Journal of
Software Engineering

ISSN 1819-4311

Academic Journals Inc.

www.academicjournals.com

Journal of Software Engineering 8 (1): 23-31, 2014
ISSN 1819-4311 / DOI: 10.3923/jse.2014.23.31

Transition Firing Rules of Logic Petri Nets

Yu Yue Du, Jing Wang and Yong Feng Zhang
Shandong University of Science and Technology, 266590, Qingdao, China

Corresponding Author: Jing Wang, Shandong University of Science and Technology, 266590, Qingdao, China

ABSTRACT

Logical Petri nets (LPNs) can well describe and analyze batch processing functions and passing value indeterminacy in cooperative systems. A new definition of the LPNs is proposed based on our initial work in this study. The standard form of logic expressions can be obtained and the logic input/output enabling vector set is defined. How to determine the corresponding relationships between logic input and output expressions is solved. In order to analyze their properties, a vector matching method is given and a therom has been proved. Finally, the feasibility of the proposed method is illustrated by an example.

Key words: Logic Petri nets, enabled vectors, equation vectors, firing rules

INTRODUCTION

Petri nets (PNs) (Aziz *et al.*, 2013) are the mathematics representation of a parallel discrete system. PNs have a rigid mathematical definition, with a well-developed mathematical theory for process analysis. They are suitable for describing the concurrent asynchronous of computer system models. With the continuous improvement of PN theory and the increasing popularity of its application, some of their extensions have been defined, such as fuzzy (Bayati and Dideban, 2012), colored (Barzegar *et al.*, 2011) and stochastic (Marin *et al.*, 2012) PNs.

Logic Petri nets (LPNs) (Du and Jiang 2002) are the abstract and extension of inhibitor arcs PNs (IPNs) and high-level PNs. The inputs and outputs of a system can be described by logic transitions in LPNs, respectively. The transitions restricted by logic input and output expressions are called logic transitions. In our initial work, LPNs have been applied efficiently to the modeling and analysis of trading systems (Du and Jiang, 2002), cooperative systems (Du *et al.*, 2009) and electronic commerce (Du and Jiang, 2008). Some weaknesses are found in the analysis of LPNs models. In this study, a new definition of LPNs are proposed. The indeterminate data transmission caused by logic output transitions is solved by introducing the matching expressions. The standard form of logic expressions are defined and some indeterminate minterms of the logic expression are deleted.

BASIC DEFINITIONS

Three basic definitions are reviewed in this section before LPNs are introduced.

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

- P is a finite set of places
- T is a finite set of transitions with $P \cup T \neq \varnothing$, $P \cap T = \varnothing$
- $F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs
- $dom(F) \cup cod(F) = P \cup T$ where:

$$dom(F) = \{x \in P \cup T \mid \exists y \in P \cup T: (x, y) \in F\}$$

$$cod(F) = \{x \in P \cup T \mid \exists y \in P \cup T: (y, x) \in F\}$$

Definition 2: $x \in P \cup T$ is called a node in N:

$${}^{\bullet}x = \{y \mid (y, x) \in F\} \text{ is called a pre-set of x}$$

$$x^{\bullet} = \{y \mid (x, y) \in F\} \text{ is called a post-set of x}$$

If $X \in P \cup T$, its pre-set and post-set are as follows:

$${}^{\bullet}X = \cup_{x \in X} {}^{\bullet}x$$

$$X^{\bullet} = \cup_{x \in X} x^{\bullet}$$

Definition 3: $PN = (P, T, F, M_0)$ is a marked PN, where:

- $N = (P, T, F)$ is a net
- $M: P \to \mathbb{N}^+$ is a marking function, where M_0 is the initial marking and $\mathbb{N}^+ = \{1, 2, \ldots\}$
- Transition firing rules:
  - t is enabled at M if $\forall p \in {}^{\bullet}t$: $M(p) = 1$, represented by $M[t>$
  - If t is enabled, it can fire and a new marking M' is generated from M, represented by $M[t>M'$, where:

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

Based on the traditional definition PN, the LPN is put forward.

Definition 4: Let $LN = (P, T, F, I, O)$. $LPN = (LN, M)$ is a logic Petri net where:

- P is a finite set of places
- $T = T_D \cup T_I \cup T_O$ is a finite set of transitions, $P \cup T \neq \varnothing$, $P \cap T = \varnothing$, $\forall t \in T_I \cup T_O$: ${}^{\bullet}t \cap t^{\bullet} = \varnothing$, where:
  - T_D denotes a set of traditional transitions

  - T_I denotes a set of logic input transitions, where $\forall t \in T_I$, the input places of t are restricted by a logic input expression $f_I(t)$ and
  - T_O denotes a set of logic output transitions, where $\forall t \in T_O$, the output places of t are restricted by a logic output expression $f_O(t)$

- $F \subseteq (P \times T) \cup (T \times P)$ is a finite set of directed arcs
- I is a mapping from a logic input transition to a logic input expression, i.e.,:

$$\forall t \in T_I,\ I(t) = f_I(t) = A_1 \vee A_2 \vee \ldots \vee A_u$$

- O is a mapping from a logic output transition to a logic input expression, i.e.,:

$$\forall t \in T_O,\ O(t) = f_O(t) = B_1 \vee B_2 \vee \ldots \vee B_v$$

- M: $P \to \{0,1\}$ is a marking function, where $\forall p \in P$, M(p) is the number of tokens in p
- Transition firing rules:

  - $\forall t \in T_D$, the firing rules of t are the same as in PNs
  - $\forall t \in T_I$, t is enabled only if $\exists A_i$, make $f_I(t)|_M = .T.$, M[t>M', where $\forall p \in {}^{\bullet}t$ and $p \in A_i$, M'(p) = M(p)-1; $\forall p \in t^{\bullet}$, M'(p) = M(p)+1; $\forall p \notin {}^{\bullet}t \cup t^{\bullet}$, M'(p) = M(p);and $\forall p \in {}^{\bullet}t$ and $p \notin A_i$, M'(p) = M(p) and
  - $\forall t \in T_O$, t is enabled only if $\forall p \in {}^{\bullet}t$: M(p) = 1. M[t>M', where $\forall p \in {}^{\bullet}t$ M'(p) = M(p)-1; $\forall p \notin {}^{\bullet}t \cup t^{\bullet}$: M'(p) = M(p); $\forall p \in t^{\bullet}$ and $\forall p \in B_i$ should satisfy $f_O(t)|_{M'} = .T.$ and $\forall p \in t^{\bullet}$ and $p \notin B_i$, M'(p) = M(p)

LPNs are the abstract and extension of inhibitor arcs PNs and high-level PNs. A logic input/output transition is restricted by the logic input/output expression $f_I(t)/f_O(t)$. All logic input/output transitions are called logic transitions. The logic expressions can describe the indeterminacy of values in input and output places. A_i and B_i represent input and output ways of logic transitions, respectively. They are not the disjunctive normal of $f_I(t)/f_O(t)$, but the elements of A_i/B_i are connected by the logic symbol "$\wedge$".

Figure 1 shows a simple LPN model. t_2 is a traditional transition and t_1 and t_3 are logic transitions restricted by the f_I/f_O, respectively, where $f_I = (p_1 \wedge p_2) \vee (p_1 \wedge p_2 \wedge p_3)$ and $f_O = (p_7 \wedge p_8) \vee (p_7 \wedge p_8 \wedge p_9)$. Each place of a logical expression has a logic value at marking M in an LPN and by substituting the values of all places into the logic expression, the expression corresponds to a logical value. In the LPN model of Fig. 1, $M=(1, 1, 0, 1, 0, 0, 0, 0, 0)^T$, $I = \{p_1, p_2, p_3\}$, having $p_1|_M = .T.$, $p_2|_M = .T.$, $p_3|_M = .F.$ and $f_I|_M = (.T. \wedge .T.) \vee (.T. \wedge .T. \wedge .F.) = .T. \vee .F. = .T.$.

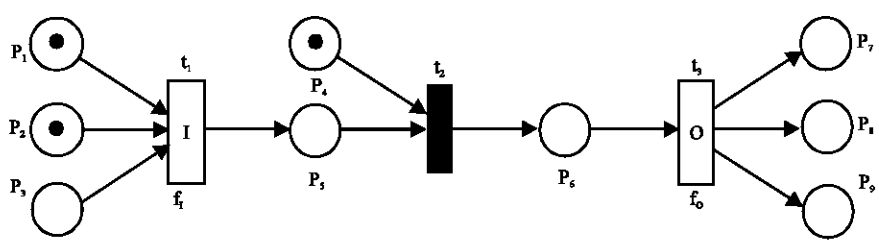

Fig. 1: An LPN model

FIRING RULES OF LPN TRANSITIONS

From Definition 4, a logic transition is restricted by the logic expression f(t), next, the standard form of f(t) is putword.

Definition 5: Suppose that a logic input/output transition t is restricted by $f_I(t)/f_O(t)$ and the standard form is as follow:

- For a logic input transition t, the standard form of $f_I(t) = A_1 \vee A_2 \vee \ldots \vee A_m$ can be obtained by:

$$A_i = \begin{cases} A_i, \text{ if } |A_i| = |{}^\bullet t| \\ A_i \wedge \neg p, \text{ if } |A_i| \neq |{}^\bullet t| \text{ and } \exists p \in {}^\bullet t \end{cases} \quad (1)$$

For a logic input transition t, the standard form of $f_O(t) = B_1 \vee B_2 \vee \ldots \vee B_n$ can be obtained by:

$$B_i = \begin{cases} B_i, \text{ if } |B_i| = |t^\bullet| \\ B_i \wedge \neg p, \text{ if } |B_i| \neq |t^\bullet| \text{ and } \exists p \in t^\bullet \end{cases} \quad (2)$$

A_i and B_i are called the standard minterms.

Example 1: In Fig. 1, transitions t_1 and t_2 are restricted by f_I/f_O, respectively, where $f_I = (p_1 \wedge p_2) \vee (p_1 \wedge p_2 \wedge p_3)$ and $f_O = (p_7 \wedge p_8) \vee (p_7 \wedge p_8 \wedge p_9)$. From Definition 5, the standard form of f_I is $f_I = (p_1 \wedge p_2 \wedge \neg p_3) \vee (p_1 \wedge p_2 \wedge p_3)$ and the standard form of f_O is $f_O = (p_7 \wedge p_8 \wedge \neg p_9) \vee (p_7 \wedge p_8 \wedge p_9)$.

A standard form of an expression consists of several standard minterms and each minterm can be represented by a vector. A standard minterm represented by a vector in this study and a standard form of an expression is described by a vector set. A logic input/output transition corresponds to an enabling vector set and a traditional transition corresponds a vector. Next, the enabling vectors are putford.

Definition 6: Let LPN = (LN, M) be a logic Petri net, where $P = \{p_1, p_2, \ldots, p_m\}$ and $T = \{t_1, t_2, \ldots, t_n\}$:

- For $t \in T_I$, the standard form of $f_I(t)$ has a minterm A_i and A_i is a logic expression made by ${}^\bullet t$. $V_i = (v_{i1}, v_{i2}, \ldots, v_{im})^T$ is called the enabling vector of t, where:

$$v_{ij} = \begin{cases} 0, \text{if } p_j \in t^\bullet \\ 1, p_j \in {}^\bullet t \text{ and } p_j|_{A_i} = .T. \\ *, \text{else} \end{cases} \quad (3)$$

- For $t \in T_O$, the standard form of $f_O(t)$ has a minterm B_i and B_i is a logic expression made by $t^\bullet$. $V_i = (v_{i1}, v_{i2}, \ldots, v_{im})^T$ is called the enabling vector of t, where:

$$v_{ij} = \begin{cases} 0, p_j \in t^\bullet \text{ and } p_j|_{B_i} = .T. \\ 1, p_j \in {}^\bullet t \\ *, \text{else} \end{cases} \quad (4)$$

For $t \in T_D$, $V_i = (v_{i1}, v_{i2}, \ldots, v_{im})^T$ is called the enabling vector of t, where:

$$v_{ij} = \begin{cases} 0, p_j \in t^{\bullet} \\ 1, p_j \in {}^{\bullet}t \\ *, \text{else} \end{cases} \tag{5}$$

In Eq. 3, 4 and 5, j belongs to the set {1, 2,..., m}. Suppose that the standard form of $f_I(t)$ consists of u minterms, from Eq. 3, the corresponding vector sets are $V_{ii,1}$, $V_{ii,2}, \ldots$, $V_{Ii,u-1}$ and $V_{Ii,u}$ and $V_{ii} = \{V_{Ii,1}, V_{Ii,2}, \ldots, V_{IiB,u}\}$ is the logic input enabling vector set of t. V_I is the logic input enabling set of the LPN and has $V_{Ii} \in V_I$. Likewise, suppose that the standard normal form of $f_O(t)$ consists of v minterms, from Eq. 4, the corresponding minterm vector sets are $V_{oi,1}$, $V_{oi,2}, \ldots$, $V_{oi,v}$ and $V_{Oi} = \{V_{oi,1B}, V_{Oi,2}, \ldots, V_{Oi,v}\}$ is the logic input enabling vector set of t. V_O is the logic output enabling set of the LPN and has $V_{Oi} \in V_O$. The common enabling set V is composed by all enabling vectors of the traditional transitions, i.e., $V_i \in V$.

The matching rule among vectors is proposed in the next.

Definition 7: Suppose that $m \in \mathbb{N}^+$, $X = (x_1, x_2, \ldots, x_m)^T$ is a 0-1vector and $Y = (y_1, y_2, \ldots, y_m)^T$, where $\forall i \in \mathbb{N}_m$ and $\mathbb{N}_m = \{1, 2, \ldots, m\}$, $y_i \in \{0, 1, *\}$. X matches Y donated by $Y \cong X$, if:

$$x_i = \begin{cases} 0, \text{if} & y_i = 0 \\ 1, \text{if} & y_i = 1 \\ 0/1, \text{if} & y_i = * \end{cases} \tag{6}$$

else $Y \neq X$.

Example 2: Let $X = (1, 0, 1, 0, 1)^T$, $Y = (1, 0, 1, *, *)^T$ and $Z = (1, 0, 0, *, *)^T$. By Eq. 4, $X \cong Y$ and $X \neq Z$.

Theorem 1: Let LPN = (LN, M) be a logic Petri net, where $P = \{p_1, p_2, \ldots, p_m\}$ and $T = \{t_1, t_2, \ldots, t_n\}$ and M_0 is its inital marking, $M \in R(M_0)$. t is enabled at M:

- For $t \in T_I$, suppose that $V_{ii} = \{V_{Ii,1}, V_{Ii,2}, \ldots, V_{Ii,u}\}$ is its logic input enabling vector set and $V_{Ii} \in V_I$. If $\exists x \in \{1, 2, \ldots, u\}$, $V_{Ii,x} \in V_{Ii}$ and $V_{Ii,x} \cong M$
- For $t \in T_O$, suppose that $V_{oi} = \{V_{Oi,1}, V_{Oi,2}, \ldots, V_{oi,v}\}$ is its logic output enabling vector set and $V_{Oi} \in V_O$. If $\exists x \in \{1, 2, \ldots, v\}$, $V_{Oi,x} \in V_{Oi}$ and $V_{Oi,x} \cong M$; or
- For $t \in T_D$, suppose V_i is its common enabling vector and $V_i \cong M$

Proof: Suppose that t is enabled at M, i.e. M[t>:

- For $t \in T_I$, $\exists V_{Ii,k} \in V_{Ii}$ and $V_{ii,k} = (v_{i1}, v_{i2}, \ldots, v_{im})^T$ is defined by Eq. 3

If $(t, p_j) \in F$, then $v_{ij} = 0$; if $(p_j, t) \in F$ and $p_j|_{Ai} = .T.$, then $v_{ij} = 1$; the left v_{ij} are marked "*".

If $(t, p_j) \in F$, then $M(j) = 0$; if $(p_j, t) \in F$ and $p_j|_{Ai} = .T.$, then $M(j) = 1$; other elements of M are marked "1/0".

Therefore, if $(p_j, t) \in F$, $M(j) = v_{ij}$; if $(t, p_j) \in F$ and $p_j|_{Ai} = .T.$, $M(j) = v_i$; the left v_{ij} are marked "*" and other elements of M are marked "1/0". According to Eq. 6, $V_{Ii,x} \cong M$:

- For $t \in T_O$, $\exists V_{Oi,k} \in V_{Oi}$ and $V_{oi,k} = (v_{i1}, v_{i2}, \ldots, v_{im})^T$ is defined by Eq. 4

If $(p_j, t) \in F$, then $v_{ij} = 1$; if $(t, p_j) \in F$ and $p_j|_{Bi} = .T.$, then $v_{ij} = 0$; the left v_{ij} are marked "*".

If $(p_j, t) \in F$, then $M(j) = 1$. If $(t, p_j) \in F$ and $p_j|_{Bi} = .T.$, then $M(j) = 0$. other elements of M are marked "1/0".

Therefore, if $(p_j, t) \in F$, $M(j) = v_{ij}$; if $(t, p_j) \in F$ and $p_j|_{Bi} = .T.$, $M(j) = v_{ij}$; the left v_{ij} are marked "*" and other elements of M are marked "1/0". According to Eq. 6, $V_{Oi,x} \cong M$:

- For $t \in T_D$, $V_i = (v_{i1}, v_{i2}, \ldots, v_{im})^T$ is defined by Eq. 5

If $(p_j, t) \in F$, then $v_{ij} = 1$; if $(t, p_j) \in F$, then $v_{ij} = 0$; if $p_j \notin {}^\bullet t \cap t^\bullet$, $v_{ij} = *$

If $(p_j, t) \in F$, then $M(j) = 1$; if $(t, p_j) \in F$, then $M(j) = 0$; if $p_j \notin {}^\bullet t \cap t^\bullet$, $M(j) = 1$ or $M(j) = 0$.

Therefore, if $(p_j, t) \in F$, then $M(j) = v_{ij}$; if $(t, p_j) \in F$, then $M(j) = v_{ij}$ and if $p_j \notin {}^\bullet t \cap t^\bullet$, $v_{ij} = *$ and $M(j) = 1$ or $M(j) = 0$. According to Eq. 6, $V_i \cong M$.

For logic transitions, Theorem 1 is the necessary condition of a transition to fire, not the necessary and sufficient condition. For all traditional transitons, Theorem is the necessary and sufficient condition. The firing of logic transitions are restricted by the enabling vectors and logic expressions. Next, the matching expression is put forward.

Definition 8: Suppose that $f_1 = A_1 \vee A_2 \vee \ldots \vee A_u$ and $f_2 = B_1 \vee B_2 \vee \ldots \vee B_u$ are standard form. f_1 and f_2 are called matching expressions, if:

- The number of standard minterm between is the same, i.e., u = v
- The standard minterm A_i of f_1 corresponds to the standard minterm B_i of f_2 and the elements of A_i corresponds to the elements of B_i, i.e., for $\forall p_m$, $p_n \in A_i$, their corresponding places p_k, $p_l \in B_i$, having k-m = l-n

Two matching expressions are causal relationship. Once a standard minterm A_i of f_1 fired, the standard minterm B_i of its matching expression could fire directionally in the running of the LPN.

Example 3: In Fig. 1, $f_I = (p_1 \wedge p_2) \vee (p_1 \wedge p_2 \wedge p_3)$ and $f_O = (p_7 \wedge p_8) \vee (p_7 \wedge p_8 \wedge p_9)$ are two matching expressions. $f_I = (p_1 \wedge p_2 \wedge \neg p_3) \vee (p_1 \wedge p_2 \wedge p_3)$ and $f_O = (p_7 \wedge p_8 \wedge p_9) \vee (p_7 \wedge p_8 \wedge \neg p_9)$ are their standard forms and they have satisfied the Definiton 8. They will be used in the section IV.

Next, the equation vector used to calculate the marking M is put forward.

Definition 9: Let LPN = (LN, M) be a logic Petri net, where $P = \{p_1, p_2, \ldots, p_m\}$ and $T = \{t_1, t_2, \ldots, t_n\}$, For $t \in T_I$, the standard form of $f_I(t)$ has a minterm A_i. $V' = (v_{i1}', v_{i2}', \ldots, v_{im}')^T$ is an equation vector of t, where:

$$v_{ij}' = \begin{cases} -1, & p_j \in {}^\bullet t \text{ and } p_j|_{A_i} = .T. \\ 1, & p_j \in t^\bullet \\ 0, & \text{else} \end{cases} \quad (7)$$

- For $t \in T_O$, the standard form of $f_O(t)$ has a minterm B_i. $V' = (v_{i1}', v_{i2}', \ldots, v_{im}')^T$ is an equation vector of t, where:

$$v_{ij}' = \begin{cases} -1, & p_j \in {}^{\bullet}t \\ 1, & p_j|_{B_i} = {}_{\bullet}T_{\bullet} \\ 0, & \text{else} \end{cases} \tag{8}$$

- For $t \in T_D$, $V_i' = (v_{i1}', v_{i2}', \ldots, v_{im}')^T$ is a equation vector of t, where:

$$v_{ij}' = \begin{cases} -1, & p_j \in {}^{\bullet}t \\ 1, & p_j \in t^{\bullet} \\ 0, & \text{else} \end{cases} \tag{9}$$

In Eq. 7-8 and 9, j belongs to the set {1, 2,..., m}. From Definitions 6 and 8, enabling vectors and equation vectors are obtained by the same expression and their number is the same. Each enabling vector corresponds to an equation vector. So $V_{ii,1}'$, $V_{Ii,2}', \ldots, V_{Ii,u-1}'$ and $V_{Ii,u}'$ are equation vectors of t belonging to T_I and $V_{ii}' = \{V_{Ii,1}', V_{Ii,2}', \ldots, V_{IiB,u}'\}$ is the input equation vector set, $V_{ii}' \in V_I'$; likewise, $V_{oi,1}'$, $V_{oi,2}', \ldots, V_{oi,v}'$ are the equation vectors of t belonging to T_O and $V_{oi}' = \{V_{Oi,1}', V_{Oi,2}', \ldots, V_{Oi,v}'\}$ is the output equation vector set, $V_{Oi}' \in V_O'$ and $V_i' \in V$ is the common equation of t.

Based on Definitions 6 and 9, a therom and an algorithm are introduced in the following.

Definition 10: Let LPN = (LN, M) be a logic Petri net, where $P = \{p_1, p_2, \ldots, p_m\}$ and $T = \{t_1, t_2, \ldots, t_n\}$ and M_0 is the initial marking of the LPN, $M \in R(M_0)$:

- For $t \in T_I$, $V_{ii}' = \{V_{Ii,1}', V_{Ii,2}', \ldots, V_{Ii,u}'\}$ is the input equation vector set of t and $V_{ii} = \{V_{Ii,1}, V_{Ii,2}, \ldots, V_{Ii,v}\}$ is its input enabling vector set. If $V_{Ii,x} \cong M$, then M[t >M' and:

$$M' = M + V_{Ii,x}' \tag{10}$$

- For $t \in T_O$, $V_{oi}' = \{V_{Oi,1}', V_{Oi,2}', \ldots, V_{Oi,v}'\}$ is its output equation vector set and $V_{oi} = \{V_{Oi,1}, V_{Oi,2}, \ldots, V_{Oi,v}\}$ is its output enabling vector set. If $V_{Oi,y} \cong M$, then M[t >M':

$$M' = M + V_{Oi,y}' \tag{11}$$

- For $t \in T_D$, V_i' is the common equation vector of t and V_i is its enabling vector. If $V_i \cong M$, then M[t >M':

$$M' = M + V_i' \tag{12}$$

Equation 10, 11 and 12 are used to calculate the marking M after a transition has fired. Based on the definitions and Theorem 1, an algorithm is proposed:

Algorithm 1: The firing of transitions in LPNs

Input: $\sum = (LPN, V_I, V_O, V)$

Output: The transitions fired at the marking M

Step 1: For $\forall V_{Ii,k} \in V_{Ii}$, $V_{Ii} \in V_I$, if $\exists V_{Ii,k}$, have $V_{Ii,k} \cong M$,

1.1 If $V_{Ii,k}$ does have corresponding minterms, then $t \in T_I$, by Eq. 10, get M′ and M[t>M′;

1.2 If $V_{Ii,k}$ has corresponding minterms and there exits a matching minterm, then then $t \in T_I$, by Eq. 10, get M′ and M[t>M′;

Step 2: For $\forall V_{Oi,k} \in V_{Oi}$, $V_{Oi} \in V_O$, if $\exists V_{Oi,k}$, have $V_{Oi,k} \cong M$,

2.1 If $V_{Oi,k}$ does have the corresponding minterm, then $t \in T_O$, by Eq. 11, get M′ and M[t>M′;

2.2 If $V_{Ii,k}$ has corresponding minterms, then $t \in T_O$, by Eq. 11, get M′ and M[t>M′;

Step 3: For $\forall V_i \in V$, if $\exists V_i$, have $V_i \cong M$, then by Eq. 12, get M′ and M[t>M′.

AN EXAMPLE

In the LPN model of Fig. 1, LPN = (P, T, F, M), where $t_1 \in T_I$, $t_2 \in T_D$, $t_3 \in T_O$, $M = (1, 1, 0, 1, 0, 0, 0, 0, 0)^T$ and t_1 and t_3 are restricted by $f_I = (p_1 \wedge p_2) \vee (p_1 \wedge p_2 \wedge p_3)$ and $f_O = (p_7 \wedge p_8) \vee (p_7 \wedge p_8 \wedge p_9)$, respectively. f_I and f_O are matching expressions.

From Example 1, get the standard form $f_I = (p_1 \wedge p_2 \wedge \neg p_3) \vee (p_1 \wedge p_2 \wedge p_3)$, $f_O = (p_7 \wedge p_8 \wedge \neg p_9) \vee (p_7 \wedge p_8 \wedge p_9)$.

By Definition 6, the input, output and common enabling vector sets can be obtained, where:

- $V_{I1} = \{V_{I1,1}, V_{I1,2}\}$ is the input enabling vector set of t_1, where $V_{I1,1} = (1, 1, *, *, 0, *, *, *, *)^T$, $V_{I1,2} = (1, 1, 1, *, 0, *, *, *, *)^T$
- V_2 is the common enabling vector of t_2 and $V_2 = (*, *, *, 1, 1, 0, *, *, *)^T$
- $V_{O3} = \{V_{O3,1}, V_{O3,2}\}$ is the output enabling vector set of t_3, where $V_{O3,1} = (*, *, *, *, *, 1, 0, 0, *)^T$, $V_{O3,2} = (*, *, *, *, *, 1, 0, 0, 0)^T$

By Definition 9, the equation vector set can be obtained, where:

- $V_{I1}' = \{V_{I1,1}', V_{I1,2}'\}$ is the input equation vector set of t_1, where $V_{I1,1}' = (-1, -1, 0, 0, 1, 0, 0, 0, 0)^T$, $V_{I1,2}' = (-1, -1, -1, 0, 1, 0, 0, 0, 0)^T$
- V_2' is the common equation vector of t_2 and $V_2' = (0, 0, 0, -1, -1, 1, 0, 0, 0)^T$
- $V_{O3}' = \{V_{O3,1}', V_{O3,2}'\}$ is the output equation vector set of t_3, where $V_{O3,1}' = (0, 0, 0, 0, 0, -1, 1, 1, 0)^T$, $V_{O3,2}' = (0, 0, 0, 0, 0, -1, 1, 1, 1)^T$

According to Algorithm 1:

- Let M match $V_{I1,1}$, $V_{I1,2}$, by Definition 7, have $V_{I1,1} \cong M$; by Eq. 10, $M_1 = M + V_{I1,1}' = (0, 0, 0, 1, 1, 0, 0, 0, 0)^T$, $M[t_1 > M_1$
- By Definition 7, have $V_2 \cong M_1$; by Eq. 12, $M_2 = M_1 + V_2' = (0, 0, 0, 0, 0, 1, 0, 0, 0)^T$, $M_1[t_2 > M_2$
- Let M_2 match $V_{O3,1}$, $V_{O3,2}$, by Definition 7, have $V_{O3,1} \cong M$ and $V_{O3,2} \cong M$. f_I and f_O are matching expressions and in (a), $V_{I1,1} \cong M$, $V_{O3,1}$ is the matching vector. By Eq. 11, $M_3 = M_2 + V_{O3,1}' = (0, 0, 0, 0, 0, 0, 1, 1, 0)^T$, $M_2[t_3 > M_3$

At the marking M_3, all enabling vectors can not match M_3 and the running of the LPN stops.

CONCLUSION

Based on our work, the definition of LPNs is redefined. By introducing the enabling vectors and matching expressions, the indeterminate data transmission caused by logic output transitions is solved. The standard form of a logic expression can delete useless minterms and the analysis of LPN models is more concisionly.

Further work will investigate the fundamental properties of LPNs according to the results proposed in this study, such as state equivalency, liveness and reachability. The Logic Petri Net Workflow will be put forward and be used in progress mining.

ACKNOWLEDGMENTS

This study is supported by the National Natural Science Foundation of China under Grants No. 61170078 and 61173042; the National Basic Research Program of China under Grant No. 2010CB328101; the Doctoral Program of Higher Education of the Specialized Research Fund of China under Grant No. 20113718110004; Basic Research Program of Qingdao City of China under Grant No. 13-1-4-116-jch; the SDUST Research Fund of China under Grant No. 2011KYTD102 and Graduate Innovation Foundation of Shaudong University of Science and Technology under Grant No. YC130107.

REFERENCES

Aziz, M.H., E.L.J. Bohez, R. Pisuchpen and M. Parnichkun, 2013. Petri Net model of repetitive push manufacturing with Polca to minimise value-added WIP. Int. J. Prod. Res., 51: 4464-4483.

Barzegar, B., M. Mehrabanian, S. Bandegan and S. Bandegan, 2011. Fuzzy logic for a traffic signal control with colored petri net. Aus. J. Basic. Applied Sci., 5: 2961-2964.

Bayati, M. and A. Dideban, 2012. Controller synthesis using the novel fuzzy petri net. Int. Rev. Autom. Control, 5: 839-843.

Du, Y.Y. and C.J. Jiang, 2002. Formal representation and analysis of batch stock trading systems by logical petri net workflows. Lect. Notes Comput. Sci., 2495: 221-225.

Du, Y.Y. and C.J. Jiang, 2008. On the design and temporal Petri net verification of grid commerce architecture. Chinese J. Electron., 17: 247-251.

Du, Y.Y., C.J. Jiang and M. Zhou, 2009. A petri net-based model for verification of obligations and accountability in cooperative systems. IEEE Trans. Syst. Man Cybern. A, 39: 299-308.

Marin, A., S. Balsamo and P.G. Harrison, 2012. Analysis of stochastic Petri nets with signals. Perform. Eval., 69: 551-572.