Petri Net Methods of Constructing Kleene-Closure Operations of Regular Languages

WenJuan Lian, Hao Fan, Yu Yue Du and YongQuan Liang
College of Information Science and Engineering, Shandong University of Science and Technology, Qingdao, Shandong, 266510, China

Abstract: It has been proved that regular language is a subclass of Petri net languages. Standard properly end Petri net, a subclass of Petri nets is defined in the related references, and the equivalency between a standard properly end Petri net and a regular language has been investigated. Thus, the Petri net constructing methods of Kleene closure operations with and without an \( \varepsilon \)-empty label in regular expressions are presented respectively in this study. By the methods, the net model of producing regular language \( L^* \) can be constructed from that of producing regular language \( L \). It is proved that a standard properly end Petri net language of Kleene closure operations is close.

Key words: Standard properly end Petri nets, Petri nets, regular language, regular expression, Kleene-closure operation

INTRODUCTION

As an important part of Petri nets (PN), PN language plays an important role in the description and analysis of system behavior (Du et al., 2007; Zeng and Duan, 2007; Scarpetta et al., 1996; Zeng, 2007). Some good research achievements of PN languages have been made. The relation between PN languages and Chomsky formal languages was analyzed (Peterson, 1981; Murata, 1989). The conceptions of a vector grammar and PN machines were presented, and the relations between vector grammar and PN language and between PN machine and PN languages were investigated (Jiang, 1996). A necessary and sufficient condition of whether a PN is a regular language or a context-free language was obtained based on the Pumping lemma in PN. The Pumping lemma describes the commonness of PN language and by it we can prove that some languages could not be produced by PN (Wu, 1994; Jiang and Lu, 2006; Wang et al., 2000).

However, it’s difficult to construct a language closure operation based on PNs, since some part of a PN model is repeatedly used and some positions may be not empty before they are used again during the construction process. According to the properties of standard properly end PNs, the PN methods of constructing closure operations in regular language are well resolved. Thus, the PN methods of constructing a kleene closure operation with and without an \( \varepsilon \)-empty label are presented in this paper. We prove that the closure operation in standard properly end PNs is close. That is, if \( L(N) \) is a formal language produced by a standard properly end PN, the PN model of producing language \( L(N)^* \) is also a standard properly end PN.

BASIC CONCEPTS AND TERMS

Some basic concepts and properties of PNs are simply reviewed in this section and other concepts of PN languages can be referred by Peterson (1981) and Murata (1989).

Definition 1: (Peterson, 1981; Murata, 1989): Let \( N = (P,T,G,M_0,\Sigma,h,F) \) be a labeled PN, where \((P,T,G,M_0)\) is a PN, \( G \in (P \times T) \cup (T \times P) \) is flow relation of PN, \( \Sigma \) is a finite alphabet, \( h: \Sigma \rightarrow \Sigma \) is a labeled function, \( F \) is an end state set, then

\[ L(N) = \{ \alpha \in \Sigma^* \mid \exists \sigma \in T^* \text{ s.t. } M_0 \Rightarrow^* M \in F \text{ and } h(\alpha) = \alpha \} \]

is a language produced by the PN.

\( L(N) \) is called an \( L \)-type language produced by \( N \) if \( F \) is any subset of \( R(M_0) \). According to different situations
of labeled function $h:T:\Sigma$, there are three types of PN languages, i.e., PN languages without labeled languages, with and without an $e$-empty label language. We mainly investigate the standard properly end Petri net methods of constructing closure operations with and without an $e$-empty label in regular languages in this study.

**Definition 2**: Peterson (1981): Let $N = (P, T, G, M_0, \Sigma, h, F)$ be a labeled Petri net, it produces language $L(N)$. It is called a standard Petri net if it satisfies the following conditions:

- $\exists p_o, p_i \in P$, such that $P_s = \emptyset$ and $p_i = \emptyset$, $p_i$ is called a start position, $p_o$ is called an end position.
- The initial marking $M_0$ satisfies
  $$M_0(p) = \begin{cases} 1, & p = p_o; \\ 0, & p \neq p_o. \end{cases}$$
- Any transition will not be enabled once there exists a token in end position $p_i$.
- There is a marking $M_i$ such that
  $$M_i(p) = \begin{cases} 1, & p = p_i; \\ 0, & p \neq p_i. \end{cases}$$
  If $\lambda \in L(N)$, then $F = \{M_0, M_i\}$, else $F = \{M_0\}$.

**Theorem 1**: Peterson (1981): Let $L$ be any Petri net language; there exists a standard Petri net which produces $L$.

**Definition 3**: Lian et al. (2008): Let $N = (P, T, G, M_0, \Sigma, h, F)$ be a standard Petri net, if $\forall M \in R(M_0)$: $M(P) = 1 - \forall p \in P, p \notin \{p_0\}$, $M(p) = 0$, then $N$ is a standard properly end Petri net.

**Definition 4**: Lian et al. (2008): If $N$ is a standard properly end Petri net, then $L(N) = \{\sigma \in \Sigma^* | \exists \sigma \in T^* \land M_0 \models \sigma \land h(\sigma) = \alpha \}$ is called a language produced by $N$.

**Theorem 2**: Lian et al. (2008): Standard properly end Petri nets must be bounded Petri nets.

**Theorem 3**: Lian et al. (2008): A standard properly end Petri net language is equivalent to a regular language.

**Definition 5**: John et al. (2004): Let $L_1$, be a language in alphabet $\Sigma$. Kleene closure operation $\ast$ is defined as follows:

$$L_1^\ast = \bigcup_{i=0}^{\infty} L_1 = L_0^\ast \cup L_1^1 \cup L_1^2 \cup \cdots$$

where, $L_1^0 = \{\lambda\}$ and $\lambda$ is an empty string.

**PN METHOD OF CONSTRUCTING KLEENE CLOSURE OPERATIONS**

**Differences between standard properly end PNs and standard PNs**: Standard properly end Petri nets are more special than common standard Petri nets from definitions 2 and 3. This can be illustrated with Fig. 1. In Fig. 1a, $N_i$ is a standard Petri net, but not a standard properly end Petri net. Since there exists $\sigma = \varepsilon aab\varepsilon b$ and $M_0(\sigma) = M'$, where

$$M' = \begin{cases} 1, & p = (p_2, p_0); \\ 0, & p \notin P - \{p_0, p_2\}. \end{cases}$$

it doesn’t satisfy $\forall M \in R(M_0)$: $M(P) = 1 - \forall p \in P, p \notin \{p_0\}$, $M(p) = 0$.

In Fig. 1b, $N_2$ is a standard properly end Petri net, and it can be seen when there exists a token in $p_0$. There isn’t any transition which can be fired and there is not any place which includes redundant tokens. When $N_2$ stops running (no transition can be fired), therefore, it is in an acceptable state $M_s$. Thus $N_2$ is called a properly end net.

In Fig. 1a, the language produced by net $N_1$ is $L(N_1) = a^*b^\ast$ and the net is in an acceptable state $M_0$ where

$$M_0(p) = \begin{cases} 1, & p = p_0; \\ 0, & p \neq p_0. \end{cases}$$

Although a transition sequence $\sigma = \varepsilon aab\varepsilon b$ can be fired by net $N_1$ and the net stops running at state $M'$, where

$$M' = \begin{cases} 1, & p = (p_2, p_0); \\ 0, & p \notin P - \{p_0, p_2\}. \end{cases}$$

$M'$ is not the acceptable state of net $N_1$, there is redundant token in $p_0$, $M(P) = 1$, thus, $h(\sigma) = h(\varepsilon aab\varepsilon b) = aab \varepsilon$ $L(N_1)$ Similarly, closure operation $\ast$ in Petri net languages is not close. When the structure of net $N_1$ is

![Diagram](a)

![Diagram](b)

Fig. 1: Standard PNs and standard properly End PNs, (a) Standard Petri Net $N_i, L(N_i) = \{a^*b^\ast | n > 0\}$ and (b): Standard properly end Petri Net $N_2, L(N_2) = (a+b)^\ast$.
Fig. 2: Net $N_1$

Fig. 3: Model of an abstract standard properly end petri net

used repeatedly to produce language $L(N_1) = (a^*b^*)^*$, it can’t ensure some positions or position sets be empty before they’re used again, such as $p_1$ in Fig. 2. For net $N_2$, there exists $M_1[\sigma \Rightarrow M_2]$ where $\sigma = a^*b^*a^*b^* (n_1 = n_1, n_2 = n_2, n_2 = n_2 + n_1) \in L(N_2)$. However, $L(N_3) \neq (a^*b^*)^*$. For a standard properly end Petri net, an acceptable state $M_1$ is reached when it stops running (no transition can be fired). That means that there is no redundant token in the net. Therefore, standard properly end Petri nets can be used to construct closure operation of regular languages.

**Petri net (with $\epsilon$-empty label) method of constructing closure operation:** For convenience, an abstract standard properly end Petri net can be represented in Fig. 3. $p_1$, $p_2$ are a start position and an end position, respectively. $T_{in} = \{t \in P \}$ is a start transition set and $T_{out} = \{t \in P \}$ is an end transition set.

**Theorem 4:** Let $N_1 = (P, T, \Sigma, M_0, \Sigma, h, F)$ be a standard properly end Petri net, it produces a language $L(N_1)$. Then there exists a standard properly end Petri net $N = (P, T, \Sigma, M_0, \Sigma, h, F)$ with an $\epsilon$-empty label, such that the net produces a language $L(N) = L(N_1)^*$.

**Proof:** Construct a net $N$ from $N_1$ (Fig. 4). Let $L(N) = L(N_1)^*$ be a language produced by $N$. Add two new places $p_1, p_2$ as a start position and an end position of $N$, respectively. In order to produce Kleene closure $^*$, we introduce four transitions $t_{11} \mid t_{12} \mid t_{21} \mid t_{22}$ with an empty label. Therefore, we have $N = (P, T, G, M_0, \Sigma, h, F)$, where

$$\begin{align*}
P &= P_1 \cup \{p_1, p_2\}, T = T_1 \cup \{t_{11}, t_{12}, t_{21}, t_{22}\} \\
G &= G_1 \cup (p_1 \times t_{11}, t_{12}) \cup (t_{12}, t_{22}) \times \{p_2\} \\
M_1(p_1) &= [1, 1, p = p_1, 0, p \neq p_1] \\
h(t_{11}) &= \begin{cases} h_1(t_{11}), & t \in T_1 \\
0, & t \in \{t_{11}, t_{12}, t_{21}, t_{22}\} \end{cases} \quad F = \{M_1\} \\
M_1(p_2) &= [1, 1, p = p_2, 0, p \neq p_2]
\end{align*}$$

$L(N) = L(N_1)^*$ is proved as follows.

If a transition sequence set $N_1(\sigma)$ is defined by $N_1(\sigma) = \sigma \epsilon \in T_1 \land M_0[\sigma \Rightarrow M_1]$. Then $h_1(N_1(\sigma)) = L(N_1)$. We define two markings $M_0$ and $M_1$ in $N$, where $M_0(p) = [1, 1, p = p_1, 0, p \neq p_1]$. According to Fig. 4, there exist only three fire transition ways which can transfer a token from $p_1$ to $p_2$.

(a) $h(t_{11}) \in L(N)$ and $\epsilon \in L(N)$ since $M_1(t_{11}) = M_2$.

(b) $M_1(t_{11}) = M_2[\sigma \Rightarrow M_3] \land M_3[\sigma \Rightarrow M_4]$ where $\sigma \in N_1(\sigma)$. Let $\alpha = h_1(\sigma)$, then $\alpha \in L(N_1)$. According to this transition sequence $\sigma$, we have $h(t_{11}, \sigma \Rightarrow \alpha) \in L(N)$ and $h(t_{12}, \sigma = \alpha \Rightarrow \epsilon) \in L(N)$.

(c) $M_1(t_{11}) = M_2[\sigma \Rightarrow M_3] \land M_3[\sigma \Rightarrow M_4] \land M_4[\sigma \Rightarrow M_5] \land M_5[\sigma \Rightarrow M_6] \land M_6[\sigma \Rightarrow M_7] \land M_7[\sigma \Rightarrow \epsilon]$ where $\sigma, \alpha, \ldots, \alpha_n$ are random elements in $N_1(\sigma)$. Let $\alpha_j = h_1(\sigma_j)$ ($j = 1, 2, \ldots$), then $\alpha \in L(N)$. According to transition sequence $t_{11} \in \sigma \Rightarrow \alpha_1 \in \sigma \Rightarrow \alpha_2 \in \sigma \Rightarrow \ldots \alpha_n \in \sigma \Rightarrow \epsilon$, we have $h(t_{11}, \sigma \Rightarrow \alpha_1 \in \sigma \Rightarrow \alpha_2 \in \sigma \Rightarrow \ldots \alpha_n \in \sigma \Rightarrow \epsilon) \in L(N)$ and $h(t_{12}, \sigma \Rightarrow \epsilon) \in L(N)$. Therefore, $h(t_{11}) \epsilon \in L(N)$.

According to the randomness of $\sigma, \sigma, \ldots \sigma_n$, it’s known that $L(N_1)^* \subseteq L(N)$, where $k$ is a positive integer and $k \geq 1$. From rules a, b and c, we get
Petri net (without ε-empty label) method of constructing closure operation: It's more difficult to construct a Petri net model without ε-empty label, since a net structure must be reused. In the following, we discuss primarily how to construct a Petri net model without an ε-empty label of the closure operation of regular languages.

Theorem 5: Let \( N_i = (P_i, T_i, G_i, M_{i0}, \Sigma_i, h_i, F_i) \) be a standard properly end Petri net without an ε-empty label, it produces a language \( L(N_i) \). Then there exists a standard properly end Petri net \( N = (P, T, G, M_0, \Sigma, h, F) \), its labeled function \( h \) is without an ε-empty label and it produces a language \( L(N) = (L(N_i))^* \).

Proof: Construct two new places \( p_0, p_1 \) in \( N \) and they are used as a start position and an end position respectively. The structure of \( N_i \) will be repeatedly used for 4 times to produce the kleene closure \( * \) of \( L(N_i) \). The structure of \( N = (P, T, G, M_0, \Sigma, h, F) \), is shown in Fig. 5 in detail and some unnecessary processes are omitted. From \( \lambda \in (L(N_i))^* \), \( F = \{M_0, M_1, M_2, M_3\} \) is the end marking set of \( N \).

The proof of \( L(N) = (L(N_i))^* \) is given as follows.

For convenience, some markings of \( N \) are defined as

\[
M_0(p) = \begin{cases} 1, & p = p_0 \\ 0, & p \neq p_0 \end{cases},
M_1(p) = \begin{cases} 1, & p = p_1 \\ 0, & p \neq p_1 \end{cases},
M_2(p) = \begin{cases} 1, & p = p_2 \\ 0, & p \neq p_2 \end{cases},
M_3(p) = \begin{cases} 1, & p = p_3 \\ 0, & p \neq p_3 \end{cases}.
\]

According to the construction of net \( N \), there exist only three ways of transferring a token from \( p_0 \) to \( p_3 \). Therefore, three types of transition firing sequences are analyzed as follows.

(a) \( M_1[σ] > M_2 \) where \( σ \) is any transition sequence in \( N_i(σ) \). Then \( L(N_i) \subset L(N) \).
(b) \( M_1[σ] > M_2[σ] > M_3[σ] \) where \( σ, σ_1, σ_2 \in N_i(σ) \). Then \( L(N_i) \subset L(N) \).
(c) \( M_1[σ] > M_2[σ] > M_3[σ] > M_4[σ] > ... > M_n[σ] > M_1[σ] \) where \( σ, σ_1, ..., σ_n \) are random transition sequences in \( N_i(σ) \). Then \( L(N_i)^* \subset L(N) \) when \( k \) is a positive integer and \( k \geq 2 \).

According to cases a), b) and c), we have \( L(N_i)^* \subset L(N) \). From \( M_i \in F \), then \( L(N_i)^* \subset L(N) \) and \( L(N) = (L(N_i))^* \). Therefore, \( N \) is a standard properly end Petri net by means of the structure of \( N \) and the previous proof.

EXAMPLES

With the respect to PNs with an ε-empty label and without an ε-empty label, Theorems 4 and 5 show respectively the PN methods of constructing kleene closure operation \( * \) in regular language. The use of the constructing methods is illustrated with the following examples.

Example 1: Let \( Σ = \{a, b\} \) and net \( N_i = (P_i, T_i, G_i, M_{i0}, \Sigma_i, h_i, F_i) \) be a standard properly end Petri net. The structure of \( N_i \) is shown in Fig. 6a and it produces a language \( L(N_i) = ab \). Construct a standard properly end Petri net \( N \) with an ε-empty label and \( L(N) = (L(N_i))^* = (ab)^* \).

According to Theorem 4, a standard end Petri net \( N \) with an ε-empty label can be constructed in Fig. 6b. We can easily verify that it satisfies \( L(N) \subset (L(N_i))^* \).

Example 2: Let \( Σ = \{a, b\} \) and net \( N_i = (P_i, T_i, G_i, M_{i0}, \Sigma_i, h_i, F_i) \) be a standard properly end Petri net. Net \( N_i \) is shown in Fig. 6a and it produces a language \( L(N_i) = ab \). Construct a standard properly end Petri net \( N \) without an ε-empty label and \( L(N) = (L(N_i))^* = (ab)^* \).
According to Theorem 5, a standard end Petri net $N$ without an $e$-empty label can be constructed in Fig. 7. We can easily check it satisfies $L(N) = (L(N_1))^* = (ab)^*$. 

CONCLUSIONS

Petri net languages play an important role in system behavior description and analysis. It is a fact that many operations of PN languages are close, such as connection, union, reverse, permutation operations. However, a closure operation is not close. It means that there is a net $N$ which produces a language $L(N)$, but the language $(L(N))^*$ is not a Petri net language. Standard properly end PNs are seen as a subclass of PNs in this paper, and its language is a regular language. We prove that closure operation of standard properly end Petri nets is close. Given a standard properly end Petri net $N$, producing a language $L(N_1)$, therefore, a standard properly end Petri net $N$ can be constructed and it produces a language $(L(N_1))^*$. The Petri net with and without an $e$-empty label methods of constructing closure operations in regular language are proposed explicitly, and the methods are illustrated with corresponding instances. This work has an important value to investigate further the Petri net language theory and applications.

ACKNOWLEDGMENTS

This study was supported in part by the National Natural Science Foundation of China under Grants 60773034, 60673053, 60603090, 60534060 and 60473094; the Taishan Scholar Construction Project of Shandong Province, China; the National Basic Research Program of China (973 Program) under Grants 2003CB316902, 2007CB316S02 and 2004CB318001-03 and the Open Project of the State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences under Grant SYSKF0604.

REFERENCES


