Proceedings of Grammar Systems Week 2004, edited by E. Csuhaj-Varjú and Gy. Vaszil, MTA SZTAKI, Budapest, pages 170 - 178, 2004.

Simple Splicing Grammar Systems

K. S. Dersanambika[*], K. Krithivasan
Department of Computer Science and Engineering
Indian Institute of Technology Madras
Chennai - 600 036, India
kamala@iitm.ernet.in

K. G. Subramanian
Department of Mathematics, Madras Christian College
Chennai - 600 059, India
kgsmani1948@yahoo.com

Abstract

Dassow and Mitrana [2] introduced a new type of grammar system called splicing grammar system in which communication is done by splicing of strings. Thus segments of sentential forms determined by given splicing rules are exchanged. In this paper, we consider simple splicing rules in these grammar systems and thus obtain four types of simple splicing grammar systems (SSGS), namely, $< 1, 3 >$, $< 2, 4 >$, $< 1, 4 >$, $< 2, 3 >$ SSGS. As $< 1, 3 >$ and $< 2, 4 >$ types of SSGS are equivalent and $< 2, 3 >$ and $< 1, 4 >$ types of SSGS become equivalent, there are essentially two types. Various properties of simple splicing grammar systems are obtained by considering different component grammars. We prove that context free simple splicing grammar systems with two components can generate context sensitive languages. Moreover systems with two regular components can generate nonlinear and context-free languages.

1 Introduction

The theory of Grammar Systems [1] is an intensively investigated area of Formal Language Theory providing an effective grammatical framework for capturing several phenomena characteristic of multi-agent systems such as cooperation, distribution, communication, parallelism etc. The basic idea in a grammar system is to consider several usual grammars and to make them cooperate in order to generate a common language. In parallel communicating grammar systems, the components are generative grammars working on their own sentential forms in parallel and communicating by request.

Motivated by the behaviour of DNA sequences under the influence of restriction enzymes and ligases, Head [3, 4] defined splicing systems that make use of a new

[*]This work is partially supported by University Grants Commision, India

operation, called splicing on strings of symbols. The idea here is that given two strings, each of these is "cut" at suitable "sites" and "pasted crosswise" yielding new strings. Dassow and Mitrana [2] introduced a new type of parallel communicating grammar systems by replacing communication by splicing of strings, thus exchanging segments of sentential forms determined by given splicing rules. Paun [6] has investigated splicing grammar systems improving the results of [2].

Mateescu et al [5] have considered simple splicing systems that make use of splicing rules that are as simple as possible. In this paper we examine splicing grammar systems by requiring the splicing rules to be simple in the sense of [5]. Various properties of the resulting simple splicing grammar systems are obtained by considering different component grammars.

2 Preliminaries

For basic results of formal language theory one can refer to [7]. For notions and results pertaining to grammar system we refer to [1]. We denote the family of regular and context-free languages by REG and CF respectively.

For an alphabet V, the set of all words over V is denoted by $V^*$and the empty word by λ; moreover $V^+ = V^* - \{\lambda\}$. We recall the definition of a simple splicing system [5].

Definition 1 A simple splicing system is a triple

$$\Gamma = (V, A, M)$$

where V is an alphabet, $A \subseteq V^*$ is a finite set of axioms and $M \subseteq V$.

The elements of M are called markers. One can consider four types of languages over V^*, corresponding to the splicing rules of the forms

$$a\#\$a\#, \#a\$\#a, a\#\$\#a, \#a\$a\#$$

where a is an arbitrary element of M. These four rules are respectively called splicing rules of type $< 1, 3 >$, $< 2, 4 >$, $< 1, 4 >$, $< 2, 3 >$.

Clearly splicing rules of types $< 1, 3 >$ and $< 2, 4 >$ yield the same result and for $x, y, z \in V^*$ and $a \in M$ we obtain
$(x, y) \vdash^a_{<(1,3>} z$ iff $x = x_1ax_2, y = y_1ay_2, z = x_1ay_2$, for some $x_1, x_2, y_1, y_2 \in V^*$

For the other types, the splicing is performed as follows:
$(x, y) \vdash^a_{<1,4>} z$ iff $x = x_1ax_2, y = y_1ay_2, z = x_1aay_2$, for some $x_1, x_2, y_1, y_2 \in V^*$
$(x, y) \vdash^a_{<2,3>} z$ iff $x = x_1ax_2, y = y_1ay_2, z = x_1y_2$, for some $x_1, x_2, y_1, y_2 \in V^*$

We now define simple splicing grammar systems.

Definition 2 A $< 1, 3 >$-simple splicing grammar system ($< 1, 3 >$-SSGS)of degree n is a construct

$$\Gamma = (N, T, (S_1, P_1), (S_2, P_2), \cdots, (S_n, P_n), M)$$

where

(i) N, T are disjoint alphabets and P_i, $1 \leq i \leq n$ are finite sets of production rules over $N \cup T$.
(ii) M is a finite subset of $(N \cup T)\#\$(N \cup T)\#$ with $\#$, $\$$ two distinct symbols which are not in $N \cup T$. Each element of M is a $< 1, 3 >$-simple splicing rule.

The sets P_i are called the components of Γ. We can consider grammars of the form $G_i = (N, T, S_i, P_i)$, $1 \leq i \leq n$. By a configuration, we mean an n-tuple consisting of words over $N \cup T$.
For Two configurations,
$x = (x_1, x_2, \cdots, x_n)$, $x_i \in (N \cup T)^* N (N \cup T)^*$, $1 \leq i \leq n$
$y = (y_1, y_2, \cdots, y_n)$, $y_i \in (N \cup T)^*$, $1 \leq i \leq n$
we define $x \Rightarrow_\Gamma y$ if and only if any of the following two conditions holds:
(i)for each $1 \leq i \leq n$, $x_i \Rightarrow_{P_i} y_i$,
(ii) there exist $1 \leq i, j \leq n$ such that
$x_i = x_i^{'} a x_i^{''}$, $x_j = x_j^{'} a x_j^{''}$,
$y_i = x_i^{'} a x_j^{''}$, $y_j = x_j^{'} a x_i^{''}$, for $a\#\$a\# \in M$, and
$y_k = x_k$, for $k \neq i, j$.

In the derivation $x \Rightarrow_\Gamma y$, in $< 1, 3 >$-SSGS, (i) defines a rewriting step, but (ii) defines a $< 1, 3 >$-splicing step, corresponding to a communication step in a parallel communicating grammar system. There is no priority of any of these operations over the other.

A $< 2, 3 >$-simple splicing grammar system is analogously defined.

Definition 3 A $< 2, 3 >$-simple splicing grammar system ($< 2, 3 >$-SSGS) of degree n is a construct

$$\Gamma = (N, T, (S_1, P_1), (S_2, P_2), \cdots, (S_n, P_n), M)$$

where
(i) N, T are disjoint alphabets and P_i, $1 \leq i \leq n$ are finite sets of production rules over $N \cup T$.
(ii) M is a finite subset of $\#(N \cup T)\$(N \cup T)\#$ with $\#$, $\$$ two distinct symbols which are not in $N \cup T$.Each element of M is a $< 2, 3 >$-simple splicing rule.

The sets P_i are called the components of Γ. We can consider grammars of the form $G_i = (N, T, S_i, P_i)$, $1 \leq i \leq n$. By a configuration, we mean an n-tuple consisting of words over $N \cup T$.
For Two configurations,
$x = (x_1, x_2, \cdots, x_n)$, $x_i \in (N \cup T)^* N (N \cup T)^*$, $1 \leq i \leq n$
$y = (y_1, y_2, \cdots, y_n)$, $y_i \in (N \cup T)^*$, $1 \leq i \leq n$
we define $x \Rightarrow_\Gamma y$ if and only if any of the following two conditions holds:
(i)for each $1 \leq i \leq n$, $x_i \Rightarrow_{P_i} y_i$,
(ii) there exist $1 \leq i, j \leq n$ such that
$x_i = x_i^{'} a x_i^{''}$, $x_j = x_j^{'} a x_j^{''}$,
$y_i = x_i^{'} x_j^{''}$, $y_j = x_j^{'} a a x_i^{''}$, for $a\#\$a\# \in M$, and
$y_k = x_k$, for $k \neq i, j$.

In the derivation $x \Rightarrow_\Gamma y$, in $< 2,3 >$-SSGS, (i) defines a rewriting step, but (ii) defines a $< 2,3 >$-splicing step, corresponding to a communication step in a parallel communicating grammar system. Again there is no priority of any of these operations over the other.

Moreover at any instant only one splicing operation can take place in the $< 1,3 >$-SSGS and $< 2,3 >$-SSGS.

Also $< 1,3 >$-SSGS and $< 2,4 >$-SSGS are essentially the same. Likewise $< 2,3 >$-SSGS and $< 1,4 >$-SSGS are the same by definition.

The language generated by the i^{th} component is defined by
$L_i(\Gamma) = \{x_i \in T^* | (S_1, S_2, \cdots, S_n) \Rightarrow^* (x_1, x_2, \cdots, x_n),\ x_j \in (N \cup T)^*,\ j \neq i\}$,
where $\Rightarrow^*$ is the reflexive and transitive closure of the relation $\Rightarrow$.

Two kinds of languages [2] can naturally be associated to a simple splicing grammar system. One of them is the language generated by a single component and, because no component is distinguished in any way , we may always choose the language generated by the first component. This language will be called the individual language of the system.

The second associated language will be the total language, namely

$$L_t(\Gamma) = \bigcup_{i=1}^{n} L_i(\Gamma)$$

Example 1 *Consider the $< 1,3 >$-SSGS with regular rewriting rules. Let*
$\Gamma_1 = (N, T, (S_1, P_1), (S_2, P_2), M)$,
$N = \{S_1, S_2, A, B\}$
$T = \{a, b, c\}$
$P_1 = \{S_1 \to aA,\ A \to aA,\ A \to c\}$
$P_2 = \{S_2 \to cB,\ B \to bB,\ B \to b\}$
$M = \{c\#\$c\#\}$
This system produces the languages
$L_1(\Gamma_1) = \{a^n cb^n | n \geq 1\} \cup \{a^n c | n \geq 1\}$
$L_2(\Gamma_1) = \{cb^n | n \geq 0\}$
Here the total language is
$L_t = \{a^n cb^n | n \geq 1\} \cup \{a^n c | n \geq 1\} \cup \{cb^n | n \geq 0\}$

Example 2 *Consider the $< 1,3 >$-SSGS with context free rewriting rules. Let*
$\Gamma_1 = (N, T, (S_1, P_1), (S_2, P_2), M)$,
$N = \{S_1, S_2, A, B\}$
$T = \{a, b, c, d\}$
$P_1 = \{S_1 \to aAbd,\ A \to aAb,\ A \to ab\}$
$P_2 = \{S_2 \to dcB,\ B \to cB,\ B \to c\}$
$M = \{d\#\$d\#\}$
This system produces the languages
$L_1(\Gamma_1) = \{a^n b^n dc^n | n \geq 1\} \cup \{a^n b^n d | n \geq 1\}$
$L_2(\Gamma_1) = \{dc^n | n \geq 0\}$
Here the total language is
$L_t = \{a^n b^n dc^n | n \geq 1\} \cup \{a^n b^n d | n \geq 1\} \cup \{dc^n | n \geq 0\}$

We denote the family of individual languages generated by simple splicing grammar systems of degree n, with components of type X by $IssgsL_n(X)$.
Similarly we denote the family of total languages generated by simple splicing grammar systems of degree n, with components of type X by $TssgsL_n(X)$.
where $X \in \{REG, CF\}$.

Remark (i) In Example 1 and Example 2, if we use the $< 2,4 >$-splicing rules instead of the $< 1,3 >$-splicing rules i.e. $M = \{\#c\$\#c\}$ in Example 1 and $M = \{\#d\$\#d\}$ in Example 2 , without changing the rewriting rules, then we obtain the same languages in Examples 1 and 2 .

(ii) In Example 1, if we use the $< 2,3 >$-splicing rule $\#c\$c\#$, instead of the $< 1,3 >$-splicing rule , without changing the rewriting rules, then we obtain the following languages:
$L_1(\Gamma_1) = \{a^n c^2 b^n | n \geq 1\} \cup \{a^n b^n | n \geq 1\} \cup \{a^n c | n \geq 1\}$
$L_2(\Gamma_1) = \{cb^n | n \geq 1\} \cup \{c^2\} \cup \{\varepsilon\}$
$L_t(\Gamma_1) = \{a^n c^2 b^n | n \geq 1\} \cup \{a^n b^n | n \geq 1\} \cup \{a^n c | n \geq 1\} \cup \{cb^n | n \geq 1\} \cup \{c^2\} \cup \{\varepsilon\}$

In Example 2, if we use the $< 2,3 >$-splicing rule $\#c\$c\#$, instead of the $< 1,3 >$-splicing rule , without changing the rewriting rules, then we obtain the following languages.
$L_1(\Gamma_2) = \{a^n b^n d^2 c^n | n \geq 1\} \cup \{a^n b^n c^n | n \geq 1\} \cup \{a^n b^n d | n \geq 1\}$
$L_2(\Gamma_2) = \{dc^n | n \geq 1\} \cup \{d^2\} \cup \{\varepsilon\}$
$L_t(\Gamma_1) = \{a^n b^n d^2 c^n | n \geq 1\} \cup \{a^n b^n c^n | n \geq 1\} \cup \{a^n b^n d | n \geq 1\}$
$\quad \cup \{dc^n | n \geq 1\} \cup \{d^2\} \cup \{\varepsilon\}$

3 The Regular and CF Cases

Lemma 1 *Let Γ be a regular $< 2,4 >$-SSGS of degree n, Then a regular $< 2,4 >$-SSGS of degree n, Γ', exists such that $L_t(\Gamma') = L_1(\Gamma)$*

Proof. Let

$$\Gamma = (N, T, (S_1, P_1), (S_2, P_2), \cdots, (S_n, P_n), M),$$

be a splicing grammar with the above mentioned property. For proving the assertion, we construct the system

$$\Gamma = (N', T, (S_1, P_1), (S_2, P_2'), \cdots (S_n, P_n'), M),$$

where

$$N' = N \cup \{X\}$$

$$P_i' = \{A \to aB \mid A \to aB \in P_i\} \cup \{A \to aX \mid A \to a \in P_i\}, 2 \leq i \leq n.$$

By this construction, all languages $L_i(\Gamma'), 2 \leq i \leq n$, become empty and $L_1(\Gamma') = L_1(\Gamma)$ or, in other words, $L_t(\Gamma') = L_1(\Gamma)$. □

The above result holds for $< 1,3 >$-SSGS systems also because these two systems are equivalent.

Theorem 1 *For $Y \in \{I, T\}$,*

1. *For $X \in \{REG, CF\}, X = Y_{<1,3>}ssgsL_1(X)$*
2. *$REG \subset Y_{<1,3>}ssgsL_2(REG)$*
3. *$CF \subset Y_{<1,3>}ssgsL_2(CF)$*
4. *$Y_{<1,3>}ssgsL_2(REG)$ contains nonlinear languages.*

Proof. The first statement immediately follows from definitions. The second statement is a consequence of example 1. The third statement is a consequence of example 2.

As far as the last statement is concerned, the individual language generated by the following $< 1,3 >$-splicing grammar system is the non-linear language
$L = \{a^n cb^n a^m db^m | n, m \geq 1\}$:

$$\Gamma = (\{S_1, S_2, A, B, X, Y, Z\}, \{a, b, c, d\}, (S_1, P_1), (S_1, P_1), M)$$

$P_1 = \{S_1 \to aA, A \to aA, A \to cX, B \to aE, E \to aE, E \to dZ\}$
$P_2 = \{S_2 \to cB, B \to bB, X \to dY, Y \to bY, Y \to b\}$
$M = \{c\#\$c\#, d\#\$d\#\}$
A derivation in Γ runs as follows:
$(S_1, S_2) \Rightarrow^+ (a^n c|X, c|b^n B)$
$\Rightarrow (a^n cb^n B, cX)$
$\Rightarrow (a^n cb^n aE, cdY)$
$\Rightarrow (a^n cb^n a^m E, cdb^{m-1} Y)$
$\Rightarrow (a^n cb^n a^m d|Z, cd|b^m)$
$\Rightarrow (a^n cb^n a^m db^m, cdZ)$,
for some n and $m \geq 1$. Hence we get
$L_1(\Gamma) = L_t(\Gamma) = \{a^n cb^n a^m db^m \mid n, m \geq 1\}$ and $L_2(\Gamma) = \phi$ □

Theorem 2 *$Y_{<2,3>}ssgsL_2(REG)$ also contains nonlinear languages.*

Proof. In the proof of the fourth statement in Theorem 1, a $< 1,3 >$-SSGS splicing grammar system was constructed which generated a nonlinear language. In this, if we use $< 2,3 >$-splicing rules instead of $< 1,3 >$-splicing rules, i.e if we change the rules in M as
$M = \{\#c\$c\#, \#d\$d\#\}$
we obtain
$L_1(\Gamma) = L_t(\Gamma) = \{a^n b^n a^m b^m \mid n, m \geq 1\} \cup \{a^n b^n a^m d^2 b^m | \ n, m \geq 1\} \cup \{a^n c^2 b^n a^m d^2 b^m | \ n, m \geq 1\} \cup \{a^n c^2 b^n a^m b^m | \ n, m \geq 1\}$ and $L_2(\Gamma) = \phi$. This proves the result. □

We show that there is an infinite hierarchy of the classes $Y_{<2,3>}ssgsL_n(CF)$for increasing n.

Theorem 3
$CF = Y_{<2,3>}ssgsL_1(CF) \subset Y_{<2,3>}ssgsL_2(CF) \subset \cdots \subset Y_{<2,3>}ssgsL_n(CF) \subset \cdots$ where Y denotes the total language or the individual language generated in the system

Proof. We consider the following $Y_{<2,3>}ssgs$ simple splicing grammar system

$$\Gamma = (\{S_1, \cdots, S_n, A_1, \cdots, A_n, D_1, \cdots, D_n\}, \{a, b\}, (S_1, P_1), (S_2, P_2), \cdots, (S_n, P_n), M)$$

with
$P_1 = \{S_1 \rightarrow aA_1bD_1, A_1 \rightarrow aA_1b, A_1 \rightarrow ab\}$

$P_2 = \{S_2 \rightarrow D_1aA_2bD_2, A_2 \rightarrow aA_2b, A_2 \rightarrow ab\}$
$\cdots$
$\cdots$
$\cdots$
$P_n = \{S_n \rightarrow D_{n-1}aA_nb, A_n \rightarrow aA_nb, A_n \rightarrow ab\}$

$M = \{\#D_i\$D_i\# : i = 1, 2, \cdots, n+1\}$
A useful sample derivation is as follows:
$(S_1, S_2, \cdots, S_n) \Rightarrow (aA_1bD_1, D_1aA_2bD_2, \cdots, D_{n-1}aA_nb)$

$\Rightarrow^+ (a^{p-1}A_1b^{p-1}D_1, D_1a^{p-1}A_2b^{p-1}D_2, \cdots, D_{n-1}a^{p-1}A_nb^{p-1})$

$\Rightarrow (a^pb^p|D_1, D_1|a^pb^pD_2, \cdots, D_{n-1}a^pb^p)$

$\Rightarrow (a^pb^pa^pb^p|D_2, , D_2|a^pb^pD_3 \cdots, D_{n-1}a^pb^p)$

$\cdots$
$\cdots$
$\cdots$
$\Rightarrow (a^pb^pa^pb^p \cdots a^pb^p, D_1^2, D_2^2, \cdots, D_{n-1}^2)$

Therefore we obtain $I_{<2,3>}ssgsL_n(CF) = T_{<2,3>}ssgsL_n(CF) = \{(a^pb^p)^n |\ p \geq 2\} \cup \{\epsilon\}$.

We can see that $L(\Gamma)$ is not in $Y_{<2,3>}ssgsL_{n-1}(CF)$. In fact in any component we can generate with CF rules only strings of the form a^nb^n prefixed with a nonterminal or suffixed with a nonterminal or both but not strings of the form $a^nb^nc^n$. When we splice strings from two components we get strings whose structure is similar to $a^nb^nc^nd^n$ but we cannot get strings with structure similar to $a^nb^nc^nd^ne^nf^n$ with two components. Continuing this argument, we find then $L(\Gamma)$ cannot be generated by a $Y_{<2,3>}ssgs$ language with $n-1$ components. □

Theorem 4 $Y_{<2,3>}ssgsL_2(REG) - CF \neq \phi$

Proof. Consider the following $< 2, 3 >$-simple splicing grammar system

$$\Gamma = (\{S_1, S_2, A, B, C, D\}, \{a, b, c, d\}, (S_1, P_1), (S_2, P_2), M)$$

$P_1 = \{S_1 \rightarrow aS_1, S_1 \rightarrow eA, X \rightarrow cX, X \rightarrow dD, C \rightarrow c\}$
$P_2 = \{S_2 \rightarrow dB, B \rightarrow bB, B \rightarrow eX, A \rightarrow cA, A \rightarrow cC, D \rightarrow dD\}$
$M = \{\#e\$e\#, \#d\$d\#\}$

A derivation in Γ runs as follows.

$(S_1, S_2) \to (aS_1, dB)$
$\quad\Rightarrow^+ (a^{n+1}S_1, db^nB)$
$\quad\Rightarrow (a^{n+1}|eA, db^ne|X)$
$\quad\Rightarrow (a^{n+1}X, db^ne^2A)$
$\quad\Rightarrow^+ (a^{n+1}c^mX, db^ne^2c^mA)$
$\quad\Rightarrow (a^{n+1}c^m|dD, d|b^ne^2c^{m+1}C)$
$\quad\Rightarrow (a^{n+1}c^mb^ne^2c^{m+1}C, d^2D)$
$\quad\Rightarrow (a^{n+1}c^mb^ne^2c^{m+2}, d^3D)$

Clearly, this language is Context-sensitive but not CF. This proves the result. □

In [2], in the conclusion, the question of whether the hierarchy $YsgsL_n(REG) \subseteq YsgsL_{n+1}(REG)$,$Y \in \{I, T\}$, is infinite, is left open. But the splicing operation is not restricted to be simple. Here we prove that the hierarchy is indeed infinite.

Theorem 5 $REG = YsgsL_1(REG) \subset YsgsL_2(REG) \subset \cdots \subset YsgsL_n(REG) \subset \cdots$

Proof. We consider the following splicing grammar system. Let

$$\Gamma = (\{S_1, \cdots, S_n, A_1, \cdots, A_n\}, \{a, b\}, (S_1, P_1), (S_2, P_2), \cdots, (S_n, P_n), M)$$

where for $i < n$ and i odd,
$P_i = \{S_i \to c_iA_i, A_i \to aA_i, A_i \to aD_{i+1}\}$
and for $i < n$ and i even,
$P_i = \{S_i \to c_iA_i, A_i \to bA_i, A_i \to bD_{i+1}\}$
For $i = n$
$P_n = \{S_n \to c_nA_n, A_n \to aA_n, A_n \to a\}$ if n is odd
$P_n = \{S_n \to c_nA_n, A_n \to bA_n, A_n \to b\}$ if n is even
$M = \{\#D_i\$c_i\# \mid 1 \leq i \leq n\}$
We prove the result for n even. Similar arguments hold when n is odd
For n even, the derivation is as follows

$(S_1, S_2, \cdots, S_n) \Rightarrow (c_1A_1, c_2A_2, \cdots, c_nA_n)$
$\quad\Rightarrow^+ (c_1a^{m-1}A_1, c_2b^{m-1}A_2, \cdots, c_nb^{m-1}A_n)$
$\quad\Rightarrow (c_1a^m|D_2, c_2|b^mD_3, \cdots, c_nb^m)$
$\quad\Rightarrow (c_1a^mb^m|D_2, c_2D_2, c_3|a^mD_3, \cdots, c_nb^m)$
$\quad\cdots$
$\quad\cdots$
$\quad\cdots$
$\quad\Rightarrow (c_1a^mb^m \cdots a^mb^m, c_2D_2, c_3D_3, \cdots, c_nD_n)$

Therefore we obtain for $n = 2, 4, 6, ...$
$YsgsL_n(REG) = \{c_1(a^mb^m)^{n/2}|m \geq 1\}$
and for $n = 1, 3, 5, ...$
$YsgsL_n(REG) = \{c_1a^m(b^ma^m)^{(n-1)/2}|m \geq 1\}$.

It can be seen that $L(\Gamma)$ is not in $YsgsL_{n-1}(REG)$ as it cannot be generated by a $Ysgs$ language with $n-1$ components. In fact with only one component strings of the form $c_i a^m$ or $c_i b^m$ can only be generated. But when there are two components simple splicing of strings generated in the components gives strings of the form $c_i a^m b^m$. Extending this idea it is seen that the language specified needs n components. $\square$

4 Conclusion

Using simple splicing rules, we have examined the power of splicing grammar systems with regular and CF component grammars. It remains to be seen how does this compare with parallel communicating grammar systems.

References

[1] E. Csuhaj-Varjú, J. Dassow, J. Kelemen and Gh. Păun, *Grammar systems: A grammatical approach to distribution and cooperation.* Gordon and Breach Science Publishers, 1994.

[2] J. Dassow and V. Mitrana, Splicing grammar systems, *Computers and Artificial Intelligence*, 15, 1996, 109-122.

[3] T. Head, Formal language theory and DNA: an analysis of the generative capacity of specific recombinant behaviours. *Bull. Math. Biology*, 49, 1987, 737-759.

[4] T. Head, Gh. Păun and D. Pixton, Language theory and molecular genetics: Generative mechanisms suggested by DNA recombination. In "Handbook of Formal Languages" Vol.2, Eds. G. Rozenberg and A. Salomaa, Springer Verlag, 1997, 295-360.

[5] A. Mateescu, Gh. Păun, G. Rozenberg and A. Salomaa, Simple splicing systems, *Discrete Applied Mathematics*, 84, 1-3, 1998, 145-163.

[6] Gh. Păun, On the power of splicing grammar systems, *Ann. Univ. Buc., Matem.-Inform. Series*, 45, 1, 1996, 93-106.

[7] A. Salomaa, *Formal languages*, Academic press, 1973.