IOSR Journal of Mathematics (IOSR-JM)
e-ISSN: 2278-5728, p-ISSN: 2319-765X. Volume 10, Issue 6 Ver. III (Nov - Dec. 2014), PP 06-11
www.iosrjournals.org

Pure 2D Context-free Puzzle P system with Conditional Communication

S.Hemalatha[1] P.S.AzeezunNisha[2]
[1](Department of Mathematics, S.D.N.B Vaishnav College for Women, Chennai, India)
[2](Department of Mathematics, J.B.A.S. College for Women, Chennai, India)

Abstract: *The feature of conditional communication in membrane computing has been introduced in symbol objects and arrays in [1], [4] and [16]. Pure context-free grammars have been introduced in [17]. In [18] this idea has been applied on array objects with an additional control on regular grammars. In this paper we use the conditional communication of membrane computing to pure 2D context free grammars and study the power of the system by comparing with the existing system.*

Keywords: *Conditional communication, context free grammar, P systems, Pure grammar, Puzzle grammar*

I. Introduction

The area of membrane computing was initiated by Păun [3]. Now, this computability model is called as P system, which is a distributed, highly parallel theoretical computing model, based on the membrane structure and the behavior of the living cells. P systems, with string-objects and string-processing rules as known in formal language theory, have been considered [8] and investigated extensively. Array languages have also been studied in [9, 10]. The problem of handling array languages using P systems has been considered by Ceterchi et al [6]. Array grammars have been introduced and investigated with the tools of P system [7, 13, 15]. Among variants of P system, conditional communication is a feature introduced in [1]. On the other hand rewriting P systems with string-objects have been extended to array-rewriting P systems with array-objects and the power of such systems has been studied in [2, 16]. Concepts of pure grammars has been introduced in [17]. Pure 2D picture grammars have been studied in [18]. In this paper, we have incorporated conditional communication in Puzzle P system with array objects [14] and further properties have been studied.

In section 2 we recall some preliminary definitions with examples. Section 3 deals with the main results of this paper which combines the Pure 2D context free grammars with P systems with conditional communication.

II. Preliminaries

We recall some basics of formal language theory as in [11]. In this section we recall some basic definitions as in [11] Let Σ be a finite alphabet. A word or string w over Σ is a sequence of symbols from Σ. The set of all words over Σ, including the empty word λ with no symbols, is denoted by Σ^*. An array over Σ consists of finitely many symbols from Σ placed in the points of Z^2 (the two-dimensional plane) and the points of the plane not marked with symbols of Σ, are assumed to have the blank symbol $\# \notin \Sigma$. We will pictorially represent the arrays indicating their non blank pixels, whenever possible. The set of all arrays over Σ will be denoted by Σ^{*2}. An array over {a} describing the picture pattern T is shown in Figure 1.

```
a a a a a a a a a
  a
  a
  a
  a
```

Fig 1 An array describing picture token T.

An array, in particular can be rectangular. A rectangular $m \times n$ array M over Σ is of the form

$$M = \begin{matrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{matrix}$$

where each $a_{ij} \in \Sigma$, $1 \leq i \leq m$, $1 \leq j \leq n$. The set of all rectangular arrays over Σ is denoted by Σ^{**}, which includes the empty array Λ.

Now we recall the definitions of pure grammars and pure context-free grammar as in [17, 18].

A pure grammar is a triple $G = (\Sigma, P, S)$ where Σ is a finite alphabet, S is a finite set of words over Σ and P is a finite set of ordered pairs (x, y) of words over Σ. The elements of P are referred to as production and denoted by $x \rightarrow y$. If in each production $x \rightarrow y$ of P the left side x is a symbol then we say that G is a pure context-free grammar. Languages generated by Pure Context Free (PCF) grammars are referred as PCF Languages.

A pure context free grammar is G = (Σ, P, set of axioms words) where Σ is a finite alphabet, a set of axiom words and P is a finite set of context free rules of the form $a \rightarrow \alpha$, $a \in \Sigma$, $\alpha \in \Sigma^*$. Derivations are done as in a context free grammar except that, unlike a context free grammar, there is only one kind of symbol, namely the terminal symbol. The language generated consists of all words generated from each axiom word. Also, we recall the notions of string rewriting P systems with conditional communication [1] and array rewriting P systems with conditional communication [4].

An extended (string) rewriting P system (of degree $m \geq 1$) with conditional communication [1] is a construct $\Pi = (V, T, \mu, M_1, M_2, \ldots, M_m, R_1, P_1, F_1, R_2, P_2, F_2, ..., R_m, P_m, F_m)$ where: V is the alphabet of the system, $T \subseteq V$ is the terminal alphabet, μ is a membrane structure with m membranes (injectively labelled by 1, 2, ..., m), M_i are finite languages over V, representing the strings initially present in the regions i, i = 1, 2, ..., m of the system, R_i are finite sets of context-free rules over V present in regions i, i = 1, 2, ..., m of the system, P_i are permitting conditions and F_i are forbidding conditions associated with regions i, i = 1, 2, ..., m. The conditions can be of the following forms:

1. empty : No restriction is imposed on strings, they either freely exit the current membrane or enter any of the directly inner membranes ; An empty permitting condition is denoted by (true, α), $\alpha \in$ {in, out}, and an empty forbidding condition by (false, not α), $\alpha \in$ {in, out}.
2. symbols checking : each P_i is a set of pairs (a, α), $\alpha \in$ {in, out} for $a \in V$ and each F_i is a set of pairs (b, not α), $\alpha \in$ {in, out} for $b \in V$: a string w can go to a lower membrane only if there is a pair (a, in) $\in P_i$ with $a \in$ alph(w), and for each (b, not in) $\in F_i$ we have $b \notin$ alph(w); similarly for the string to go out of membrane i, it is necessary to have $a \in$ alph(w) for at least one pair (a, out) $\in P_i$ and $b \notin$ alph(w) for all (b, notout) $\in F_i$
3. substrings checking : each P_i is a set of pairs (u, α), $\alpha \in$ {in, out} for $u \in V^+$ and each F_i is a set of pairs (v, not α), $\alpha \in$ {in, out} for $v \in V^+$: a string w can go to a lower membrane only if there is a pair (u, in) $\in P_i$ with $u \in$ Sub(w), and for each (v, notin) $\in F_i$ we have $v \notin$ Sub(w); similarly for the string to go out of membrane i, it is necessary to have $u \in$ Sub(w) for at least one pair (u, out) $\in P_i$ and $v \notin$ Sub(w) for all (v, not out) $\in F_i$.

Thus we have conditions of the types empty, symb, sub_k, respectively, where k is the length of the longest string in all P_i, F_i; when no upper bound is imposed we replace the subscript by *. A system is said to be non-extended if V = T.

The transitions in a system as above are defined in the following way. In each region, each string which can be rewritten by a rule from that region is rewritten. The rule to be applied and the symbol rewritten by it are non-deterministically chosen. Each string obtained in this way is checked against the conditions P_i, F_i from that region. If it fulfils the requested conditions, then it will be immediately sent out of the membrane or to an inner membrane, if any exists; if it fulfils both in and out conditions, then it is sent either out of the membrane or to a lower membrane, non-deterministically choosing the direction- and non-deterministically choosing the inner membrane in the case when several directly inner membrane exist. If a string does not fulfil any condition, or it fulfills only in conditions and there is no inner membrane, then the string remains in the same region. If a string cannot be rewritten (this can be the case with strings from other membranes), then it is directly checked against the communication conditions, and as above, it leaves the membrane (or remains inside forever) depending on the result of this checking.

The language generated by the above system is denoted by $L(\Pi)$. The family of all languages $L(\Pi)$ generated by the system Π of degree at most $m \geq 1$ with permitting conditions of type α and forbidding conditions of type β, is denoted by $[E]LSP_m(rw, \alpha, \beta)$, $\alpha, \beta \in$ {empty, symb} $\cup$ {sub_k | $k \geq 2$}. If the degree of the systems is not bounded, then the subscript m is replaced by *.

Example 1. [1] Consider the system
$\Pi = (\{a, b\}, \{a, b\}, [_1 [_2]_2]_1, \{a\}, \phi, R_1, P_1, F_1, R_2, P_2, F_2)$

$R_1 = \{a \rightarrow bb\}$, $P_1 = \{(\text{true}, \text{in}), (\text{true}, \text{out})\}$ $F_1 = \{(a, \text{notin}), (a, \text{notout})\}$
$R_2 = \{b \rightarrow a\}$, $P_2 = \{(\text{true}, \text{out})\}$, $F_2 = \{(b, \text{notout})\}$
generates the language $L(\Pi(= \{b^{2^n} \mid n \geq 0\}$
An array is a mapping $A : Z^2 \rightarrow V \cup \#$ with a finite support, supp(A), where supp(A)= $\{v \in Z^2 \mid A(v) \neq \#\}$. In order to specify an array it is sufficient to specify the set (v, A(v)), for $v \in$ supp(A).

For example the L-shaped angle with equal arms is given by the sets, {((0, 0), a), ((1, 0), a), ((2, 0), a), ((3, 0), a), ((0, 1), a), ((0, 2), a),)) 0, 3), a)} all the other elements have # symbol. The empty array is denoted by λ and the set of all arrays over V is given by $V^{*2} = V^{+2} \cup \{\lambda\}$. Any subset of V^{*2} is an array language.

An array production $\wp$ over V is a triple $\wp = (W, A, B)$, where W is a finite subset of Z^2 and A, B are arrays with the supports included in W. For two arrays C, D over V and a production $\wp$ as above, we write $C \Rightarrow_{\wp} D$ if D can be obtained by replacing a sub array of C identical to A with B; all pixels of W which are blank in A should be blank also in C.

An Array P System [2] is a construct $\Pi = (V, T, \mu, F_1, ..., F_m, R_1, ..., R_m, i_0)$,where: V is the alphabet of the system, $T \subseteq V$ is the terminal alphabet, μ is a membrane structure with m membranes (labelled injectively by 1, 2, ..., m), F_i are finite set of arrays over V, representing the arrays initially present in the regions i, i = 1, 2, ..., m, of the system, $R_1, R_2, ..., R_m$ are finite sets of array-rewriting rules over V of the form $A \rightarrow B$ (tar); and i_0 is the output elementary membrane of μ. The system is said to be non-extended if V = T.

A computation in an array P system is defined in the same way as in a string rewriting P system with the successful computations being the halting ones. A computation is successful only if it stops, a configuration is reached where no rule can be applied to the existing arrays, The result of a halting computation consists of the arrays composed only of symbols from T placed in the membrane with label i_0 in the halting configuration.

Example 2**.**Consider the non-extended context-free system [2]

$\Pi = (\{a\}, \{a\}, \#, [_1 [_2 [_3]_3]_2]_1, \begin{matrix} a \\ a \end{matrix}, \phi, \phi, R_1, R_2, 3)$

$R_1 = \left\{ \begin{matrix} \# \\ \# \ a \end{matrix} \rightarrow \begin{matrix} a \\ \# \ a \end{matrix} (\text{in}) \right\}$, $R_2 = \left\{ \begin{matrix} a \ \# \\ \# \end{matrix} \rightarrow \begin{matrix} a \ a \\ \# \end{matrix} (\text{out}), \begin{matrix} a \ \# \ \# \\ \# \end{matrix} \rightarrow \begin{matrix} a \ a \ a \\ \# \end{matrix} (\text{in}) \right\}$, $R_3 = \phi$

The above system generates all L-shaped angles with equal arms, each arm being of length at least three.

An extended array-rewriting P system (of degree $m \geq 1$) with conditional communication is a construct $\Pi = (V, T, \mu, M_1, ..., M_m, R_1, P_1, F_1, ..., R_m, P_m, F_m)$, where V is the alphabet of the system, $T \subseteq V$ is the terminal alphabet, μ is a membrane structure with m membranes (injectively labelled by 1, 2, ..., m), $M_1, ..., M_m$ are finite set of arrays over V, representing the arrays initially present in the regions 1, 2, ..., m of the system, R_1, R_2, ..., R_m are finite sets of array-rewriting rules over V present in regions 1, 2, ..., m of the system, P_i are permitting conditions and F_i are forbidding conditions associated with region i, $1 \leq i \leq m$.

The conditions can be in the forms empty, symbol checking, subarray checking, which are defined analogous to the corresponding forms in the string case. The difference is that the objects are arrays. Thus we have the conditions of the types empty, symb, subarr in all P_i, F_i.

The transitions in an array-rewriting P system with conditional communication are analogous to the string case. But the result of a halting computation is as defined for array-rewriting P systems.

The set of all arrays computed by an array-rewriting P system Π with conditional communication is denoted by [E]AL(Π) with E being omitted when the system is non-extended. The family of array languages generated by systems as above is denoted by $[E]ALP_m(\text{arw}, \alpha, \beta)$, $\alpha, \beta \in \{\text{empty}, \text{symb}, \text{subarr}\}$. We illustrate with an example.

Example 3**.**Consider the non-extended system

$\Pi = (\{a\}, \{a\}, \#, [_1]_1, \begin{matrix} a \\ a \ a \end{matrix}, R_1, P_1, F_1, 1)$

$R_1 = \left\{ a \ \# \rightarrow a \ a, \begin{matrix} \# \\ a \end{matrix} \rightarrow \begin{matrix} a \\ a \end{matrix} \right\}$,

$P_1 = \{(a, \text{in})\}$, $F_1 = \{a, \text{notin}\}$,

generates all L-shaped pictures over a.

III. Pure 2d Context Free P System With Conditional Communication

A Pure 2D Context free P system with conditional communication is a construct $\Pi = (\Sigma \cup \#, \mu, M_1, M_2, ..., M_m, (R_1, P_1, F_1), (R_2, P_2, F_2), ..., (R_m, P_m, F_m), O)$ where: Σ is the alphabet of the system with only terminals, # is the special symbol for representing blanks, μ is a membrane structure with m membranes (injectively labelled by 1, 2, ..., m), M_i are finite languages over V, representing the strings initially present in the regions i, i = 1, 2, ..., m of the system, R_i are finite sets of pure context-free rules over Σ^{**} present in regions i, i = 1, 2, …, m of the system, P_i are permitting conditions and F_i are forbidding conditions associated with regions i, i = 1, 2, ..., m as discussed in the preliminaries and O the output membrane. We denote the new system as $P2DCFP_m(\alpha, \beta)$ where $\alpha, \beta \in$ {empty, symbol, subarray} the conditional communicating conditions. The languages generated by this system is given by $L(\Pi)$.

Example 4**.**Consider the construct

$\Pi = (\Sigma \cup \#, [_1 [_2]_2]_1, M_1, M_2, (R_1, P_1, F_1), (R_2, P_2, F_2), 2)$

with # = {a, b, c, d}, $M_1 = \begin{matrix} a & \bullet & a \\ b & c & b \\ d & \bullet & d \end{matrix}$, $M_2 = \phi$

$P_1 = \{\text{true, in}\}$, $F_1 = \left\{ \begin{matrix} \bullet & \# \bullet & \bullet c & \# c & c \\ c & c'c & \#'\bullet & \bullet'\bullet & \#' \end{matrix}, \text{notin, notout} \right\}$

$P_2 = \{\text{true, in}\}$, $F_2 = \left\{ \begin{matrix} a & \# \# & a\ b & \bullet\ \bullet & b\ b & c\ a & \bullet \\ b & \bullet'\bullet & b'd & \#'\# & d'd & \bullet'b & c' \end{matrix}, \text{notin, notout} \right\}$

Applying the above rules the output generates picture pattern of token H.

Theorem 1**.**The $P2DCFP_3$(subarr, subarr) generates token T patterns

Proof**.** Consider the construct

$\Pi = (\Sigma \cup \#, [_1 [_2]_2]_1, M_1, M_2, (R_1, P_1, F_1), (R_2, P_2, F_2), 2)$

with # = {a, b, c, d}, $M_1 = \begin{matrix} a & b & c \\ \bullet & d & \bullet \\ \bullet & e & \bullet \end{matrix}$, $M_2 = \phi$

$P_1 = \left\{ \begin{matrix} b & b\ b & b \\ \# & d'd & \# \end{matrix}, \text{in} \right\}$, $F_1 = \left\{ \begin{matrix} a & b\ b & c\ a & b\ b & c \\ \# & d'd & \#'\# & \bullet'\bullet & \#' \end{matrix}, \text{notin, notout} \right\}$

$P_2 = \{\text{true, in}\}$, $F_2 = \left\{ \begin{matrix} a & \# \# & a\ b & \bullet\ \bullet & b\ b & c\ a & \bullet \\ b & \bullet'\bullet & b'd & \#'\# & d'd & \bullet'b & c' \end{matrix}, \text{notin, notout} \right\}$

Applying the above rules the output generates picture pattern of token T.

IV. Comparisons And Closure Properties

Theorem 2**.**The class $P2DCFP_m\{\alpha, \beta\}$ intersects the class P2DCFL with regular control.

Proof. The result follows from example 4. □

Theorem 3.The class $P2DCFPP_m\{\alpha, \beta\}$ is incomparable with the families of RML and CFML.

Proof: $L_1 = (a)_{m \times n}$ can be generated by a $P2DPP_2$(subarr, aubarr) with

The corresponding permitting and forbidding rules can be framed as earlier to avoid irregular patterns. This language can also be generated by regular Siromoney Matrix grammar. But H tokens and T tokens cannot be generated CFML [10]. Consider $L_2 = \{a^n b^n / n \geq 1\}$ is a CFML but cannot be generated by the system $P2DCFPP_m(\alpha, \beta)$. Also $L_3 = aaabbb^n (ab)^*$ is a RML but not a pure CFL.

Theorem 4.The class $P2DCFPP_m\{\alpha, \beta\}$ intersects with LOC and REC.

Proof: Consider the picture languages $L_4 = I\ m\times m$, where I is the identity matrix is in LOC and hence in REC. This L_4 can be generated by a construct
$\Pi = (\Sigma \cup \#, [_1 [_2]_2]_1, M_1, M_2, (R_1, P_1, F_1), (R_2, P_2, F_2), 2)$
with $\Sigma = \{0, 1\}$, $M_1 = \begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix}$, $M_2 = \phi$

$P_1 = \{\text{true, in}\}$, $F_1 = \left\{ \begin{matrix} 0 & 0 & \# & 1 \\ 1 & \#' & 0 & 0 \end{matrix}, \text{notin}, \text{notout} \right\}$

$P_2 = \{\text{true, in}\}$, $F_2 = \left\{ \begin{matrix} 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & \#' & 0 & \#' & 0 & 0' & 0 & \#' & \# & 0 \end{matrix}, \text{notin}, \text{notout} \right\}$

Theorem 5. The class $P2DCFP_m\{\alpha, \beta\}$ is closed under column concatenation and row concatenation

Proof**:** Consider $L_5 = \{(a)^n c(b)^n / n \geq 1\}$ and $L_6 = \{(x)^n c(y)^n / n \geq 1\}$ both can be generated by $P2DCFPP_m(\alpha, \beta)$. It can be proved that $L_5 \circ L_6$ can be generated by the system $P2DCFPP_2$(empty, substring).

V. Conclusion

It has been found that the power of the new system has increased with the puzzle and conditional communication feature. The new system also satisfies certain closure properties. Further investigations like comparison with Tabled 0L systems [12] can also be studied in future work.

Acknowledgement

The first author is thankful to the University Grants Commission-SERO: Project no.F. MRP-4254/12 Link No.4254 under which this work was done by her.

References

[1] P. Bottoni, A. Labella, C. Martin-vide, Gh. Păun, "Rewriting P Systems with Conditional Communication." Lecture Notes in Computer Science, Springer, Berlin, vol. 2300, 2002, 325–353.
[2] R. Ceterchi, M. Mutyam, Gh. Păun, K.G. Subramanian, "Array - Rewriting P Systems." Natural Computing, vol. 2, 2003. 229–249.
[3] Gh. Păun, Membrane Computing (An Introduction. Springer, 2002).
[4] S. Hemalatha, K.S. Derasanambika, K.G. Subramanian, C. Sri Hari Nagore, "Array-Rewriting P Systems with Conditional Communication," Lecture Note Series, Ramanujan Mathematical Society, vol. 3,2006, 155–160.
[5] K.G. Subramanian, R. Siromoney, V.R. Dare, A, Saoudi, "Basic Puzzle Languages," Int. Journal of Pattern Recognition and Artificial Intelligence, vol. 9,1995. 763–775.
[6] R. Ceterchi, M. Mutyam, Gh. Paun, K.G. Subramanian, "Array-rewriting P systems," Natural Computing, vol. 2, 2003, 229–249.
[7] M. Nivat, A. Saoudi, K.G. Subramanian, R. Siromoney, V.R. Dare, "Puzzle Grammars and Context-free Array Grammars," Int. Journal of Pattern Recognition and Artificial Intelligence, vol. 5, 1991, 663–6761.
[8] Gh. Păun, Membrane Computing : An Introduction,(Springer-Verlag Berlin, Heidelbrg, 2000).
[9] Rosenfeld, "Picture Languages," (Academic Press, 1979).
[10] Rosenfeld and R. Siromoney, "Picture languages - a survey," Languages of design, vol. 1, 1993, 229–245.
[11] Salomaa, Formal languages,(Academic Press, London, 1973).

[12] R. Siromoney and G. Siromoney, "Extended Controlled Tabled L-arrays," Information and Control, vol. 35, no. 2,1977,119–138.
[13] K.G. Subramanian, M. Geethalakshmi, P. Helen Chandra, "Array rewriting P systems generating rectangular arrays," Paper presented at the National conference on Intelligent Optimization Modeling, Gandhigram Rural Institute-Deemed University, Gandhigram, India, March 2006.
[14] K.G. Subramanian, R. Saravanan, K. Rangarajan, "Array P systems and Basic puzzle grammars," Paper presented at the National conference on Intelligent Optimization Modeling, Gandhigram Rural Institute-Deemed University, Gandhigram, India, March 2006.
[15] Y. Yamamoto, K. Morita, K. Sugata, Context-sensitivity of two-dimensional array grammars,in P.S.P. Wang(Ed), Array grammars, Patterns and recognizers, (World Scientific, 1989), 17–41.
[16] K.G. Subramanian, S. Hemalatha, C. Sri Hari Nagore, M. Margernstern, "On the power of P systems with parallel rewriting and conditional communication," Romanian Journal of Information Science and Technology, vol. 10, no. 2,2007, 137–144.
[17] H.A. Maurer, A, Salomaa, D. Wood, "Pure Grammars," Information and Control, vol. 44, 1980, 47–72.
[18] K.G. Subramanian, Atulya K. Nagar, M. Geethalakshmi, Pure 2D Picture Grammars (P2DPG) and P2DPG with Regular Control.