The Special Issue of the ScienceAsia
publisher@iccscm.com

INVOICE

To: Nurhidaya Mohamad Jan,

Greetings; It is my pleasure to inform you that your paper entitled "*Place-Labeled Petri Net Controlled Grammars*" has been accepted for publication in the special issue of the **ScienceAsia** journal hosted by the **Science and Knowledge Research Society**.

The publication processing fee is **RM 2300**.

This invoice is valid until 13/09/2016.

The payment can be done using the online link provided through email.

Kamal Mohd.
SandKRS Director

This special issue is hosted by the Science and Knowledge Research Society (registered in Malaysia as SandKRS sdn bhd.)
Tel: 0060327306577-Ext. 13369, Fax:0060327262819
Addr: Lot 9.C00, Wisma Central, Jalan Ampang, 50450, Kuala Lumpur, Malaysia.

Place-Labeled Petri Net Controlled Grammars

N. Mohamad Jan[a], **S. Turaev**[b,*], **W.H. Fong**[c], **N.H. Sarmin**[a]

[a] Department of Mathematical Sciences, Faculty of Science, Universiti Teknologi Malaysia, 81310 UTM Johor Bahru, Johor, Malaysia

[b] Department of Computer Science, Kulliyyah of Information and Communication Technology, International Islamic University Malaysia, 53100 Kuala Lumpur, Malaysia

[c] Ibnu Sina Institute for Fundamental Science Studies, Universiti Teknologi Malaysia, 81310 UTM Johor Bahru, Johor, Malaysia

*Corresponding author, e-mail: sherzod@iium.edu.my

ABSTRACT: A *place-labeled Petri net (pPN) controlled grammar* is a context-free grammar equipped with a Petri net and a function which maps places of the net to the productions of the grammar. The language consists of all terminal strings that can be obtained by simultaneously applying of the rules of multisets which are the images of the sets of the input places of transitions in a successful occurrence sequence of the Petri net. In this paper, we study the generative power and structural properties of pPN controlled grammars. We show that pPN controlled grammars have the same generative power as matrix grammars. Moreover, we prove that for each pPN controlled grammar, we can construct an equivalent place-labeled *ordinary* net controlled grammar.

KEYWORDS: Petri nets, context-free grammars, Petri net controlled grammars, computational power, structural properties

INTRODUCTION

Petri nets[1], "dynamic" bipartite directed graphs with two sets of nodes, called *places* and *transitions*, provide an elegant and powerful mathematical formalism for modeling concurrent systems and their behavior. Since Petri nets successfully describe and analyze the flow of information and the control of action in such systems, they can be very suitable tools for studying the properties of formal languages. If Petri nets are initially used as language generating/accepting tools[2–8], in recent studies, they have been widely applied as regulation mechanisms for grammar systems[9], automata[10–15], and grammars[16–32].

A *Petri net controlled grammar* is, in general, a context-free grammar equipped with a (place/transition) Petri net and a function which maps transitions of the net to productions of the grammar. Then, the language consists of all terminal strings that can be obtained by applying of the sequence of productions which is the image of an occurrence sequence of the Petri net under the function. Several variants of Petri net controlled grammars have been introduced and investigated:

Refs. 18, 19, 24 introduce k*-Petri net controlled grammars* and study their properties including generative power, closure properties, infinite hierarchies, etc.

Refs. 20, 22 consider a generalization of regularly controlled grammars: instead of a finite automaton a Petri net is associated with a context-free grammar and it is required that the sequence of applied rules corresponds to an occurrence sequence of the Petri net, i.e., to sequences of transitions which can be fired in succession.

Refs. 21, 23 investigate grammars controlled by the structural subclasses of Petri nets, namely state machines, marked graphs, causal nets, free-choice nets, asymmetric choice nets and ordinary nets. it was proven that the family of languages generated by (arbitrary) Petri net controlled grammars coincide with the family of languages generated by grammars controlled by free-choice nets.

Refs. 26–28 continue the research on Petri net controlled grammars by restricting to (context-free, extended or arbitrary) Petri nets with place capacities. A Petri net with place capacity regulates the defining grammar by permitting only those derivations where the number of each nonterminal in each sentential form is bounded by its capacity. It was shown that several families of languages generated by grammars controlled by extended cf Petri nets with place capacities coincide with the family of matrix languages of finite index.

In all above-mentioned variants of Petri net controlled grammars, the production rules of a core grammar are associated only with transitions of a control Petri net. Thus, it is also interesting to consider the *place labeling strategies* with Petri net controlled grammars. Theoretically, it would complete the node labeling cases, i.e., we study the cases where the production rules are associated with places of a Petri net, not only with its transitions. Moreover, the place labeling makes possible to consider parallel application of production rules in Petri net controlled grammars, which allows to develop formal language based models for synchronized/parallel discrete event systems.

Informally, a *place-labeled Petri net controlled grammar* (a *pPN controlled grammar* for short) is a context-free grammar with a Petri net and a function which maps places of the net to productions of the grammar. The language consists of all terminal strings that can be obtained by parallelly applying of the rules of *multisets* which are the images of the sets of the input places of transitions in a successful occurrence sequence of the Petri net. In this paper, we study the effect of the place labeling strategies to the computational power, establish the lower and upper bounds for the families of languages generated by pPN controlled grammars, and investigate their structural properties.

PRELIMINARIES

We assume that the reader is familiar with the basic concepts of formal language theory and Petri nets. In this section we only recall some notions, notations and results directly related to the current work. For more details we refer the reader to Ref. 33 and Refs. 4, 5, 34.

Throughout the paper we use the following general notations. The symbol $\in$ denotes the membership of an element to a set while the negation of set membership is denoted by $\notin$. The inclusion is denoted by $\subseteq$ and the strict (proper) inclusion is denoted by $\subset$. The empty set is denoted by $\varnothing$. The cardinality of a set X is denoted by $|X|$.

Grammars

Let Σ be an alphabet. A *string* over Σ is a sequence of symbols from the alphabet. The *empty* string is denoted by λ which is of length 0. The set of all strings over the alphabet Σ is denoted by Σ^*. A subset L of Σ^* is called a *language*. If $w = w_1w_2w_3$ for some $w_1, w_2, w_3 \in \Sigma^*$, then w_2 is called a *substring* of w. The *length* of a string w is denoted by $|w|$, and the number of occurrences of a symbol a in a string w by $|w|_a$.

A *multiset* over an alphabet Σ is a mapping $\pi : \Sigma \to \mathbb{N}$. The alphabet Σ is called the *basic set* of a multiset π and the elements of Σ is called the *basic elements* of a multiset π. A multiset π over $\Sigma = \{a_1, a_2, \ldots a_n\}$ is denoted by

$$\pi = [\underbrace{a_1, \ldots, a_1}_{\pi(a_1)}, \underbrace{a_2, \ldots, a_2}_{\pi(a_2)}, \ldots, \underbrace{a_n, \ldots, a_n}_{\pi(a_n)}].$$

We also "abuse" the set–membership notation by using it for multisets. We write $a \in [a, a, a, b]$ and $c \notin [a, a, a, b]$. The set of all multisets over Σ is denoted by $\Sigma^\oplus$.

A *context-free grammar* is a quadruple $G = (V, \Sigma, S, R)$ where V and Σ are disjoint finite sets of *nonterminal* and *terminal* symbols, respectively, $S \in V$ is the *start* symbol and a finite set $R \subseteq V \times (V \cup \Sigma)^*$ is a set of *(production) rules*. Usually, a rule (A, x) is written as $A \to x$. A rule of the form $A \to \lambda$ is called an *erasing rule*. A string $x \in (V \cup \Sigma)^+$ *directly derives* a string $y \in (V \cup \Sigma)^*$, written as $x \Rightarrow y$, iff there is a rule $r = A \to \alpha \in R$ such that $x = x_1Ax_2$ and $y = x_1\alpha x_2$. The reflexive and transitive closure of $\Rightarrow$ is denoted by $\Rightarrow^*$. A derivation using the sequence of rules $\pi = r_1r_2 \cdots r_n$ is denoted by $\xRightarrow{\pi}$ or $\xRightarrow{r_1r_2\cdots r_n}$. The *language* generated by G is defined by $L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$.

A *matrix grammar* is a quadruple $G = (V, \Sigma, S, M)$ where V, Σ, S are defined as for a context-free grammar, M is a finite set of *matrices* which are finite strings over a set of context-free rules (or finite sequences of context-free rules). The language generated by G is $L(G) = \{w \in \Sigma^* \mid S \xRightarrow{\pi} w \text{ and } \pi \in M^*\}$. The families of languages generated by matrix grammars without erasing rules and by matrix grammars with erasing rules are denoted by $\mathbf{MAT}$ and $\mathbf{MAT}^\lambda$, respectively.

Theorem 1 (Ref. 35)

$$\mathbf{CF} \subset \mathbf{MAT} \subset \mathbf{CS} \text{ and } \mathbf{MAT} \subseteq \mathbf{MAT}^\lambda \subset \mathbf{RE}$$

where $\mathbf{CF}$, $\mathbf{CS}$ *and* $\mathbf{RE}$ *denote the families of context-free, context-sensitive and recursively enumerable languages, respectively.*

Petri Nets

A *Petri net* (PN) is a construct $N = (P, T, F, \phi)$ where P and T are disjoint finite sets of *places* and *transitions*, respectively, $F \subseteq (P \times T) \cup (T \times P)$ is the set of *directed arcs*, $\phi : F \to \mathbb{N}$ is a *weight function*.

A Petri net can be represented by a bipartite directed graph with the node set $P \cup T$ where places are drawn as *circles*, transitions as *boxes* and arcs as *arrows*. The arrow representing an arc $(x, y) \in F$ is labeled with $\phi(x, y)$; if $\phi(x, y) = 1$, then the label is omitted.

An *ordinary net* (ON) is a Petri net $N = (P, T, F, \phi)$ where $\phi(x, y) = 1$ for all $(x, y) \in F$. We omit ϕ from the definition of an ordinary net, i.e., $N = (P, T, F)$.

A mapping $\mu : P \to \mathbb{N}_0$ is called a *marking*. For each place $p \in P$, $\mu(p)$ gives the number of *tokens* in p. Graphically, tokens are drawn as small solid *dots* inside circles. The sets ${}^\bullet x = \{y \mid (y, x) \in F\}$ and $x^\bullet = \{y \mid (x, y) \in F\}$ are called *pre-* and *post-sets* of $x \in P \cup T$, respectively. For $X \subseteq P \cup T$, define ${}^\bullet X = \bigcup_{x \in X} {}^\bullet x$ and $X^\bullet = \bigcup_{x \in X} x^\bullet$. For $t \in T$ ($p \in P$), the elements of ${}^\bullet t$ (${}^\bullet p$) are called *input* places (transitions) and the elements of $t^\bullet$ ($p^\bullet$) are called *output* places (transitions) of t (p).

A sequence of places and transitions $\rho = x_1 x_2 \cdots x_n$ is called a *path* if and only if no place or transition except x_1 and x_n appears more than once, and $x_{i+1} \in x_i^\bullet$ for all $1 \leqslant i \leqslant n - 1$.

A transition $t \in T$ is *enabled* by marking μ if and only if $\mu(p) \geqslant \phi(p, t)$ for all $p \in {}^\bullet t$. In this case t can *occur* (*fire*). Its occurrence transforms the marking μ into the marking μ' defined for each place $p \in P$ by $\mu'(p) = \mu(p) - \phi(p, t) + \phi(t, p)$. We write $\mu \xrightarrow{t}$ to denote that t may fire in μ, and $\mu \xrightarrow{t} \mu'$ to indicate that the firing of t in μ leads to μ'. A marking μ is called *terminal* if in which no transition is enabled. A finite sequence $t_1 t_2 \cdots t_k \in T^*$, is called *an occurrence sequence* enabled at a marking μ and finished at a marking μ' if there are markings $\mu_1, \mu_2, \ldots, \mu_{k-1}$ such that $\mu \xrightarrow{t_1} \mu_1 \xrightarrow{t_2} \ldots \xrightarrow{t_{k-1}} \mu_{k-1} \xrightarrow{t_k} \mu'$. In short this sequence can be written as $\mu \xrightarrow{t_1 t_2 \cdots t_k} \mu'$ or $\mu \xrightarrow{\nu} \mu'$ where $\nu = t_1 t_2 \cdots t_k$. For each $1 \leqslant i \leqslant k$, marking μ_i is called *reachable* from marking μ. $\mathcal{R}(N, \mu)$ denotes the set of all reachable markings from a marking μ.

A *marked* Petri net is a system $N = (P, T, F, \phi, \iota)$ where (P, T, F, ϕ) is a Petri net, ι is the *initial marking*.

A Petri net *with final markings* is a construct $N = (P, T, F, \phi, \iota, M)$ where (P, T, F, ϕ, ι) is a marked Petri net and $M \subseteq \mathcal{R}(N, \iota)$ is set of markings which are called *final* markings. An occurrence sequence ν of transitions is called *successful* for M if it is enabled at the initial marking ι and finished at a final marking τ of M. If M is understood from the context, we say that ν is a successful occurrence sequence.

A Petri net N is said to be k-*bounded* if the number of tokens in each place does not exceed a finite number k for any marking reachable from the initial marking ι, i.e., $\mu(p) \leqslant k$ for all $p \in P$ and for all $\mu \in \mathcal{R}(N, \iota)$. A Petri net N is said to be *bounded* if it is k-bounded for some $k \geqslant 1$.

DEFINITIONS AND EXAMPLES

In this section, we define a place-labeled Petri net controlled grammar, a derivation step, a successful derivation and the language of a place labeled Petri net controlled grammar.

Definition 1 A *place labeled Petri net controlled grammar* (a *pPN controlled grammar* for short) is a 7-tuple $G = (V, \Sigma, R, S, N, \beta, M)$ where (V, Σ, R, S) is a context-free grammar, N is a (marked) Petri net, $\beta : P \to R \cup \{\lambda\}$ is a place labeling function and M is a set of final markings.

Let $A \subseteq P$. We use the notations $\beta(A)$ and $\beta_{-\lambda}(A)$ to denote the multisets $[\beta(p) \mid p \in A]$ and $[\beta(p) \mid p \in A \text{ and } \beta(p) \neq \lambda]$, respectively. Further, we define the notions of a *successful derivation step* and a *successful derivation*.

Definition 2 $x \in (V \cup \Sigma)^*$ *directly derives* $y \in (V \cup \Sigma)^*$ with a multiset $\pi = [A_{i_1} \to \alpha_{i_1}, \ldots, A_{i_k} \to \alpha_{i_k}] \subseteq R^\oplus$ written as $x \stackrel{\pi}{\Longrightarrow} y$, if and only if

$$x = x_1 A_{i_1} x_2 A_{i_2} \cdots x_k A_{i_k} x_{k+1} \text{ and } y = x_1 \alpha_{i_1} x_2 \alpha_{i_2} \cdots x_k \alpha_{i_k} x_{k+1}$$

where $x_j \in (V \cup \Sigma)^*$, $1 \leqslant j \leqslant k + 1$, and $\pi = \beta_{-\lambda}({}^\bullet t)$ for some $t \in T$ enabled at a marking $\mu \in \mathcal{R}(N, \iota)$.

Definition 3 A derivation

$$S \stackrel{\pi_1}{\Longrightarrow} w_1 \stackrel{\pi_2}{\Longrightarrow} w_2 \stackrel{\pi_3}{\Longrightarrow} \cdots \stackrel{\pi_n}{\Longrightarrow} w_n = w \in \Sigma^*, \quad (1)$$

where $\pi_i \subseteq R^{\oplus}, 1 \leqslant i \leqslant n$, is called *successful* if and only if $\pi_i = \beta_{-\lambda}({}^{\bullet}t_i)$ for some $t_i \in T$, $1 \leqslant i \leqslant n$, and $t_1 t_2 \cdots t_n \in T^*$ is a successful occurrence sequence in N. For short, (1) can be written as $S \xRightarrow{\pi_1 \pi_2 \cdots \pi_n} w$.

Definition 4 The *language* generated by pPN controlled grammar G consists of strings $w \in \Sigma^*$ such that there is a successful derivation $S \xRightarrow{\pi_1 \pi_2 \cdots \pi_n} w$ in G.

With respect to different labeling strategies and the definition of final marking sets, we can define various variants of place labeled Petri net controlled grammars. In this work, we define the following variants:

Definition 5 A pPN controlled grammar $G = (V, \Sigma, S, R, N, \beta, M)$ is called

- *free* (denoted by f) if a different label is associated to each place, and no place is labeled with the empty string,
- λ-*free* (denoted by $-\lambda$) if no place is labeled with the empty string,
- *arbitrary* (denoted by λ) if no restriction is posed on the labeling function β.

Definition 6 A pPN controlled grammar $G = (V, \Sigma, S, R, N, \beta, M)$ is called

- r-*type* if M is the set of all reachable markings from the initial marking i, i.e. $M = \mathcal{R}(N, \iota)$.
- t-*type* if $M \subseteq \mathcal{R}(N, \iota)$ is a finite set.

We use the notation (x, y)-*pPN controlled grammar* where $x \in \{f, -\lambda, \lambda\}$ shows the type of a labeling function and $y \in \{r, t\}$ shows the type of a set of final markings. We denote by $p\mathbf{PN}(x, y)$ and $p\mathbf{PN}^{\lambda}(x, y)$ the families of languages generated by (x, y)-pPN controlled grammars without and with erasing rules, respectively, where $x \in \{f, -\lambda, \lambda\}$ and $y \in \{r, t, g\}$. We also use bracket notation $p\mathbf{PN}^{[\lambda]}(x, y)$, $x \in \{f, -\lambda, \lambda\}$, $y \in \{r, t\}$, in order to say that a statement holds both in case with erasing rules and in case without erasing rules.

LOWER AND UPPER BOUNDS

The following inclusions immediately follow from the definitions of place-labeled Petri net controlled grammars.

Lemma 1 *For* $x \in \{f, -\lambda, \lambda\}$ *and* $y \in \{r, t\}$, $p\mathbf{PN}(x, y) \subseteq p\mathbf{PN}^{\lambda}(x, y)$.

Example 1 Let $G_1 = (\{S, A, B, C\}, \{a, b, c\}, S, R)$ be a context-free grammar where R consists of the following productions:

$$r_0 : S \to ABC,\ r_1 : A \to aA,\ r_2 : A \to bB,\ r_3 : AC \to cC,\ r_4 : A \to a,\ r_5 : B \to b,\ r_6 : C \to c.$$

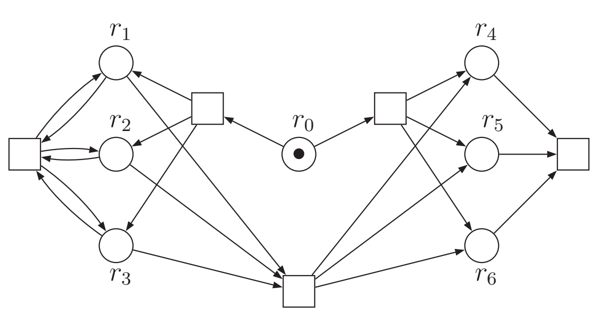

Fig. 1 Petri net N_1.

Figure 1 illustrates a Petri net N_1 with respect to G_1. Obviously,

$$L(G_1) = \{a^n b^n c^n \mid n \geqslant 1\} \in p\mathbf{PN}(f, t).$$

Example 2 Let G_2 be a context-free grammar with the rules:

$$r_0 : S \to AB, r_1 : A \to aA, r_2 : B \to aB, r_3 : A \to bA, r_4 : B \to bB, r_5 : A \to \lambda, r_6 : B \to \lambda$$

Figure 2 illustrates a Petri net N_2 with respect to G_2. It is not difficult to see that

$$L(G_2) = \{ww \mid w \in \{a, b\}^*\} \in p\mathbf{PN}(\lambda, t).$$

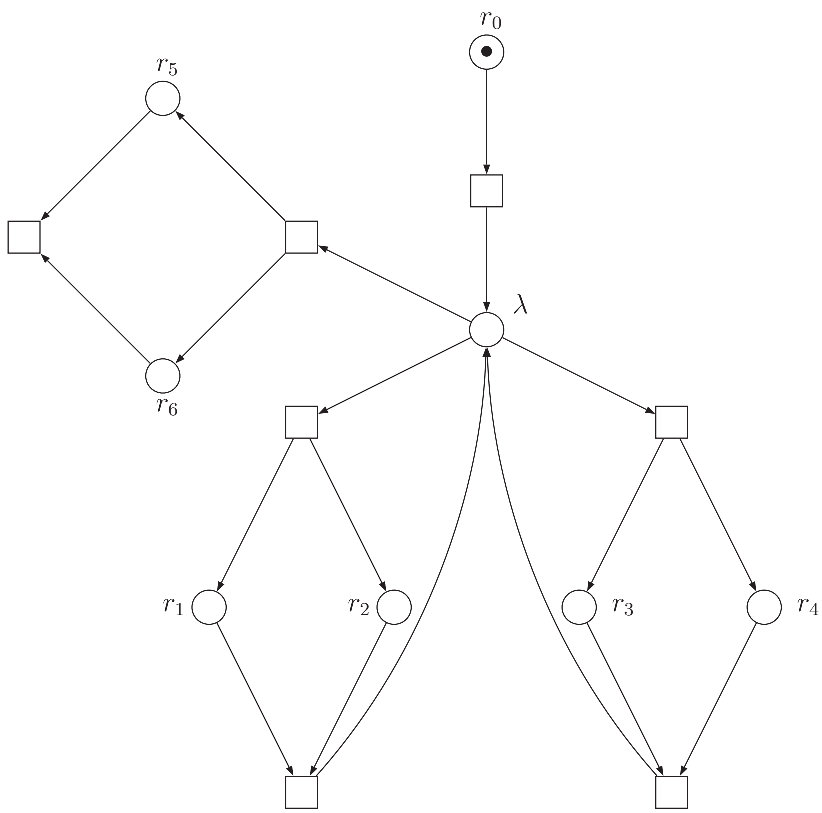

Fig. 2 Petri net N_2.

Further, we discuss the upper bound for the families of languages generated by pPN controlled grammars.

Lemma 2 *For* $y \in \{r, t\}$, $p\mathbf{PN}^{[\lambda]}(-\lambda, y) \subseteq \mathbf{MAT}^{\lambda}$.

Proof: Let $G = (V, \Sigma, S, R, N, \beta, M)$ be an $(-\lambda, y)$-pPN controlled grammar (with or without erasing rules) and $N = (P, T, F, \phi, \iota)$ where $y \in \{r, t\}$. Let $P = \{p_1, p_2, ..., p_s\}$ and $T_\varnothing = \{t \in T \mid {}^\bullet t = \varnothing\}$. Suppose, $T - T_\varnothing = \{t_1, t_2, \ldots, t_n\}$. We define the sets of new nonterminals as

$$\overline{P} = \{\overline{p} \mid p \in P\} \text{ and } \overline{V} = \{\overline{A} \mid A \in V\},$$

and set the homomorphism $h : (V \cup \Sigma)^* \to (\overline{V} \cup \Sigma^*)$ as

$$h(a) = a \text{ for all } a \in \Sigma, \text{ and } h(A) = \overline{A} \text{ for all } A \in V.$$

Consider $t \in T - T_\varnothing$, and let ${}^\bullet t = \{p_{i_1}, p_{i_2}, \ldots, p_{i_k}\}$. We assume that $\beta(p_{i_j}) = A_{i_j} \to \alpha_{i_j} \in R$, $1 \leqslant j \leqslant k$. Let

$$h(\alpha_{i_1}\alpha_{i_2}\cdots\alpha_{i_k}) = x_1\overline{B}_1x_2\overline{B}_2\cdots x_l\overline{B}_lx_{l+1}$$

where $x_i \in \Sigma^*, 1 \leqslant i \leqslant l+1$ and $\overline{B}_j \in \overline{V}, 1 \leqslant j \leqslant l$.

We associate the following sequences of rules with each transition $t \in T - T_\varnothing$:

$$\delta_{t,\lambda} : \underbrace{\overline{p}_{i_1} \to \lambda, \ldots, \overline{p}_{i_1} \to \lambda}_{\phi(p_{i_1},t)}, \underbrace{\overline{p}_{i_2} \to \lambda, \ldots, \overline{p}_{i_2} \to \lambda}_{\phi(p_{i_2},t)}, \ldots, \underbrace{\overline{p}_{i_k} \to \lambda, \cdots, \overline{p}_{i_k} \to \lambda}_{\phi(p_{i_k},t)}$$

$$\delta_{t,h} : A_{i_1} \to h(\alpha_{i_1}), A_{i_2} \to h(\alpha_{i_2}), \ldots, A_{i_k} \to h(\alpha_{i_k})$$

$$\delta_{t,B} : \overline{B}_1 \to B_1, \overline{B}_2 \to B_2, \ldots, \overline{B}_l \to B_l$$

and define the matrix

$$m_t = (\delta_{t,\lambda}, \delta_{t,h}, \delta_{t,B}, \delta_{t,X}). \tag{2}$$

where

$$\delta_{t,X} : X \to \overline{p}_1^{|\phi(t,p_1)|} \cdot \overline{p}_2^{|\phi(t,p_2)|} \cdots \overline{p}_s^{|\phi(t,p_s)|} \cdot X.$$

We also add the starting matrix

$$m_0 = (S' \to S \cdot \prod_{p \in P} \overline{p}^{|\iota(p)|} \cdot X) \tag{3}$$

According to types of the sets of final markings, we consider two cases of erasing rules:

Case $y = r$. Then

$$m_{p,\lambda} = (\overline{p} \to \lambda) \text{ for each } p \in P \text{ and } m_{X,\lambda} = (X \to \lambda). \tag{4}$$

Case $y = t$. For each $\mu \in M$,

$$m_{\mu,\lambda} = (\underbrace{\overline{p}_1 \to \lambda, \ldots, \overline{p}_1 \to \lambda}_{\mu(p_1)}, \ldots, \underbrace{\overline{p}_s \to \lambda, \ldots, \overline{p}_s \to \lambda}_{\mu(p_s)}, X \to \lambda). \tag{5}$$

We consider the matrix grammar $G' = (V', \Sigma, S', M)$ where M consists of all matrices (2) and (3) and matrices (4) for case $y = r$ or matrix (5) for case $y = t$.

Let

$$D : S \xRightarrow{\pi_1} w_1 \xRightarrow{\pi_2} w_2 \cdots \xRightarrow{\pi_d} w_d = w \in \Sigma^*$$

be a derivation in G. Then, $t_1 t_2 \cdots t_d$ where $\beta({}^\bullet t_i) = \pi_i, 1 \leqslant i \leqslant d$, is a successful occurrence sequence in N. We construct the derivation D' in the grammar G' simulating the derivation D as follows: we start the derivation D' by applying the matrix (3) and get

$$D' : S' \xRightarrow{m_0} S \prod_{p \in P} \overline{p}^{|\iota(p)|} X.$$

Then, for each transition t_i in the successful occurrence sequence $t_1 t_2 \cdots t_d$, we choose the matrix m_{t_i}, $1 \leqslant i \leqslant d$, in D':

$$D' : S' \xRightarrow{m_0} S \prod_{p \in P} \overline{p}^{|\iota(p)|} X \xRightarrow{m_{t_1}} w_1 z_1 X \xRightarrow{m_{t_2}} w_2 z_2 X \cdots \xRightarrow{m_{t_d}} w_d z_d X = w z_d X$$

where $z_i \in \overline{P}^*, 1 \leqslant i \leqslant d$.

The rules $\delta_{t_i,h}$ and $\delta_{t_i,B}$, $1 \leqslant i \leqslant d$, simulate the rules in the multiset π_i whereas the homomorphism h controls that all rules in $\delta_{t_i,h}$ are applied only to w_{i-1}, $2 \leqslant i \leqslant d$.

By construction, the rules $\delta_{t_i,\lambda}$ and $\delta_{t_i,X}$, $1 \leqslant i \leqslant d$, simulate the numbers of tokens consumed and produced in the occurrence of transition t_i. The number of occurrences of each $\overline{p} \in \overline{P}$ in string z_i is the same as the number of tokens in place $p \in P$ after the occurrence of t_i. Moreover, the number of occurrences of $\overline{p} \in \overline{P}$ in string z_d and the number of tokens in place $p \in P$ in a final marking $\mu \in M$ are the same.

Further, to erase z_d and X, we use the matrices (4) or (5) depending on $y \in \{r, t\}$. Thus, $L(G') \subseteq L(G)$. Using the similar arguments in backward manner, one can show that the inverse inclusion also holds. □

With slight modification of the arguments of the proof of the lemma above, we can also show that

Lemma 3 *For $y \in \{r,t\}$, $p\mathbf{PN}^{[\lambda]}(\lambda, y) \subseteq \mathbf{MAT}^{\lambda}$.*

Next, we show that every matrix language can be generated by (f,t)- and (f,r)-pPN controlled grammars.

Lemma 4 *For $y \in \{r,t\}$, $\mathbf{MAT}^{[\lambda]} \subseteq p\mathbf{PN}^{[\lambda]}(f, y)$.*

Proof: Let $G = (V, \Sigma, S, M)$ be a matrix grammar with $M = \{m_1, m_2, \ldots, m_n\}$ where $m_i : (r_{i1}, r_{i2}, \ldots, r_{ik_i}), 1 \leqslant i \leqslant n, 1 \leqslant j \leqslant k_i$. We construct an (f,t)-place labeled Petri net controlled grammar $G' = (V \cup \{S_0\}, \Sigma, R \cup \{S_0 \to S\}, S_0, N, \beta, M)$ where the Petri net $N = (P, T, F, \phi, \iota)$, the place labeling function $\beta : P \to R \cup \{S_0 \to S\}$ and the final marking set M are defined as follows

- the sets of places, transitions and arcs:

$$
\begin{aligned}
P =& \{p_0\} \cup \{p_{ij} \mid 1 \leqslant i \leqslant n, 1 \leqslant j \leqslant k_i\},\\
T =& \{t_{0i} \mid 1 \leqslant i \leqslant n\} \cup \{t_{ij} \mid 1 \leqslant i \leqslant n, 1 \leqslant j \leqslant k_i\},\\
F =& \{(p_0, t_{0i}), (t_{0i}, p_{i1}), (p_{ik_i}, t_{ik_i}), (t_{ik_i}, p_0) \mid 1 \leqslant i \leqslant n\}\\
& \cup \{(p_{ij}, t_{ij}), (t_{ij}, p_{ij+1}) \mid 1 \leqslant i \leqslant n, 1 \leqslant j \leqslant k_{i-1}\};
\end{aligned}
$$

- the weight function: $\phi(x,y) = 1$ for all $(x,y) \in F$;
- the initial marking: $\iota(p_0) = 1$ and $\iota(p) = 0$ for all $p \in P - \{p_0\}$;
- the transition labeling function: $\beta(p_0) = S_0 \to S$ and $\beta(p_{ij}) = r_{ij}, 1 \leqslant i \leqslant n, 1 \leqslant j \leqslant k_i$;
- the final marking set: $M = \mathcal{R}(N, \iota)$.

Remark 1 By definition of the Petri net N, it is not difficult to see that $\mathcal{R}(N, \iota)$ is a finite set. Thus, the cases $y = r$ and $y = t$ coincide.

Let

$$w_1 \xrightarrow{r_{i1}} w_2 \xrightarrow{r_{i2}} \cdots \xrightarrow{r_{ik_i}} w_k, \tag{6}$$

where $m_i : (r_{i1}, r_{i2}, \cdots, r_{ik_i}) \in M$, be derivation steps of a successful derivation $S \overset{*}{\Rightarrow} w \in \Sigma^*$ in G. Then,

$$w_1 \xrightarrow{[r_{i1}]} w_2 \xrightarrow{[r_{i2}]} \cdots \xrightarrow{[r_{ik_i}]} w_k \tag{7}$$

simulates by (6) and $t_{0i}t_{i1}t_{i2} \cdots t_{ik_i}$ is a subsequence of a successful occurrence sequence $\nu \in \mathcal{R}(N, \iota)$. Thus, $L(G) \subseteq L(G')$. The inclusion $L(G') \subseteq L(G)$ can also be shown by backtracking the arguments above. □

From the lemmas above,

Theorem 2 *For $x \in \{f, -\lambda, \lambda\}$ and $y \in \{r,t\}$,*

$$\mathbf{MAT} \subseteq p\mathbf{PN}(x,y) \subseteq \mathbf{MAT}^{\lambda}, \ \textit{and} \ p\mathbf{PN}^{\lambda}(x,y) = \mathbf{MAT}^{\lambda}.$$

THE EFFECT OF LABELING STRATEGIES

In this section, we study the labeling effect to the computational power of pPN controlled grammars. The following lemma follows immediately from the definition of the languages determined by labeling functions.

Lemma 5 *For $y \in \{r,t\}$, $p\mathbf{PN}^{[\lambda]}(f, y) \subseteq p\mathbf{PN}^{[\lambda]}(-\lambda, y) \subseteq p\mathbf{PN}^{[\lambda]}(\lambda, y)$.*

Further, we prove that the reverse inclusions also hold.

Lemma 6 *For $y \in \{r,t\}$, $p\mathbf{PN}^{[\lambda]}(-\lambda, y) \subseteq p\mathbf{PN}^{[\lambda]}(f, y)$.*

Proof: Let $G = (V, \Sigma, R, S, N, \beta, M)$ be a $(-\lambda, y)$-pPN controlled grammar (with or without erasing rules) where $N = (P, T, F, \phi, \iota)$. Let $R = \{r_i : A_i \to \alpha_i \mid 1 \leqslant i \leqslant n\}$, and let

$$P^+ = \{p \in P \mid (p,t) \in F\} \text{ and } P^- = \{p \in P \mid (p,t) \notin F\}.$$

We set the following sets of places, transitions and arcs:

$$\begin{aligned}\overline{P} =& \{c_{p,t}, c'_{p,t} \mid (p,t) \in F\},\\ \overline{T} =& \{d_{p,t}, d'_{p,t} \mid (p,t) \in F\},\\ \overline{F} =& \{(p, d_{p,t}), (d_{p,t}, c_{p,t})(c_{p,t}, d'_{p,t}), (d'_{p,t}, c'_{p,t}), (c'_{p,t}, t) \mid (p,t) \in F\}.\end{aligned}$$

We also introduce the new nonterminals and productions for each pair $(p,t) \in F$:

$$\begin{aligned}\overline{V} =& \{A_p, A_{p,t} \mid (p,t) \in F\},\\ \overline{R} =& \{A \to A_p, A_p \to A_{p,t}, A_{p,t} \to \alpha \mid (p,t) \in F,\ \beta(p) = A \to \alpha \in R \text{ and } A_{p,t} \in \overline{V}\}.\end{aligned}$$

We define the weight function $\overline{\phi} : \overline{F} \to \mathbb{N}$ as follows:

$$\overline{\phi}(p, d_{p,t}) = \overline{\phi}(d_{p,t}, c_{p,t}) = \overline{\phi}(c_{p,t}, d'_{p,t}) = \overline{\phi}(d'_{p,t}, c'_{p,t}) = \overline{\phi}(c'_{p,t}, t) = \phi(p,t)$$

where $(p,t) \in F$.

Using the sets and function defined above, we construct an (f, y)-place-labeled Petri net controlled grammar $G' = (V', \Sigma, R', S, N', \beta', M')$ with

$$\begin{aligned}V' =& V \cup \overline{V},\\ R' =& (R - \{A \to \alpha \in R \mid \beta(p) = A \to \alpha \text{ and } (p,t) \in F\}) \cup \overline{R}.\end{aligned}$$

The set components of the Petri net $N' = (P', T', F', \phi', \iota')$ are defined as

- the sets of places, transitions and arcs:

$$P' = P \cup \overline{P},\ T' = T \cup \overline{T} \text{ and } F' = (F - \{(p,t)\} \in F) \cup \overline{F};$$

- the weight function $\phi' : F' \to \mathbb{N}$:

$$\phi'(x,y) = \begin{cases}\phi(x,y) & \text{if } (x,y) \in F - \{(p,t) \in F\},\\ \overline{\phi}(x,y) & \text{if } (x,y) \in \overline{F};\end{cases}$$

- the initial marking $\iota' : P' \to \mathbb{N}_0$:

$$\iota'(p) = \begin{cases}\iota(p) & \text{if } p \in P,\\ 0 & \text{if } p \in \overline{P};\end{cases}$$

- the place labeling function $\beta' : P' \to R'$:

$$\beta'(p) = \begin{cases}\beta(p) & \text{if } p \in P^-,\\ A \to A_p & \text{if } p \in P^+,\end{cases}$$

  and, for each $c_{p,t}$ and $c'_{p,t}$ in $\overline{P}$:

$$\beta'(c_{p,t}) = A_p \to A_{p,t} \text{ and } \beta'(c'_{p,t}) = A_{p,t} \to \alpha,$$

  where $\beta(p) = A \to \alpha \in R$;

- if $y = r$, then the final marking set M' is defined as $M' = \mathcal{R}(N', \iota')$, and if $y = t$, then for every $\mu \in M$, we set $\nu_\mu \in M'$ where
$$\nu_\mu(p) = \begin{cases} \mu(p) & \text{if } p \in P, \\ 0 & \text{if } p \in \overline{P}. \end{cases}$$

Let us now consider a successful derivation in G:

$$S \xRightarrow{E_1} w_1 \xRightarrow{E_2} w_2 \xRightarrow{E_3} \cdots \xRightarrow{E_n} w_n = w \in \Sigma^* \tag{8}$$

where $E_i = [r_{i_1}, r_{i_2}, ..., r_{i_{k_i}}] \subseteq R^{\oplus}$, $r_{i_j} : A_{i_j} \to \alpha_{i_j}$, with $\beta(p_{i_j}) = r_{i_j}$, $p_{i_j} \in P$, $1 \leqslant i \leqslant n$, $1 \leqslant j \leqslant k_i$. Let $P'_i = \{p_{i_j} \mid 1 \leqslant j \leqslant k_i\} \subseteq {}^\bullet t_i$ for some $t_i \in T$, $1 \leqslant i \leqslant n$ (t_i and t_j, $1 \leqslant i \neq j \leqslant n$ are not necessarily distinct). Hence, by definition,
$$\iota \xrightarrow{t_1 t_2 \cdots t_n} \mu, \ \mu \in M, \tag{9}$$
is the successful occurrence of transitions in N. Then, by definition of the set R' of the rules, each derivation step $w_{i-1} \xRightarrow{E_i} w_i$, $1 \leqslant i \leqslant n$, where $w_0 = S$, in (8) can be simulated with the following sequence of the derivation steps in the grammar G':

$$\begin{aligned} w_{i-1} &\xRightarrow{(A \to A_{i_1}) \cdot (A \to A_{i_2}) \cdots (A \to A_{i_{k_i}})} w'_{i-1} \\ &\xRightarrow{(A_{i_1} \to A_{i_1, t_i}) \cdot (A_{i_2} \to A_{i_2, t_i}) \cdots (A_{i_{k_i}} \to A_{i_{k_i}, t_i})} w''_{i-1} \\ &\xRightarrow{(A_{i_1, t_i} \to \alpha_{i_1}) \cdot (A_{i_2, t_i} \to \alpha_{i_2}) \cdots (A_{i_{k_i}, t_i} \to \alpha_{i_{k_i}})} w_i. \end{aligned} \tag{10}$$

Correspondingly, by construction of the Petri net N', each transition t_i, $1 \leqslant i \leqslant n$, in (9) is extended with the occurrence sequence
$$d_{i_1, t_i} d_{i_2, t_i} \cdots d_{i_{k_i}, t_i} \cdot d'_{i_1, t_i} d'_{i_2, t_i} \cdots d'_{i_{k_i}, t_i} t_i \tag{11}$$
where
$${}^\bullet d_{i_j, t_i} = p_{i_j}, \ d^\bullet_{i_j, t_i} = {}^\bullet d'_{i_j, t_i} = \{c_{i_j, t_i}\} \ \text{ and } \ d'^\bullet_{i_j, t_i} = \{c'_{i_j, t_i}\} \subseteq t_i.$$
for all $1 \leqslant i \leqslant n$, $1 \leqslant j \leqslant k_i$. Thus, $L(G) \subseteq L(G')$.

Consider some successful derivation
$$S \Rightarrow^* w, \ w \in \Sigma^* \tag{12}$$
in the grammar G' with
$$\iota' \xrightarrow{\cdots\ t\ \cdots} \mu, \ \mu \in M' \tag{13}$$
where $t \in T$. By construction of N', in order to enable the transition t, the transition $d'_{p,t} \in {}^\bullet c'_{p,t}$, for each $c'_{p,t} \in {}^\bullet t$ and the transition $d_{p,t} \in {}^\bullet c_{p,t}$, for each $c_{p,t} \in {}^\bullet({}^\bullet t)$ must be fired. Thus, if ${}^\bullet t = \{c'_{p_1,t}, c'_{p_2,t}, \ldots, c'_{p_k,t}\}$, then, (13) will contain all the transitions

$$d_{p_1,t}, d_{p_2,t}, \ldots, d_{p_k,t}, d'_{p_1,t}, d'_{p_2,t}, \ldots, d'_{p_k,t}. \tag{14}$$

Accordingly, (12) contains the rules

$$A_i \to A_{p_i}, A_{p_i} \to A_{p_i,t}, A_{p_i,t} \to \alpha_i, \tag{15}$$

where $\beta(p_i) = A_i \to \alpha_i$, $1 \leqslant i \leqslant k$. Without loss of generality, we can rearrange the order of the occurrence of the transitions in (14) and correspondingly, the order of the application of the rules in (15), and as the result, we construct the occurrence steps and the derivation steps similar to (11) and (10), respectively. Thus, the transitions (14) can be replaced with t in the grammar G and the rules (15) can be replaced with the rules $A_i \to \alpha_i$, $1 \leqslant i \leqslant k$, which results in $L(G') \subseteq L(G)$. □

Lemma 7 *For* $y \in \{r, t\}$, $p\mathbf{PN}^{[\lambda]}(\lambda, y) \subseteq p\mathbf{PN}^{\lambda}(-\lambda, y)$.

Proof: Let $G = (V, \Sigma, R, S, N, \beta, M)$ be a (λ, y)-pPN controlled grammar (with or without erasing rules). Let

$$P_\lambda = \{p \mid \beta(p) = \lambda\} \text{ and } P_S = \{p \mid \beta(p) = S \to \alpha \in R\}.$$

We define $(-\lambda, y)$-pPN controlled grammar

$$G' = (V \cup \{S_0, X\}, \Sigma, S_0, R \cup \{S_0 \to SX, X \to X, X \to \lambda\}, N', \beta', M')$$

where $N' = (P \cup \{p_0, p_\lambda\}, T \cup \{t_0, t_\lambda\}, F', \phi', \iota')$ with the set of arcs

$$F' = F \cup \{(p_0, t_0), (t_0, p_\lambda), (p_\lambda, t_\lambda)\} \cup \{(t_0, p) \mid \beta(p) = S \to \alpha \in R\},$$

the weight function

$$\phi'(x, y) = \begin{cases} \phi(x, y) & \text{if } (x, y) \in F, \\ 1 & \text{if } (x, y) \in \{(p_0, t_0), (t_0, p_\lambda), (p_\lambda, t_\lambda)\}, \\ \iota(p) & \text{if } (x, y) = (t_0, p),\ p \in P_S, \end{cases}$$

and the initial marking

$$\iota'(x, y) = \begin{cases} 1 & \text{if } p = p_0, \\ 0 & \text{if } p \in P_S, \\ \iota(p) & \text{if } p \in P - P_S. \end{cases}$$

The place labeling function β is modified as

$$\beta'(p) = \begin{cases} \beta(p) & \text{if } p \notin P_\lambda, \\ X \to X & \text{if } p \in P_\lambda, \\ X \to \lambda & \text{if } p = p_\lambda. \end{cases}$$

Lastly, when $y = t$, for each final marking $\mu \in M$, we set $\nu_\mu \in M'$ as

$$\nu_\mu(p) = \begin{cases} \mu(p) & \text{if } P, \\ 0 & \text{if } p \in \{p_0, p_\lambda\}. \end{cases}$$

Further, it is not difficult to see that $L(G) = L(G')$. □

The following theorem summarizes the results obtained above.

Theorem 3

$$p\mathbf{PN}(f, y) = p\mathbf{PN}(-\lambda, y) \subseteq p\mathbf{PN}(\lambda, y) \subseteq p\mathbf{PN}^{\lambda}(f, y) = p\mathbf{PN}^{\lambda}(-\lambda, y) = p\mathbf{PN}^{\lambda}(\lambda, y).$$

By combining the results in Theorems 1, 2 and 3, we obtain the hierarchy of the family of languages generated by place-labeled Petri net controlled grammars:

Theorem 4 *The relations in Figure 3 hold; the lines (arrows) denote inclusions (proper inclusions) of the lower families into the upper families.*

STRUCTURAL PROPERTIES

In this section, we investigate structural properties of place labeled Petri net controlled grammars.

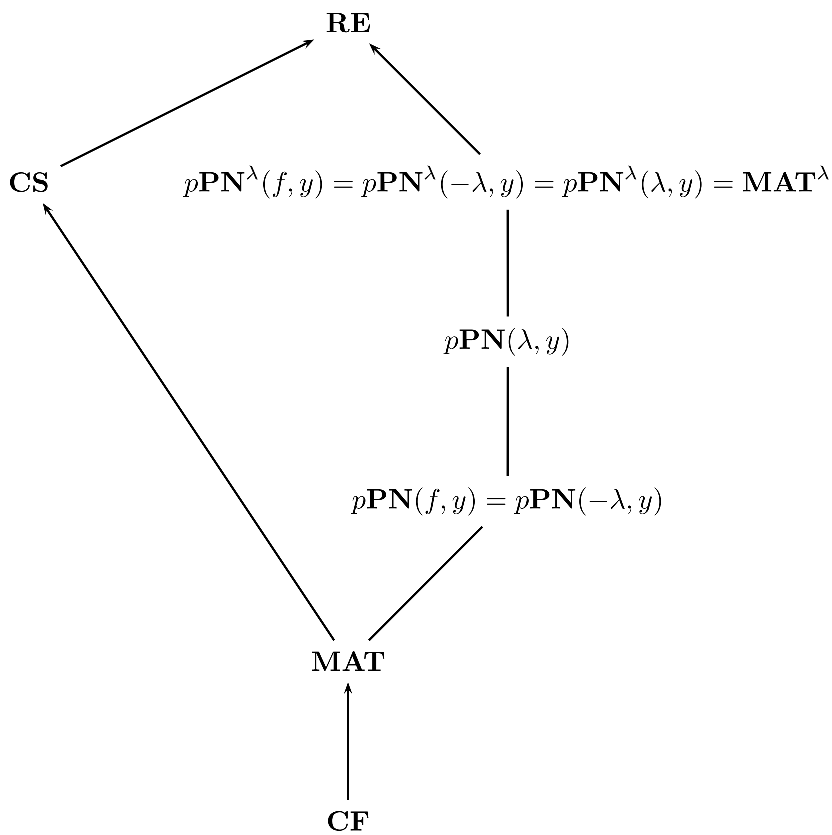

Fig. 3 The hierarchy of the family of languages generated by place-labeled Petri net controlled grammars

A single start place

Definition 7 Let $G = (V, \Sigma, R, S, N, \beta, M)$ with $N = (P, T, F, \phi, \iota)$ be an (x, y)-pPN controlled grammar where $x \in \{f, -\lambda, \lambda\}$ and $y \in \{r, t\}$. We say that N has a single start place p_0 if $\iota(p_0) = 1$ and $\iota(p) = 0$ for all $p \in P - \{p_0\}$.

Lemma 8 *For every (x, y)-place-labeled PN controlled grammar $G = (V, \Sigma, R, S, N, \beta, M)$ with a Petri net $N = (P, T, F, \phi, \iota)$, where $x \in \{f, -\lambda, \lambda\}$ and $y \in \{r, t\}$, there exists an equivalent (x, y)-pPN controlled grammar $G' = (V', \Sigma, R', S', N', \beta', M')$ such that the Petri net $N' = (P', T', F', \phi', \iota')$ has a single start place.*

Proof: Let $G = (V, \Sigma, S, R, B, \beta, M)$ is a (x, y)-pPN controlled grammar (with or without erasing rules). We introduce a new place p_0, a new transition t_0 and new arcs

$$\overline{F} = \{(p_0, t_0)\} \cup \{(t_0, p) \mid p \in P, \iota(p) > 0\}$$

and define the (x, y)-pPN controlled grammar $G' = (V \cup \{S_0\}, \Sigma, S_0, R \cup \{S_0 \to S\}, N', \beta', M')$ with the Petri net $N' = (P \cup \{p_0\}, T \cup \{t_0\}, F \cup \overline{F}, \phi', \iota)$, where

- the weight function $\phi' : F \cup \overline{F} \to \mathbb{N}$:

$$\phi'(x, y) = \begin{cases} \phi(x, y) & \text{for all } (x, y) \in F, \\ \iota(p) & \text{for all } (x, y) \in \overline{F}; \end{cases}$$

- the initial marking $\iota' : P \cup \{p_0\} \to \{0, 1, 2, \ldots\}$:

$$\iota'(p) = \begin{cases} 1 & \text{if } p = p_0, \\ 0 & \text{if } p \in P. \end{cases}$$

Further,

- the place labeling function $\beta' : P \cup \{p_0\} \to R \cup \{S_0 \to S\}$ is defined as

$$\beta'(p) = \begin{cases} S_0 \to S & \text{if } p = p_0, \\ \beta(p) & \text{if } p \in P; \end{cases}$$

- for every $\mu \in M$, we set $\nu_\mu \in M'$ with $\nu_\mu(p_0) = 0$ and $\nu_\mu(p) = \mu(p), \mu \in M$ for all $p \in P$.

Then, it is not difficult to see that $L(G) = L(G')$. □

Removal of dead places

Definition 8 Let $N = (P, T, F, \phi, \iota)$ be a marked Petri net. A place $p \in P$ is said to be dead if $p^\bullet = \varnothing$.

Lemma 9 *For an (x, y)-pPN controlled grammar $G = (V, \Sigma, S, R, N, \beta, M)$, $x \in \{\lambda, -\lambda, f\}$ and $y \in \{r, t\}$, there exists an equivalent (x, y)-pPN controlled grammar $G' = (V, \Sigma, S, R, N', \beta', M')$ where N' is without dead places.*

Proof: Let $G = (V, \Sigma, R, S, N, \beta, M)$ be an (x, y)-place-labeled Petri net controlled grammar with $N = (P, T, F, \phi, \iota)$ where $x \in \{f, \lambda, -\lambda\}$ and $y \in \{r, t\}$. Let

$$P_\varnothing = \{p \in P \mid p^\bullet = \varnothing\} \text{ and } F_\varnothing = \{(t, p) \in F \mid p^\bullet = \varnothing\}.$$

We construct an (x, y)-place-labeled Petri net controlled grammar in normal form $G' = (V, \Sigma, S, R, N', \beta', M')$ where the Petri net N' is obtained from N by removing its dead places and the incoming arcs to these places, i.e., $N' = (P - P_\varnothing, T, F - F_\varnothing, \phi', \iota')$ where

$$\phi'(x, y) = \phi(x, y) \text{ for all } (x, y) \in F - F_\varnothing,$$

and

$$\iota'(p) = \iota(p) \text{ for all } p \in P - P_\varnothing.$$

We define the labeling function $\beta' : (P - P_\varnothing) \to R$ by setting

$$\beta'(p) = \beta(p) \text{ for all } p \in P - P_\varnothing.$$

For every $\mu \in M$, we set $\nu_\mu \in M'$ as

$$\nu_\mu(p) = \mu(p) \text{ for all } p \in P - P_\varnothing.$$

Let

$$\iota \xrightarrow{t_1 t_2 \cdots t_n} \mu,\ \mu \in M \tag{16}$$

be a successful occurrence sequence of transitions in N. Then, for any place $p \in {}^\bullet t_i$, $1 \leqslant i \leqslant n$, we have $p \notin P_\varnothing$. Thus, (16) is also successful occurrence sequence in N'. □

A reduction to ordinary nets

Here, we show that for each pPN controlled grammar we can construct an equivalent place-labeled ordinary net (pON) controlled grammar.

Lemma 10 *Let* $G = (V, \Sigma, R, S, N, \beta, M)$ *with* $N = (P, T, F, \phi, \iota)$ *be an* (x, y)*-pPN controlled grammar, where* $x \in \{f, -\lambda, \lambda\}$ *and* $y \in \{r, t\}$*. Then, there exists an equivalent* (λ, y)*-place labeled ordinary net controlled grammar* $G' = (V', \Sigma, R', S', N', \beta', M')$.

Proof: Let $G = (V, \Sigma, S, R, N, \beta, M)$ with $N = (P, T, F, \phi, \iota)$ be an (x, y)-pPN controlled grammar (with or without erasing rules) where $x \in \{f, -\lambda, \lambda\}$ and $y \in \{r, t\}$. We set

$$P^+ = \bigcup_{(p,t)\in F} \{b^i_{pt} \mid 1 \leqslant i \leqslant \phi(p, t)\},$$

$$P^- = \bigcup_{(t,p)\in F} \{b^i_{tp} \mid 1 \leqslant i \leqslant \phi(t, p)\},$$

$$T^+ = \bigcup_{(p,t)\in F} \{d^i_{pt} \mid 1 \leqslant i \leqslant \phi(p, t)\},$$

$$T^- = \bigcup_{(t,p)\in F} \{d^i_{tp} \mid 1 \leqslant i \leqslant \phi(t, p)\},$$

and

$$F^+ = \bigcup_{(p,t)\in F} \{(p, d^i_{pt}), (d^i_{pt}, b^i_{pt}), (b^i_{pt}, t) \mid 1 \leqslant i \leqslant \phi(p, t)\},$$

$$F^- = \bigcup_{(t,p)\in F} \{(t, b^i_{tp}), (b^i_{tp}, d^i_{tp}), (d^i_{tp}, p) \mid 1 \leqslant i \leqslant \phi(t, p)\}.$$

We define the (λ, y)-pPN controlled grammar $G' = (V, \Sigma, S, R, N', \beta', M')$ with the Petri net $N = (P', T', F', \phi', \iota')$ where

- the set of places, transitions and arcs are constructed as

$$P' = P \cup P^+ \cup P^-, \; T' = T \cup T^+ \cup T^-, \text{ and } \; F' = F^+ \cup F^-;$$

- the weight function $\phi' : F' \to \mathbb{N}$ is set as $\phi'(x, y) = 1$ for all $(x, y) \in F'$;
- the initial marking is defined as

$$\iota'(p) = \begin{cases} \iota(p) & \text{if } p \in P, \\ 0 & \text{otherwise.} \end{cases}$$

Further, we set

- we set the place labeling function $\beta' : P' \to R$ as $\beta'(b^1_{pt}) = \beta(p)$ for each $(p, t) \in F$ and $\beta'(p) = \lambda$ if $p \in P \cup P^- \cup (P^+ - \{b^1_{pt} \mid (p, t) \in F\}$, and
- define the final markings $\nu_\mu \in M'$ when $y = t$ as:

$$\nu_\mu(p) = \begin{cases} \mu(p) & \text{if } p \in P, \\ 0 & \text{otherwise.} \end{cases}$$

Further, one can easily show that $L(G) = L(G')$. □

CONCLUSION

In this paper, we defined place-labeled Petri net (pPN) controlled grammars, and investigated their computational power and some structural properties. We showed that

- pPN controlled grammars have at least the computational power of matrix grammars *without* erasing rules and at most the computational power of matrix grammars *with* erasing rules;
- the labeling strategies do not effect to the generative capacities of pPN controlled grammars with erasing rules. Though free- and lambda-free-pPN controlled grammars without erasing rules have the same computational power, the "lambda" case remains open;
- control Petri nets can be reduced to "canonical forms" without effecting to the generative capacity of pPN controlled grammars.

The strictness of the inclusions in Theorem 4 is an interesting topic for future research, since it may lead to the solution of a classical open problem $\mathbf{MAT} \overset{?}{\subset} \mathbf{MAT}^{\lambda}$.

Acknowledgements: The first author would like to thank Universiti Teknologi Malaysia for the UTM Zamalah Scholarship and International Islamic University Malaysia for the financial funding through Endownment B Fund **EDW B13-053-0983**. The second author is grateful to the Ministry of Education and Research Management Center, International Islamic University Malaysia for the financial funding through **FRGS13-066-0307**. The third and fourth authors would like to thank the Ministry of Education and Research Management Center, UTM for the financial funding through Research University Fund Vote No. **08H45**.

REFERENCES

1. Petri C (1962) Kommunication mit Automaten. PhD thesis, University of Bonn.
2. Ginzburg A, Yoeli M (1980) Vector addition systems and regular languages. *J Comput System Sci* **20**, 227–284.
3. Hack M (1975) Petri net languages. Computation Structures Group Memo, Project MAC 124, MIT.
4. Jantzen M (1979) On the hierarchy of Petri net languages. *RAIRO Theoretical Informatics* **13**, 19–30.
5. Jantzen M (1987) Language theory of Petri nets. In: Advances in Petri Nets, Springer, vol 254-I of *LNCS*, pp 397–412.
6. Jantzen M, Petersen H (1994) Cancellation in context-free languages: Enrichment by induction. *Theoretical Computer Science* **127**, 149–170.
7. Valk R, Vidal-Naquet G (1981) Petri nets and regular languages. *J Comput System Sci* **23**, 229–325.
8. Yen H (1996) On the regularity of petri net languages. *Information and Computation* **124**, 168–181.
9. Beek Mt, Kleijn H (2002) Petri net control for grammar systems. In: Formal and Natural Computing, Springer, vol 2300 of *LNCS*, pp 220–243.
10. Farwer B, Jantzen M, Kudlek M, Rölke H, Zetzsche G (2008) Petri net controlled finite automata. *Fundamenta Informaticae* **85**, 111–121.
11. Farwer B, Kudlek M, Rölke H (2006) Petri-net-controlled machine models. Tech Rep 274, FBI-Bericht.
12. Farwer B, Kudlek M, Rölke H (2007) Concurrent Turing machines. *Fundamenta Informaticae* **79**, 303–317.
13. Jantzen M, Kudlek M, Zetzsche G (2008) Language classes defined by concurrent finite automata. *Fundamenta Informaticae* **85**, 267–280.
14. Zetzsche G (2009) Erasing in petri net languages and matrix grammars. In: Proceedings of the 13th International Conference on Developments in Language Theory, DLT '09, Springer-Verlag, Berlin, Heidelberg, pp 490–501.
15. Zetzsche G (2011) A sufficient condition for erasing productions to be avoidable. In: Mauri G, Leporati A (eds) *Developments in Language Theory, 15th International Conference, DLT 2011, Milan, Italy, July 19-22, 2011. Proceedings*, Springer, vol 6795 of *LNCS*, pp 452–463.
16. Marek V, Češka M (2001) Petri nets and random-context grammars. In: Proc. of the 35th Spring Conference: Modelling and Simulation of Systems, MARQ Ostrava, Hardec nad Moravicí, pp 145–152.
17. Hauschildt D, Jantzen M (1994) Petri net algorithms in the theory of matrix grammars. *Acta Informatica* **31**, 719–728.
18. Dassow J, Turaev S (2008) k-Petri net controlled grammars. Pre-proceedings of Second International Conference on Language and Automata Theory and Applications. Report 36/08, Research Group on Mathematical Linguistics, Universitat Rovira i Virgili.

19. Dassow J, Turaev S (2008) k-Petri net controlled grammars. In: Martín-Vide C, Otto F, Fernau H (eds) *Language and Automata Theory and Applications. Second International Conference, LATA 2008. Revised Papers*, Springer, vol 5196 of *LNCS*, pp 209–220.
20. Dassow J, Turaev S (2008) Arbitrary Petri net controlled grammars. In: Bel-Enguix G, Jiménez-López M (eds) *Linguistics and Formal Languages. Second International Workshop on Non-Classical Formal Languages In Linguistics, Tarragona, Spain*, pp 27–39, iSBN 978-84-612-6451-3.
21. Dassow J, Turaev S (2009) Grammars controlled by special Petri nets. In: Dediu A, Ionescu AM, Martín-Vide C (eds) *Language and Automata Theory and Applications, Third International Conference, LATA 2009*, Springer, vol 5457 of *LNCS*, pp 326–337.
22. Dassow J, Turaev S (2009) Petri net controlled grammars: the power of labeling and final markings. *Romanian Jour of Information Science and Technology* **12**, 191–207.
23. Dassow J, Turaev S (2009) Petri net controlled grammars: the case of special Petri nets. *Journal of Universal Computer Science* **15**, 2808–2835.
24. Dassow J, Turaev S (2010) Petri net controlled grammars with a bounded number of additional places. *Acta Cybernetica* **19**, 609–634.
25. Selamat M, Turaev S (2010) Grammars controlled by petri nets with place capacities. In: 2010 International Conference on Computer Research and Development, pp 51–55.
26. Stiebe R, Turaev S (2009) Capacity bounded grammars and Petri nets. *EPTCS* **3**, 193–203.
27. Stiebe R, Turaev S (2009) Capacity bounded grammars and Petri nets. In: Dassow J, Pighizzini G, Truthe B (eds) *Eleventh International Workshop on Descriptional Complexity of Formal Systems, Magdeburg, Germany*, pp 247–258.
28. Stiebe R, Turaev S (2009) Capacity bounded grammars. *Journal of Automata Languages and Combinatorics* **15**, 175–194.
29. Turaev S (2006) Semi-matrix grammars. In: Second Doctoral Workshop on Mathematical and Engineering Methods in Computer Science, MEMICS 2006, Mikulov, Czechia, pp 245–252, iSBN 80-214-3287-X.
30. Turaev S (2007) Petri net controlled grammars. In: Third Doctoral Workshop on Mathematical and Engineering Methods in Computer Science, MEMICS 2007, Znojmo, Czechia, pp 233–240, iSBN 978-80-7355-077-6.
31. Turaev S, Krassovitskiy A, Othman M, Selamat M (2012) Parsing algorithms for regulated grammars. *Mathematical Models Methods in Applied Science* (to appear).
32. Turaev S, Krassovitskiy A, Othman M, Selamat M (2011) Parsing algorithms for grammars with regulated rewriting. In: Zaharim A, Sopian K, Mostorakis N, Mladenov V (eds) *Recent Researches in Applied Informatics and Remote Sensing. The 11th WSEAS International Conference on APPLIED COMPUTER SCIENCE*, pp 103–109.
33. Rozenberg G, Salomaa A (eds) (1997) *Handbook of formal languages*, Springer.
34. Reisig W, Rozenberg G (eds) (1998) *Lectures on Petri Nets I: Basic Models*, vol 1491 of *LNCS*, Springer, Berlin.
35. Dassow J, Păun G (1989) *Regulated rewriting in formal language theory*, Springer-Verlag, Berlin.