Bulletin of Pure and Applied Sciences
Section - E - Mathematics & Statistics

Website : https : //www.bpasjournals.com/

Bull. Pure Appl. Sci. Sect. E Math. Stat.
38E(Special Issue)(2S), 18–23 (2019)
e-ISSN:2320-3226, Print ISSN:0970-6577
DOI: 10.5958/2320-3226.2019.00073.0
©Dr. A.K. Sharma, BPAS PUBLICATIONS,
387-RPS- DDA Flat, Mansarover Park,
Shahdara, Delhi-110032, India. 2019

Generating pictures using P system on matrix grammars [*]

Emerald Princess Sheela J.D.[1], Christopher Kezia Parimalam[2], Sharon Philomena V.[3] and A. Jency[4]

1, 2. Department of Mathematics, Queen Mary's College,
Affiliated to University of Madras, Chennai, India.
3, 4. P.G. Department of Mathematics, Women's Christian College,
Affiliated to University of Madras, Chennai, India.
1. E-mail: emeraldsolomon96@gmail.com , 3. E-mail: sharonphilo2008@gmail.com

Abstract The study of image or picture generation has been of great interest for researchers as they find a variety of applications such as character recognition, pictorial information system design, pattern recognition and so on. Matrix grammars are an extension of string grammars to two dimensions in formal language theory, to generate picture array languages. P system is an area of membrane computing which was proposed by Păun inspired by the structure and functioning of living cells. P systems are useful in dealing with different problems, including the problem of handling picture array generation. In this paper we define P systems for matrix grammars with regular and context free rules which brings out the suitability of P system model for picture array generation.

Key words Matrix grammars, P system, Picture Languages, Membrane Computing.

2010 Mathematics Subject Classification 05C90, 05C99.

1 Introduction

Picture processing is one of the areas of computer science which is concerned with the analysis and generation of pictures. Pictures are regarded as 'concatenations' of subpictures which in turn are built up out of still smaller parts. A digitized picture is a finite rectangular array of 'points' and 'elements' each of which has associated with it one of a discrete finite set of 'values'. Thus a picture can be represented as a $m \times n$ matrix in which each entry a_{ij}, $1 \le i \le m$, $1 \le j \le n$ has one of the values, say $V_1, \ldots, V_k$ (see, [4]). In this paper a linguistic model called matrix grammars is used for the generation of the rectangular arrays. Some interesting classes of pictures including certain letters of the alphabet, kolam, (traditional picture patterns used to decorate the floor in south Indian homes) and wall paper designs (repetitive patterns) can be generated by these grammars (see, [3]). The theory of P system (or membrane computing) is a recent and intensively investigated area of Natural computing, a field of research that tries to imitate nature in the way it computes. Neural networks, generic algorithms and DNA computing are three areas of Natural computing already well established. A new

[*] Communicated, edited and typeset in Latex by *Lalit Mohan Upadhyaya* (Editor-in-Chief).
Received March 16, 2019 / Revised October 28, 2019 / Accepted November 24, 2019. Online First Published on June 03, 2020 at https://www.bpasjournals.com/.
Corresponding author Sharon Philomena V., E-mail: sharonphilo2008@gmail.com
Refereed Proceedings of the National Conference on Mathematics and Computer Applications held at the Department of Mathematics, Women's Christian College, Chennai, India from January 29, 2019 to January 30, 2019.

computability model, called a P system, which is a distributed highly parallel theoretical model based on membrane structure and the behaviour of living cells is introduced by Păun [1]. One of the branches of membrane computing is the rewriting P system in which objects in membranes are described by strings and these strings are processed by rewriting rules or other string manipulating operations. In this paper, we generate matrix languages for various pictures using the P system with different types of rewriting rules.

2 Preliminaries

Definition 2.1. A phrase-structure grammar denoted by G, consists of four components (N, T, P, S), where $N \cap T = \emptyset$. N is a finite sets of symbols, called non-terminals. T is a finite sets of symbols, called the terminals. S is a distinguished element of N and is called a start symbol. P is a finite set of productions which are ordered pairs of the form (u, v), where $u \in V^+$ containing at least one non-terminal and $v \in V^*$, where $V = N \cup T$, V^+ contains strings over V and $V^* = V^+ \cup$empty word λ.

Definition 2.2. If a phrase-structure grammar consists of only rules of the form $A \to aB$ or $A \to a$, where $A, B \in N$, $a \in T$ then the grammar is called a regular grammar (or a right-linear grammar).

Definition 2.3. A context- free grammar (CFG) is a 4- tuple $G = (N, T, P, S)$ where, N is a finite non-empty set of symbols, called the non-terminals (or variables), T is the terminal alphabet (or terminals), $N \cap T = \emptyset$, $S \in N$ is the start symbol and P is a finite non-empty set of rules of the form $A \to \alpha$, where $A \in N$ and $\alpha \in (N \cup T)^*$.

Definition 2.4. [5] A Phrase Structure Grammar(PSG), Context Sensitive Grammar(CSG), Context Free Grammar(CFG), Right Linear Grammar (RLS) is a two tuple $M = (G, G')$, where $G = (V, I, P, S)$ is a PSG, CSG, CFG, RLS with V = a finite set of horizontal non-terminals, I = a finite set of intermediates $= (S_1, S_2, \ldots, S_k)$, P = finite set of production rules called horizontal production rules, S = start symbol $S \in V$ and $\mathrm{V} \cap \mathrm{I} = \emptyset$. $G' = \cup_{i=1}^{k} G_i{}'$ where $G_i{}' = (V_i, T_i, P_i, S_i)$ are Right Linear Grammar with T_i = a finite set of terminals, V_i = a finite set of vertical non- terminals, S_i = start symbol, P_i = a finite set of right linear production rules $V_i \cap V_j = \emptyset$ if $i \neq j$.

Definition 2.5. The set of all matrices generated by M is defined to be

$$L(M) = \left\{ m \times n \text{ arrays } [a_{ij}], i = 1, \ldots, m; j = 1, \ldots, n; m, n \geq 1 / S_1 \ldots S_n \overset{*}{\Downarrow} [a_{ij}] \right\}.$$

Definition 2.6. [1] A rewriting P system $\Pi = (V, \mu, w_1, \ldots, w_n, R_1, \ldots, R_n, i_0)$ where V is an alphabet, μ is a membrane structure, $w_1, \ldots, w_n$ are finite languages over V, $R_1, \ldots, R_n$ are finite sets of context-free evolution rules of the form $X \to v(\text{tar})$, with $X \in V$, $v \in V^*$, tar $\in \{\text{here , out}\} \cup \{\text{in}_j | 1 \leq j \leq n\}$and i_o is the output membrane. The language generated by a system Π is denoted by $L(\Pi)$ and it consists of all strings placed in the output membrane when at the end the computations halt. Each string is processed by one rule only, the parallelism refers here to processing simultaneously all available strings by all applicable rules. If several rules can be applied to a string, may be in several places each, then we take only one rule and only one possibility to apply it and consider the obtained string as the next state of the object described by the string.

Definition 2.7. [2] In a maximal parallelism rewriting step (M), all occurrences of all symbols (which can be the subject of a rewriting rule) are simultaneously rewritten by rules which are non-deterministically chosen in the set of all applicable rewriting rules. That is, if the string, $w = x_1a_1x_2a_2x_3a_3 \ldots x_na_nx_{n+1}$ with $w \in V^+$, $a_1 \ldots, a_n \in V$ and $x_i \in (V - \{a_1, \ldots, a_n\})^*$, $i = 1, \ldots, n+1$ is such that there are no rules defined over symbols in the strings $x_1, \ldots, x_{n+1}$ and there are some rules $r_1 : a_1 \to \alpha_1, \ldots, r_m : a_m \to \alpha_m$ not necessarily distinct, then we obtain in one maximally parallel rewriting step the string $w' = x_1\alpha_1x_2\alpha_2x_3\alpha_3 \ldots x_m\alpha_mx_{m+1}$.

Definition 2.8. [2] A rewriting operation with unique parallelism (U) consists in the substitution of all occurrences of exactly one symbol according to exactly one rule, which is non-deterministically chosen between all rules that can be applied to that symbol. That is, given a string $w = x_1ax_2ax_3 \ldots x_nax_{n+1}$

with $x_i \in (V - \{a\})^*, i = 1, \ldots, n+1$ and one context- free rule $r_1 : a_1 \to \alpha_1, r_2 : a_2 \to \alpha_2$, in one parallel rewriting step we obtain the string $w = x_1 a_1 x_2 a_2 x_3 a_1 x_4 \alpha_1 x_5 \alpha_2 x_6$, (we non-deterministically choose the third occurrence of symbol a_1 and the first occurrence of symbol a_2).

Definition 2.9. [2] With an occurrence parallelism rewriting step (O) , only one occurrence for each symbol in the string that can be rewritten is non-deterministically chosen to be rewritten. For example, given two symbols $a_1,\ a_2$ in V and a string $w = x_1 a_1 x_2 a_2 x_3 a_1 x_4 a_1 x_5 a_2 x_6$, with $x_j \in (V - \{a_1, a_2\})^*$, $j = 1, \ldots, 6$, and given the context-free rules $r_1 : a_1 \to a_1,\ r_2 : a_2 \to a_2$, in one step we could obtain the string $w = x_1 a_1 x_2 a_2 x_3 a_1 x_4 a_1 x_5 a_2 x_6$ (we non-deterministically choose the third occurrence of symbol a_1 and the first occurrence of symbol a_2).

3 Main result

Theorem 3.1. *For every right linear matrix grammar there exists a rewriting P system that generates the corresponding matrix language.*

Proof. Consider the regular matrix grammar $M = (G, G')$, where $G = (V, T, S, P)$ and $G_i^{'} = \left(V_i^{'}, T_i^{'}, S_i^{'}, P_i^{'}\right)$ then $G' = \cup G_i{}'$ and both G and G' being right linear matrix grammar.
Now we construct the rewriting P system $\Pi = (V, T, \mu, w_1, \ldots, w_4, R_1, R_2, R_3, R_4, i_0)$ where, $V = \cup V_i{}'$, $T = \cup T_i{}'$, $\mu = [_1[_2[_3]_3[_4]_4]_2]_1$, $w_1 = S$, $w_2 = w_3 = w_4 = \phi$, $R_1 = \{P\}$, i.e., set of all productions in grammar G,

$$R_2 = \Big\{\text{all rules of } G_i^{'} \text{ of the form } S_i \to a_i A_i\,(\text{tar})\,,\ \text{i.e., all the rules with } S_i \text{ on the LHS where } a_i \in T_i^{'}, A_i \in V_i^{'} \text{ and } S_i \in V\Big\}$$

$$R_3 = \Big\{\text{all the right linear rules in } G_i^{'} \text{ of the form } A_i \to a_i B_j\,(\text{tar})\,,\ \text{where } a_i \in T_i^{'} \text{ and } A_i, B_j \in V_i^{'}\Big\}$$

$R_4 = \Big\{$all the rules of the form $A_i \to a_i\,(\text{tar})$,where $a_i \in T_i^{'}$ and $A_i \in V_i^{'}\Big\}$, i.e., all the terminate rules in the grammar G_i's.
Initially there are no words in the membranes except in the membrane 1, which contains the initial word 'S', that corresponds to the start symbol in the grammar G . So membrane 1 becomes active and the rules of R_1 are applied, which results in the intermediate language (horizontal languages) which was generated by G. At this stage, the resulting string is pushed to membrane 2 and the rules are applied to the intermediate symbols, resulting in words which is a string on $V_i' \cup T_i'$. Now the string is pushed to membrane 3, where recursive rules for the language are applied, the process can go on repeating itself until it is pushed out using the rule with target out. Now the string reaches the membrane 4 where the rules of R_4 are applied and process gets terminated since there are no more rules that can be applied. Thus the process terminates resulting in words which are the words of the matrix language M. Thus $L(\Pi) = L(M)$.

□

Example 3.2. Consider a regular matrix grammar $M = (G, G')$ where G represents the horizontal rules and G' represents the vertical rules of the grammar.

$$G = (\{S, B\}, \{S_1, S_2, S_3\}, \{S \to S_1 B, B \to S_2 B, B \to S_3\}, S);$$

$G' = G_1{}' \cup G_2{}' \cup G_3{}'$ where

$$G_1{}' = (\{S_1, A_1\}, \{a_1\}, s_1 \to a_1 A_1, A_1 \to b_1 A_1, A_1 \to c_1, S_1)$$

$$G_2{}' = (\{S_2, A_2\}, \{a_2\}, s_2 \to a_2 A_2, A_2 \to b_2 A_2, A_2 \to c_2, S_2)$$

$$G_3{}' = (\{S_3, A_3\}, \{a_3\}, s_3 \to a_3 A_3, A_3 \to b_3 A_3, A_3 \to c_3, S_3)$$

are right linear grammars.

Then

$$\begin{array}{llll}
S \Rightarrow S_1S_2S_3 & & & \\
\Downarrow & & & \\
 & & & \begin{array}{l} a_1a_2a_3 \\ b_1b_2b_3 \end{array} \\
\quad \ldots & \ldots & \ldots & \\
\Downarrow & & & \\
 & & & \begin{array}{l} a_1a_2a_3 \\ b_1b_2b_3 \\ c_1c_2c_3 \end{array}
\end{array}$$

In the above example the set of all matrices generated by M is

$$L\left(M\right) = \{S_1S_2^mS_3,\ m \geq 1\} = \left\{ \begin{array}{ccc} a_1 & a_2 & a_3 \\ b_1^n & b_2^n & b_3^n \\ c_1 & c_2 & c_3 \end{array} \right\}, n \geq 0.$$

The P system for matrix grammar:
Consider the P system of degree 4 (Fig. 1) given by $\Pi = (V, T, [_1[_2[_3]_3[_4]_4]_2]_1, R_1, R_2, R_3, R_4, 4)$

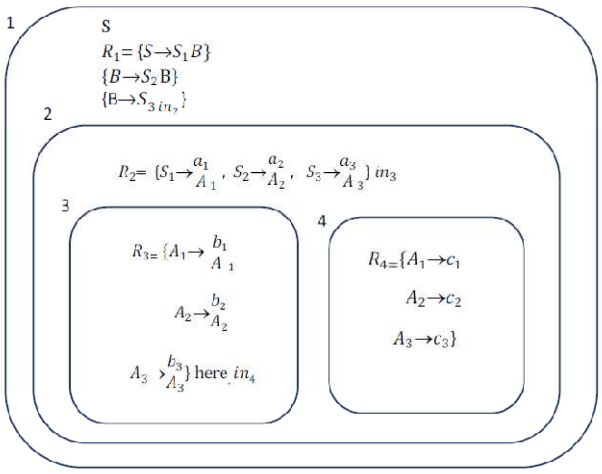
1
S
$R_1 = \{S \to S_1B\}$
$\{B \to S_2B\}$
$\{B \to S_{3\,in_2}\}$
2
$R_2 = \{S_1 \to \begin{array}{c} a_1 \\ A_1 \end{array},\ S_2 \to \begin{array}{c} a_2 \\ A_2 \end{array},\ S_3 \to \begin{array}{c} a_3 \\ A_3 \end{array}\}\ in_3$
3
$R_3 = \{A_1 \to \begin{array}{c} b_1 \\ A_1 \end{array}$
$A_2 \to \begin{array}{c} b_2 \\ A_2 \end{array}$
$A_3 \to \begin{array}{c} b_3 \\ A_3 \end{array}\}$ here in_4
4
$R_4 = \{A_1 \to c_1$
$A_2 \to c_2$
$A_3 \to c_3\}$

Fig. 1:

The application picture for above language is the kolam given below in Fig. 2.

Theorem 3.3. *For every context free matrix grammar there exists a rewriting P system that contains it.*

Proof. Consider the context free matrix grammar, $M = (G, G')$, where G is a context free matrix grammar and G' is a length equal right linear grammar. The construction of P system is the same as in the above result. But by applying different types of rewriting rules defined in section 2, we obtain the different languages for each semantics some of which may coincide depending on the number of variables and their occurrences in the rules defined. In a right linear matrix grammar only one variable occurs at any stage of computations in the P system. Therefore all the types of the rewriting rules will result in the same language. In context free matrix grammar the computation can have more than one variable either recurring or not as the case may be. We are able to apply the different types of rewriting rules which generates languages other than the context free matrix language some of which may coincide with each other.

□

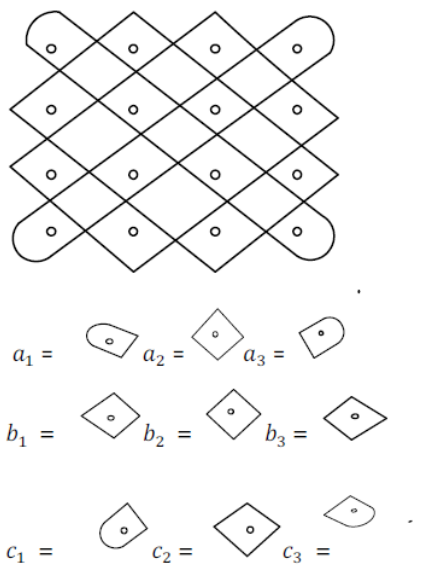

Fig. 2:

Example 3.4. Consider a context free matrix grammar $M = (G, G')$ where G represents horizontal rules and G' represents the vertical rules of the grammar.

$$G = (\{S, A\}, \{S_1, S_2\}, \{S \to AS_1A, A \to AS_2, A \to S_2\}, S), G' = G_1^{'} \cup G_2^{'}$$

where

$$G_1^{'} = (\{S_1, B\}, \{b\}, \{S_1 \to bB, B \to bB, B \to b\}, S_1)$$
$$G_2^{'} = (\{S_2, A\}, \{a\}, \{S_2 \to aA, A \to aA, A \to a\}, S_2)$$

are right linear grammars.
The horizontal rule is $L(G) = \{S_2^n S_1 S_2^n | n, m \geq 1\}$and the vertical rule is

$$L(G') = \left\{(a^n b a^n)^T | n \geq 1\right\}$$

The P system for matrix grammar:
Consider the P system of degree 4, $\Pi = (V, T, [_1[_2[_3]_3[_4]_4]_2]_1, R_1, R_2, R_3, R_4, 4)$
where, $V = (\{S_1, S_2\}, \{A, B\})$, $T = \{a, b\}$,

$$w_1 = S, w_2 = \phi, w_3 = \phi, w_4 = \phi; R_1 = \{S \to AS_1A, A \to AS_2, A \to S_2\};$$
$$R_2 = \{(A \to A_{\text{out}}) > (S_1 \to S_{1\text{in}_3})\}; R_3 = \{S_1 \to bB, B \to bB, S_2 \to aA, A \to aA\};$$
$$R_4 = \{A \to a, B \to b\}$$

In the above P system of matrix grammar we generate two different rewriting languages that is
Unique parallel rewriting: $L(M) = \{S_2^n S_1 S_2^n | n \geq 1\} = \left\{(a^n b a^n)^T | n \geq 1\right\}$
Occurrence parallel rewriting: $L(M) = \{S_2^n S_1 S_2^m | n, m \geq 1\} = \left\{(a^n b a^m)^T | n, m \geq 1\right\}$.
The application picture for above language is the wall paper design is given below in Fig. 3.

4 Conclusion

In this paper, we generated regular and context free matrix languages using P system with different types of rewriting rules and considered their applications to picture generation. Our future work focuses on using different types of P systems on different types of array grammars which can be used in picture processing.

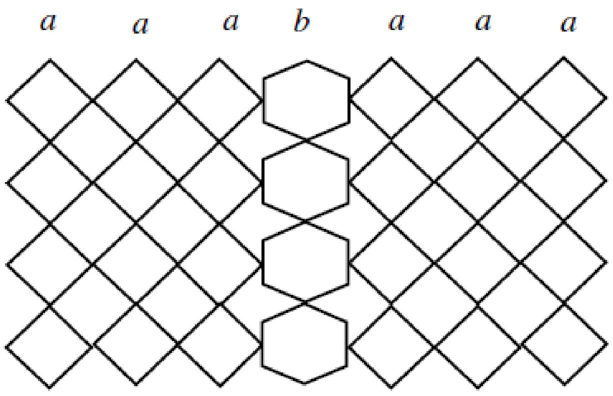

Fig. 3: Unique parallel rewriting.

References

[1] Păun, Gheorghe (2000). Computing with Membranes, *Journal of Computer and System Sciences*, 61, 108–143 and Turku Center for Computer Science- TUCS Report no. 208 (1998).

[2] Păun, Gheorghe, Rozenberg, Grzegorz and Salomaa, Arto (2010). The Oxford Handbook of Membrane Computing, Oxford University Press, Oxford, 168–197.

[3] Siromoney, G., Siromoney, R. and Krithivasan, K. (1972). Abstract families of Matrices and Pictures Languages, *Comput. Graph. Image Proces.*, 284–307.

[4] Siromoney, G., Siromoney, R. and Krithivasan, K. (1973). Picture languages with array rewriting rules, *Information and Control*, 22, 447–470.

[5] Siromoney, Rani (1969). On equal matrix languages, *Information and Control*, 14(2), 135–151.