e_1 \in M_E({}^{\bullet\bullet}p_{cOk})$, $k = \mathbb{N}^+_{|PcO|}$, ${}^{\bullet\bullet}p_{cOk} \in P_2$ and ${}^{\bullet\bullet}p_{cOk} \neq p_{E2}$, p_{E2} is the end place in ECLPN_2. From ${}^{\bullet\bullet}p_{cOk} \neq \emptyset$, ECLPN_2 is not robust. Therefore, situations a) - c) cannot ensure that ECLPN_C is robust;
2) The situation d) does not exist, because $Elem_1 \supsetneq Elem_2$ which conflicts with Condition a). If $\exists e_1 \in Elem_1$: $e_1 \notin Elem_2$, then $h_1(e_1) > h_2(e_1)$;
3) If $Elem_1 \subseteq Elem_2$ and Condition a) holds, then $Sum_1 \subseteq Sum_2$. There is no redundant token in ECLPN_2. When the system reaches the end marking, the robustness of ECLPN_1 and ECLPN_2ensures that $M_E(p_E) = Init(p_0)$, $M_E(p) = \emptyset$, and $p \neq p_E$. Therefore, ECLPN_C is robust under situation e); and
4) If $\exists e_1 \in Elem_1$: $e_1 \notin Elem_2$, $\exists e_2 \in Elem_2$: $e_2 \notin Elem_1$, and $D_1 = \{d_1 | d_1 \in Elem_1 \backslash Elem_2\}$, then the type set of $M_E({}^{\bullet\bullet}p_{cOk})$ is D_1, $k = \mathbb{N}^+_{|PcO|}$. For ${}^{\bullet\bullet}p_{cOk} \in P_2$ and ${}^{\bullet\bullet}p_{cOk} \neq p_{E2}$, p_{E2} is the end place in ECLPN_2. From ${}^{\bullet\bullet}p_{cOk} \neq \emptyset$, ECLPN_2 is not robust. Therefore, the situation f) cannot ensure ECLPN_C be robust.

In 3), if $\exists e_2 \in Elem_2$: $e_2 \notin Elem_1$, there is no token colored as e_2 that flows to ECLPN_2 from ECLPN_1, a transition${}^\bullet p_{cOk}$, $k \in \mathbb{N}^+_{|PcO|}$, is still enabled. When it fires, it generates no token colored as e_2 to p_{cOk} according to the firing rule of ECLPNs. Thus, ECLPN_1 is robust. ■

In an ECLPN, colored tokens represent data packages. Data are processed in different subsystems. The color indicates the source of a token. After the data processing, tokens return to the source according to their colors. The strict conservativeness of an ECLPN_C reveals that types and quantities of tokens remain static from the initial marking to any reachable marking.

The strict conservativeness for an ECLPN is defined as follows.

Definition 15: An ECLPN = $\{\sum, P, T, A, C, G, E, Init, I, O\}$ is strictly conservative if

1) M_0 is the initial marking, where $M_0(p) = \begin{cases} Init(p), & p = p_0 \\ \emptyset, & p \in P \backslash \{p_0\} \end{cases}$; and
2) $\forall M \in R(M_0)$: $\Sigma_{j=1}^m M(p_j) = \Sigma_{j=1}^m M_0(p_j)$.

Similar to the robustness, given that each system is strictly conservative, the inheritance of strict conservativeness also depends on arc expressions which are connected to channel places. Then we have the following theorem.

Theorem 2: Let ECLPN_C be composed by $\text{ECLPN}_l = (\sum_l, P_l, T_l, A_l, C_l, G_l, E_l, Init_l, I_l, O_l, M_l)$, $l \in \{1, 2\}$. ECLPN_C is strictly conservative if

1) ECLPN_1 and ECLPN_2 are robust and strictly conservative; and
2) For the binding b of ECLPN_C, there is a set of channel places $P_{cI} \subseteq P_c$, $P_{cI} = \{p_{cIj} | {}^\bullet p_{cIj} \in \text{ECLPN}_1\}$, $j \in \mathbb{N}^+_{|PcI|}$ and $a_{Ij} = ({}^\bullet p_{cIj}, p_{cIj})$. Similarly, there is a channel place set $P_{cO} \subseteq P_c$, $P_{cO} = \{p_{cOk} | p^\bullet_{cOk} \in \text{ECLPN}_1\}$, $k \in \mathbb{N}^+_{|PcO|}$ and $a_{Ok} = (p_{cOk}, p^\bullet_{cOk})$, and then,
$\Sigma_{j=1}^{|P_{cI}|} E(a_{Ij})<b> \subseteq \Sigma_{k=1}^{|P_{cO}|} E(a_{Ok})<b>$;

Proof: In ECLPN_l, $l \in \{1, 2\}$, M_{l0} is the initial marking, and $\forall M_l \in R(M_{l0})$. M_{lE} is the end marking. p_{l1} is the initial place, and p_{lE} is the end place. $Elem_l$ is the type set for $M_{l0}(p_{l1})$. $H(Elem_l) = \{h(e)|e \in Elem_l\}$ is the coefficient set for $Elem_l$.

Given that ECLPN_1 is strictly conservative, we have $\Sigma_{j=1}^{m_1} M_1(p_{1j}) = \Sigma_{j=1}^{m_1} M_{10}(p_{1j})$. Given that ECLPN_1 is robust, we have $\Sigma_{j=1}^{m_1} M_1(p_{1j}) = M_{10}(p_{11}) = M_{1E}(p_{1E})$, and by the sum form, we have $M_{10}(p_{11}) = M_{1E}(p_{1E}) = \sum_{e \in Elem1} h(e) \cdot e = \sum_{e \in Elem1} h(e) \cdot e$.

Similarly, in ECLPN_2, we have $M_{20}(p_{21}) = M_{2E}(p_{2E}) = \sum_{e \in Elem2} h(e) \cdot e = \sum_{e \in Elem2} h(e) \cdot e$.

Like Theorem 1, let Sum_1 denote $\Sigma_{j=1}^{|P_{cI}|} E(a_{Ij})<b>$ and Sum_2 denote $\Sigma_{k=1}^{|P_{cO}|} E(a_{Ok})<b>$. The theorem is also proven from 6 situations regarding the relation between Sum_1 and Sum_2.

1) When Condition a) fails, $\exists e_1 \in Elem_1$, such that $h_1(e_1) > h_2(e_1)$. Let $r = h_1(e_1)$-$h_2(e_1)$. There is no transition in ECLPN$_2$ consuming the remaining r tokens colored as e_1, and then $r \cdot e_1 \in M_E(^{\bullet\bullet}p_{cOk})$, $k = \mathbb{N}^+_{|PcO|}$, where $^{\bullet\bullet}p_{cOk} \in P_2$ and $^{\bullet\bullet}p_{cOk} \neq p_{E2}$. Colored tokens in an ECLPN flow independently by the firing rule. Then in the marking M_{2E}, the sum of tokens is $\sum_{e\in Elem2} h(e)\cdot e + \sum_{e\in Elem1} h_r(e)\cdot e$, where $\sum_{e\in Elem1} h_r(e)\cdot e$ is the sum of tokens in place $^{\bullet\bullet}p_{cOk}$, and $h_r(e)$ is the coefficient of the type which is remaining in place $^{\bullet\bullet}p_{cOk}$. We have that $\sum_{e\in Elem1} h_r(e)\cdot e \supseteq r\cdot e_1$. Therefore, $\sum_{e\in Elem2} h(e)\cdot e + \sum_{e\in Elem1} h_r(e)\cdot e \supseteq \sum_{e\in Elem2} h(e)\cdot e$. Thus, ECLPN$_2$ is not strictly conservative. Therefore, situations a) - c) cannot ensure that ECLPN$_C$ is strictly conservative;
2) The situation d) does not exist, and the reason is the same with 2) in the proof for Theorem 1;
3) In situation e), if $Elem_1 \subseteq Elem_2$ and Condition a) holds, then $Sum_1 \subseteq Sum_2$. Given that ECLPN$_1$ and ECLPN$_2$ are strictly conservative, we have the sum of tokens stay constant in ECLPN$_1$ and ECLPN$_2$. $\forall p_c \in P_c$: $E_c(^{\bullet}p_c, p_c)<b> = E_c(p_c, p_c^{\bullet})<b>$ (Definition 13). It shows that the sum of tokens does not change when tokens transferring between ECLPNs. Besides, $Sum_1 \subseteq Sum_2$ ensures that there is no redundant token in ECLPN$_2$. Therefore, ECLPN$_C$ is strictly conservative; and
4) If $\exists e_1 \in Elem_1$: $e_1 \notin Elem_2$, $\exists e_2 \in Elem_2$: $e_2 \notin Elem_1$, and $D_1 = \{d_1 | d_1 \in Elem_1 \backslash Elem_2\}$, then the type set of $M_E(^{\bullet\bullet}p_{cOk})$ is D_1, $k = \mathbb{N}^+_{|PcO|}$. There is no transition in ECLPN$_2$ consuming the remaining tokens in $M_E(^{\bullet\bullet}p_{cOk})$, $k = \mathbb{N}^+_{|PcO|}$. In the marking M_{2E}, the sum of tokens is $\sum_{e\in Elem2} h(e)\cdot e + \sum_{e\in D1} h_r(e)\cdot e$, where $\sum_{e\in D1} h_r(e)\cdot e$ is the sum of tokens in place $^{\bullet\bullet}p_{cOk}$, and $h_r(e)$ is the coefficient of the element which is remaining in place $^{\bullet\bullet}p_{cOk}$. Then we have $\sum_{e\in D1} h_r(e)\cdot e \neq \emptyset$, and therefore, $\sum_{e\in Elem2} h(e)\cdot e + \sum_{e\in D1} h_r(e)\cdot e \supseteq \sum_{e\in Elem2} h(e)\cdot e$. Thus, ECLPN$_2$ is not strictly conservative. As a result, situation f) cannot ensure that ECLPN$_C$ is strictly conservative. ■

D. COMPOSITION PROCEDURE OF ECLPN$_S$

Suppose that a complicated system is composed of multiple subsystems, and information transfer between subsystems. Then the composing procedure of an ECLPN$_C$ model is presented below.

Step 1: Build a CPN subnet sn_1 for a subsystem by sequential and choice CPN modules, and use colored token c_1 to simulate the running process of the subsystem;

Step 2: For another subsystem, if it shares the same structure with sn_1, then it uses a colored token c_2 in sn_1 to simulate its running process. We then transform ordinary transitions into logical transitions by attaching logical expressions. Arc expressions also make changes accordingly, and sn_1 becomes an ECLPN. If the subsystem has a different structure from sn_1, then we construct another CPN subnet $sn_i, i = \{2, 3, \ldots\}$, and use colored token c_i to simulate its running process. Repeat Step 2 until all subsystems are simulated in subnets;

Step 3: For a subnet sn_i, $i \in \mathbb{N}^+_n$, if it receives tokens from sn_j, $j = \{1, 2, \ldots, i-1, i+1, \ldots\}$, then we attach logic expressions to the transition in sn_j. The expression includes all types in sn_j. Arc expressions also make changes accordingly, and sn_j becomes an ECLPN; and

Step 4: Connect all sn_i, $i = \{1, 2, \ldots\}$, by channel places and set arc expressions. The Composition of ECLPNs is complete.

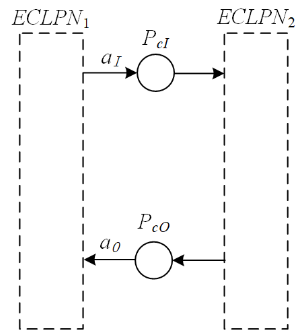


FIGURE 5. An ECLPN$_C$ example.

Remark 1: In Step 1, building a CPN subnet begins with a simple, bounded, live, reversible CPN. It is elaborated by sequential and choice modules in a Top-down manner [30].

Remark 2: In Step 2, the purpose of attaching logic expressions is to keep c_1 and c_2 independent.

Remark 3: In Step 3, if there is a CPN subnet, it has only one type of token, and receives no token from any other subnets, and then, it can be regarded as an ECLPN, whose logic expression of all transitions is the Boolean constant *true*.

IV. ECLPN MODEL OF AN E-COMMERCE SYSTEM

In this section, An ECLPN model $ECLPN_{EC}$ is presented to describe an E-commerce system [29] and make a comparison with the LPN model LPN_{EC} in [29].

A CLPN [27] model is similar to LPN in scale. Therefore, we compare $ECLPN_{EC}$ with LPN_{EC} instead of a CLPN model.

A. INTRODUCTION OF AN E-COMMERCE SYSTEM

The E-commerce system is composed of 3 subsystems, i.e., customers, a merchant and a third-party. Customers accomplish the following operations: placing an order, making the payment or being refused, confirming the receipt of the commodity, and waiting for the confirmation of the receipt of payment from the merchant. The merchant sequentially receives an order, checks the inventory to decide whether to accept the order or not, checks the payment, delivers the product and confirms the receipt of payment. The third-party keeps the

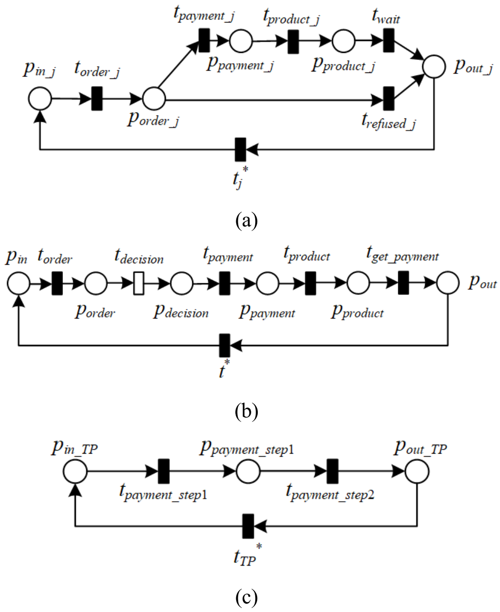


FIGURE 6. **Three subnets: (a) the seller subnet sn_c, (b) the merchant subnet sn_m, and (c) the third-party subnet sn_{tp}.**

payment until the customer has confirmed the receipt of the commodity.

B. ECLPN$_{EC}$ MODEL

The composing procedure of ECLPN$_{\text{EC}}$ according to Section III is given below.

Step 1: Construct the customer subnet sn_c by sequential and choice CPN modules. For all the customer subsystems that share the same structure, their operation processes are all described in sn_c by different colored tokens;

Step 2: Construct the merchant subnet sn_m and the third-party subnet sn_{tp} by sequential CPN modules. sn_c, sn_m and sn_{tp} are shown in Fig. 6(a), (b), and (c), respectively;

In sn_c, the binding is $b_1 = <con_1 = c, v_1 = c_1, v_2 = c_2, \ldots v_n = c_n>$. $\{v_1, v_2, \ldots, v_n\}$ denotes the set of tokens that are carrying data in customer subsystems. $\{c_1, c_2, \ldots, c_n\}$ denotes the color of tokens, and colors implicate customers in sn_c. $\{con_1\}$ denotes the control token, which is colored as $\{c\}$. The control token controls and records the operation process of other tokens. p_{in_j} is the initial place and p_{out_j} is the end place.

Let set_c be the set of MS_c, where MS_c makes the logic expression $\cup_{i=1}^{n} c_i$ be *true*. There are 2^n elements in set_c. In sn_c, all arc expressions are $\{c\} \cup set_c$.

In sn_m, the binding is $b_2 = <con_2 = m>$, and $\{con_2\}$ denotes the control token in sn_m, which is colored as $\{m\}$. All arc expressions are m. p_{in} is the initial place, and p_{out} is the end place.

In sn_{tp}, the binding is $b_3 = <con_3 = tp>$, and $\{con_3\}$ denotes the control token in sn_{tp}, which is colored as $\{tp\}$. All arc expressions are tp. p_{in_TP} is the initial place, and p_{out_TP} is the end place.

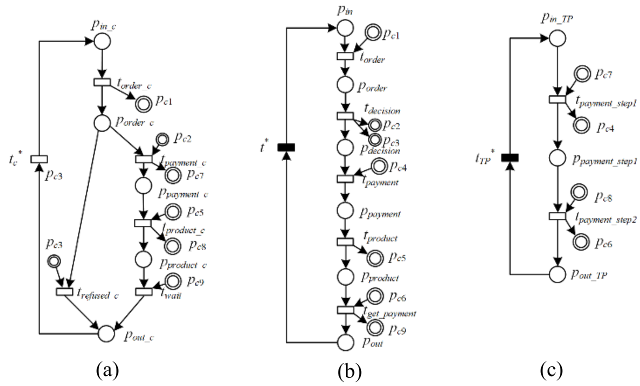


FIGURE 7. **The composition of three ECLPNs model ECLPN$_{\text{EC}}$. (a) the customer subnet sn_c, (b) the merchant subnet sn_m, and (c) the third-party subnet sn_{tp}.**

Step 3: Connect the three subnets by channel places $\{p_{c1}, p_{c2}, \ldots, p_{c9}\}$, and the composition of ECLPNs is built as Fig. 7.

In sn_m, transition $t_{decision}$ checks whether the product is in stock or not. If it is in stock, related tokens flow to p_{c2}, otherwise they flow to p_{c3}. Tokens flowing into sn_m via p_{c1} are the same with tokens flowing out of sn_m through p_{c2} and p_{c3}.

In Fig. 7, there are 23 places, 15 transitions, and 48 directed arcs.

The binding of $ECLPN_{EC}$ is $b_M = <con_1 = c, con_2 = m, con_3 = tp, v_1 = c_1, \ldots v_n = c_n>$, con_1- con_3 are control tokens for sn_c, sn_m, and sn_{tp}, respectively. $\{v_1, v_2, \ldots, v_n\}$ is the set of tokens carrying data in customer subsystems. They are produced in sn_c, flow into sn_m and sn_{tp}, and return to sn_c in the end.

Arc expressions in $ECLPN_{EC}$ are updated as follows.

All tokens in the set $\{v_1, v_2, \ldots, v_n\}$ transmit between subsystems. Therefore, all arc expressions which are related with channel places $\{p_{c1}, p_{c2}, \ldots, p_{c9}\}$ are set_c.

In sn_c, expressions on these arcs are $\{c\}$, and they are $(t_{order\_c}, p_{order\_c})$, $(p_{order\_c}, t_{payment\_c})$, $(t_{payment\_c}, p_{payment\_c})$, $(p_{payment\_c}, t_{product\_c})$, $(t_{product\_c}, p_{product\_c})$, $(p_{product\_c}, t_{wait})$, and $(p_{order\_c}, t_{refused\_c})$. Expressions on arcs (p_{in_c}, t_{order_c}), (t_{wait}, p_{out_c}), $(t_{refused\_c}, p_{out\_c})$, (p_{out_c}, t_c*) and (t_c*, p_{in_c}) remain $\{c\} \cup set_c$.

In sn_m, expressions on these arcs are $\{m\} \cup set_c$, and they are (t_{order}, p_{order}), $(p_{order}, t_{decision})$, $(t_{payment}, p_{payment})$, and $(p_{payment}, t_{product})$. Expressions on arcs (p_{in}, t_{order}), $(t_{decision}, p_{decision})$, $(p_{decision}, t_{payment})$, $(t_{product}, p_{product})$, $(p_{product}, t_{get\_payment})$, $(t_{get\_payment}, p_{out})$, (p_{out}, t^*), and (t^*, p_{in}) remain $\{m\}$.

In sn_{tp}, expressions on arcs $(p_{in\_TP}, t_{payment\_step1})$, $(t_{payment\_step1}, p_{payment\_step1})$, $(p_{payment\_step1}, t_{payment\_step2})$, $(t_{payment\_step2}, p_{out\_TP})$, $(p_{out\_TP}, t_{TP}*)$, and $(t_{TP}*, p_{in\_TP})$ remain $\{tp\}$.

Logic transitions and logical expressions are set below.

In sn_c, all transitions are logic transitions, and input and output expressions are $f_I = f_O = \{c\} \cup set_c$;

$$M_0 \xrightarrow{t_{order\_c}} M_1 \xrightarrow{1)} M_2 \xrightarrow[t_{payment\_c}]{t_{refused\_c}} M_3 \xrightarrow{t_c^*} M_4$$

$$M_4 \xrightarrow{2)} M_5 \xrightarrow{t_{product\_c}} M_6 \xrightarrow{3)} M_7 \xrightarrow{t_{wait}} M_8 \xrightarrow{t_c^*} M_0$$

FIGURE 8. The running process of subsystem sn_c.

$$M_0 \xrightarrow{1)} M_1 \xrightarrow{t_{order}} M_2 \xrightarrow{t_{decision}} M_3 \xrightarrow{2)} M_4 \xrightarrow{t_{payment}} M_5$$

$$M_5 \xrightarrow{t_{product}} M_6 \xrightarrow{3)} M_7 \xrightarrow{t_{get\_payment}} M_8 \xrightarrow{t^*} M_9 \xrightarrow{4)} M_0$$

FIGURE 9. The running process of subsystem sn_m.

In sn_m, all transitions are logic transitions except t^*, and input and output expressions are $f_I = f_O = \{m\} \cup set_c$. t^* is an ordinary transition; and

In sn_{tp}, all transitions are logic transitions except t_{TP}^*, and input and output expressions are $f_I = f_O = \{tp\} \cup set_c$. t_{TP}^* is an ordinary transition.

Suppose that there are three customers c_1, c_2 and c_3 in the system. c_1 and c_2 can get the goods from the merchant, whereas c_3 is refused, because the goods are sold out. Three processes are needed, representing three subsystems sn_c, sn_m and sn_{tp}, respectively. Let set_c be the set of MS_c, where MS_c makes the logic expression $c_1 \cup c_2 \cup c_3$ be *true* in this instance. While the system is running, different transitions fire concurrently when enabled.

The running process of subsystem sn_c is presented below in Fig. 8.

The operations 1)-3) in Fig. 8 are explained as follows.

1) Process sn_c initializes process sn_m, sends tokens in p_{c1} to sn_m and suspends process sn_c. Process sn_c awakes when tokens in p_{c2} or p_{c3} are received from sn_m;
2) Process sn_c initializes process sn_{tp}, sends tokens in p_{c7} to sn_{tp} and suspends process sn_c. Process sn_c awakes when tokens in p_{c5} are received from sn_m; and
3) Process sn_c sends tokens in p_{c8} to sn_{tp}, suspends process sn_c and wakes up process sn_{tp}. Process sn_c awakes when tokens in p_{c9} are received from sn_m.

With the column label being [p_{in_c}, p_{c1}, p_{order_c}, p_{c2}, p_{c7}, $p_{payment\_c}$, p_{c5}, p_{c8}, $p_{product\_c}$, p_{c9}, p_{c3}, p_{out_c}], markings in Fig.8 are given below.

$M_0 = (3c + c_1 + c_2 + c_3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$,
$M_1 = (0, c_1 + c_2 + c_3, 3c, 0, 0, 0, 0, 0, 0, 0, 0, 0)$,
$M_2 = (0, 0, 3c, c_1 + c_2, 0, 0, 0, 0, 0, 0, c_3, 0)$,
$M_3 = (0, 0, 0, 0, c_1 + c_2, 2c, 0, 0, 0, 0, 0, c + c_3)$,
$M_4 = (c + c_3, 0, 0, 0, c_1 + c_2, 2c, 0, 0, 0, 0, 0, 0)$,
$M_5 = (c + c_3, 0, 0, 0, 0, 2c, c_1 + c_2, 0, 0, 0, 0, 0)$,
$M_6 = (c + c_3, 0, 0, 0, 0, 0, 0, c_1 + c_2, 2c, 0, 0, 0)$,
$M_7 = (c + c_3, 0, 0, 0, 0, 0, 0, 0, 2c, c_1 + c_2, 0, 0)$, and
$M_8 = (c + c_3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2c + c_1 + c_2)$.

The running process of subsystem sn_m is presented in Fig. 9.

$$M_0 \xrightarrow{1)} M_1 \xrightarrow{t_{payment\_step1}} M_2 \xrightarrow{2)} M_3 \xrightarrow{t_{payment\_step2}} M_4 \xrightarrow{t_{TP}^*} M_5 \xrightarrow{3)} M_0$$

FIGURE 10. The running process of subsystem sn_{tp}.

The operations 1)-4) in Fig. 9 are explained as follows.

1) Process sn_m receives tokens from sn_c in p_{c1};
2) Process sn_m wakes up process sn_c, sends tokens in p_{c2} and p_{c3} to sn_c, and suspends process sn_m. Process sn_m awakes when tokens in p_{c4} are received from sn_{tp};
3) Process sn_m wakes up process sn_c, sends tokens in p_{c5} to sn_c, and suspends process sn_m. Process sn_m awakes when tokens in p_{c6} are received from sn_{tp}; and
4) Process sn_m wakes up process sn_c, and sends tokens in p_{c9} to sn_c.

With the column label being [p_{in}, p_{c1}, p_{order}, p_{c2}, p_{c3}, $p_{decision}$, p_{c4}, $p_{payment}$, p_{c5}, $p_{product}$, p_{c6}, p_{c9}, p_{out}], markings in Fig. 9 are given below.

$M_0 = (3m, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$,
$M_1 = (3m, c_1 + c_2 + c_3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$,
$M_2 = (0, 0, 3m + c_1 + c_2 + c_3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$,
$M_3 = (0, 0, 0, c_1 + c_2, c_3, 3m, 0, 0, 0, 0, 0, 0, 0)$,
$M_4 = (0, 0, 0, 0, 0, 3m, c_1 + c_2, 0, 0, 0, 0, 0, 0)$,
$M_5 = (0, 0, 0, 0, 0, 0, 0, 3m + c_1 + c_2, 0, 0, 0, 0, 0)$,
$M_6 = (0, 0, 0, 0, 0, 0, 0, 0, c_1 + c_2, 3m, 0, 0, 0)$,
$M_7 = (0, 0, 0, 0, 0, 0, 0, 0, 0, 3m, c_1 + c_2, 0, 0)$,
$M_8 = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, c_1 + c_2, 3m)$, and
$M_9 = (3m, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, c_1 + c_2, 0)$.

The running process of subsystem sn_{tp} is presented below in Fig.10.

The operations 1)-3) in Fig. 10 are explained as follows.

1) Process sn_{tp} receives tokens from sn_c in p_{c7};
2) Process sn_{tp} wakes up process sn_m, sends tokens in p_{c4} to process sn_m and suspends process sn_{tp}. Process sn_{tp} awakes when tokens in p_{c8} are received from sn_c; and
3) Process sn_{tp} wakes up process sn_m, and sends tokens in p_{c6} to process sn_m.

With the column label being [p_{in_TP}, p_{c7}, p_{c4}, $p_{payment\_step1}$, p_{c8}, p_{c6}, p_{out_TP}], markings in Fig. 10 are given below.

$M_0 = (2\ tp, 0, 0, 0, 0, 0, 0)$, $M_1 = (2\ tp, c_1 + c_2, 0, 0, 0, 0, 0)$, $M_2 = (0, 0, c_1 + c_2, 2\ tp, 0, 0, 0)$, $M_3 = (0, 0, 0, 2\ tp, c_1 + c_2, 0, 0)$, $M_4 = (0, 0, 0, 0, 0, c_1 + c_2, 2\ tp)$, and $M_5 = (2\ tp, 0, 0, 0, 0, c_1 + c_2, 0)$.

The property of $ECLPN_{EC}$ is analyzed as follows.

Before the composition, sn_c is robust and strictly conservative. After the composition, sn_c's output channel places are $\{p_{c1}, p_{c7}, p_{c8}\}$, input channels are $\{p_{c2}, p_{c3}, p_{c5}, p_{c9}\}$, and the following conclusions 1) - 3) hold in $ECLPN_{EC}$.

1) Tokens flowing out from sn_c via p_{c1} are the same with tokens flowing in sn_c through p_{c2} and p_{c3};
2) Tokens flowing out from sn_c via p_{c7} are the same with tokens flowing in sn_c through p_{c5}; and
3) Tokens flowing out from sn_c via p_{c8} are the same with tokens flowing in sn_c through p_{c9};

TABLE 1. The comparison between $ECLPN_{EC}$ and LPN_{EC}.

Item	$ECLPN_{EC}$	LPN_{EC}	Reduction rate
Number of places	23	$12n+10$	$(12n-13)/(12n+10)$
Number of transitions	15	$5n+10$	$(n-1)/(n+2)$
Number of directed arcs	48	$16n+30$	$(8n-9)/(8n+15)$

TABLE 2. The comparison between $ECLPN_{EC}$ and LPN_{EC} when $n = 3$.

Item	$ECLPN_{EC}$	LPN_{EC}	Reduction rate
Number of places	23	46	50%
Number of transitions	15	25	40%
Number of directed arcs	48	78	38.5%

For sn_m, it is robust and strictly conservative before the composition, and its input channel places are $\{p_{c1}, p_{c4}, p_{c6}\}$ and output channel places are $\{p_{c2}, p_{c3}, p_{c5}, p_{c9}\}$ in the composed net, where conclusions 4) to 6) hold.

4) Tokens flowing in sn_m via p_{c1} is the same with tokens flowing out from sn_m through p_{c2} and p_{c3};
5) Tokens flowing in sn_m via p_{c4} is the same with tokens flowing out from sn_m through p_{c5}; and
6) Tokens flowing in sn_m via p_{c6} is the same with tokens flowing out from sn_m through p_{c9};

For sn_{tp}, it is robust and strictly conservative before the composition, and its input channel places are $\{p_{c7}, p_{c5}\}$ and output channel places are $\{p_{c4}, p_{c6}\}$ in the composed net, where conclusions 7) and 8) hold.

7) Tokens flowing in sn_{tp} via p_{c7} is the same with tokens flowing out from sn_{tp} through p_{c4}; and
8) Tokens flowing in sn_{tp} via p_{c5} is the same with tokens flowing out from sn_{tp} through p_{c6}.

As a result, $ECLPN_{EC}$ is robust and strictly conservative by 1) to 8), according to Theorems 1 and 2.

C. COMPARISON BETWEEN $ECLPN_{EC}$ AND LPN_{EC}

Suppose that the complicated E-commerce system is composed of n customers, $n \in \mathbb{N}^+$, a seller and a third-party participant. The comparison between $ECLPN_{EC}$ in this work and LPN_{EC} in [29] are presented in Table 1.

The reduction rate in Tables 1-2 is defined as:

$$\text{Reduction rate} = 1 - \frac{\text{The item of } ECLPN_{EC}}{\text{The item of } LPN_{EC}}.$$

From Table 1, we can see that the number of places, transitions and directed arcs are constant in $ECLPN_{EC}$, whereas LPN_{EC} needs to build a subnet for each subsystem. With the increase in the number of subsystems, the scale of LPN_{EC} continues to expand.

In the offseason, for instance, suppose that there are 3 customers participating the E-commerce system at the same time, and the comparison on the scale is shown in Table 2.

TABLE 3. The comparison between $ECLPN_{EC}$ and LPN_{EC} when $n = 15$.

Item	$ECLPN_{EC}$	LPN_{EC}	Reduction rate
Number of places	23	190	87.9%
Number of transitions	15	85	82.4%
Number of directed arcs	48	270	82.2%

In the peak season, suppose that there are 15 customers participating the E-commerce system at the same time, and the comparison on the scale is shown in Table 3.

According to Tables 2 and 3, with the increase of n, the reduction rate of $ECLPN_{EC}$ is increasing, and thus $ECLPN_{EC}$ is kept in a small scale.

If there are more sellers and third-party participants, $ECLPN_{EC}$ can describe the process in those subsystems by using more colored tokens, without any change in the system structure and scale. The size of model can be further reduced.

V. CONCLUSION

In this paper, we give the method of the composition of ECLPNs and analyze its properties. The formal definition of the composition of ECLPNs is given. Conditions for the inheritance of robustness and strict conservativeness are discussed. We give the procedure of building a composition of ECLPNs from basic design modules. Besides, we apply $ECLPN_C$ in modeling and analyzing an E-commerce system, and make a comparison with an LPN model. The result shows that the ECLPN model has a simpler structure and lower complexity. Future work will study the maintenance of other properties, and apply ECLPNs in other fields, such as manufacturing systems [39], [40], knowledge-based systems [41], and transportation systems [42].

REFERENCES

[1] 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.
[2] W. Luan, L. Qi, and Y. Du, "Composition of logical Petri nets and compatibility analysis," *IEEE Access*, vol. 5, pp. 9152–9162, 2017.
[3] C. A. Petri, "Kommunikation mit automaten. Bonn: Institute fur Instrumentelle Mathematik, Schriften des IIM Nr.3, 1962. Also, English translation: Communication with automata," Griffiss Air Force Base, New York, NY, USA, Tech. Rep. RADC-TR-65–377, 1966, vol. 1, no. 1.
[4] 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.
[5] X. Guo, S. Wang, D. You, Z. Li, and X. Jiang, "A siphon-based deadlock prevention strategy for S3PR," *IEEE Access*, vol. 7, pp. 86863–86873, 2019.
[6] 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.
[7] L. Wang, 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.
[8] Z. He, Y. Du, L. Wang, L. Qi, and H. Sun, "An alpha-FL algorithm for discovering free loop structures from incomplete event logs," *IEEE Access*, vol. 6, pp. 27885–27901, 2018.

[9] H. Sun, Y. Du, L. Qi, and Z. He, "A method for mining process models with indirect dependencies via Petri nets," *IEEE Access*, vol. 7, pp. 81211–81226, 2019.

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

[11] Y. Du and C. Jiang, "Formal representation and analysis of batch stock trading systems by logical Petri net workflows," in *Formal Methods and Software Engineering*. Berlin, Germany: Springer, 2002.

[12] 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.

[13] 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.

[14] W. Zheng, Y. Du, L. Qi, and L. Wang, "A method for repairing process models containing a choice with concurrency structure by using logic Petri nets," *IEEE Access*, vol. 7, pp. 13106–13120, 2019.

[15] 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.

[16] K. Jensen, "An introduction to the theoretical aspects of coloured Petri nets," in *Proc. Workshop/School/Symp. REX Project, Res. Educ. Concurrent Syst.* Berlin, Germany: Springer, 1993.

[17] E. Kucera, O. Haffner, and S. Kozak, "Modelling and control of AS/RS using coloured Petri nets," in *Proc. Cybern. Informat. (K&I)*, Levoča, Slovakia, Feb. 2016, pp. 1–6.

[18] A. C. Fauzan, R. Sarno, and M. A. Yaqin, "Performance measurement based on coloured Petri net simulation of scalable business processes," in *Proc. 4th Int. Conf. Electr. Eng., Comput. Sci. Informat. (EECSI)*, Yogyakarta, Indonesia, Sep. 2017, pp. 1–6.

[19] G. R. Petrosyan and L. A. Ter-Vardanyan, "Modelling of identification and secret-key generation system with colored Petri net," in *Proc. Int. Conf. Control, Decis. Inf. Technol. (CoDIT)*, Saint Julian's, Malta, Apr. 2016, pp. 239–245.

[20] L. Zhu, S. Tan, W. Zhang, Y. Wang, and X. Xu, "Validation of pervasive cloud task migration with colored Petri net," *Tsinghua Sci. Technol.*, vol. 21, no. 1, pp. 89–101, Feb. 2016.

[21] C. G. F. Junior, J. M. Soares, and G. C. Barroso, "CacheSIM: A Web cache simulator tool based on coloured Petri nets and java programming," *IEEE Latin America Trans.*, vol. 13, no. 5, pp. 1511–1519, May 2015.

[22] S. Du, P. Wu, G. Wu, C. Yao, and L. Zhang, "The collaborative system workflow management of industrial design based on hierarchical colored Petri-net," *IEEE Access*, vol. 6, pp. 27383–27391, 2018.

[23] N. Akhtar, A. Rehman, M. Hussnain, S. Rohail, M. S. Missen, M. Nasir, A. Hayder, N. Salamat, and M. Pasha, "Hierarchical coloured Petri-net based multi-agent system for flood monitoring, prediction, and rescue (FMPR)," *IEEE Access*, vol. 7, pp. 180544–180557, 2019.

[24] H. Macia, V. Valero, G. Diaz, J. Boubeta-Puig, and G. Ortiz, "Complex event processing modeling by prioritized colored Petri nets," *IEEE Access*, vol. 4, pp. 7425–7439, 2016.

[25] O. T. Baruwa, M. A. Piera, and A. Guasch, "Deadlock-free scheduling method for flexible manufacturing systems based on timed colored Petri nets and anytime heuristic search," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 45, no. 5, pp. 831–846, May 2015.

[26] T. H. Chang, T. E. Lee, and C. H. Lin, "A fuzzy color Petri net for repair scheduling and dispatching of large-scale contingencies in distribution system," in *Proc. Int. Conf. Appl. Syst. Innov. (ICASI)*, Sapporo, Japan, May 2017, pp. 1036–1039.

[27] J. Wang, Y. Du, and S. Yu, "Coloured logic Petri nets and analysis of their reachable trees," *Enterprise Inf. Syst.*, vol. 9, no. 8, pp. 900–919, Feb. 2014.

[28] Z. Wang, Y. Du, and L. Qi, "Extended colored logic Petri nets and reachability analysis," *J. Shandong Univ. Sci. Technol. Natur. Sci.*, vol. 39, no. 3, 2020.

[29] 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.

[30] M. Zhou, F. DiCesare, and A. A. Desrochers, "A hybrid methodology for synthesis of Petri net models for manufacturing systems," *IEEE Trans. Robot. Autom.*, vol. 8, no. 3, pp. 350–361, Jun. 1992.

[31] 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.

[32] 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.

[33] 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.

[34] 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.

[35] 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.

[36] 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.

[37] 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.

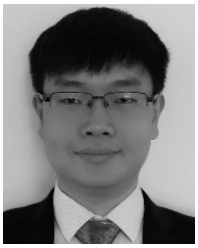
[38] X. Y. Lu, M. C. Zhou, A. C. Ammari, and J. C. Ji, "Hybrid Petri nets for modeling and analysis of microgrid systems," *IEEE/CAA J. Autom. Sinica*, vol. 3, no. 4, pp. 349–356, Oct. 2016.

[39] 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.

[40] 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. Autom. Sci. Eng.*, to be published.

[41] 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.

[42] 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.



ZHEN WANG received the B.S. degree in mathematics and applied mathematics from the Shandong University of Science and Technology, Qingdao, China, in 2014, where he is currently pursuing the master's degree in computer science. His current research interest is in Petri nets' theory and applications.

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. 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. She is currently a Lecturer of computer science and technology with the Shandong University of Science and Technology. 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) at the 13th IEEE International Conference on Networking, Sensing and Control (ICNSC 2016).

YUYUE DU received the B.S. degree from Shandong University, Jinan, China, in 1982, the M.S. degree from the 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 with the College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao, China. He has taken in over 20 projects supported by the National Nature Science Foundation, the National Key Basic Research Developing Program, and other important and key projects at provincial levels. He has published over 200 articles in domestic and international academic publications, and they are embodied over 150 times by SCI and EI. His research interests are in formal engineering, process mining, Petri nets, real-time systems, web services, and workflows.

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. He has published more than 50 articles in journals and conference proceedings. His research interests include Petri nets, machine learning, optimization, and intelligent transportation systems.