International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-8 Issue-12, October 2019

Generation of Two-Dimensional Grammar Based Pattern Languages using Array Rewriting P System

Christopher Kezia Parimalam, J.D.Emerald

Abstract—The theory of membrane computing was formulated by Paun as an attempt to formulate a computational model inspired by the way in which the living cells function. P systems which is a highly distributed, parallel, theoretical model and is an area of special interest in recent times. P systems have various application one such area of research is the generation of array grammars using them. In this study we define a model of P system to generate a new class of languages called grammar based two-dimensional pattern languages and their picture generation.

Key Words: Two-dimensional patterns, Two-dimensional axioms, Factorization of array, Array rewriting P system for grammar based 2D patterns

I. INTRODUCTION

String grammars are used to describe various forms of language constructs, hence the study of string grammars find their application in various fields such as Linguistics, Mathematics and Computer Science. String grammars also play an important role in analysis of high level languages. In this context, a pattern $'\alpha'$ is a string over an alphabet set $\{x_1, x_2, x_3, \cdots\}$ of variables. For some finite alphabet Σ of terminal symbols, the pattern language described by α (with respect to Σ) is the set of all words over Σ that can be derived from α by uniformly substituting the variables in α by non-empty terminal words was introduced by Angluin [2]. Dassow et.al., [5] introduced pattern grammars which is a generative device to modify the pattern languages defined by Angluin [2] namely, not allowing the replacing of variables by arbitrary strings, but to adopt the following strategy more usual in formal language theory: start from a finite set of given strings (axioms), replace them by variables in a given set of pattern(s), all strings generated (identified) in this way constitute the associated languages. This way of obtaining languages by substituting all occurrences of the same variable by the same string is related to rewriting in parallel.

Revised Manuscript Received on October 30, 2019.
***** Correspondence Author
Christopher Kezia Parimalam*, Research Scholar, Department of Mathematics, Queen Mary's College, Chennai, Tamilnadu, India. Email: keziamaths@gmail.com
J.D.Emerald, Assistant Professor, Department of Mathematics, Queen Mary's College, Chennai, Tamilnadu, India. Email: emeraldsolomon96@gmail.com

In [4] the concept of pattern as language descriptors was generalized to two-dimensional case, preserving the simplicity and compactness of the descriptors.

The theory of P system (or membrane computing) introduced by Paun [6] is a theoretical model which is processed in a parallel manner. Recently Natural computing, a field of research which deals with the computation to imitate nature is of great interest. Already a lot of work has been done in the areas of DNA computing, Generic Algorithms and Neural networks. P systems introduced by Paun deals with how to process multi-sets by evolution rules in a hierarchical arrangement of membranes that represent the living cell. Rewriting P systems is a branch of membrane computing where objects are replaced by strings and are processed using rewriting rules.

Pictures are described as finite arrays in two-dimensional plane [3] which are connected and has been an area of great interest for many researchers. The study of pictures is in an attempt to extend recognisability in one dimension to two dimensions. Array grammars are extension of string grammars to pictures in two-dimension, where the points with integer coordinates of the plane represent symbols. It is easily seen that array grammars are immediate extensions of the string grammars, where the strings in the membranes are replaced by arrays and string operations by array rewriting rules.

In this paper we define an array rewriting P system to generate languages defined by two-dimensional grammars based on patterns defined in [1]. We also to analyse the picture generated by two-dimensional grammars based on patterns.

II. PRELIMINARIES

For a finite alphabet Σ, a string or word (over Σ) is a finite sequence of symbols fromΣ, andϵ stands for the empty string. The notation Σ^+ denotes the set of all nonempty strings over Σ, and $\Sigma^* = \Sigma^+ \cup \{\epsilon\}$. We say that a string $v\epsilon\Sigma^*$ is a factor of a string $\omega\epsilon\Sigma^*$ if there are $u_1, u_2\epsilon\Sigma^*$ such that $\omega = u_1.v.u_2$. If $u_1$or $u_2$is empty string then v is a prefix (or a suffix respectively) of ω .the notation$|\omega|$ stands for the length of a string ω.

A two-dimensional word (or array) over Σ is a tuple W= $\big((a_{11}, a_{12}, \cdots, a_{1n}), (a_{21}, a_{22}, \cdots, a_{2n}), \cdots, (a_{m1}, a_{m2}, \cdots, a_{mn})\big)$, where m, n$\epsilon\mathbb{N}$ and, for every i, $1 \leq i \leq m$, $1 \leq j \leq n$,$a_{ij}\epsilon\,\Sigma$.

We define width to be the number of columns and height to be the number of rows of W. λ, denotes the empty array.For the sake of convenience, we denote W by $[a_{ij}]_{m,n}$ or by a matrix of one of the following form:

$$\begin{bmatrix} a_{11}a_{12} & \cdots & a_{1n} \\ a_{21}a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots \ \vdots \\ a_{m1}a_{m2} & \cdots & a_{mn} \end{bmatrix} \begin{matrix} a_{11}a_{12} & \cdots & a_{1n} \\ a_{21}a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots \ \vdots \\ a_{m1}a_{m2} & \cdots & a_{mn} \end{matrix}$$

If we want to refer to the j[th] symbol in row i of the array When we use W[i,j] =$a_{i,j}$.By Σ^{++}, we denote the set of all nonempty arrays over Σ, and $\Sigma^{**} = \Sigma^{++} \cup \{\lambda\}$. An array language is every subset of languages that belong to Σ^{**}

Definition 2.1[1] Two-dimensional Pattern Grammar is a four tuple $G = (\Sigma, \Delta, A, P)$ where

$\Sigma =$ Set of terminal alphabets

$\Delta =$ Set of pattern variables

A = Set of two-dimensional axioms which satisfy the patterns defined by their factorization

P = The two-dimensional pattern, defined by pattern variables only and someδ_{ij} the same.

i.e., P = $\begin{bmatrix} \delta_{11}\delta_{12} \cdots & \delta_{1n} \\ \delta_{21}\delta_{22} \cdots & \delta_{2n} \\ \vdots \quad \vdots & \ddots \quad \vdots \\ \delta_{n1}\delta_{n2} \cdots & \delta_{mn} \end{bmatrix}$ all the m x n entries are defined by a pattern variable.

The pattern follows one of the concatenation (factorization) (i) row, (ii) column (iii) row-column (iv) column-row (v) proper (where it satisfies both row-column and column-row).

Given pattern P and initial axiom to start with $P(L)$ is the set of two-dimensional arrays obtained by replacing the variables in the pattern P by the arrays in A, the same variable being replaced by the same array in all their occurrences.

The language generated by G, denoted by $L(G)$, is the smallest language $L \subseteq \Sigma^{**}$ for which we have:

(i) $A \subseteq L$

(ii) P(L) $\subseteq L$.

Arrays are obtained starting from the axioms and using them in the patterns finite number of times. The language generated $L(G) = P \cup P(A) \cup P\big(P(A)\big) \cup P\left(P\big(P(A)\big)\right) \cdots$

A language $L(G)$ thus obtained is called a two-dimensional pattern Language and we denote it as $L_{p,f}$ where f denotes the array factorization type of the pattern defined ; and its family is denoted by $\mathcal{L}_{P,f}$.

A two-dimensional pattern is a mxn array, and the axiom set is a set of mxn dimensional arrays which satisfy the concatenation rules for arrays either by row, column, row-column or column-row or proper factorization.

Example 2.1 $G_1 = (\{a\}, \{\delta\ \}, \{[a]\}, P), P = \begin{bmatrix} \delta & \delta \\ \delta & \delta \end{bmatrix}_p$

$L(G_1) = L_{P,p}$

$= A \cup set\ of\ all\ arrays\ of\ a's\ with\ dimension\ 2^n \times 2^n$

$where\ n = 1,2 \cdots$

III. ARRAY REWRITING P SYSTEM FOR GRAMMAR BASED TWO-DIMENSIONAL PATTERN LANGUAGES

Definition 3.1 An array-rewriting P systemfor two-dimensional patterns ARPS2DP (of degree m≥ 1) is a construct

$\Pi_R = (N, T, \mu, A_1, \cdots A_m, P_1, \cdots P_m, i_{0ut})$,

Where N is the set of pattern variables and terminals, T⊆ V is the terminal alphabet, μ represents the way in which the 'm' membranes are arranged in the system, $A_1, \cdots, A_m$ are the arrays present initially in the regions of μ and are finite in number, $P_1, \cdots P_m$, are finite set of rewriting rules in the m regions of μ; the pattern rewriting rules are of the form $A \to B_{(tar)}$; where the target can be in, out, here, $A, B \in (N \cup T)^*$, i_{0ut} is the label of the output membrane.

A computation in an array rewriting P system for 2D patterns is defined in the same way as in a string rewriting P system. Each variable in the array from each region can be rewritten by exactly one rewriting rule applicable to it in the given region, the same variable being rewritten with the same rule and is sent to the target specified in the rewriting rule. When there are no more rewriting rule to be applied and if the array reaches the output membrane the computation stops. A successful computation consists of arrays composed only of symbols from T placed in the membrane with label i_{0ut}in the halting configuration. The set of all such arrays generated by the system Π_R is denoted by APL(Π_R).

We now give an example of an array rewriting P system that can generate a given two-dimensional pattern language based on grammars that defines it. We consider the case where the pattern is a two-dimensional array in particular a 2×2 array and the axioms are over a single alphabet.

Example 3.1 Consider the two-dimensional pattern grammar G_2= (Σ, Δ, A, P)

Where, $\Sigma = \{a, b\}$; $\Delta = \{\delta_1, \delta_2\}; A = \{a, b\}$;

P=$\begin{bmatrix} \delta_1 & \delta_2 \\ \delta_1 & \delta_2 \end{bmatrix}_{cr}$;

$L(G_2)$

$= \{\begin{bmatrix} w_1 & w_2 \\ w_1 & w_2 \end{bmatrix} / \ w_1, w_2\ are\ square\ matrices\ of\ order\ 2^m \times 2^m, m = 0, 1, \cdots\}$

The array rewriting P system to generate the above two-dimensional pattern language is defined as follows: $\Pi_{R2} = (N, T, A_1 \cdots A_3, P_1 \cdots P_3, i_1)$; where

N= $\{\eta_{11}, \eta_{12}, \eta_{21}, \eta_{22}\}$;T= {a,b}

$\mu = [[[\ ]_3]_2]_1$

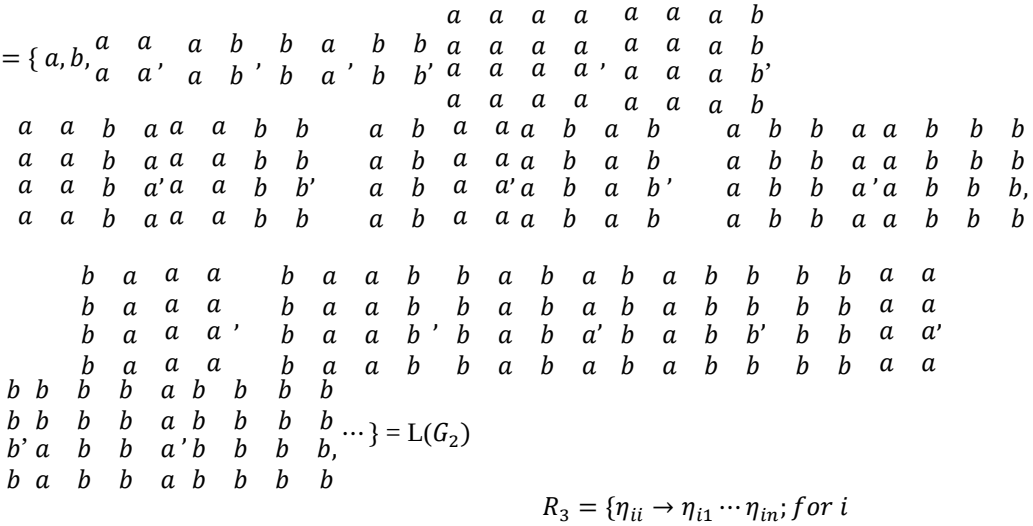
$A_1 = \{a, b, \begin{matrix} \eta_{11} & \eta_{22} \\ \eta_{11} & \eta_{22} \end{matrix}\}; A_2 = \phi;\ A_3 = \begin{matrix} \eta_{11} & \eta_{22} \\ \eta_{11} & \eta_{22} \end{matrix}\}$

P_1

$= \left\{\begin{matrix} \eta_{11} \to a, \eta_{11} \to b, \eta_{12} \to a, \eta_{12} \to b, \eta_{21} \to a, \eta_{21} \to b, \\ \eta_{22} \to a, \eta_{22} \to b \end{matrix}\right\}$

$$P_2 = \left\{ \begin{array}{l} \eta_{11}\eta_{11} \rightarrow \begin{matrix} \eta_{11}\eta_{11} \\ \eta_{11}\eta_{11_{in,out}} \end{matrix};\ \eta_{11}\eta_{12} \rightarrow \begin{matrix} \eta_{11}\eta_{12} \\ \eta_{11}\eta_{12_{in,out}} \end{matrix}; \\ \eta_{12}\eta_{11} \rightarrow \begin{matrix} \eta_{12}\eta_{11} \\ \eta_{12}\eta_{11_{in,out}} \end{matrix};\ \eta_{12}\eta_{12} \rightarrow \begin{matrix} \eta_{12}\eta_{12} \\ \eta_{12}\eta_{12_{in,out}} \end{matrix}; \\ \eta_{21}\eta_{21} \rightarrow \begin{matrix} \eta_{21}\eta_{21} \\ \eta_{21}\eta_{21_{in,out}} \end{matrix};\ \eta_{21}\eta_{22} \rightarrow \begin{matrix} \eta_{21}\eta_{22} \\ \eta_{21}\eta_{22_{in,out}} \end{matrix}; \\ \eta_{22}\eta_{21} \rightarrow \begin{matrix} \eta_{22}\eta_{21} \\ \eta_{22}\eta_{21_{in,out}} \end{matrix};\ \eta_{22}\eta_{22} \rightarrow \begin{matrix} \eta_{22}\eta_{22} \\ \eta_{22}\eta_{22_{in,out}} \end{matrix} \end{array} \right\}$$

$$P_3 = \left\{ \begin{array}{l} \eta_{11} \rightarrow \eta_{11}\eta_{11_{in}}/\eta_{11}\eta_{12_{in}}/\eta_{12}\eta_{11_{in}}/\eta_{12_{in}}; \\ \eta_{12} \rightarrow \eta_{11}\eta_{11_{in}}/\eta_{11}\eta_{12_{in}}/\eta_{12}\eta_{11_{in}}/\eta_{12_{in}}; \\ \eta_{21} \rightarrow \eta_{21}\eta_{21_{in}}/\eta_{21}\eta_{22_{in}}/\eta_{22}\eta_{21_{in}}/\eta_{22}\eta_{22_{in}}; \\ \eta_{22} \rightarrow \eta_{21}\eta_{21_{in}}/\eta_{21}\eta_{22_{in}}/\eta_{22}\eta_{21_{in}}/\eta_{22}\eta_{22_{in}} \end{array} \right\}$$

$APL(\Pi_{R2}) = \{ a, b,$  $\cdots \} = \mathrm{L}(G_2)$

IV. CONSTRUCTION OF AN ARRAY REWRITING P SYSTEM MODEL FOR TWO-DIMENSIONAL PATTERNS

Theorem 4.1 For every two-dimensional grammar based on pattern we have an Array Rewriting P System for two-dimensional pattern $ARPS2DP_3$ that generates it.

Proof: We give the construct for the Array rewriting P system for two-dimensional pattern in two cases. The first being the 2D pattern satisfying column-row factorization and the second when the 2D pattern satisfies the row-column factorization.

Case 1: Given a 2D pattern grammar $G = (\Sigma, \Delta, \mathrm{A}, \mathrm{P})$, in which the 2D pattern satisfies the column-row factorization, we define the P-system model where

$$\Pi_R = (N, T, \mu, A_1, \cdots A_3, P_1, \cdots P_3, i_1)$$

$$N = \left\{ \begin{array}{c} \eta_{ij}, i = 1,2 \cdots n, for\ every\ \delta_i \in \Delta, i = 1 \cdots n; \\ j = 1 \cdots m, where\ m\ is\ the\ number\ of\ columns\ in\ the\ pattern \end{array} \right\}$$

$T = A$

$\mu = [_1[_2[_3]_3]_2]_1$

$A_1 = \{A, P\}$; where P is replaced with the η_{ii} in the place of δ_i.

$A_2 = \phi$; $A_3 = P\ as\ defined\ in\ A_1$

$R_1 = \{\eta_{ij} \rightarrow a_t, for\ i, = 1, \cdots n, j = 1, \cdots m,$ for each $a_t \in A\}$

$$R_2 = \left\{ \begin{array}{c} [\eta_{i1} \cdots \eta_{is}]^T \rightarrow [\eta_{i1} \cdots \eta_{it}]^T; if\ [\delta_1 \cdots \delta_s]^T \rightarrow [\delta_1 \cdots \delta_t]^T \\ is\ a\ row\ catenation\ rule \\ in\ the\ given\ 2D\ pattern\ for\ the\ i^{th}\ column\ and\ \eta_{in} = \eta_{im}\ if\ \ \delta_n = \delta_m \end{array} \right\}_{in,out}$$

P = $(\delta_1\delta_2) \ominus (\delta_1\delta_2)$; Once we have a rule in the third membrane which first grows each pattern variable according to the column catenation rule in the first parenthesis of the pattern rule and the second membrane has a rule to grow the pattern according to the row catenation rule. Thus we form the frame work to grow the patterns and finally terminate them using the termination rules in the first membrane which is the output membrane.

$R_3 = \{\eta_{ii} \rightarrow \eta_{i1} \cdots \eta_{in}; for\ i = 1 \cdots n, if\ \delta_1 \cdots \delta_n\ is\ the\ first\ row\ of\ the 2D\ pattern;$ $and\ \eta_{ij} \rightarrow \eta_{i1} \cdots \eta_{in}\ for\ every\ \eta_{ij}\ on\ the\ RHS\ of\ the\ productions\ of\ \eta_{ii}\ \}$

To begin with the P system contains the initial words in membrane 1 and 3. The set of axioms are present in membrane 1 which are the initial words in the pattern language. The pattern is present at the beginning in membrane 1 and 3. Applying the rules of R_1 to the pattern variables in membrane 1 we get the resultant as P(A). In membrane three applying the rules of R_3, each pattern variable expands in the column where each variable is expanded to form the first row of the array in P(P(A)) and sent into membrane 2. In this membrane the pattern is expanded according to the row catenation rules in the pattern and is sent to membrane 1 and 3. In membrane1 the terminating rules are applied to the pattern variables which result in P(P(A)). While the membrane 3 further builds the array in column and then again sends it membrane two where the process is repeated and the resultant is P(P(P(A))). Thus $ARPS2DP_3$ generates L(G).

Case 2: We now give the construction for 2D patterns which satisfy row-column factorization rule. T, μ, A_1, A_2, A_3 are as defined in case1.

$N = \left\{ \begin{matrix} \eta_{ij}, i = 1,2 \cdots n, for\ every\ \delta_i \in \Delta, i = 1 \cdots n; \\ j = 1 \cdots m, where\ m\ is\ the\ number\ of\ columns\ in\ the\ pattern \end{matrix} \right\}$

The array rewriting rules for the three membranes are defined as follows

$R_1 = \{\eta_{ij} \to a_t, for\ i, = 1, \cdots n, j = 1, \cdots m,$ for each $a_t \in A\}$

$R_2 = \left\{ \begin{matrix} \eta_{i1} \cdots \eta_{is} \to \eta_{i1} \cdots \eta_{it}; if\ \delta_1 \cdots \delta_s \to \delta_1 \cdots \delta_t \\ is\ a\ column\ catenation\ rule \\ in\ the\ given\ 2D\ pattern\ for\ the\ i^{th}\ row\ and\ \eta_{in} = \eta_{im}\ if\ \delta_n = \delta_m \end{matrix} \right\}_{in,out}$

$R_3 = \{\eta_{ii} \to [\eta_{i1} \cdots \eta_{in}]^T\ foreach\ i = 1 \cdots n,$
$if\ [\delta_1 \cdots \delta_n]^T is\ the\ first\ column\ of\ the 2D\ pattern;$
$and\ \eta_{ij} \to [\eta_{i1} \cdots \eta_{in}]^T for\ every\ \eta_{ij}\ on\ the\ RHS\ of\ the\ productions\ of\ \eta_i \}.$

The working is similar as in case 1, and we see that the P system generates L(G).

A 2D pattern defined by proper factorization can be generated either by $ARPS2DP_3$ in case 1 or case 2.

Thus $ARPS2DP_3$ = L(G).

Example4.1

$G_3 = \left(\{a, b\}, \{\delta_1, \delta_2\}, \left\{[a\ a], \begin{bmatrix} b & b \\ b & b \end{bmatrix}, P\right\}\right), P = \begin{bmatrix} \delta_1 & \delta_2 \\ \delta_2 & \delta_1 \end{bmatrix}_{rc}$; i.e., P = $(\delta_1 \ominus \delta_2)(\delta_2 \ominus \delta_1)$

$L(G_3) = \mathcal{L}_{p,rc} = \begin{bmatrix} w_i & w_j \\ w_j & w_i \end{bmatrix}$
$= A \cup set\ of\ all\ arrays\ of\ dimension\ (sm_1 + lm_2) \times 2k nover\ the\ given\ axioms.$

The $ARPS2DP_3$ to generate the G_3 is defined by

$\Pi_{R3} = (N, T, \mu, A_1, A_2, A_3, P_1, P_2, P_3, i_1)$

$N = \{\eta_{11}, \eta_{12}, \eta_{21}, \eta_{22},\}$

$T = \{a, b\}$

$\mu = [_1[_2[_3]_1]_2]_3,$

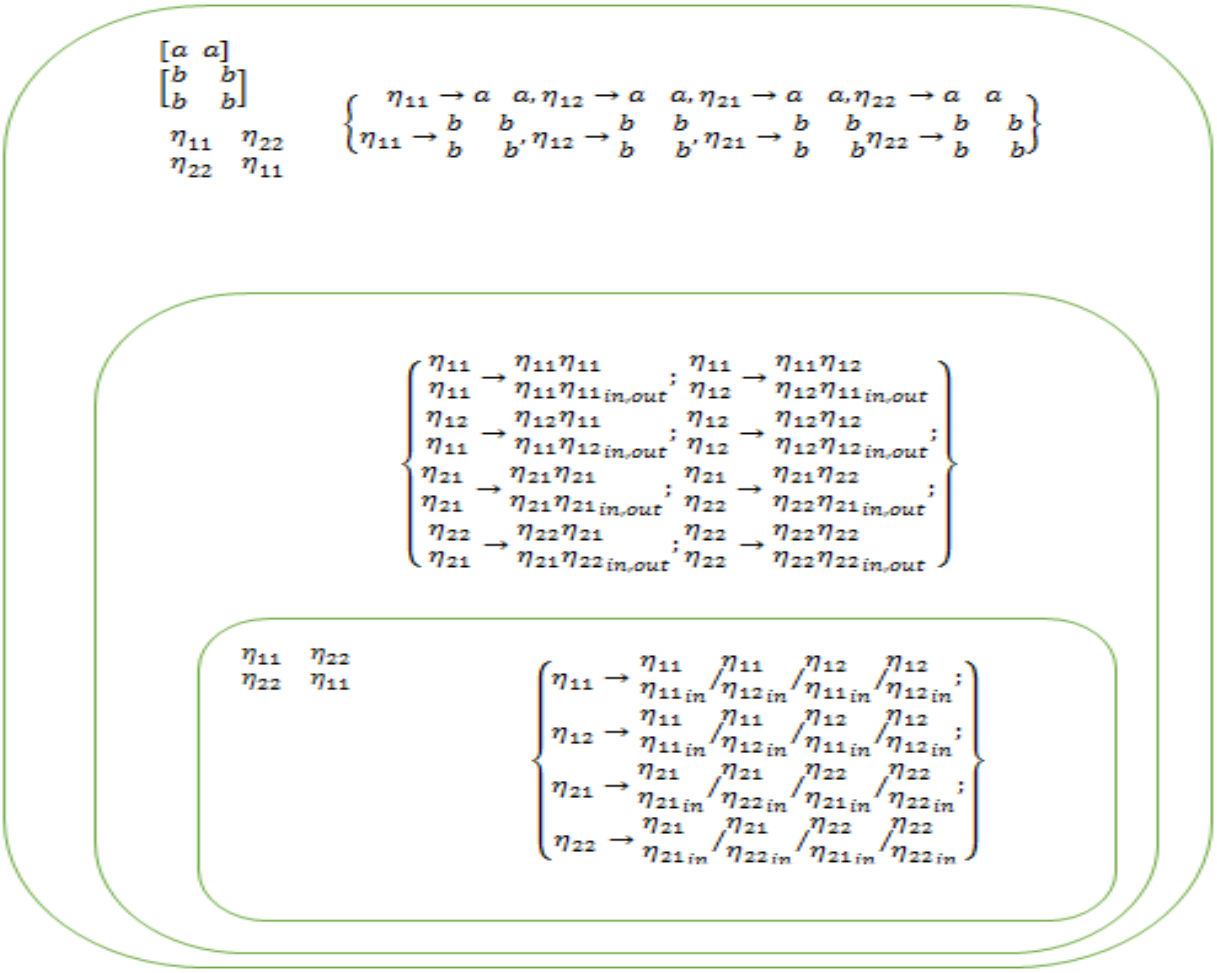


Fig 1 Array Rewriting P System generating the 2D Pattern Language

V. PICTURE GENERATION FOR TWO-DIMENSIONAL PATTERN LANGUAGES& RESULTS

In this section, we generate pictures for the grammar based two-dimensional pattern languages defined above using pasting rules in Array rewriting P system for 2D pattern $ARPS2DP_3$. Consider the example 3.1, we associate two tiles for the two axioms, namely $'a'$ a hexagon and $'b'$ a rhombus. The$ARPS2DP_3$with pasting rules to generate the pictures associated with the 2D pattern language is given in Fig 2.

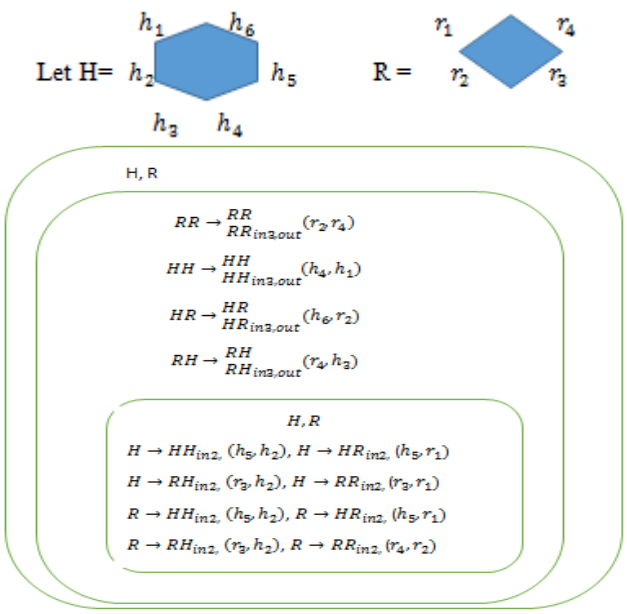


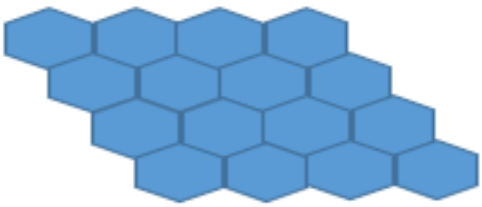
Fig 2 $ARPS2DP_3$ with pasting rules for picture generation

The P system initially contains the hexagon and the rhombus which are the axioms in the two-dimensional language. The second membrane is empty with no initial words, hence no rules can be applied at thi stage in this membrane. The third membrane has the hexagon and the rhombus and the one among the four rules is applied. Suppose if the first rule is applied then the resultant would be two hexagons pasted with their edges h_5, h_2 pasted together and pushed into membrane 2. In membrane two the first rule is used and the resultant is pushed into membrane three and mem

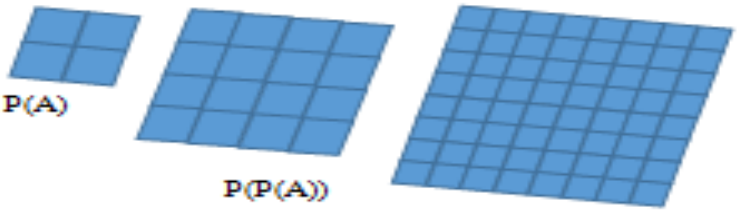
brane 1 which is the output membrane.

In membrane 3, the pasting rule

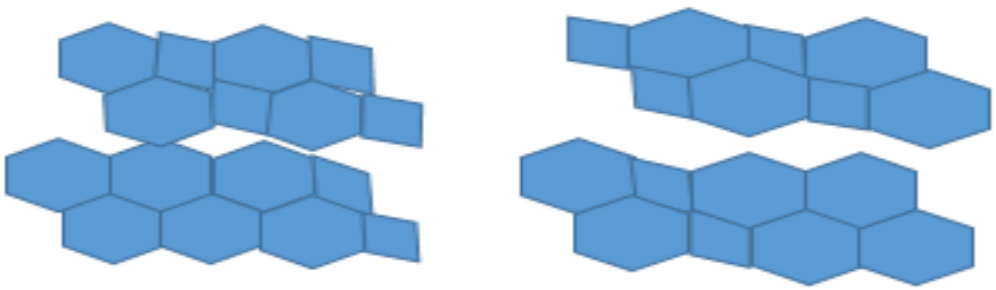
In membrane 2, the pasting is  and is the output which belongs to P(A)).

The next output in the series is  belongs to P(P(A)).

Applying the second rule in both membranes the first, second and third output will be



Some of the outputs for P(A) when other rules are applied in membrane three is given below



VI. CONCLUSION

In this paper we have defined an array rewriting P system model to generate two-dimensional pattern languages based on pattern grammar with two-dimensional axioms. We also generated pictures associated with these languages. The study would be further extended to make a comparative study of 2D pattern languages with other array languages and picture languages.

REFERENCES

1 C.K. Parimalam, J.D. Emerald Two-dimensional grammars based on Patterns, International Journal for Computer Science and Engineering,Vol.07, Issue.05 , pp.216-220, 2019.

2 D. Angluin, Finding Pattern common to set of strings, Journal of Computer and System Sciences 21, 46-62, 1980.

3 3. D. Giammarresi, A. Restivo, Recognizable picture languages, International Journal of Pattern Recognition and Artificial Intelligence. 6:31-46, 1992.

4 H. Fernau, M.L. Schmid, K.G. Subramanian, Two-Dimensional Pattern Languages, In S.Bensch, F. Drewes, R. Freund, and F.Otto, editors, Fifth Workshop on Non-Classical Models for Automata and Applications, NCMA, Volume 294 of books@ocg.at, pp.117-132. Osterreichische Computer Gesellschaft, 2013.

5 J. Dassow, Gh. Paun, Arto Salomaa, Grammars based on Patterns, International Journal of Foundations of Computer Science, Vol 4: 1-14, 1993.

6 Gh. Paun , Computiong with Membranes, Journal of Computer and System Sciences,61,108- 143, 2000.