A new variant of Petri net controlled grammars

Nurhidaya Mohamad Jan, Sherzod Turaev, Wan Heng Fong, and Nor Haniza Sarmin

Citation: AIP Conference Proceedings **1682**, 040015 (2015); doi: 10.1063/1.4932488
View online: http://dx.doi.org/10.1063/1.4932488
View Table of Contents: http://scitation.aip.org/content/aip/proceeding/aipcp/1682?ver=pdfcov
Published by the AIP Publishing

A New Variant of Petri Net Controlled Grammars

Nurhidaya Mohamad Jan[a], Sherzod Turaev[b], Wan Heng Fong[c] and Nor Haniza 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
nurhidayamj@gmail.com, sherzod@iium.edu.my, fwh@utm.my, nhs@utm.my

Abstract. A Petri net controlled grammar is a Petri net with respect to a context-free grammar where the successful derivations of the grammar can be simulated using the occurrence sequences of the net. In this paper, we introduce a new variant of Petri net controlled grammars, called a *place-labeled Petri net controlled grammar*, which is a context-free grammar equipped 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 multisets of the rules which are the images of the sets of the input places of transitions in a successful occurrence sequence of the Petri net. We study the effect of the different labeling strategies to the computational power and establish lower and upper bounds for the generative capacity of place-labeled Petri net controlled grammars.

Keywords: Petri net, context-free grammar, Petri net controlled grammar, computational power.
PACS: 87.14.gk

INTRODUCTION

Petri nets are "dynamic" bipartite directed graphs with two sets of nodes, called *places* and *transitions* [1], which 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 suitable tools for studying the properties of formal languages. Petri nets have been widely used as language generating/accepting tools (for instance, see [2,3]), and as regulation mechanisms for grammar systems [4], automata [5], and grammars [6,7].

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. The language generated consists of all terminal strings that can be obtained by applying 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.

For instance, the papers [8,9] introduced *k-Petri net controlled grammars*, and studied their properties including generative power, closure properties, infinite hierarchies. A generalization of regularly controlled grammars was considered in [10,11]: 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. 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 were investigated in [12,13]. 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. Papers [14,15] studied grammars controlled by (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 context-free 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 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 enables us to consider parallel application of production rules in Petri net controlled grammars, which allows the development of 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 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 mainly study the effect of the place labeling strategies to the computational power, and establish the lower and upper bounds for the families of languages generated by pPN controlled grammars.

This paper is organized as follows: the first section gives necessary definitions and results from the theories of formal languages and Petri nets that are used in the sequel. The second section defines place-labeled Petri net controlled grammars and illustrates the concept in examples. In the third section, the lower and upper bounds for the families of languages generated by pPN controlled grammars are established. The effect of labelling strategies to the computational power of pPN controlled grammars is presented in the fourth section. The fifth section discusses the obtained results, open problems and possible topics for future research.

PRELIMINARIES

We assume that the reader is familiar with basic concepts of formal language theory and Petri nets. For details, the reader is referred to [2, 4, 7, 12, 13, 14, 16, 17]. In this section we only recall some notions, notations and results directly related to the current work.

Throughout the paper we use the following general notation. 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 $\emptyset$. 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 and 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 π: $\Sigma \to \mathbb{N}$. The alphabet Σ is called the *basic set* of the multiset π and the elements of Σ is called the *basic elements* of the 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 $G = (V, \Sigma, R, S)$ where V and Σ are disjoint finite sets of *nonterminal* and *terminal* symbols, respectively, $R \subseteq V \times (V \cup \Sigma)^*$ is a finite set of *(production) rules* and $S \in V$ is the *start* symbol. 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 $x \Rightarrow y$, if and only if 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 $\overset{\pi}{\Rightarrow}$ or $\overset{r_1r_2\cdots r_n}{\Rightarrow}$. The *language* generated by G is defined by $L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$.

A matrix grammar is a quadruple $G = (V, \Sigma, M, S)$ 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 $L(G) = \{w \in \Sigma^* \mid S \overset{\pi}{\Rightarrow} w \text{ and } \pi \in M^*\}$.

The families of languages generated by matrix grammars without erasing rules and by matrix grammars with erasing rules are denoted by MAT and MAT^{λ}, respectively.

Theorem 1 [13] $\mathrm{CF} \subset \mathrm{MAT} \subset \mathrm{CS}$ and $\mathrm{MAT} \subseteq \mathrm{MAT}^{\lambda} \subset \mathrm{RE}$
where CF, CS and 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.

A mapping μ: $P\to\mathbb{N}_0$ 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 symbols ${}^{\bullet}x=\{y\,|(y,x)\in F\}$ and $x^{\bullet}=\{y\,|(x,y)\in F\}$ are called *pre*- and *post-sets* of $x\in P\cup T$, respectively. For $X\subseteq P\cup T$, ${}^{\bullet}X=\bigcup_{x\in X}{}^{\bullet}x$ 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 transition $t\in T$ is enabled by a marking μ if and only if $\mu(p)\geq\phi(p,t)$ 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 $\mu'(p)=\mu(p)-\phi(p,t)+\phi(t,p)$ where $(p,t),(t,p)\in F$. We write $\mu\xrightarrow{t}\mu'$ to indicate that the firing of t in μ leads to μ'. A marking μ in which no transition is enabled is called *terminal*.

A finite sequence $t_1t_2\cdots t_k\in T^*$, 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 $\mu\xrightarrow{t_1t_2\cdots t_k}\mu'$ or $\mu\xrightarrow{\nu}\mu'$ where $\nu=t_1t_2\cdots t_k$. For each $1\leq i\leq k$, marking μ_i is called *reachable* from marking μ. The symbol $\mathrm{R}(N,\mu)$ 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* $N=(P,T,F,\phi,\iota,M)$ is a construct where (P,T,F,ϕ,ι) is a marked Petri net and $M\subseteq\mathrm{R}(N,\iota)$ is a 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.

DEFINITIONS

In this section, we introduce a new variant of Petri net controlled grammar known as place-labeled Petri net controlled grammar, and present some examples to illustrate their power.

Definition 1 (Place-Labeled Petri Net Controlled Grammar)
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 $(V,\,\Sigma,R,S)$ 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 *direct derivation* and a *successful derivation*.

Definition 2 (Direct Derivation)
The symbol $x \in (V \cup \Sigma)^*$ *directly derives* $y \in (V \cup \Sigma)^*$ with a multiset $\pi = \left[A_{i_1} \to \alpha_{i_1}, A_{i_2} \to \alpha_{i_2}, \ldots, A_{i_k} \to \alpha_{i_k}\right]$ $\subseteq R^{\oplus}$ written as $x \overset{\pi}{\Rightarrow} y$, if and only if $x = x_1 A_{i_1} x_2 A_{i_2} \cdots x_k A_{i_k} x_{k+1}$ and $y = x_1 \alpha_{i_1} x_2 \alpha_{i_2} \cdots x_k \alpha_{i_k} x_{k+1}$ where

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

Definition 3 (Successful Derivation)
A derivation

$$S \overset{\pi_1}{\Rightarrow} w_1 \overset{\pi_2}{\Rightarrow} w_2 \overset{\pi_3}{\Rightarrow} \cdots \overset{\pi_n}{\Rightarrow} w_n = w \in \Sigma^*, \tag{1}$$

where $\pi_i \subseteq R^{\oplus}, 1 \leq i \leq n$, is called *successful* if and only if

1) $\pi_i = \beta_{-\lambda}({}^{\bullet}t_i)$ for some $t_i \in T, 1 \leq i \leq n$,
2) $t_1 t_2 \cdots t_n \in T^*$ is a successful occurrence sequence in N.

For short, (1) can be written as $S \overset{\pi_1 \pi_2 \cdots \pi_n}{\Rightarrow} w$.

Definition 4 (Language Generated by Grammar *G*)
The *language* $L(G)$ *generated by a grammar* G consists of strings $w \in \Sigma^*$ such that there is a successful derivation $S \overset{\pi_1 \pi_2 \cdots \pi_n}{\Rightarrow} w$ 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 (Language Determined by Labeling Function)
A *p*PN controlled grammar $G = (V, \Sigma, R, S, N, \beta, M)$ 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 based on the labeling function β.

Definition 6 (Language Determined by a Set of Final Markings)
A *p*PN controlled grammar $G = (V, \Sigma, R, S, N, \beta, M)$ is called

- *r-type* if *M* is the set of all reachable markings from the initial marking ι, i.e. $M = \mathrm{R}(N, \iota)$,
- *t-type* if $M \subseteq \mathrm{R}(N, \iota)$ 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 $p\mathrm{PN}(x, y)$ and $p\mathrm{PN}^{\lambda}(x, y)$ the families of languages generated by $(x, y) - p$PN controlled grammars without and with erasing rules, respectively, where $x \in \{f, -\lambda, \lambda\}$ and $y \in \{r, t\}$.

We also use bracket notation $p\mathrm{PN}^{[\lambda]}(x, y), x \in \{f, -\lambda, \lambda\}, y \in \{r, t\}$, in order to say that a statement holds both in case with erasing rules and in case without erasing rules.

Further, we give two examples of pPN controlled grammars to demonstrate their generative capacities.

Example 1. Let G_1 be a context-free grammar with the rules:

$$S \to ABC,\ A \to aA,\ B \to bB,\ C \to cC,\ A \to a,\ B \to b,\ C \to c,$$

(the other components of the grammar can be seen from these rules). Figure 1 illustrates a Petri net N_1 with respect to G_1. Obviously, $L(G_1) = \{a^n b^n c^n \mid n \geq 1\} \in p\text{PN}(f,t)$.

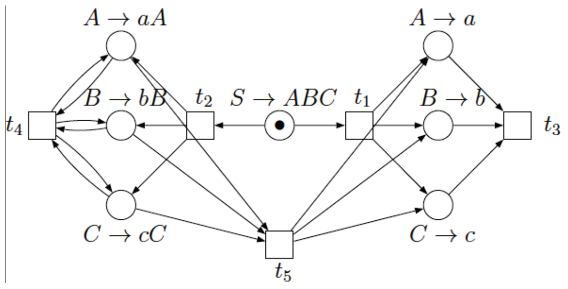

FIGURE 1. A Petri Net N_1.

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

$$S \to AB,\ A \to aA,\ B \to aB,\ A \to bA,\ B \to bB,\ A \to \lambda,\ B \to \lambda$$

(the other components of the grammar can be seen from these rules). Figure 2 illustrates a Petri net N_2 with respect to G_2. Obviously, $L_2 = \{ww \mid w \in \{a,b\}^*\} \in p\text{PN}(\lambda,t)$.

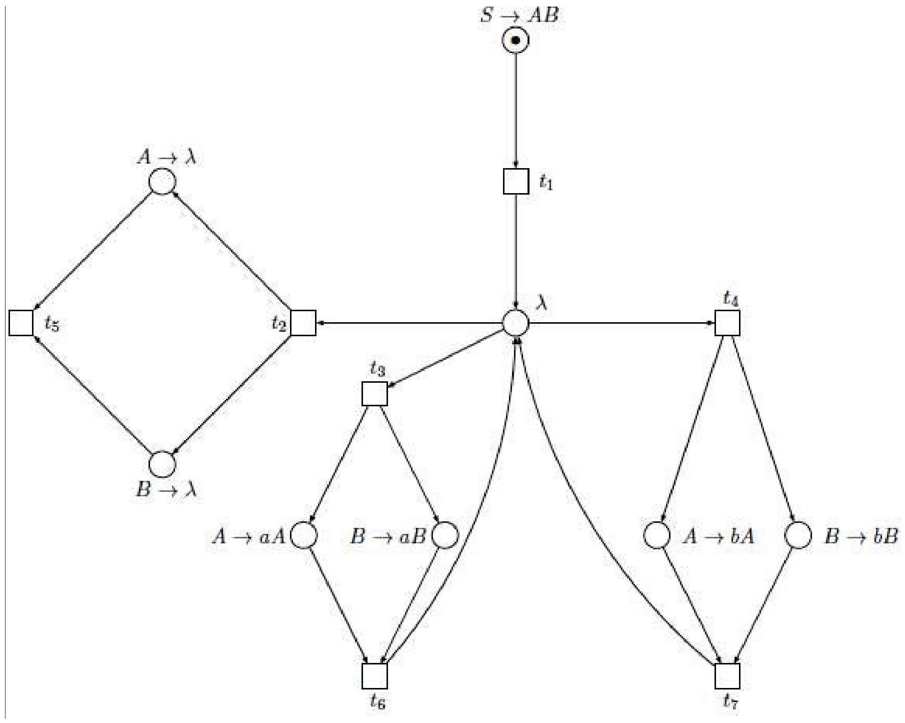

FIGURE 2. A Petri Net N_2.

LOWER AND UPPER BOUNDS

In this section, we establish lower and upper bounds for the generative capacity of pPN controlled grammars. The following inclusions are obvious.

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

Further, we show that context-free languages can be generated by $(f, t)-$ and $(f, r)-p$PN controlled grammars.

Lemma 2. For $y \in \{r, t\}$, $\mathrm{CF} \subseteq p\mathrm{PN}^{[\lambda]}(f, y)$.

Proof. Let $G = (V, \Sigma, R, S)$ be a context-free grammar with $R = \{r_i : A_i \to \alpha_i \mid 1 \le i \le n\}$. We construct $(f, t)-$place 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:

- $P = \{p_i \quad 0 \le i \le n\}$,
- $T = \{t_{ij} \quad 0 \le i, j \le n\}$,
- $F = \{(p_i, t_{ij}), (t_{ij}, p_j) \quad 0 \le i, j \le n\}$,
- $\phi(x, y) = 1$ for all $(x, y) \in F$,
- $\iota(p_0) = 1$ and $\iota(p) = 0$ for all $p \in P - \{p_0\}$,
- $\beta(p_0) = S_0 \to S$ and $\beta(p_i) = A_i \to \alpha_i, 1 \le i \le n$,
- $M = \mathrm{R}\,(N, \iota)$.

Remark: By definition of the Petri net N, it is not difficult to see that $\mathrm{R}\,(N, \iota)$ is a finite set. Indeed, in the occurrence of any transition $t \in T$, only one place $p \in P$ has a token, and the other places have no tokens. Thus, $|\,\mathrm{R}\,(N, \iota)| = n + 1$.

Next, we show that $L(G) = L(G')$. Let

$$S \overset{r_{i_1}}{\Rightarrow} w_1 \overset{r_{i_2}}{\Rightarrow} w_2 \overset{r_{i_3}}{\Rightarrow} \cdots \overset{r_{i_n}}{\Rightarrow} w_m = w \in \Sigma^* \tag{2}$$

be a derivation in G. We can also construct the derivation

$$S_0 \overset{[r_0]}{\Rightarrow} S \overset{[r_{i_1}]}{\Rightarrow} w_1 \overset{[r_{i_2}]}{\Rightarrow} w_2 \overset{[r_{i_3}]}{\Rightarrow} \cdots \overset{[r_{i_n}]}{\Rightarrow} w_m = w \in \Sigma^*, \tag{3}$$

which simulates (2), and it is not difficult to see that the occurrence sequence corresponding to the derivation (3),

$$t_{0i_1} t_{i_1 i_2} t_{i_2 i_3} \cdots t_{i_{n-1} i_n} t_{i_n i_{n+1}}, \tag{4}$$

where $\bullet t_{0i_1} = \{r_0\}$ and $\bullet t_{i_j i_{j+1}} = \{r_{i_j}\}, 1 \le j \le n$, starts at the initial marking ι and ends at a final marking $\mu \in M$. Thus, $L(G) \subseteq L(G')$.

The inclusion $L(G') \subseteq L(G)$ can also be shown by backtracking the arguments above. Hence, $\mathrm{CF}^{[\lambda]} \subseteq p\mathrm{PN}^{[\lambda]}(f, y)$ where $y \in \{t, r\}$. □

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

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

Proof. Let $G = (V, \Sigma, R, S, N, \beta, M)$ be an $(f, y)-p$PN 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 $T_\varnothing=\{t\in T\mid {}^\bullet t=\varnothing\}$. Then, let $T-T_\varnothing=\{t_1,t_2,\dots,t_n\}$. We define the sets of new nonterminals as $\overline{P}=\{\overline{p}\mid p\in P\}$ and $\overline{V}=\{\overline{A}\mid A\in V\}$.

We set the homomorphism $h:(V\cup\Sigma)^*\to(\overline{V}\cup\Sigma^*)$ as $h(a)=a$ for all $a\in\Sigma$, and $h(A)=\overline{A}$ for all $A\in V$. Consider $t\in T-T_\varnothing$, and let ${}^\bullet t=\{p_{i_1},p_{i_2},\dots,p_{i_k}\}$. We assume that $\beta(p_{i_j})=A_{i_j}\to\alpha_{i_j}\in R, 1\le j\le k$. Let $h(\alpha_{i_1}\alpha_{i_2}\cdots\alpha_{i_k})=x_1\overline{B_1}x_2\overline{B_2}\cdots x_l\overline{B_l}x_{l+1}$ where $x_i\in\Sigma^*, 1\le i\le l+1$ and $\overline{B_j}\in\overline{V}, 1\le j\le l.$

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

$$\begin{aligned}
&\delta_{t,\lambda}:\underbrace{\overline{p_{i_1}}\to\lambda,\dots,\overline{p_{i_1}}\to\lambda}_{\phi(p_{i_1},t)},\underbrace{\overline{p_{i_2}}\to\lambda,\dots,\overline{p_{i_2}}\to\lambda}_{\phi(p_{i_2},t)},\dots,\underbrace{\overline{p_{i_k}}\to\lambda,\dots,\overline{p_{i_k}}\to\lambda}_{\phi(p_{i_k},t)},\\
&\delta_{t,h}:A_{i_1}\to h(\alpha_{i_1}),A_{i_2}\to h(\alpha_{i_2}),\dots,A_{i_k}\to h(\alpha_{i_k}),\\
&\delta_{t,B}:\overline{B_1}\to B_1,\overline{B_2}\to B_2,\dots,\overline{B_l}\to B_l,
\end{aligned}$$

and define the matrix

$$m_t=(\delta_{t,\lambda},\delta_{t,h},\delta_{t,B},\delta_{t,X}) \tag{5}$$

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

$$m_0=(S'\to S\prod_{p\in P}\overline{p}^{|\iota(p)|}X). \tag{6}$$

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

Case $y=r$. Then

$$m_{p,\lambda}=(\overline{p}\to\lambda)\ \text{ for each }\ p\in P \text{ and } m_{X,\lambda}=(X\to\lambda). \tag{7}$$

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

$$m_{\mu,\lambda}=(\underbrace{\overline{p_1}\to\lambda,\dots,\overline{p_1}\to\lambda}_{\mu(p_1)},\dots,\underbrace{\overline{p_s}\to\lambda,\dots,\overline{p_s}\to\lambda}_{\mu(p_s)},X\to\lambda). \tag{8}$$

We consider the matrix grammar $G'=(V',\Sigma,S',M)$ where M consists of all matrixes of (5) and (6) and matrix (7) for case $y=r$ or matrix (8) for case $y=t$.

Let $D:S\overset{\pi_1}{\Rightarrow}w_1\overset{\pi_2}{\Rightarrow}w_2\cdots\overset{\pi_d}{\Rightarrow}w_d=w\in\Sigma^*$ be a derivation in G. Then, $t_1t_2\cdots t_d$ where $\beta({}^\bullet t_i)=\pi_i, 1\le i\le d,$ 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 (6) and get

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

Then, for each transition t_i in the successful occurrence sequence $t_1t_2\cdots t_d$, we choose the matrix $m_{t_i}, 1\le i\le d,$ in D':

$$D':S'\overset{m_0}{\Rightarrow}S\prod_{p\in P}\overline{p}^{|\iota(p)|}X\overset{m_{t_1}}{\Rightarrow}w_1z_1X\overset{m_{t_2}}{\Rightarrow}w_2z_2X\cdots\overset{m_{t_d}}{\Rightarrow}w_dz_dX=wz_dX$$

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

The rules $\delta_{t_i,h}$ and $\delta_{t_i,B}, 1\le i\le d,$ 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 $w_{i-1}, 2\le i\le d.$

By construction, the rules $\delta_{t_i,\lambda}$ and $\delta_{t_i,X}, 1\le i\le d,$ simulate the number 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 (7) or (8) 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. Let $D': S' \overset{m_0}{\Rightarrow} S\prod_{p\in P}\overline{p}^{|\iota(p)|}X \overset{m_{t_1}}{\Rightarrow} w_1 z_1 X \overset{m_{t_2}}{\Rightarrow} w_2 z_2 X \cdots \overset{m_{t_d}}{\Rightarrow} w_d z_d X = wz_d X$ be a derivation in G'. Then we simulate the derivation of G, $D: S \Rightarrow w_1 \Rightarrow w_2 \cdots \Rightarrow w_n = w$ by applying $w_i z_i X \overset{m_t}{\Rightarrow} w_{i+1} z_{i+1} X$ in D' where the production rules of G have the same form of m_t in (5). □

From Example 1, it follows that the inclusion represented in Lemma 2 is strict. On the other hand, from Lemma 3, *p*PN controlled grammars can generate at most matrix languages. Thus, we have the lower and upper bounds for *p*PN controlled grammars:

Theorem 2. For $x \in \{f, -\lambda, \lambda\}$ and $y \in \{r, t\}$, $\mathrm{CF} \subset p\mathrm{PN}^{[\lambda]}(x, y) \subseteq \mathrm{MAT}^{\lambda}$.

THE EFFECT OF LABELING STRATEGIES

The following lemma follows immediately from the definition of the labeling functions.

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

We now prove that the reverse inclusions also hold.

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

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

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

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

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

- $\overline{V} = \{A_p, A_{p,t} \mid (p,t) \in F, p \in P, t \in T\}$,
- $\overline{R} = \{A \to A_p, A_p \to A_{p,t}, A_{p,t} \to \alpha \mid (p,t) \in F, p \in P, t \in T, \beta(p) = A \to \alpha \in R$ and $A_{p,t} \in \overline{V}\}$.

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 $(p,t) \in F, p \in P, t \in T$.

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

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

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

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

The weight function $\phi' : F' \to \mathbb{N}$ is constructed as:

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

The initial marking $\iota' : P' \to \mathbb{N}_0$ is defined as:

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

Next, we set 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 $\overline{P}$, we set

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

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

If $y = r$, then the final marking set M' is defined as $M' = \mathrm{R}(N', \iota')$, 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:

$$S \overset{\pi_1}{\Rightarrow} w_1 \overset{\pi_2}{\Rightarrow} w_2 \cdots \overset{\pi_n}{\Rightarrow} w_n = w \in \Sigma^* \tag{9}$$

where $\pi_i = \langle r_{i_1}, r_{i_2}, \ldots, r_{i_{k_i}} \rangle, r_{i_j} : A_{i_j} \to \alpha_{i_j}$ and $\beta_{-\lambda}(p_{i_j}) = r_{i_j}, p_{i_j} \in P$, $1 \le i \le n$, $1 \le j \le k_i$.

Let $P'_i = \{p_{i_j} \mid 1 \le j \le k_i\} \subseteq {}^\bullet t_i$ for some $t_i \in T, 1 \le i \le n$ (t_i and $t_j, 1 \le i \ne j \le n$ are not necessarily distinct). Hence, by definition,

$$\iota \xrightarrow{t_1 t_2 \cdots t_n} \mu, \mu \in M, \tag{10}$$

is the successful occurrence of transitions in N.

Then, by definition of the set R' of the rules, each derivation step $w_{i-1} \overset{\pi_i}{\Rightarrow} w_i, 1 \le i \le n$, where $w_0 = S$, in (9) can be simulated with the following sequence of the derivation steps in the grammar G':

$$w_{i-1} \overset{(A \to A_{i_1}) \cdot (A \to A_{i_2}) \cdots (A \to A_{i_{k_i}})}{\Rightarrow} w'_{i-1} \overset{(A_{i_1} \to A_{i_1, t_i}) \cdot (A_{i_2} \to A_{i_2, t_i}) \cdots (A_{i_{k_i}} \to A_{i_{k_i}, t_i})}{\Rightarrow} w''_{i-1} \overset{(A_{i_1, t_i} \to \alpha_{i_1}) \cdot (A_{i_2, t_i} \to \alpha_{i_2}) \cdots (A_{i_{k_i}, t_i} \to \alpha_{i_{k_i}})}{\Rightarrow} w_i. \tag{11}$$

Correspondingly, by construction of the Petri net N', each transition $t_i, 1 \le i \le n$, in (10) is extended with the occurrence sequence

$$d_{i_1, t_i} d_{i_2, t_i} \cdots d_{i_{k_i}, t_i} d'_{i_1, t_i} d'_{i_2, t_i} \cdots d'_{i_{k_i}, t_i} t_i \tag{12}$$

where ${}^\bullet d_{i_j, t_i} = p_{i_j}$, $d^\bullet_{i_j, t_i} = {}^\bullet d'_{i_j, t_i} = \{c_{i_j, t_i}\}$ and $d'^\bullet_{i_j, t_i} = \{c'_{i_j, t_i}\} \subseteq t_i$ for all $1 \le i \le n, 1 \le j \le k_i$. Thus, $L(G) \subseteq L(G')$.

Consider some successful derivation

$$S \Rightarrow^* w, w \in \Sigma^* \tag{13}$$

in the grammar G' with

$$\iota' \xrightarrow{\cdots t \cdots} \mu, \tag{14}$$

where $\mu \in M', t \in T$. 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, (14) will contain all the transitions

$$d_{p_1, t}, d_{p_2, t}, \ldots, d_{p_k, t}, d'_{p_1, t}, d'_{p_2, t}, \ldots, d'_{p_k, t}. \tag{15}$$

Accordingly, (13) contains the rules

$$A_i \to A_{p_i}, A_{p_i} \to A_{p_i, t}, A_{p_i, t} \to \alpha_i, \tag{16}$$

where $\beta(p_i) = A_i \to \alpha_i, 1 \le i \le k$. Without loss of generality, we can rearrange the order of the occurrence of the transitions in (15) and correspondingly, the order of the application of the rules in (16), and as a result we construct the occurrence steps and the derivation steps similar to (12) and (11), respectively.

Thus, the transitions in (15) can be replaced with t in the grammar G and the rules in (16) can be replaced with the rules $A_i \to \alpha_i$, $1 \le i \le k$, which results in $L(G') \subseteq L(G)$. □

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

Proof. Let $G = (V, \Sigma, R, S, N, \beta, M)$ be a $(\lambda, y)-p$PN controlled grammar (with or without erasing rules). Let $P_\lambda = \{p \in P \mid \beta(p) = \lambda\}$ and $P_S = \{p \in P \mid \beta(p) = S \to \alpha \in R\}$.

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

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

where $N' = (P \cup \{p_0, p_\lambda\}, T \cup \{t_0, t_\lambda\}, F', \phi', \iota')$ where p_0, p_λ are new places and t_0, t_λ are new transitions. 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

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

For any derivation $S \Rightarrow w_1 \Rightarrow w_2 \cdots \Rightarrow w_n = w \in \Sigma^*$ in G, we construct the derivation

$$S_0 \Rightarrow SX^{i_1} \Rightarrow w_1 X^{i_2} \Rightarrow w_2 X^{i_3} \Rightarrow \cdots \Rightarrow w_n X^{i_{n+1}} \overset{*}{\Rightarrow} w \in \Sigma^*$$

in G' and vice versa. Hence, it is not difficult to see that $L(G) = L(G')$. □

The following theorem summarizes the results obtained in Teorem 2, Lemma 5 and Lemma 6.

Theorem 3. For $y \in \{r,t\}$,

$$CF \subset pPN(f,y) = pPN(-\lambda,y) \subseteq pPN(\lambda,y) \subseteq pPN^{\lambda}(f,y) = pPN^{\lambda}(-\lambda,y) = pPN^{\lambda}(\lambda,y) \subseteq MAT^{\lambda}.$$

CONCLUSION

We introduced a new variant of Petri net controlled grammars, called a place-labeled Petri net controlled grammar, by considering place labeling strategy instead of transition labeling strategy. We studied the effect of the labeling strategies to the computational power and established that pPN controlled grammars with erasing rules and different labeling strategies have the same power. The case without erasing rules has not been solved completely yet. We also showed that the family of context-free languages are strictly included in the family of languages generated by *p*PN controlled grammars, in its turn, it is included in the family of matrix languages with erasing rules. Further, we need to establish the tight bounds for the generative capacity of *p*PN controlled grammars.

ACKNOWLEDGMENTS

The first author would like to thank Universiti Teknologi Malaysia for the UTM Zamalah Scholarship. The second author is grateful to International Islamic University Malaysia and Ministry of Education for the financial funding through **FRGS13-066-0307** and Endownment B Fund **EDW B13-053-0983**. The third and fourth authors would also like to thank the Ministry of Education and Research Management Centre, UTM for the financial funding through Research University Fund Vote No. **08H45** and Fundamental Research Grant Scheme Vote No. **4F590**.

The authors would like to thank the reviewers for taking the time to read our paper and giving good and encouraging comments.

REFERENCES

1. A. Petri, "Kommunication mit Automaten", Ph.D. Thesis, University of Bonn, 1962.
2. M. Hack, "Petri net languages", Computation Structures Group Memo, Project MAC 124, MIT, Cambridge Mass, 1975.
3. M. Jantzen, *Rairo-Inf Theor* **13**(1), 19–30 (1979).
4. M. H. ter Beek and H. C. M. Kleijn, *Formal and Natural Computing*, Berlin: Springer-Verlag, 2002, pp. 220-243.
5. M. Jantzen, M. Kudlek and G. Zetzsche, *Fundam Inform* **85**, 1-4, 267–280 (2008).
6. V. Marek and M. Čěska, Proceeding of the 35th Spring Conference: Modelling and Simulation of Systems, 2001, pp. 145–152.
7. S. Turaev, In Third Doctoral Work-shop on Mathematical and Engineering Methods in Computer Science, MEMICS 2007, 2007, pp. 233–240.
8. J. Dassow, and S. Turaev, Proceedings of the Second International Conference on Language and Automata Theory and Applications 2008 (LATA2008), 2008, pp. 221-232.
9. J. Dassow and S. Turaev, *J Acta Cybernet* **19**, pp. 609-634 (2010).
10. J. Dassow and S. Turaev, Proceeding of the Second International Workshop on Non-Classical Formal Languages in Linguistics, pp. 27–39, 2008.
11. J. Dassow and S. Turaev, *Rom J Inf Sci Tech* **12**(2), pp. 91-207 (2009).
12. J. Dassow and S. Turaev, Proceedings of the Third International Conference on Language and Automata Theory and Applications 2009 (LATA2009), 2009. pp. 326–337.
13. J. Dassow and S. Turaev, *J Univers Comput Sci* **15**(14), pp. 2808–2835 (2009).
14. R. Stiebe and S. Turaev, *Electronic Proceedings of Theoretical Computer Science* (*EPTCS*) **3**, pp. 193-203 (2009).
15. R. Stiebe and S. Turaev, *J. Autom. Lang. Comb.***15**(1/2), pp. 175-194 (2010**).**
16. J. Dassow and Gh. Paun, *Regulated Rewriting in Formal Language Theory*, Berlin: Springer-Verlag, 1989, pp. 25-42.
17. P. Linz, *An Introduction to Formal Languages and Automata*, 3rd. ed., USA: Jones and Bartlett Publishers, Inc. 2001, pp. 15-24.