基于同步合成的结构复杂 Petri 网的行为描述*

曾庆田+

(中国科学院 计算技术研究所,北京 100080)
(山东科技大学 计算机科学与技术系,山东 泰安 271019)
(中国科学院 研究生院,北京 100039)

Behavior Descriptions of Structure-Complex Petri Nets Based on Synchronous Composition

ZENG Qing-Tian+

(Institute of Computing Technology, The Chinese Academy of Sciences, Beijing 100080, China)
(Department of Computer Science, Shandong University of Science and Technology, Taian 271019, China)
(Graduate School, The Chinese Academy of Sciences, Beijing 100039, China)
+ Corresponding author: Phn: +86-10-62565533 ext 5668, E-mail: zqtian@ict.ac.cn, zqtian@163.com, http://www.ict.ac.cn
Received 2003-03-12; Accepted 2003-07-14

Zeng QT. Behavior descriptions of structure-complex Petri nets based on synchronous composition. *Journal of Software*, 2004,15(3):327~337.
http://www.jos.org.cn/1000-9825/15/327.htm

Abstract: In order to specify the behaviors of structure-complex Petri nets, the concept of synchronous composition is extended and a method is presented, with which a given structure-complex Petri net can be obtained through the synchronous composition of a set of structure-simple Petri nets, namely S-nets. Firstly, the language characters of S-nets are analyzed with details and the methods to obtain their language expressions are presented. With the synchronous intersection operation of Petri net languages, the language relationships between the structure-complex Petri net and the set of S-nets can be expressed. Based on these works, an algorithm to specify the behaviors of Petri nets especially structure-complex systems is obtained.
Key words: Petri net; S-net; Petri net language; synchronous composition; synchronous intersection

摘 要: 首先分析了一类结构简单的 Petri 网——S-网的语言性质,得到了它们的行为描述方法.拓展了 Petri 网同步合成的概念,证明了给定一个结构复杂的 Petri 网都可通过一组 S-网的同步合成运算而得到,并给出了相应的求解算法.引入语言的同步交运算,分析了结构复杂的 Petri 网与其同步合成子网之间的行为关系,给出了结构复杂 Petri 网的行为描述算法,为利用网语言分析实际系统的行为特征提供了可靠的理论依据和方法.
关键词: Petri 网;S-网;Petri 网语言;同步合成;同步交

* Supported by the National Natural Science Foundation of China under Grant Nos.60173053, 60274063 (国家自然科学基金); the Excellent Young Scientist Foundation of Shandong Province of China under Grant No.02BS069 (山东省优秀青年科学家奖励基金)

ZENG Qing-Tian was born in 1976. He is a Ph.D. candidate at the Institute of Computing Technology, the Chinese Academy of Sciences. His research interests include large-scale knowledge processing and ontology analysis, V&V of knowledge-based system, Petri net theory and applications, and mathematical knowledge acquisition and analysis.

中图法分类号: TP301 文献标识码: A

1 Introduction

As models of modeling and analyzing physical systems, Petri nets have shown their abilities to deal with concurrency and conflict. Petri net language, the method to specify the behaviors of a net system itself, has attracted more attention since the late seventies[1~8], because it is powerful for describing the dynamical behaviors of the net system among the analysis methods in net theory. Four different types of Petri net languages, named *L*-type, *G*-type, *T*-type and *P*-type, are defined based on the difference of the set of reachable states[2]. Hack[2] analyzes the computing ability and proves that the computing ability of Petri nets extended with prohibit arcs is equal to Turing machine. Garg[4] introduces the concept of concurrent regular expression and proves that the computing ability of this kind of expression is equal to Petri net. Wu[5] presents the relationships between Petri net language and the formal language. A set of necessary and sufficient conditions for determining a Petri net language to be a regular language, context-free language or a context-sensitive language is given[5]. The determining conditions are obtained by transforming the Pumping Lemma in formal language theory into the structured properties of the standard Petri net corresponding to a given Petri net. Recently, Jiang and Lu[6] analyze the properties of concurrent systems based on Petri net languages. Two language characterizations for weak liveness (free-deadlock) and liveness of Petri nets are given[6]. In order to analyze the behaviors during the decomposition and composition of Petri nets, Zeng and Wu introduce the concept of synchronous intersection operation of Petri net languages[7].

Given a Petri net model of a physical system, if its language expression can be obtained easily, it is convenient to model and analyze the properties of the physical system[12~15]. However, it is not easy to obtain the language expression of a given Petri net, especially a structure-complex net. Traditionally, in order to overcome this difficulty, some solutions including decomposition, reduction, composition and net operation are introduced by many researchers[9~12]. Kwang[9] gives several generalized reduction methods of Petri nets and Suzuki[10] presents a method for stepwise refinement and abstraction of Petri net. Recently, Zeng and Wu give a decomposition method of structure-complex Petri net based on the index of places[11], and obtain a method to analyze the behaviors of structure-complex Petri nets[12].

Another method for property analysis of structure-complex Petri nets is the composition of structure-simple Petri nets. Jiang[13,14] gives the concept of synchronous composition of Petri nets and analyzes the sate and behavior properties during the process of composition, and some conditions to keep states and behaviors invariant are obtained[13~15]. The operation of synchronous composition of Petri nets plays an important role in property analysis of systems. Jiang[13,14] has presented many theoretical results about the synchronous composition of Petri nets. Some conditions to keep the behaviors invariant completely are obtained in Refs.[13,14], and the process characters are analyzed in Ref.[15] for the synchronous composition of Petri nets. All the obtained results about synchronous composition of Petri nets are only between two nets. In the real physical system modeling and analysis with Petri net, it is necessary to obtain a structure-complex Petri net by the synchronous composition of more than two nets. In the previous work[13~15], how to find the Petri nets to compose a given Petri net from synchronous composition is not given.

In this paper, the concept of synchronous composition is extended to more than two Petri nets and it is proved that any structure-complex Petri net can be composed from the synchronous composition of a set of structure-simple Petri nets such that $|{}^{\bullet}t|\leq 1$ and $|t^{\bullet}|\leq 1$. The language characters of these structure-simple nets are analyzed with details, by which the method to specify their languages is presented. The relationships of the languages during the synchronous composition are expressed with the synchronous intersection operation of languages. Finally, an algorithm to describe the behavior of Petri net, especially a structure-complex net, is obtained, which is a benefit for

the modeling and analysis of physical systems based on Petri net languages.

This paper is organized as follows. Section 2 presents the basic notations of the Petri net languages. Section 3 analyzes the languages of a set of structure-simple Petri nets in which $|{}^\bullet t|\le 1$ and $|t^\bullet|\le 1$ hold for all transitions. Section 4 extends the concept of synchronous composition and introduces the concept of synchronous intersection operation of languages. Then, how to find the S-nets to compose a structure-complex Petri net with synchronous composition, and the language relationships during the process of composition are given in Section 5. Based on the relationships, an algorithm to obtain the language expression of a Petri net is represented in this section. Section 6 gives an example and Section 7 concludes the whole paper.

2 Basic Concepts and Notations

It is assumed that readers are familiar with the basic concepts of Petri net[1,2]. Some of the essential terminology and notations about the Petri net languages used in this paper are presented as follows.

In the theory of Petri net languages, four different types of languages named, *L*-type, *G*-type, *T*-type and *P*-type, are defined based on the difference of the set of reachable states[2]. Depending on the choice of transition labeling (free, λ-free, arbitrary), each type is divided into three classes. We use free *L*-type language as an example in the following discussions and the reachable states set are defined as:

$Q_T \subseteq R(M_0) \wedge (\forall M_e \in Q_T, \forall p \in P - P_f : M_e(p) = 0)$, where M_0 is the initial marking, P is the place set of a given Petri net, and $P_f \subseteq P$ is an appointed place set.

Now we present the formal definition of the language.

Definition 2.1. Let $\Sigma = (P,T;F,M_0)$ be a Petri net, $P_f \subseteq P$, $L(\Sigma)$ is the language of Σ iff $L(\Sigma) = \{\sigma \mid \sigma \in T^* \wedge M_0[\sigma > M_e \wedge (\forall p \in P - P_f, M_e(p) = 0)\}$. P_f is namely the end place set of Σ.

In the following of this paper, a Petri net is denoted directly as $\Sigma = (P,T;F,M_0,P_f)$ where P_f is the end place set. In practice, the end place set can be decided by the sates of the real systems.

The common language operations include "$\bullet$" (connection) operation, "+" (choice) operation, "*" (Kleen-closure) operation, "||" (parallel) operation. Now we introduce the concepts of "||" and "α-closure" operation. See others in Ref.[10].

Definition 2.2[2]. The parallel operation (||) on the alphabet Ω is formally defined as follows.

(1) for all $a \in \Omega$, $a \| \varepsilon = \varepsilon \| a = \{a\}$, where ε is the empty word of Ω.

(2) for all $a,b \in \Omega$ and all $\sigma_1, \sigma_2 \in \Omega^*$, $a \bullet \sigma_1 \| b \bullet \sigma_2 = a \bullet (\sigma_1 \| b \bullet \sigma_2) \cup b \bullet (a \bullet \sigma_1 \| \sigma_2)$.

Let $L(\Sigma_i)$ be the language of Petri net $\Sigma_i = (P_i,T_i;F_i,M_{0i},P_{fi})$ ($i \in \{1,2\}$). The operation || to languages $L(\Sigma_1)$ and $L(\Sigma_2)$ is defined as $L(\Sigma_1) \| L(\Sigma_2) = \{\sigma_1 \| \sigma_2 \mid \sigma_1 \in L(\Sigma_1), \sigma_2 \in L(\Sigma_2)\}$.

Based on Definition 2.2, $L \| L = \{\sigma_1 \| \sigma_2 \mid \sigma_1, \sigma_2 \in L\}$, denoted as $L^{(2)} = L \| L$, and $L^{(n)}$ is defined as $L^{(n)} = L \| L^{(n-1)}$ ($n \ge 2$).

Definition 2.3[4]. The α-closure of language L is defined as $L^\alpha = \bigcup_{i=0,1,\ldots} L^{(i)}$, where $L^{(i)} = L \| L^{(i-1)}$ ($i \ge 2$).

Example. $L = \{a \bullet b \bullet c\}$, then

$L^\alpha = \{w \mid \forall s \in Pref(w), \#(a,s) \ge \#(b,s) \ge \#(c,s) \wedge \#(a,w) = \#(b,w) = \#(c,w)\}$, where $Pref(w)$ is the prefix of w and $\#(a,w)$ is the number of a occurring in w.

3 Behavior Descriptions of Structure-Simple Petri Nets

In this section, we analyze the languages of a set of structure-simple Petri nets named as S-net[1] at first, and the methods to specify the behaviors of this kind of Petri nets are presented.

Definition 3.1[1]. $N = (P,T;F)$ is called an S-net if $|{}^\bullet t| \le 1$ and $|t^\bullet| \le 1$ for any transition $t \in T$.

Definition 3.2. Let $N = (P,T;F)$ be a net. A transition $t \in T$ is called a primitive transition of N if

$|{}^{\bullet}t|=0$; a transition $t\in T$ is called a terminal transition of N if $|t^{\bullet}|=0$.

$\Sigma=(P,T;F,M_0,P_f)$ is a Petri net, and N is the underlying net of Σ. If a transition $t\in T$ is the primitive (terminal) transition of N, we also say that transition t is the primitive (terminal) transition of Σ.

Definition 3.3[1]. An S-net is said to be an S-graph, if there are no primitive transitions and no terminal transitions in this net.

We will pay more attention to S-net than to S-graph in this section, and S-net is regarded as structure-simple net. Next, we will analyze their language characters in order to obtain their language expressions.

Theorem 3.1[1]. $\Sigma=(P,T;F,M_0,P_f)$ is a net system, where $N=(P,T;F)$ is an S-graph. If $|P|=m$ and $q=\sum_{p\in P}M_0(p)$ ($q\le m$), $L(\Sigma)$ is a regular expression produced by a finite automaton with C_m^q states.

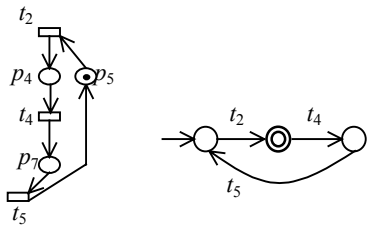

Fig.1 A Petri net Σ_1 and the automaton

Example 3.1. A Petri net Σ_1 is shown in Fig.1, where p_4 is the end place, and $L(\Sigma_1)$ can be produced by the automaton shown in the right of Fig.1. The state marks of the automaton are ignored. With Theorem 3.1, $L(\Sigma_1)=t_2(t_4t_5t_2)^*$.

Theorem 3.2. $\Sigma=(P,T;F,M_0,P_f)$ is a net system, where the underlying net of Σ is an S-net with some terminal transitions, then there exists another S-net Σ' without any terminal transitions holding $L(\Sigma)=L(\Sigma')$.

Proof. Construct another Petri net Σ' with adding a new place p_e to Σ, and add arcs from all the terminal transitions to p_e. It is not difficult to prove that the underlying net of Σ' is an S-graph, and $L(\Sigma)=L(\Sigma')$.

Corollary 3.1. $\Sigma=(P,T;F,M_0,P_f)$ is a net system, where the underlying net of Σ is an S-net without primitive transitions but with some terminal transitions. If $|P|=m$ and $q=\sum_{p\in P}M_0(p)$ ($q\le m$), $L(\Sigma)$ is a regular expression produced by a finite automaton with C_{m+1}^q states.

Theorem 3.2 indicates that an S-net Σ with some terminal transitions can be transformed into another S-net Σ' without terminal transitions, and $L(\Sigma)=L(\Sigma')$. In the following discussions, it is supposed that all the S-nets are without terminal transitions.

Theorem 3.3. $\Sigma=(P,T;F,M_0,P_f)$ is a net system, where the underlying net of Σ is an S-net holding

(1) there is one and only one primitive transition in it; and

(2) $\forall p\in P$, $M_0(p)=0$,

$L(\Sigma)$ is an α-closure of a regular expression.

Proof. Construct a finite automaton $FSM=(\Gamma,Q,\delta,q_0,Q_f)$, where

(1) $\Gamma=T$;

(2) $Q=P\cup\{p_s\}$;

(3) $\delta\leftarrow\varnothing$, then use the following steps to obtain δ.

If transition t is not a primitive transition of Σ and $(\exists p_1,p_2\in P)\wedge((p_1,t)\in F\wedge(t,p_2)\in F)$, then $\delta\leftarrow\delta\cup\{(p_1,t,p_2)\}$.

If transition t is a primitive transition of Σ and $(\exists p'\in P)\wedge((t,p')\in F)$, then $\delta\leftarrow\delta\cup\{(p_s,t,p')\}$;

(4) $q_0=p_s$;

(5) $Q_f=P_f\cup\{p_s\}$.

The language accepted by FSM is denoted as $L(FSM)$. Now we prove that $L(\Sigma)=L(FSM)^{\alpha}$.

First, we prove that $L(\Sigma)\subseteq L(FSM)^{\alpha}$.

For all $\sigma\in L(\Sigma)$, because Σ has no initial marks, there must be a primitive transition t fired with finite times. So, a finite number of tokens of Σ, supposing the number is n, move into the output places of t, and then

forward, and finally reach the set of the end places P_f. Let $\sigma_1,\sigma_2,...,\sigma_n$ trace token $1,2,...,n$. Thus, $\sigma=\sigma_1\|\sigma_2\|...\|\sigma_n$. It is obvious that each σ_j ($j\in\{1,2,...,n\}$) can be accepted by *FSM*, therefore, $\sigma_1\|\sigma_2\|...\|\sigma_n\in L(FSM)^\alpha$. So $\sigma\in L(FSM)^\alpha$.

Then, we prove that $L(FSM)^\alpha\subseteq L(\Sigma)$.

$\forall\sigma\in L(FSM)^\alpha$, then $\sigma=\sigma_1\|\sigma_2\|...\|\sigma_n$ and $\forall\sigma_j\in L(FSM)$ ($j\in\{1,2,...,n\}$), σ_j can be traced by firing one primitive transition t in Σ, and the tokens produced by transition t move to the output places and finally reach the set of the end places P_f, so $\sigma_j\in L(\Sigma)$. Therefore $\sigma=\sigma_1\|\sigma_2\|...\|\sigma_n$ traces the behaviors of firing primitive transition and the tokens produced move into the end place set with n times, so $\sigma\in L(\Sigma)$.

Based on the above proof, $L(\Sigma)=L(FSM)^\alpha$. It is also said that $L(\Sigma)$ can be described by an expression like A^α, where A is a regular expression accepted by the finite automaton *FSM*.

Example 3.2. The net system shown on the left of Fig.2 holds the conditions of Theorem 3.3, and the end place set $P_f=\{p_2\}$. Based on the proof of Theorem 3.3, construct an automation *FSM* shown on the right of Fig.2. It is easy to prove that $L(\Sigma_3)=L(FSM)^\alpha=(t_1t_2(t_3t_2)^*)^\alpha$.

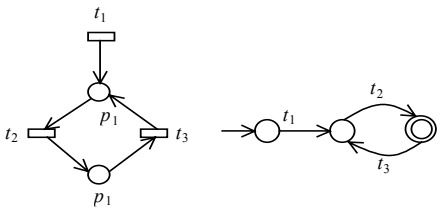

Fig.2 A Petri net Σ_2 and automaton FSM

Theorem 3.4. $\Sigma=(P,T;F,M_0,P_f)$ is a net system, where the underlying net of Σ is an S-net such that

(1) There is one and only one primitive transition in it; and

(2) $\exists p\in P$, $M_0(p)\neq 0$,

$L(\Sigma)=L_1^\alpha\|L_2$, where L_j ($j=1,2$) is a regular expression.

Proof. Based on $\Sigma=(P,T;F,M_0,P_f)$, we construct two nets Σ_1 and Σ_2, where $\Sigma_1=(P,T;F,M_{01},P_f)$ such that $\forall p\in P: M_{01}(p)=0$, and $\Sigma_2=(P\cup\{p_s\},T,F',M_{02},P_f\cup\{p_s\})$ such that $F'=F\cup\{(p_s,t)|t\in T\wedge{}^\bullet t=\varnothing\}$; and $\forall p\in P\cup\{p_s\}$, $M_{02}(p)=\begin{cases}M_0(p), & p\neq p_s\\ 0, & p=p_s\end{cases}$.

It is easy to know that $L(\Sigma)=L(\Sigma_1)\|L(\Sigma_2)$. With Theorem 3.1, $L(\Sigma_2)$ is a regular expression. With Theorem 3.3, $L(\Sigma_1)$ is an α-closure of a regular expression.

Example 3.3. A Petri net Σ_3, holding the conditions of Theorem 3.4, is shown on the left of Fig.3, where p_6 is the end place. With Theorem 3.4, Σ_3' and Σ_3' are constructed as shown on the right of Fig.3. It is obvious that $L(\Sigma_3)=L(\Sigma_3')\|L(\Sigma_3'')$. The place p_s in Σ_3' with wide edge is added newly, thus Σ_3' is an S-net. It is easily to obtain that $L(\Sigma_3')=t_1(t_5t_1)^*$. Σ_3' has a primitive transition t_3 but has no initial markings, so $L(\Sigma_3'')=(t_3+t_3(t_5t_1)^*)^\alpha$.

So, $L(\Sigma_3)=L(\Sigma_3')\|L(\Sigma_3'')=t_1(t_5t_1)^*\|(t_3+t_3(t_5t_1)^*)^\alpha$.

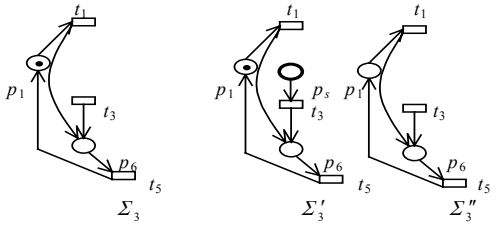

Fig.3 Petri nets Σ_3, Σ_3' and Σ_3''

Remarks. Theorems 3.3 and 3.4 discuss the behaviors of the net systems with one and only one primitive transition. Given an S-net system with more than one primitive transitions, we can transform it into some systems with one and only one primitive transition and obtain its language expression based on the transformations with the

following methods.

Let $\Sigma=(P,T;F,M_0,P_f)$ be a net system, in which $\exists t_1,t_2,...,t_k\in T$ are primitive transitions and $k\geq 2$,

Case (1). If $\forall p\in P$, $M_0(p)=0$, then $L(\Sigma)=L(\Sigma_1)\|L(\Sigma_2)\|...\|L(\Sigma_k)$, where $\Sigma_i=(P,T_i;F_i,M_0,P_f)$ such that $T_i=T-\{t_j\mid j\in\{1,2,...,k\}\wedge j\neq i\}$ and $F_i=F\cap((P\times T_i)\cup(T_i\times P))$ ($i\in\{1,2,...,k\}$).

Case (2). If $\exists p\in P$, $M_0(p)\neq 0$, then $L(\Sigma)=L(\Sigma_1)\|L(\Sigma_2)$, where

(1) $\Sigma_1=(P,T;F,M_{01},P_f)$ is a net system satisfying $\forall p\in P: M_{01}(p)=0$, and

(2) $\Sigma_2=(P\cup\{p_s\},T,F',M_{02},P_f\cup\{p_s\})$ such that $F'=F\cup\{(p_s,t)\mid t\in T\wedge {}^\bullet t=\varnothing\}$ and $\forall p\in P\cup\{p_s\}$ such that $M_{02}(p)=\begin{cases}M_0(p), & p\neq p_s\\ 0, & p=p_s\end{cases}$.

In Case (1), Σ is transformed into $\Sigma_1,\Sigma_2,...,\Sigma_k$, each of which has one and only one primitive transition and has no initial markings, so the language expression of Σ_i can be obtained with Theorem 3.3, so $L(\Sigma)$ can be expressed. In Case (2), Σ is transformed into Σ_1 holding Case (1), and Σ_2 being an S-graph, so $L(\Sigma)$ can also be expressed.

Example 3.4. The net system Σ_4, with $P_f=\{p_2\}$ as the end place set, shown on the left of Fig.4, holds Case (1). Σ_{41}, Σ_{42} and Σ_{43} shown on the right of Fig.4 are the systems constructed newly with the method of Case (1). It is obvious that $L(\Sigma_4)=L(\Sigma_{41})\|L(\Sigma_{42})\|L(\Sigma_{43})$ and $L(\Sigma_{4i})$ ($i=1,2,3$) can be obtained based on Theorem 3.3.

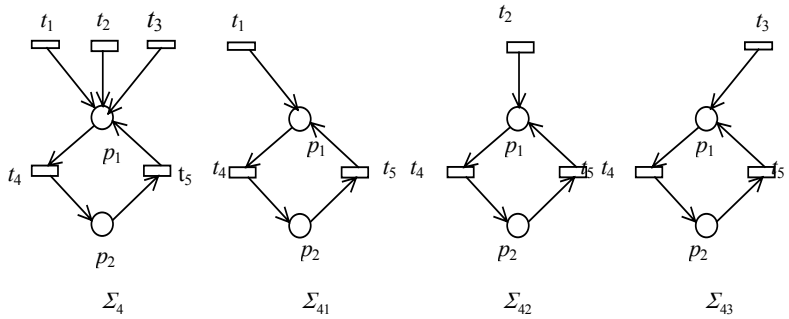

Fig.4 Petri net Σ_4, Σ_{41}, Σ_{42} and Σ_{43}

Example 3.5. The net system Σ_5, with $P_f=\{p_2\}$ as the end place set, shown on the left of Fig.5 holds Case (2). Σ_{51} and Σ_{52} shown on the right of Fig.5, are the systems constructed newly with the method of Case (2). It is obvious that $L(\Sigma_5)=L(\Sigma_{51})\|L(\Sigma_{52})$ and $L(\Sigma_{51})$ can be obtained with the method of Case (1). Σ_{52} is an marked S-graph, so its language can be obtained with Theorem 3.1.

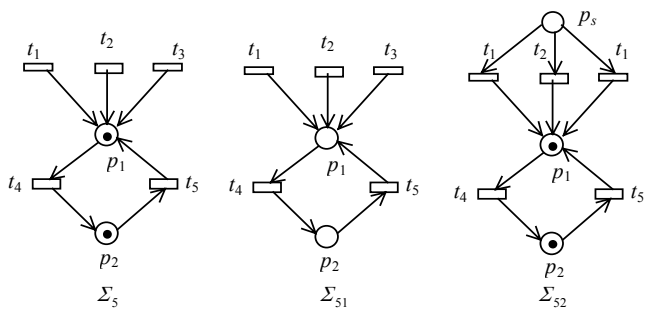

Fig.5 Petri nets Σ_5, Σ_{51} and Σ_{52}

4 Synchronous Composition of Petri Nets

Definition 4.1[13]. $\Sigma_i=(P_i,T_i;F_i,M_{0i})$ ($i\in\{1,2\}$) is a Petri net. If a Petri net $\Sigma=(P,T;F,M_0)$ holds the

follows

(1) $P=P_1\cup P_2, P_1\cap P_2=\varnothing$;

(2) $T=T_1\cup T_2, T_1\cap T_2=\varnothing$;

(3) $F=F_1\cup F_2$;

(4) $M_0(p)=\begin{cases}M_{01}(p), & p\in P_1\\ M_{02}(p), & p\in P_2\end{cases}$,

then Σ is the synchronous composition net system of Σ_1 and Σ_2, denoted as $\Sigma=\mathop{O}_{T}^{2}_{i=1}\Sigma_i$.

The synchronous composition operation of Petri nets plays an important role in the property analysis of real physical systems. Jiang[13~15] has analyzed the state invariant and behavior invariant in the process of synchronous composition. In Definition 4.1, the operation of synchronous composition is only between two Petri nets. When we analyze large-scaled net system, it is necessary to compose more than two net systems, we extend the concept of synchronous composition to the case of $k(k\geq 2)$ nets.

Definition 4.2. $\Sigma_i=(P_i,T_i;F_i,M_{0i},P_{fi})$ ($i=1,...,k$) is a Petri net. If a Petri net $\Sigma=(P,T;F,M_0,P_f)$ holds

(1) $P=\bigcup_{i=1}^{k}P_i$, and for all i,j such that $i\neq j$, then $P_i\cap P_j=\varnothing$;

(2) $T=\bigcup_{i=1}^{k}T_i$, and for each i there is at least one j such that $T_i\cap T_j=\varnothing$;

(3) $F=\bigcup_{i=1}^{k}F_i$;

(4) If $p\in P_i$, $M_0(p)=M_{0i}(p)$;

(5) $P_f=\bigcup_{i=1}^{k}P_{fi}$,

then Σ is the synchronous composition net of $\Sigma_1,\Sigma_2,...,\Sigma_k$, denoted as $\Sigma=\mathop{O}_{T}^{k}_{i=1}\Sigma_i$.

It is not difficult to prove that Definition 4.1 is the special case of Definition 4.2. Next, we mainly discuss the language relationships between Σ_i ($i=1,...,k$) and Σ. The structure properties and other behavior properties can be analyzed with the same method as Refs.[13,14].

In order to specify the behaviors of structure-complex Petri nets, the synchronous intersection operation of Petri net languages is introduced[7].

Definition 4.3[7]. Let X be a finite alphabet and $Y\subseteq X$. We define the projection $\Gamma_{X\to Y}: X^*\to Y^*$ from X to Y, for all $\sigma\in X^*$, $\Gamma_{X\to Y}\sigma$ is the sub-string of σ with deletion of all $a\in(X-Y)$ from σ.

Definition 4.4[7]. $L(\Sigma_i)$ is the language of Petri net $\Sigma_i=(P_i,T_i;F_i,M_{0i},P_{fi})$ ($i\in\{1,2\}$). The synchronous intersection language of $L(\Sigma_1)$ and $L(\Sigma_2)$ is defined as $L(\Sigma_1)[]L(\Sigma_2)=\{\sigma\in T^*\mid(\Gamma_{T\to T_i}\sigma)\in L(\Sigma_i), i=1,2\}$, where $T=T_1\cup T_2$.

We also denote $\mathop{[]}_{i=1}^{2}L(\Sigma_i)=L(\Sigma_1)[]L(\Sigma_2)$.

Similarly, the synchronous intersection language of $L(\Sigma_i)$ ($(k\geq 3)$ and $i\in\{1,2,...,k\}$ can be defined as $L(\Sigma)=(\mathop{[]}_{i=1}^{k-1}L(\Sigma_i))[]L(\Sigma_k)=\mathop{[]}_{i=1}^{k}L(\Sigma_i)$.

Theorem 4.1[7]. Let $L(\Sigma_i)$ be the language of $\Sigma_i=(P_i,T_i;F_i,M_{0i},P_{fi})$ ($i\in\{1,2\}$). If $T_1=T_2$, then $L(\Sigma_1)[]L(\Sigma_2)=L(\Sigma_1)\cap L(\Sigma_2)$.

Based on Theorem 4.1, the synchronous intersection operation of languages is different from the intersection operation of languages. The intersection operation is a special case of the synchronous intersection operation in case $T_1=T_2$.

Theorem 4.2. If $\Sigma=(P,T;F,M_0,P_f)$ is the synchronous composition net of Petri net, $\Sigma_i=(P_i,T_i;F_i,M_{0i},P_{fi})$ ($i\in\{1,2,...,k\}$), $L(\Sigma)=\prod_{i=1}^{k}L(\Sigma_i)$.

Proof. We prove the theorem by induction on $|\sigma|$.

(1) If $|\sigma|=1$, then $\sigma_i=\Gamma_{T\to T_i}(\sigma)=\begin{cases}\sigma, & \sigma\in T_i\\ \varepsilon, & \sigma\notin T_i\end{cases}$ ($i\in\{1,2,...,k\}$).

With $\sigma\in L(\Sigma)$ iff $M_0[\sigma>M_1\wedge M_1\in M_f$, and iff $\Gamma_{P\to P_i}(M_0)[\sigma_i>\Gamma_{P\to P_i}(M_1)\wedge\Gamma_{P\to P_i}(M_1)\in M_{fi}$, $\sigma_i\in L(\Sigma_i)$ ($i\in\{1,2,...,k\}$). Based on Definition 4.4, $L(\Sigma)=\prod_{i=1}^{k}L(\Sigma_i)$.

(2) Suppose that when $|\sigma|=n$ the assertion is correct, where x is an integer. Next, we prove that the assertion is also correct when $|\sigma|=n+1$.

Let $\sigma=\sigma'\bullet t'$, where $|\sigma'|=n$ and t' is the $(n+1)th$ element of σ.

With $\sigma\in L(\Sigma)$ iff $M_0[\sigma'\bullet t'>M_{n+1}\wedge M_{n+1}\in M_f$, let $M_0[\sigma'>M_n[t'>M_{n+1}$, then $M_n\in R(M_0)$; and $M'_f=\{M_n\mid M_0[\sigma'>M_n[t'>M_{n+1}\wedge M_{n+1}\in M_f\}$; $M'_{fi}=\{\Gamma_{P\to P_i}M_n\mid M_n\in M'_f\}$ ($i\in\{1,2,...,k\}$).

With the supposition, $\exists\sigma'_i=\Gamma_{T\to T_i}(\sigma')$ and $\sigma'_i\in L(\Sigma_i)$ such that $\sigma_i=\Gamma_{T\to T_i}(\sigma)=\begin{cases}\sigma'_i\bullet t', & t'\in T_i\\ \sigma'_i, & t'\notin T_i\end{cases}$ and $\sigma_i\in L(\Sigma_i)$ iff $M_{0i}[\sigma'_i>M'_i\wedge M'_i\in M'_{fi}$; iff $M_{0i}[\sigma_i>M_{(n+1)\,i}\wedge M_{(n+1)i}\in M_{fi}$, $\sigma_i\in L(\Sigma_i)$ and $\sigma_i=\Gamma_{T\to T_i}(\sigma)$, $i\in\{1,2,...,k\}$.

So, $L(\Sigma)=\prod_{i=1}^{k}L(\Sigma_i)$.

5 Behavior Descriptions of Structure-Complex Petri Nets

Definition 5.1. Let $\Sigma=(P,T;F,M_0,P_f)$ be a Petri net. $\forall M\in R(M_0)$, and $P_1\subseteq P$. The projection of M on P_1, denoted as $\Gamma_{P\to P_1}(M)$, is defined as $\forall p\in P_1$, $\Gamma_{P\to P_1}(M)(p)=M(p)$.

Theorem 5.1. Let $\Sigma=(P,T;F,M_0,P_f)$ be a Petri net. There exist a set of S-nets $\Sigma_1,\Sigma_2,...,\Sigma_k$ ($k\geq 2$) such that $\Sigma=\mathop{O}_{T\,i=1}^{k}\Sigma_i$.

Proof. Follow these steps to prove this theorem.

(1) Define a function $f:P\to\{1,2,...,k\}$ such that $\forall p_1,p_2\in P$,

$$(p_1^{\bullet}\cap p_2^{\bullet}\neq\varnothing)\vee({}^{\bullet}p_1\cap{}^{\bullet}p_2\neq\varnothing)\to f(p_1)\neq f(p_2).$$

(2) Decompose Σ into $\Sigma_1,\Sigma_2,...,\Sigma_k$ based on f defined in (1). Each $\Sigma_i=(P_i,T_i;F_i,M_{0i},P_{fi})$ holding

(2.1) $P_i=\{p\in P\mid f(p)=i\}$;

(2.2) $T_i=\{t\in T\mid\exists p\in P_i,t\in{}^{\bullet}p\cup p^{\bullet}\}$;

(2.3) $F_i=\{(P_i\times T_i)\cup(T_i\times P_i)\}\cap F$;

(2.4) $M_{0i}=\Gamma_{P\to P_i}M_0$;

(2.5) $P_{fi}=P_i\cap P_f$.

(3) Let $\Sigma_i=(P_i,T_i;F_i,M_{0i},P_{fi})$ ($i\in\{1,2,...,k\}$) be the decomposed nets of Σ obtained in (2), then for all $i\in\{1,2,...,k\}$, it is easy to prove that Σ_i is an S-net.

(4) Let $\Sigma_i=(P_i,T_i;F_i,M_{0i},P_{fi})$ ($i\in\{1,2,...,k\}$) be the decomposed nets of Σ obtained in (2), then

(4.1) For all $i,j\in\{1,2,...k\},i\neq j$, $P_i\cap P_j=\varnothing,\bigcup_{i=1}^{k}P_i=P$.

(4.2) If $k>1$, then $\forall i\in\{1,2,...,k\},\exists j\in\{1,2,...,k\},i\neq j$ such that $T_i\cap T_j\neq\varnothing,\bigcup_{i=1}^{k}T_i=T$.

Based on Definition 4.2, it is easy to prove that $\Sigma = \mathop{O}_{T}^{k}{}_{i=1} \Sigma_i$.

Theorem 5.1 indicates that every Petri net especially a structure-complex net is a synchronous composition net of a set of S-nets. With the proof process, Algorithm 5.1 gives the method to obtain the S-nets that compose a given structure-complex Petri net.

Algorithm 5.1.

Input: A Petri net $\Sigma = (P, T; F, M_0, P_f)$

Output: A set of S-nets that compose Σ

Step 1. Define a function $f: P \to \{1,2,...,k\}$ such that $\forall p_1, p_2 \in P$,

$$(p_1^{\bullet} \cap p_2^{\bullet} \neq \varnothing) \vee ({}^{\bullet}p_1 \cap {}^{\bullet}p_2 \neq \varnothing) \to f(p_1) \neq f(p_2).$$

Step 2. Decompose Σ into $\Sigma_1, \Sigma_2, ..., \Sigma_k$ based on f. Every $\Sigma_i = (P_i, T_i; F_i, M_{0i}, P_{fi})$ holds

(2.1) $P_i = \{p \in P \mid f(p) = i\}$; and

(2.2) $T_i = \{t \in T \mid \exists p \in P_i, t \in {}^{\bullet}p \cup p^{\bullet}\}$; and

(2.3) $F_i = \{(P_i \times T_i) \cup (T_i \times P_i)\} \cap F$; and

(2.4) $M_{0i} = \Gamma_{P \to P_i} M_0$; and

(2.5) $P_{fi} = P_i \cap P_f$.

Step 3. Output $\Sigma_1, \Sigma_2, ..., \Sigma_k$.

Theorem 5.2. Let $\Sigma = (P, T; F, M_0, P_f)$ be a Petri net. There exist a set of S-nets $\Sigma_1, \Sigma_2, ..., \Sigma_k$ ($k \geq 2$) such that $L(\Sigma) = \mathop{[]}_{i=1}^{k} L(\Sigma_i)$.

Proof. Based on Theorem 5.1, there exist S-net $\Sigma_1, \Sigma_2, ..., \Sigma_k$ ($k \geq 2$) such that $\Sigma = \mathop{O}_{T}^{k}{}_{i=1} \Sigma_i$. With Theorem 4.2, $L(\Sigma) = \mathop{[]}_{i=1}^{k} L(\Sigma_i)$.

Theorem 5.2 proves that the language of a given Petri net can be expressed by the synchronous intersection operation of the languages of a set of S-nets, which can be obtained by Algorithm 5.1, and which language can be obtained with the methods presented in Section 3. So, a method to specify the behaviors of a structure-complex Petri net has been obtained. Algorithm 5.2 is presented for the behavior description of a structure-complex Petri net.

Algorithm 5.2.

Input: a Petri net $\Sigma = (P, T; F, M_0, P_f)$

Output: $L(\Sigma)$

Step 1. Obtain the S-nets $\Sigma_1, \Sigma_2, ..., \Sigma_k$ with Algorithm 5.1.

Step 2. For $i = 1$ to k do

(1) If Σ_i has more than one primitive transitions, then using the methods shown in Cases (1) and (2) to transform Σ_i into a set of S-nets with one and only one primitive transition add them to the subnets set by Step 1, else go to (2).

(2) If Σ_i is a marked S-graph, obtain the expression $L(\Sigma_i)$ based on Theorem 3.1. And go to (3), else:

(2.1) If Σ_i is an S-net with some terminal transitions, obtain the expression $L(\Sigma_i)$ based on Theorem 3.2. And go to (3), else:

(2.2) If there are no initial markings in Σ_i, then obtain the expression $L(\Sigma_i)$ based on Theorem 3.3. And go to (3), else:

(2.3) If $\exists p \in P$ such that $M_0(p) \neq 0$, then obtain the expression $L(\Sigma_i)$ based on Theorem 3.4.

(3) $i \leftarrow i + 1$;

Step 3. $L(\Sigma) \leftarrow \mathop{[]}_{i=1}^{k} (L(\Sigma_i))$

Step 4. Output $L(\Sigma)$, and end.

The termination and correctness of Algorithm 5.2 can be proved by the above analysis.

6 Examples

An unbounded Petri net Σ_6 with complex structures is shown in Fig.6. Now we analyze its behaviors and present its language expression with the method of this paper. It is supposed that the end place set is $P_f = \{p_2, p_4, p_6\}$. With Algorithm 5.1, a function f is defined as:

$f(p_1) = f(p_6) = 1$, $f(p_2) = f(p_3) = 2$ and $f(p_4) = f(p_5) = f(p_7) = 3$.

It is obvious that f holds the conditions of Step 2 in Algorithm 5.1. Based on f, Σ_6 is decomposed into three nets Σ_{61}, Σ_{62} and Σ_{63}, as shown in Fig.6.

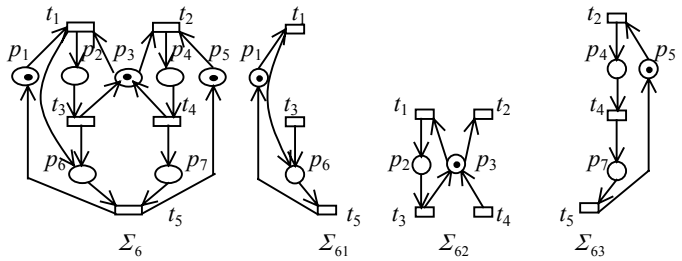

Fig.6 A Petri net Σ_6 ant its three decomposed nets Σ_{61}, Σ_{62} and Σ_{63}

There are one primitive transition t_3 in Σ_{61}, one primitive transition t_4 and one terminal transition t_2 in Σ_{62}. Σ_{63} is an S-graph.

Now, we present the language expressions of the decomposed net systems.

(1) Σ_{61} is the net system shown in Example 3.3. So $L(\Sigma_{61}) = L(\Sigma\Sigma_3) = t_1(t_5t_1)^* \,\|\, (t_3 + t_3(t_5t_1)^*)^\alpha$.

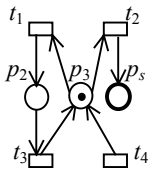

Fig.7 Petri net Σ'_{62}

(2) In Σ_{62}, p_2 is the end place and t_2 is a terminal transition. With Theorem 3.2, Σ_{62} is first transformed into Σ'_{62} shown on the right in Fig.7 with adding a new place p_s to the wide edge. Σ'_{62} holds the conditions of Theorem 3.4. With the same method of obtaining $L(\Sigma_3)$, $L(\Sigma_{62}) = ((t_1t_3)^*t_2 + t_1(t_3t_1)^*) \,\|\, (t_4(t_1t_3)^*t_2 + t_4t_1(t_3t_1)^*)^\alpha$.

(3) Σ_{63} is the net system Σ_1 shown in Example 3.1, where p_4 is the end place. With Example 3.1, $L(\Sigma_{63}) = L(\Sigma_1) = t_2(t_4t_5t_2)^*$.

Based on Theorem 5.2, the expression of Σ_6 is

$$L(\Sigma_6) = L(\Sigma_{61})[]L(\Sigma_{62})[]L(\Sigma_{63}) = t_1(t_5t_1)^* \,\|\, (t_3 + t_3(t_5t_1)^*)^\alpha [] ((t_1t_3)^*t_2 + t_1(t_3t_1)^*) \,\|\, (t_4(t_1t_3)^*t_2 + t_4t_1(t_3t_1)^*)^\alpha [] t_2(t_4t_5t_2)^*.$$

7 Conclusions

To analyze large-scaled complex physical systems based on Petri net, it becomes difficult because the structure is very complex. In order to overcome these difficulties, Jiang[13~15] presents the concept of synchronous composition of Petri nets and has conducted some deep research on it. Based on the previous work[13~15], the concept of synchronous composition is extended and a method is presented, with which a given Petri net is the synchronous composition net of several S-nets. An algorithm to find the S-nets to compose a given Petri net by synchronous composition is obtained. Because the structure of S-nets is simple, the language characters of S-nets can be analyzed easily and the methods to obtain their language expressions are given. With these results, the behavior of a given structure-complex can be expressed based on synchronous composition of Petri nets and synchronous intersection of languages. In the last section, a method to obtain the language expression of a structure-complex Petri net is obtained, which will benefit for analyzing physical system using Petri net languages.

The main contributions of this paper include: (1) The behavior description methods for structure-simple nets,

S-nets. (2) A method to obtain the S-nets to compose any given structure-complex net. (3) A method to specify the behaviors of a given structure-complex Petri net.

Compared with the results of Ref.[12], the algorithm to obtain the language expression of a structure-complex Petri net is more executive and convenient with the synchronous intersection operation of Petri net languages[7].

The following works are under way: (1) The relations between the language expression containing synchronous intersection and the traditional formal language expression[9]. (2) How to analyze the physical system based on the language expression obtained with the method presented in this paper.

Reference:

[1] Yuan CY. Principle of Petri Net. Beijing: Publishing House of Electronics Industry, 1998 (in Chinese).

[2] Peterson J. Petri Net Theory and the Modeling of Systems. Englewood Cliffs: Prentice-Hall, Inc., 1981.

[3] Murata T. Petri nets, properties, analysis and applications. Proc. of the IEEE, 1989,77(4):541~577.

[4] Garg VK, Ragunath MT. Concurrent regular expressions and their relationship to Petri nets. Theoretical Computer Science, 1992, 96(2):285~304.

[5] Wu ZH. Petri net description of pumping lemma−A set of conditions for determining the type of Petri net language. Chinese Journal of Computers, 1994,17(11):852~858 (in Chinese with English abstract).

[6] Jiang CJ, Lu WM. On properties of concurrent system based on Petri net language. Journal of Software, 2001,12(4):512~520 (in English with Chinese abstract).

[7] Zeng QT, Wu ZH. Synchronous intersection operation of Petri net languages. Mini-Micro Systems, 2004,25(2):216~220 (in English with Chinese abstract).

[8] Hopcroft J, Ullman J. Introduction to Automaton Theory Languages and Computation. Reading: Addison-Wesley, 1979.

[9] Lee KH, Favrel J, Baptiste P. Generalized Petri net reduction method. IEEE Trans. on Systems Man and Cybernetics, 1987,17(2): 297~303.

[10] Suzuki I, Murata T. A method for stepwise refinement and abstraction of Petri nets. Journal of Computer and System Science, 1983, 27(1):51~76.

[11] Zeng QT, Wu ZH. Decomposition method of Petri net based on index of places. Journal of Computer Science, 2002,29(4):15~17 (in Chinese with English abstract).

[12] Zeng QT, Wu ZH. Language behavior description of structure-complex Petri net based on decomposition. Journal of System Engineering, 2004,19(3):231~237 (in English with Chinese abstract).

[13] Jiang CJ. Research of process characters of synchronous composition nets. Journal of Electronics, 1996,25(2):57~60 (in Chinese with English abstract).

[14] Jiang CJ. Petri net dynamic invariance. Science in China (Series E), 1997,27(4): 605~611(in Chinese with English abstract).

[15] Jiang CJ. Complete sequence behavior invariance of synchronous composition nets. Journal of Applied Science, 2000,18(3): 271~275 (in Chinese with English abstract).

附中文参考文献:

[1] 袁崇义.Petri 网原理.北京:电子工业出版社,1998.

[5] 吴哲辉.Pumping 引理的 Petri 网描述——Petri 网语言属型的一组判定条件.计算机学报,1994,17(11):852~858.

[6] 蒋昌俊,陆维明.基于 Petri 网语言的并发系统性质研究.软件学报,2001,12(4):512~520.

[7] 曾庆田,吴哲辉.Petri 网语言的同步交运算.小型微型计算机系统,2003,2004,25(2):216~220.

[11] 曾庆田,吴哲辉.基于库所指标的 Petri 网分解方法.计算机科学,2002,29(4):15~17.

[12] 曾庆田,吴哲辉.基于分解的结构复杂 Petri 网的行为描述.系统工程学报,2003,2004,19(3):231~237.

[13] 蒋昌俊.同步合成网的进程特性研究.电子学报,1996,25(2):57~60.

[14] 蒋昌俊.Petri 网的动态不变性.中国科学(E 辑),1997,27(4):605~611.

[15] 蒋昌俊.同步合成网的完全顺序行为不变性.应用科学学报,2000,18(3):271~275.