Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.18576/amis/100515

Context-Free Petri Net Controlled Grammars under Parallel Firing Strategy

Gairatzhan Mavlankulov[1,*], *Laula Zhumabayeva*[2], *Mohamed Othman*[1,*], *Tamara Zhukabayeva*[2], *Mohd Hasan Selamat*[1] *and Sherzod Turaev*[3]

[1] Department of Communication Technology and Network, Universiti Putra Malaysia, 43400 UPM Serdang, Selangor D.E., Malaysia
[2] Department of Information Technologies, L.N. Gumilyov Eurasian National University, 010008, Astana, Kazakhstan
[3] Department of Computer Science,International Islamic University Malaysia 53100 Gombak, Selangor D.E., Malaysia

Received: 12 Apr. 2016, Revised: 11 Jun. 2016, Accepted: 12 Jun. 2016
Published online: 1 Sep. 2016

Abstract: Petri nets are becoming one of the most important mathematical tools in Computer Science. In this paper we propose a new firing strategy in Petri Nets called ***a parallel firing strategy*** and study some mathematical properties of ***concurrent grammars*** which are controlled by Petri nets under parallel firing strategies. We propose some ***modes*** on this strategy and a notion of concurrent context-free grammar which is a similar to the context-free Petri nets under parallel firing strategy, where parallel firing modes of context-free Petri nets were converted to rule applications in context-free grammars. Moreover, we investigate some their properties.

Keywords: Petri nets, parallel firing, controlled grammars, parallel computing

1 Introduction

The rapid evolution of computing, communication, network, and sensor technologies has brought about the proliferation of computer-integrated systems, mostly technological and often highly complex. These systems, indispensable for our modern life, include air traffic control systems; automated manufacturing systems; computer and communication networks; embedded and networked systems; and software systems. The challenge for the researchers and engineers is the planning, modeling, analysis, verification, control, scheduling, and control implementation. Petri nets are increasingly becoming one of the most important mathematical tools to handle the above problems.

As Petri nets combine a well defined mathematical theory with a graphical representation of the dynamic behavior of systems, they have become a powerful modeling formalism in computer science, system engineering and many other disciplines. The theoretic outlook of Petri nets allows exact modeling and analysis of system behavior, while the graphical representation of Petri nets enable visualization of the modeled system state changes. This combination is the main reason for the great success of Petri nets. Hence, Petri nets have been used to model various kinds of dynamic event-driven systems such as computer networks [1], communication systems [2], manufacturing plants [3], command and control systems [4], real-time computing systems [5], logistic networks [6], and workflows [7] to mention only a few important examples. This wide spectrum of applications is accompanied by wide spectrum different aspects which have been considered in the research on Petri nets. One of the fundamental approaches in this area is to consider Petri nets as language generators. If the transitions in a Petri net are labeled with a set of symbols, a sequence of transition firing generates a string of symbols. The set of strings generated by all possible firing sequences defines a language called a Petri net language. With different kinds of labeling functions and different kinds of final marking sets, various classes of Petri net languages were introduced and investigated by Hack [8] and Peterson [9].

Recently in [10,11,12,13] different variants of a Petri net controlled grammar were introduced, which is a context-free grammar equipped with a Petri net, whose transitions are labeled with rules of the grammar or the empty string, and the associated language consists of all terminal strings which can be derived in the grammar. The sequence of rules in every terminal derivation corresponds to some occurrence sequence of transitions

* Corresponding author e-mail: gairatjon@gmail.com, mothman@upm.edu.my

of the Petri net which is enabled at the initial marking and finished at a final marking of the net. It can be considered as mathematical models for the study of concurrent systems appearing in systems biology and automated manufacturing systems. The distinguished feature of all of these variants is that the transitions of a Petri net fire sequentially. The concept of maximal parallelity in Petri nets was studied by Burkhard in [14]. Another different viewpoint on the parallel firing of transitions in Petri nets was taken by Farwer, Kudlek and Rolke in [15,16], where Turing Machines (called Concurrent Turing Machines) with Petri nets as a finite control were introduced. The variant of the Concurrent Finite Automation (CFA) has been defined and studied in [16]. In [17]were compared some modes of firing transitions in Petri nets and investigated classes of languages specified by them. In this paper we extend a new variant of theoretical models for parallel computation using Petri nets under parallel firing strategies, which were introduced in [18,19,**?**]. These grammars are called grammars controlled by Petri nets under parallel firing strategies (concurrent grammars), i.e. the transitions of a Petri net fire simultaneously in different modes. We investigate some properties of the concurrent context-free languages and compare with other known grammars. For instance, show some examples of concurrent context-free grammars which can generate non-context free languages. Noted, these languages can not be generated by Petri Net controlled grammars in the sequential case.

2 Preliminaries

2.1 Grammars and Languages

Let $\mathbb{N}$ be the set of all non-negative integers and $\mathbb{N}^k$ be the set of all vectors of k non-negative integers. The cardinality of a set X is denoted by $|X|$. Let Σ be an *alphabet* which is a finite nonempty set of symbols. A *string* over the alphabet Σ is a finite sequence of symbols from Σ. The *empty* string is denoted by λ. The set of all strings over the alphabet Σ is denoted by Σ^*. A subset of Σ^* is called a *language*. The *length* of a string w, denoted by $|w|$, is the number of occurrences of symbols in w. The number of occurrences of a symbol a in a string w is denoted by $|w|_a$.

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 $R \subseteq V \times (V \cup \Sigma)^*$ is a finite 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*. $x \in (V \cup \Sigma)^+$ *directly derives* $y \in (V \cup \Sigma)^*$, written as $x \Rightarrow y$, iff there is a rule $r = A \to \alpha \in R$ such that $x = x_1 A x_2$ and $y = x_1 \alpha x_2$. The rule $r : A \to \alpha \in R$ is said to be *applicable* in sentential form x, if $x = x_1 A x_2$, where $x_1, x_2 \in (V \cup \Sigma)^*$ The reflexive and transitive closure of $\Rightarrow$ is denoted by $\Rightarrow^*$. A derivation using the sequence of rules $\pi = r_1 r_2 \cdots r_n$ is denoted by $\Rightarrow\ \pi$ or $\Rightarrow\ r_1 r_2 \cdots r_n$. The *language* generated by G is defined by $L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$. The family of context-free languages is denoted by CF. Context-free grammar is called *linear* if all production rules have a form $R \subseteq V \times (\Sigma^* V \Sigma^* \cup \Sigma^*)$. The family of linear languages is denoted by LIN.

2.2 Multisets

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

$$\mu = (\mu(a_1)a_1, \mu(a_2)a_2, \ldots, \mu(a_n)a_n)$$

where $\mu(a_i)$, $1 \le i \le n$, is the multiplicity of a_i, or as a vector

$$\mu = (\mu(a_1), \mu(a_2), \ldots, \mu(a_n)),$$

or as the set in which each basic element $a \in \Sigma$ occurs $\mu(a)$ times

$$\mu = \{\underbrace{a_1, \ldots, a_1}_{\mu(a_1)}, \underbrace{a_2, \ldots, a_2}_{\mu(a_2)}, \ldots, \underbrace{a_n, \ldots, a_n}_{\mu(a_n)}\}.$$

The empty multiset is denoted by ε, that is $\varepsilon(a) = 0$ for all $a \in \Sigma$. The set of all multisets over Σ is denoted by $\Sigma^\oplus$. Since Σ is finite, $\Sigma^\oplus = \mathbb{N}^{|\Sigma|}$. The power (or cardinality) of a multiset $\mu = (\mu(a_1), \mu(a_2), \ldots, \mu(a_n))$ denoted by $|\mu|$, is $\sum_{i=1}^{n} \mu_i$. A multiset μ is a *set* if and only if $\mu(a) \le 1$ for all $a \in \Sigma$. For two multisets μ and ν over the same alphabet Σ, we define

–the *inclusion* $\mu \subseteq \nu$ by

$$\mu \subseteq \nu \text{ if and only if } \mu(a) \le \nu(a) \text{ for all } a \in \Sigma;$$

–the *sum* $\mu \oplus \nu$ by

$$(\mu \oplus \nu)(a) = \mu(a) + \nu(a) \text{ for each } a \in \Sigma,$$

and we denote the sum of multisets $\mu_1, \mu_2, \ldots, \mu_k$ by $\sum_{i=1}^{k} \mu_i$, i.e.,

$$\sum_{i=1}^{k} \mu_i = \mu_1 \oplus \mu_2 \oplus \cdots \oplus \mu_k;$$

–the *difference* $\mu \ominus \nu$ by

$$(\mu \ominus \nu)(a) = \max\{0, \mu(a) - \nu(a)\} \text{ for each } a \in \Sigma.$$

2.3 Petri nets

A Petri net is a triple (P,T,δ) where P and T are finite disjoint sets of *places* and *transitions*, respectively, and $\delta : T \to P^{\oplus} \times P^{\oplus}$ is a mapping which assigns to each transition $t \in T$ a pair $\delta(t) = (\alpha,\beta)$. Graphically, a Petri net is represented by a bipartite directed graph with the node set $P \cup T$ where places are drawn as *circles*, transitions as *boxes*. For each transition $t \in T$ with $\delta = (\alpha,\beta)$, the multiplicities $\alpha(p)$, $\beta(p)$ of a place $p \in P$, give the number of arcs from p to t and from t to p, respectively. A multiset $\mu \in P^{\oplus}$ is called a *marking*. For each $p \in P$, $\mu(p)$ gives the number of *tokens* in p. Graphically, tokens are drawn as small solid *dots* inside circles.

A *place/transition net* (*p/t net* for short) is a quadruple $N = (P,T,\delta,\mu_0)$ where (P,T,δ) is a Petri net, $\iota \in P^{\oplus}$ is the *initial marking*.

A transition $t \in T$ with $\delta(t) = (\alpha,\beta)$ is *enabled* at a marking $\mu \in P^{\oplus}$ if and only if $\alpha \sqsubseteq \mu$. In this case we say that t can *occur* (*fire*). Its occurrence transforms the marking μ into the marking $\mu' \in P^{\oplus}$ defined by $\mu' = \mu \ominus \alpha \oplus \beta$. 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 finite sequence $t_1t_2\cdots t_k$, $t_i \in T, 1 \le i \le k$, is called *an occurrence sequence* enabled at a marking μ and finished at a marking μ_k if there are markings $\mu_1,\mu_2,\ldots,\mu_{k-1}$ such that

$$\mu \xrightarrow{t_1} \mu_1 \xrightarrow{t_2} \ldots \xrightarrow{t_{k-1}} \mu_{k-1} \xrightarrow{t_k} \mu_k.$$

In short this sequence can be written as $\mu \xrightarrow{t_1t_2\cdots t_k} \mu_k$ or $\mu \xrightarrow{\nu} \mu_k$ where $\nu = t_1t_2\cdots t_k$. For each $1 \le i \le k$, marking μ_i is called *reachable* from marking μ. $\mathscr{R}(N,\mu) \subseteq P^{\oplus}$ denotes the set of all reachable markings from a marking μ.

Let $N = (P,T,\delta,\iota)$ be a p/t net and $F \subseteq \mathscr{R}(N,\iota)$ be a set of markings which are called *final markings*. An occurrence sequence ν of transitions is called *successful* for F if it is enabled at the initial marking ι and finished at a final marking τ of F. If F is understood from the context, we say that ν is a *successful occurrence sequence*.

A *labeled Petri net* is a tuple $K = (\Delta,N,\gamma,F)$ where Δ is an alphabet, $N = (P,T,\delta,\iota)$ is a p/t net, $\gamma : T \to \Delta \cup \{\lambda\}$ is a transition labeling function and $F \subseteq \mathscr{R}(N,\iota)$.

The labeling function γ is extended to occurrence sequences in natural way, i.e., if $\nu t \in T^*$ is an occurrence sequence then $\gamma(\nu t) = \gamma(\nu)\gamma(t)$ and $\gamma(\lambda) = \lambda$. For an occurrence sequence $\nu \in T^*$, $\gamma(\nu)$ is called a *label sequence*.

A *Petri net language* of K with respect to a transition labeling function γ and a final marking set F is defined by

$$L(K) = \{\gamma(\nu) \in \Delta^* \mid \iota \xrightarrow{\nu} \mu \text{ where } \nu \in T^* \text{ and } \mu \in F\}.$$

2.4 Context-Free Petri Nets

A context-free (cf) Petri net is a Petri net $N = (P,T,F,\phi,\beta,\gamma,\iota)$ where

• labeling function $\beta : P \to V$ and $\gamma : T \to R$ are bijections;
• there is an arc from place p to transition t if and only if $\gamma(t) = A \to \alpha$ and $\beta(p) = A$. The weight of the arc (p,t) is 1;
• there is an arc from transition t to place p if and only if $\gamma(t) = A \to \alpha$ and $\beta(p) = \chi$ where $|\alpha|_\chi > 0$. The weight of the arc (t,p) is $|\alpha|_\chi$;
• the initial marking ι is defined by $\iota(b^{-1}(S)) = 1$ and $\iota(p) = 0$ for all $p \in P - \{\beta^{-1}(S)\}$

*Example 1.*Let G_1 be a context-free grammar with the rules: $r_0 : S \to bSbb, r_1 : S \to A, r_2 : A \to aA, r_3 : A \to a$

(the other components of the grammar can be seen from these rules). Figure 1 illustrates a cf Petri net with respect to the grammar G_1. Obviously, $L(G_1) = \{b^n a^m b^{2n} \mid m \ge 1, n \ge 0\}$.

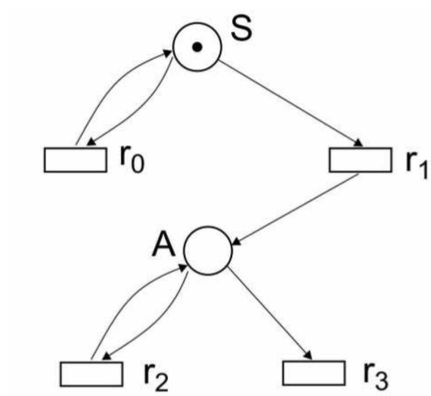

Figure 1. A Context-Free Petri net

2.5 Multisteps

Let $G = (V,\Sigma,S,R)$ be context-free grammar. $K = (\Delta,N,\gamma,F)$, $N = (P,T,\delta,\iota)$, be a labeled Petri net such that $\Delta = R$. Let $A = \{t_1,t_2,\ldots,t_k\} \subseteq T$ with $\delta(t_i) = (\alpha_i,\beta_i)$ for $1 \le i \le k$.

Definition 1.*The transitions of a multiset $\nu \in A^{\oplus}$ are simultaneously/parallelly enabled/firable at a marking $\mu \in \mathscr{R}(N,\iota)$ if and only if*

$$\sum_{i=1}^{k} \nu(t_i)\alpha_i \sqsubseteq \mu.$$

Then the transitions of ν *parallelly fire* resulting in the new marking μ' defined by

$$\mu' = \mu \ominus \sum_{i=1}^{k} \nu(t_i)\alpha_i \oplus \sum_{i=1}^{k} \nu(t_i)\beta_i.$$

A multiset ν whose transitions fire parallelly is called a *multistep*. We write $\mu \xrightarrow[m]{\nu} \mu'$ to denote that the a multistep ν at μ leads to μ'. Let $X = \{t_1, t_2, \ldots, t_k\} \subseteq T$ with $t_i = (\alpha_i, \beta_i)$, $1 \leq i \leq k$, and let a multistep $\nu \in X^{\oplus}$ be enabled at a marking $\mu \in P^{\oplus}$. We will define some special types(modes) of multisteps with respect to the basic sets and multisets.

1. The multistep ν is called in k *mode* if $|\nu| = k$. Similarly, ν is called in $\leq k$ mode ($\geq k$ mode) if $|\nu| \leq k (|\nu| \geq k)$.
2. Let $A \in V$. The multistep ν is called in *A-nonterminal labeled mode* if $T_A = \{t \in T : \gamma(t) = A \to \alpha$ for some $A \to \alpha \in$R$\}$.
3. Let $r \in R$. The multistep ν is called in *r-rule labeled mode* if $T_r = \{t \in T : \gamma(t) = r\}$
4. The multistep ν is called in *wide* mode if $\nu(t) > 0$ for all $t \in X$ and
   $X = T$ or
   for all $\nu' \in Y^{\oplus}$, where $X \subset Y \subseteq T$,
$$\sum_{t \in Y} \nu'(t)\alpha \not\subseteq \mu.,$$
5. The multistep ν is called in *global* mode if and only if for all $\eta \in X^{\oplus}$,
$$\sum_{i=1}^{k} \eta(t_i)\alpha_i \subseteq \mu \text{ imply } \eta = \nu.$$
6. The multistep ν is called a in *step* mode if ν is a set, i.e., $\nu \subseteq X$.

2.6 Parallel Firing Strategy in Context-Free Petri Nets

Definition 2. *Let $R' = \{r_1, r_2, \cdots r_n\} \subseteq R$, where $r_i = A_i \to \alpha_i (1 \leq i \leq n)$ are applicable rules in the sentential form x.*
Multiset $R'^{\oplus} = \{\rho(r_1)r_1, \rho(r_2)r_2 \cdots \rho(r_t)r_t\} (t \leq n)$ is called parallelly applicable in the sentential form x if x can be represented as $x = x_1 A_{i_1} x_2 A_{i_2} \cdots x_k A_{i_k} x_{k+1}$ where $\{A_{i_j}, 1 \leq j \leq k\} = \{\rho(r_1)A_1, \rho(r_2)A_2, \cdots \rho(r_t)A_t, t \leq k\}$. A set of all multisets of parallelly applicable rules in the sentential form x is denoted by $\Re'_{app(x)}$

Definition 3. *Let $x = x_1 A_1 x_2 A_2 \cdots x_m A_m x_{m+1}$ and $y = x_1 u_1 x_2 u_2 \cdots x_m u_m x_{m+1}$, where $x_i \in (V \cup \Sigma)^* (1 \leq i \leq m+1)$, $A_j \in V^*$, $u_j \in (V \cup \Sigma)^* (1 \leq j \leq m)$, and $\{r_i : r_i = A_i \to u_i, 1 \leq i \leq m\} \subseteq R$. Let $\nu \subseteq \Re'_{app(x)}$ is a multiset. We say that x directly derives y.*

(i) in *a multistep mode*, denoted by m, if a multiset $\nu \subseteq \Re'_{app(x)}$
(ii) in *a step mode*, denoted by s, if $\nu \subseteq R'$.
(iii) in k *mode*, denoted by mode k, if $|\nu| \leq k$.
(iv) *in a nonterminal labeled mode*, denoted by n, if $n \in \Re'_{app(x)}$ and $n = \{r : r = A_i \to u_i\}$, where $A_j = A_i$ for any $1 \leq j \leq m$;
(v) *in a rule labeled mode*, denoted by r, if $r \in \Re'_{app(x)}$ and $r = \{r : r = A_i \to u_i\}$, where $A_j = A_i$ and $u_j = u_i$ for any $1 \leq j \leq m$
(vi) *in a global mode*, denoted by g, if $g \in \Re'_{app(x)}$ and $g \cup r \notin \Re'_{app(x)}$ for any $r \in R'$
(vii) *in a wide mode*, denoted by w, if $w \in \Re'_{app(x)}$ and

- the multiset w consists all rules $r_i \in R'$
  or
- the multiset $(\rho \cup r_i) \notin \Re'_{app(x)}$ for any $r_i \in R' (\notin w)$
  and
  $\rho = \{\rho_1(r_1), \rho_2(r_2), \cdots \rho_t(r_t)\} \sqsubseteq w$, where $\rho_i(r_i) \geq 1$ for all $1 \leq i \leq t$

It is also of interest to consider some combined cases of these modes.
We denote by $ws, wg, wk, wn, ng, nk, rg, rk, kg$, respectively *wide step, wide global, wide k, wide nonterminal labeled, nonterminal labeled global, nonterminal labeled k, rule labeled global, rule labeled k* and *k global* modes.
Let $F = \{m, s, k, n, r, g, w, ws, wg, wk, wn, ng, nk, rg, rk, kg\}$.
We use a general notion $x \Rightarrow [\ f\ ] y$ if x *directly derives* y in f mode, where $f \in F$. The reflexive and transitive closure of $\Rightarrow [\ f\ ]$ is denoted by $\Rightarrow [\ f\ ]*$.

Definition 4. *A concurrent context-free grammar in f mode is a tuple $\mathscr{G} = (V, \Sigma, S, R, f)$ where $G = (V, \Sigma, S, R)$ is a context-free grammar and $f \in F$.*

Definition 5. *The language $L(\mathscr{G})$ generated by concurrent context-free grammar in f mode is defined by $L(\mathscr{G}) = \{w \in \Sigma^* \mid S \Rightarrow [\ f\ ] * w\}$.*

The family of languages generated by concurrent context-free grammars in f mode is denoted by fCF, where $f \in F$.

2.7 Results

In this section we investigate some properties of the concurrent context-free grammars. Based on the previous definitions and examples, the instance results are as follows:
Let $F = \{m, s, k, n, r, g, w, ws, wg, wk, wn, ng, nk, rg, rk, kg\}$ is set of modes.

Theorem 1. *$LIN = fLIN$, where $f \in F$.*

Proof.
By the definition, every rule of the linear grammar has a form $R \subseteq V \times (\Sigma^* V \Sigma^* \cup \Sigma^*)$, therefore in every derivation step a sentential form of the grammar has at

most one nonterminal. So all modes of the concurrent linear grammar has the same derivation step with linear grammar.

Theorem 2. *$CF = xCF$, where $x \in \{s,m,k\}$.*

Proof.
Now we show that $CF = sCF$
a) $CF \subseteq sCF$
We suppose $G = (V,\Sigma,S,R)$ is context free grammar and $L(G)$ is context free language. Let $G' = (V',\Sigma',S',R',s)$ concurrent context free grammar in s mode and $L(G')$ is concurrent context free language in s mode. Let $D \in G$ and $D' \in G'$ are derivations of corresponding grammars.
First, we show that any derivation $D \in G$ can be simulated by some derivation $D' \in G'$.
It follows directly from definitions of CF and sCF. Since only one single rule is used in every derivation step of D we can choose a derivation D' same as with derivation D.
Second, we show that any derivation $D' \in G'$ can also be simulated by some derivation $D \in G$.
Let $D' : S \Rightarrow s_1 D'_1 \Rightarrow s_2 D'_2 \Rightarrow s_3 D'_3 \dots \Rightarrow s_k D'_k = w(D')$, where $s_i \subseteq R = \{r_1, r_2, ..., r_n\}$. ($r_j \neq r_l$ for any $j \neq l, 1 \leq j,l \leq n$).
Let $s_i = \{s_i^1, s_i^2, \cdots s_i^{k_i}\} \subseteq R$,
where $s_i^j \in R$ $(1 \leq i \leq k, 1 \leq j \leq k_i)$.
We construct D from D' by changing each derivation step $\Rightarrow s_i D'_i \in D'$ to the sequence of derivation steps $\Rightarrow s_1^i D_{i_1} \Rightarrow s_2^i D_{i_2} \dots \Rightarrow s_k^i D_{i_k}$ in D.
b) the proof of the inclusion $sCF \subseteq CF$ is the similar to the proof of $CF \subseteq sCF$.

Theorem 3. *$rgCF - CF = \emptyset$.*

Proof.
Let $G_1 = (V,\Sigma,S,R,rg)$ is concurrent context-free grammar in rg mode , where R=$\{r_1 : S \to SS$, $r_2 : S \to a\}$ and $\Sigma = \{a\}$. It is clear, using the rule r_1 increases number of **S**'s two times in each derivation step.
$S \Rightarrow r_1 S^2 \Rightarrow r_1 S^4 \Rightarrow r_1 S^8 .... \Rightarrow r_1 S^{2^k}$.
Application of the r_2 rule in any step replaces all **S**'s with a's, consequently $S \Rightarrow *a^{2^k}$. Therefore $L(G_2) = \{a^{2^n} : n \geq 0\}$ which is not context-free.
Another example which shows $rgCF$ is not context free grammar is $G_2 = (V,\Sigma,S,R,rg)$, where $R = \{S \to AA, A \to aA, A \to a$, for all $a \in \Sigma\}$. It can be easily seen that the grammar generate the language $L(G_1) = \{ww : w \in \Sigma\}$ which is not context-free. For example, if $\Sigma = \{a,b\}$, the set of labeled rules
r_1: $S \to AA$
r_2: $A \to aA$
r_3: $A \to bA$
r_4: $A \to a$
r_5: $A \to b$.
For example, derivation steps for generating word **aaabaaab** would be
$S \Rightarrow r_1 AA \Rightarrow r_2 aAaA \Rightarrow r_2 aaAaaA \Rightarrow r_2 aaaAaaaA \Rightarrow r_5 aaabaaab$

3 Conclusion

We have proposed a new variant of theoretical models for parallel computation using Petri nets under parallel firing strategies, called grammars controlled by Petri nets under parallel firing strategies (i.e., concurrent grammars), which are natural formal models of concurrent, asynchronous, distributed, parallel, nondeterministic and stochastic systems. Various concurrent grammars were defined with respect to classes of Petri nets, firing modes, labeling strategies and final marking sets. We consider a context-free Petri net under parallel strategy and define parallel firing modes. Moreover we convert these firing modes to the rule application in context-free grammar and introduced a conception of the concurrent grammars. Some properties of the concurrent context-free grammars are also investigated.

References

[1] J. Wang. Charging information collection modeling and analysis of GPRS networks. Man and Cybernetics, Part C, IEEE Transactions on Systems, vol.37:473-481,2007.

[2] M. Ajmone Marsan, Gianfranco Balbo, Gianni Conte. Performance models of multiprocessor systems. MIT Press Cambridge, USA,1986.

[3] Alan A. Desrochers, Robert Y. Al-Jaar. Applications of Petri Nets in Manufacturing Systems: Modeling, Control, and Performance Analysis,IEEE, 1995.

[4] Andreadakis, Stamos K; Levis, Alexander H.Synthesis of distributed command and control for the outer air battle, MIT Press Cambridge, USA,1988.

[5] Andreadakis, Mandrioli, D., A. Morzenti, M. Pezze. A Petri net and logic approach to the specification and verification of real time systems,FormalMethods for Real Time Computing,1996.

[6] Rik Van Landeghem and Rik Van L and Carmen-veronica Bobeanu. Formal Modelling of Supply Chain: An Incremental Approach Using Petri Nets, 2002.

[7] Lin, Chuang, Liqin Tian and Yaya Wei. Performance equivalent analysis of workflow systems,Journal of Software,vol.13(8), 1472-1480, 2002.

[8] M. Hack. Petri net languages. Computation Structures Group Memo, Project MAC 124, MIT, Cambridge Mass., 1975.

[9] J.L. Peterson. Petri net theory and modeling of systems. Prentice-Hall, Englewood Clis, NJ, 1981.

[10] J. Dassow and S. Turaev. k-Petri net controlled grammars. Language and Automata Theory and Applications. Second International Conference, LATA 2008. Revised Papers, volume 5196 of LNCS, pages 209-220. Springer, 2008.

[11] J. Dassow and S. Turaev. Petri net controlled grammars: the power of labeling and final markings. Romanian Jour. of Information Science and Technology, 12(2):191-207, 2009.

[12] J. Dassow, G. Mavlankulov, M. Othman, S. Turaev, M.H. Selamat and R. Stiebe, Grammars Controlled by Petri Nets, In: P. Pawlewski (ed.) Petri nets, INTECH, 2012, ISBN 978-953-51-0700-2.

[13] S. Turaev. Petri net controlled grammars. In Third Doctoral Workshop on Mathematical and Engineering Methods in Computer Science, MEMICS 2007, pages 233-240, Znojmo, Czechia, 2007. ISBN 978-80-7355-077-6.
[14] Hans-Dieter Burkhard. Ordered Firing in petri nets. Information Processing and Cybernetcs, 2/3:71-86, 1989.
[15] B. Farwer, M. Jantzen, M. Kudlek, H. Rolke, and G. Zetzsche. Petri net controlled finite automata. Fundamenta Informaticae, 85(1-4):111-121, 2008.
[16] B. Farwer, M. Kudlek, and H. Rolke. Concurrent Turing machines. Fundamenta Informaticae, 79(3-4):303-317, 2007.
[17] M. Jantzen and G. Zetzsche. Labeled step sequences in Petri Nets. In Petri Nets 2008, volume 5062 of LNCS, pages 270-283. Springer, 2008.
[18] G. Mavlankulov, M. Othman, S. Turaev, M.H. Selamat, Some properties of the concurrent grammar, International Conference on Mathematical Sciences and Statistics 2013, pp 223-231, Springer. 2014
[19] G. Mavlankulov, M. Othman, S. Turaev, M.H. Selamat, Concurrent Context-Free Grammars, Proceedings of the First International Conference on Advanced Data and Information Engineering, 521-528, LNEE, Springer. 2013
[20] G. Mavlankulov, S. Turaev, L. Zumabaeva,T. Zhukabayeva, Parallel firing strategy on Petri nets: A review, Proceedings of the International Conference On Mathematics, Engineering And Industrial Applications , AIP Publishing, 2015

Gairatzhan Mavlankulov PhD Candidate, Faculty of Computer Science and Information Technology, University Putra Malaysia, Malaysia Research interests: Computer Networks, Petri net controlled grammars, Formal Languages, Parallel Computing, Automata Theory

Laula Zhumabayeva PhD student, Department of Computer Engineering, Eurasian National University of L.N. Gumilyov, Kazakhstan, Research Interests: Formal languages, controlled grammars, Petri nets under parallel firing strategy

Mohamed Othman received his PhD from the Universiti Kebangsaan Malaysia with distinction He is currently a Professor in the Faculty of Computer Science and Information Technology, Universiti Putra Malaysia (UPM). He is also an associate researcher at the Lab of Computational Science and Mathematical Physics, Institute of Mathematical Research (INSPEM), UPM. He is a member of IEEE Malaysia Section, IEEE Communications Society (ComSoc), IEEE Computer Society and Malaysian Mathematical Sciences Society. Research interests: High speed network, parallel and distributed algorithms, software dened networking, network design and management, wireless network (MPDU- and MSDU-Frame aggregation, TCP Performance, MAC layer, resource management, network security

Tamara Zhukabayeva Assistant Professor, Department of Computer Engineering, Eurasian National University of L.N. Gumilyov, Kazakhstan. Research interests: Systems and control, software engineering, intelligent systems, system security

Mohd Hasan Selamat Associate Professor, Faculty of Computer Science and Information Technology, University of Putra Malaysia, Malaysia. Research Interests: Software engineering, Information Systems

Sherzod Turaev obtained his PhD in 2010 from Universitat Rovira i Virgili, Tarragona, Spain. He is currently an Assistant Professor at Department of Computer Science, Faculty of Information and Communication Technology, International Islamic University Malaysia. Research interests: Petri net controlled grammars, descriptional complexity of formal languages and automata, weighted DNA computing.