Theoretical Computer Science 968 (2023) 114027

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

An array P system based on a new variant of pure 2D context-free grammars [☆], [☆☆]

Somnath Bera [a], Atulya K. Nagar [b], Sastha Sriram [c], K.G. Subramanian [b,*]

[a] School of Advanced Sciences-Mathematics, Vellore Institute of Technology, Chennai, Tamil Nadu 600 127 India
[b] School of Mathematics, Computer Science and Engineering, Liverpool Hope University, Hope Park, Liverpool L16 9JD, UK
[c] Department of Mathematics, School of Arts, Sciences, Humanities and Education, SASTRA Deemed University, Thanjavur, Tamil Nadu 613 401 India

A R T I C L E   I N F O

Article history:
Received 19 December 2022
Received in revised form 31 May 2023
Accepted 7 June 2023
Available online 12 June 2023

Keywords:
Membrane computing
P system
Two-dimensional languages
Pure 2D context-free grammars

A B S T R A C T

Pure 2D context-free grammar ($P2DCFG$) with an independent mode of array rewriting, was recently introduced and named as $IP2DCFG$. Here we consider a variant of $IP2DCFG$, called $(l/u)IP2DCFG$, by requiring rewriting of the leftmost (respy. uppermost) symbol in every row (respy. column) of an array, with the symbol having a rewriting rule in a given set of pure context-free rules. We introduce an array P system with $(l/u)IP2DCFG$ kind of rules and array rewriting in its membranes. When two membranes are used in the array P system, the array generative power is increased compared to using a single membrane.

1. Introduction

Motivated by the seminal work [10] of Păun, especially, the variant of a P system with string objects and evolution rules involving the rewriting operation, the two areas of membrane computing [11,12,23] and two-dimensional formal language theory [7,13,14,22] were linked in [4], by defining a cell-like array P system with array-objects in the compartments of a membrane structure and array-rewriting rules of the isometric variety to evolve the arrays. Several kinds of array P systems were subsequently introduced and investigated involving rectangular or non-rectangular array-objects and isometric or non-isometric kind of rewriting rules (see, for example, [18,21] and references therein). Unlike the non-isometric array rewriting rules, the isometric array rewriting rules preserve the geometric shape of the rewritten subarray. In the recent past, a 2D grammar model, called pure 2D context-free grammar ($P2DCFG$), which was introduced in [19,20], has been a simple and at the same time an effective non-isometric 2D grammar model generating two-dimensional picture languages consisting of rectangular picture arrays. In a $P2DCFG$, all symbols in any column or any row of the rectangular array are rewritten at a time by applicable rules in a set of pure context-free rules, called table of rules, with equal length strings on the right sides of the rules, thus maintaining the array to be rectangular. In [2,3,19,20], several properties of $P2DCFG$ have been established while a variant of $P2DCFG$, called $(l/u)P2DCFG$, has been introduced in [8], with the feature that rewriting of all symbols only in the leftmost column or in the uppermost row of an array, is done in a $(l/u)P2DCFG$. Recently,

☆ This article belongs to Section C: Theory of natural computing, Edited by Lila Kari.
☆☆ This work is a revised and enhanced version of a paper presented in the Asian conference on membrane computing held at the University of Philippines Diliman during September 2022.
* Corresponding author.
E-mail address: kgsmani1948@gmail.com (K.G. Subramanian).

https://doi.org/10.1016/j.tcs.2023.114027

another variant of $P2DCFG$, called pure 2D context-free grammar with independent mode of rewriting, called $IP2DCFG$ was introduced in [1]. In an $IP2DCFG$ rules of a column table are applied to symbols in all the rows rewriting one symbol in each row but not necessarily in the same column unlike in a $P2DCFG$. Likewise rules of a row table are applied to symbols in all the columns rewriting one symbol in each column but not necessarily in the same row.

Motivated by an (l/u) mode of rewriting of a picture array, here we consider pure 2D context-free grammars in independent mode ($IP2DCFG$) and require rewriting of the leftmost (respy. uppermost) symbol in every row (respy. column) of an array, with the symbol having a rewriting rule in a column table (respy. a row table) of pure context-free rules [9]. We call the resulting array grammar as $(l/u)IP2DCFG$. The resulting class $(l/u)IP2DCFL$ of picture array languages is shown to be incomparable but not disjoint with each of the families of $P2DCFL$ and $IP2DCFL$. We then consider an array P system with $(l/u)IP2DCFG$ kind of rewriting using tables of rules as in a $P2DCFG$. We show that the use of two membranes gives more picture array generative power with respect to the same kind of P systems, but having only one membrane. We also construct an array P system with three membranes and $(l/u)IP2DCFG$ kind of rules in its regions, generating a context-sensitive two-dimensional picture array language in the class $CSML$ introduced in [17].

2. Preliminaries

A finite sequence $w = a_1a_2 \cdots a_n$, $(n \geq 1)$ of symbols $a_i, 1 \leq i \leq n$, belonging to a finite alphabet T is called a non-empty word (or non-empty string) over T and the length of the word w is denoted by $|w|$. The set of all words (also called as *row words*) over T is denoted by T^* which includes the empty word λ with no symbols. Given a word $w = a_1a_2 \cdots a_n$, we denote by w^t the word w written vertically as follows and call it a *column word*:

$$\begin{matrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{matrix}.$$

Note that $(w^t)^t = w$ and if w^t is a column word, then $(w^t)^t$ is a row word.

An $m \times n$ rectangular array p over T (also called picture array), is of the form

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

where each $p_{ij} \in T, 1 \leq i \leq m,\ 1 \leq j \leq n$. We denote by T^{**} the set of all picture arrays over T, including the empty array λ and $T^{++} = T^{**} - \{\lambda\}$.

For notions on formal languages and array grammars, the reader can refer to [13–16] while for concepts concerning P Systems we refer to [10,11]. Pure 2D context-free grammar with independent mode of rewriting, introduced in [19,20] is now recalled.

Definition 1. A pure 2D context-free grammar in independent mode
($IP2DCFG$) is a quadruple $G = (T, P_1, P_2, I)$ where

i) T is a finite set of symbols;
ii) P_1 is a finite set of column tables c, where c is a finite set of pure context-free rules of the form $a_1 \to x_1, a_1 \in T, x_1 \in T^*$ with the property that the words x_1 and x_2 have equal length for any two rules $a_1 \to x_1, a_2 \to x_2$ in c, *i.e* $|x_1| = |x_2|$;
iii) P_2 is a finite set of row tables r, where r is a finite set of pure context-free rules of the form $b_1 \to y_1^t, b_1 \in T, y_1 \in T^*$ such that for any two rules $b_1 \to y_1^t, b_2 \to y_2^t$ in r, we have $|y_1| = |y_2|$;
iv) $I \subseteq T^{**} - \{\lambda\}$ is a finite set of axiom arrays.

A direct derivation in a $IP2DCFG$ G is defined as in a $P2DCFG$ [19,20] but rewriting of a picture array is done as given below and we call the rewriting as an independent mode of rewriting. Also at each derivation step we can apply either a column table of rules, or a row table of rules, whenever such a table of rules is available and applicable. A picture array p_2 is obtained directly from an $m \times n$ picture array p_1, written as $p_1 \Rightarrow_i p_2$, as follows: In applying the rules of a column table c to the $m \times n$ picture array p_1, only one symbol in each of the m rows is rewritten at a time and it can be any symbol in that row. Likewise in an application of the rules of a row table r to p_1, only one symbol in each of the n columns is rewritten at a time and again it can be any symbol in that column. All the symbols chosen for rewriting should have rules in the respective table; Otherwise, the table of rules is not applicable. If a picture array p' is obtained from a picture array p using a $IP2DCFG$, through a sequence of direct derivation steps, we write $p \Rightarrow_i^* p'$. The array derived is also a rectangular array since the lengths of the right sides of all the rules in a column table or a row table, are the same.

The picture array language generated by a $IP2DCFG$ G is the set of picture arrays $L(G) = \{p \in T^{**} \mid p_0 \Rightarrow_i^* p$ for some $p_0 \in I\}$. The family of picture array languages generated by $IP2DCFGs$ is denoted by $IP2DCFL$. We illustrate with an example.

Example 2.1. Consider the $IP2DCFG$ $G_1 = (T, P_1, P_2, \{p_0\})$ where
$T = \{a, b, x, d\}$, $P_1 = \{c\}$, $P_2 = \{r\}$, where

$$c = \{a \to ab, x \to xx, d \to bd\}, r = \left\{ a \to \begin{matrix} a \\ b \end{matrix}, x \to \begin{matrix} x \\ x \end{matrix}, d \to \begin{matrix} b \\ d \end{matrix} \right\},$$

and

$$p_0 = \begin{matrix} a & b & b \\ b & x & b \\ b & b & d \end{matrix}.$$

G_1 generates a picture array language L_1 consisting of picture arrays p of size $m \times n$, $m, n \geq 3$ with $p(1,1) = a$, $p(1, j) = p(i, 1) = p(m, k) = p(l, n) = b$, for $2 \leq j \leq n$, $2 \leq i \leq m$, $2 \leq k \leq n-1$, $2 \leq l \leq m-1$, $p(m,n) = d$ and $p(i, j) = x$, otherwise. We note that a derivation in G_1, starting from the axiom array p_0, generates picture arrays of the form

$$\begin{matrix} a & b & b & \cdots & b & b \\ b & x & x & \cdots & x & b \\ b & x & x & \cdots & x & b \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ b & x & x & \cdots & x & b \\ b & b & b & \cdots & b & d \end{matrix}.$$

Note that when a column table of rules is used, a symbol in each row (not necessarily in the same column) is rewritten while a symbol in each column (not necessarily in the same row) is rewritten when a row table of rules is used. For example, if the rules of the column table c are applied to the axiom array p_0, then the symbol a in the first row, an x in the second row and the d in the third row can be rewritten to yield the array

$$\begin{matrix} a & b & b & b \\ b & x & x & b \\ b & b & b & d \end{matrix}.$$

3. Pure 2D context-free grammar in independent mode and (l/u) kind of rewriting

We now introduce the (l/u) kind of rewriting in the pure 2D context-free grammar in independent mode, resulting in another variant of $P2DCFG$, which we call as $(l/u)IP2DCFG$.

Definition 2. A pure 2D context-free grammar in independent mode with (l/u) kind of rewriting ($(l/u)IP2DCFG$) $G = (T, P_1, P_2, I)$ has its components
T, P_1, P_2, I as in the $IP2DCFG$ in Definition 1, with a difference in the mode of rewriting of a picture array and which is done as given below:
A direct derivation of a picture array p_2 from an $m \times n$ picture array p_1, written as $p_1 \Rightarrow p_2$, is done in the following manner: In applying the rules of a column (respy. row) table c (respy. r) to the $m \times n$ picture array p_1, among the symbols in the rows (respy. columns) which can be rewritten by rules in c (respy. r), only the leftmost (respy. uppermost) symbol in each of the m rows (respy. n columns) is rewritten at a time. If a picture array M_2 is obtained from a picture array M_1 using the $(l/u)IP2DCFG$, through a sequence of direct derivation steps, we write $M_1 \Rightarrow^* M_2$. Note that the lengths of the right sides of all the rules in a column table or a row table, are the same and so the derived array is also a rectangular array.
The picture language generated by a $(l/u)IP2DCFG$ G is the set of picture arrays $L(G) = \{M \in T^{**} \mid M_0 \Rightarrow^* M$ for some $M_0 \in I\}$. The family of picture languages generated by $(l/u)IP2DCFGs$ is denoted by $(l/u)IP2DCFL$.

We illustrate with an example.

Example 3.1. Consider the $(l/u)IP2DCFG$ $G_2 = (T, P_1, P_2, \{M_0\})$ where
$T = \{a, b, d, e, x, y\}$, $P_1 = \{c_1, c_2\}$, $P_2 = \{r\}$, where

$$c_1 = \{a \to ab, x \to xy, y \to yx, d \to bd\}, c_2 = \{a \to a, x \to x, y \to y, d \to e\},$$

$$\begin{array}{cccccc} a & b & b & \cdots & b & b \\ b & x & y & \cdots & y & b \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ b & x & y & \cdots & y & b \\ b & x & y & \cdots & y & b \\ b & y & x & \cdots & x & b \\ b & b & b & \cdots & b & d \end{array} \qquad \begin{array}{cccccc} a & b & b & \cdots & b & b \\ b & x & y & \cdots & y & b \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ b & x & y & \cdots & y & b \\ b & x & y & \cdots & y & b \\ b & y & x & \cdots & x & b \\ b & b & b & \cdots & b & e \end{array}$$

Fig. 1. Picture arrays of Example 3.1.

$$M_0 = \begin{array}{cccc} \boxed{a} & b & b & b \\ b & \boxed{x} & y & b \\ b & \boxed{y} & x & b \\ b & b & b & \boxed{d} \end{array} \Rightarrow \begin{array}{ccccc} \boxed{a} & b & b & b & b \\ b & \boxed{x} & \boxed{y} & \boxed{y} & b \\ b & y & x & x & b \\ b & b & b & b & \boxed{d} \end{array} \Rightarrow$$

$$\begin{array}{ccccc} \boxed{a} & b & b & b & b \\ b & \boxed{x} & y & y & b \\ b & \boxed{x} & y & y & b \\ b & \boxed{y} & x & x & b \\ b & b & b & b & \boxed{d} \end{array} \Rightarrow \begin{array}{ccccc} a & b & b & b & b \\ b & x & y & y & b \\ b & x & y & y & b \\ b & y & x & x & b \\ b & b & b & b & e \end{array} = M$$

Fig. 2. Derivation $M_0 \Rightarrow^* M$.

$$r = \left\{ a \rightarrow \begin{array}{c} a \\ b \end{array}, x \rightarrow \begin{array}{c} x \\ x \end{array}, y \rightarrow \begin{array}{c} y \\ y \end{array}, d \rightarrow \begin{array}{c} b \\ d \end{array} \right\}$$

and

$$M_0 = \begin{array}{cccc} a & b & b & b \\ b & x & y & b \\ b & y & x & b \\ b & b & b & d \end{array}.$$

G_2 generates a picture language L_2 consisting of picture arrays p of size $m \times n$, $m, n \geq 4$ with $p(1,1) = a$, $p(1, j) = p(m,k) = p(i,1) = p(l,n) = b$, for $2 \leq j \leq n$, $2 \leq k \leq n-1$, $2 \leq i \leq m$ and $2 \leq l \leq m-1$, $p(i,2) = p(m-1, j) = x$, for $2 \leq i \leq m-2$ and $3 \leq j \leq n-1$, $p(m,n) = d$ or e, $p(i,j) = y$, otherwise. We note that a derivation in G_2, starting from the axiom array M_0, generates picture arrays of the form shown in Fig. 1. A sample derivation $M_0 \Rightarrow^* M$ using the tables c_1, r, c_2 in this order is shown in Fig. 2. We have indicated the symbols rewritten by enclosing these in rectangular boxes. Note that when a column table of rules is used, in each row, a symbol which can be rewritten by a rule of the table and which is the leftmost symbol among such symbols in the row, is rewritten. If any of the rows has no symbol for which there is a rule in the table which can rewrite it, then the table of rules is not applicable. Likewise a row table of rules is used. For example, in applying the rules of the table c_1 to the array M_0, the symbol a in the first row, is rewritten as this symbol is the leftmost symbol having a rule in the table c_1 that can be used to rewrite it. The symbol x in the second row is the leftmost symbol in this row that has a rule in the table c_1 which can be used to rewrite it and so this symbol is rewritten. Likewise, the symbol y in the third row and d in the fourth row are the leftmost symbols in the respective rows having rewriting rules in the table c_1 and so these are rewritten, thus completing the application of the rules of the table c_1 to the array M_0, yielding the second array in the derivation. Likewise, the subsequent steps of the derivation are done.

Lemma 1. (i) $(l/u)IP2DCFL \setminus P2DCFL \neq \emptyset$
(ii) $(l/u)IP2DCFL \setminus IP2DCFL \neq \emptyset$

Proof. It is clear from Example 3.1 that $L_2 \in (l/u)IP2DCFL$. We show that $L_2 \notin P2DCFL$. If we assume that L_2 can be generated by a $P2DCFG$, we first note that in a $P2DCFG$ all the symbols in a single column are to be rewritten if the symbols have rules in a column table. The first column or the last column of a picture array in L_2 has many consecutive $b's$ with b as the last symbol in the first column and the first symbol in the last column. One such array needs to be an axiom array in the $P2DCFG$ to be constructed. Then the column table of rules used to rewrite the first column or the last column of such an array should have a rule for b of the form $b \rightarrow b \cdots b$. But then this kind of a rule can be applied to any other b in the first column and likewise in the last column. Obviously this will yield pictures not in L_2. On the other hand if symbols in a column other than the first and last columns in a picture array L_2 are rewritten, we have to include suitable rules for x, y and b. In fact in order to rewrite symbols in the second column, we require a column table t of rules which includes a rule for x of the form $x \rightarrow y \cdots y$. But then such a table of rules can be used to rewrite symbols in another column. So if symbols in a column (other than the first column, second column and the last column) of the picture array, are rewritten, then the rule for x in t could be used for rewriting the symbol x in the column chosen. Again this will result in picture arrays not in the language L_2. This proves that $(l/u)IP2DCFL \setminus P2DCFL \neq \emptyset$.

The argument for $IP2DCFG$ is not much different from the argument mentioned above for the case of $P2DCFG$. The main difference is that not all symbols in a single column need be rewritten at a time. So a little reflection will tell that any kind of rule for b or x or y included in a column table, will only yield picture arrays that do not belong to L_2 while inclusion of rules only for a and e in a column table, is not enough to rewrite an array in $IP2DCFG$ mode. This proves that $(l/u)IP2DCFL \setminus IP2DCFL \neq \emptyset$. □

Lemma 2. *(i)* $P2DCFL \setminus (l/u)IP2DCFL \neq \emptyset$
(ii) $IP2DCFL \setminus (l/u)IP2DCFL \neq \emptyset$

Proof. Consider the picture array language generated by the $P2DCFG$ with axiom $\begin{matrix} a & b & a \\ b & c & b \end{matrix}$ and a column table of rules $\{b \to aba, c \to bcb\}$. The picture arrays generated by this $P2DCFG$ are of the form $\begin{matrix} a & \cdots & a & b & a & \cdots & a \\ b & \cdots & b & c & b & \cdots & b \end{matrix}$. But this language cannot be generated by any $(l/u)IP2DCFG$. In fact in order to generate an equal number of $a's$ to the left and right of the b in the first row, we need a rule of the form $b \to a^m b a^m$ for some $m \geq 1$. But then this rule can be applied to the first b in the second row in a $(l/u)IP2DCFG$. This will yield arrays not in the language. Hence this language cannot be generated by any $(l/u)IP2DCFG$. This proves that $P2DCFL \setminus (l/u)IP2DCFL \neq \emptyset$.

Consider the picture language generated by the $IP2DCFG$ with axiom $\begin{matrix} a & a \\ a & b \end{matrix}$, column table of rules $\{a \to aa, b \to bb\}$. Picture arrays generated are of the form $\begin{matrix} a & a & \cdots & a \\ a & b & \cdots & b \end{matrix}$. This language cannot be generated by any $(l/u)IP2DCFG$. In fact to generate the first row of the arrays, a rule of the form $a \to a^m$, $m \geq 1$ is needed. But then in a $(l/u)IP2DCFG$, the first and the only a in the second row of the array is to be rewritten. This will then yield arrays not in the language. □

Theorem 3.1. *The family $(l/u)IP2DCFL$ is incomparable but not disjoint with each of the families $P2DCFL$ and $IP2DCFL$.*

Proof. Consider the picture array language L_a consisting of $m \times n$ $(m, n \geq 2)$ picture arrays p over $\{a\}$ where $p(i, j) = a$, for all $1 \leq i \leq m, 1 \leq j \leq n$. This language L_a is generated by a $P2DCFG$ as well as a $IP2DCFG$ and a $(l/u)IP2DCFG$ with a column table of rules $c = \{a \to aa\}$, a row table of rules $\left\{ a \to \begin{matrix} a \\ a \end{matrix} \right\}$ and axiom array $\begin{matrix} a & a \\ a & a \end{matrix}$.

The incomparability of $(l/u)IP2DCFL$ with $P2DCFL$ follows from Lemma 1 while the incomparability of $(l/u)IP2DCFL$ with $IP2DCFL$ follows from Lemma 2. □

4. Array P system based on $(l/u)IP2DCFG$

An array P system is considered now with the membranes of the P system containing picture array objects and column or row tables of rules as in a $P2DCFG$ but rewriting of arrays is done as in a $(l/u)IP2DCFG$ in the sense that the rewriting is in independent mode with the leftmost (respy. uppermost) symbol in every row (repy. column) which can be rewritten by a rule of the respective table, being rewritten.

Definition 3. An array P system (of degree $m \geq 1$) with $(l/u)IP2DCFG$ kind of rules is a construct

$$\Pi = (T, \mu, F_1, \cdots, F_m, P_1, \cdots, P_m, i_o),$$

where T is the alphabet consisting of terminal symbols, μ is a membrane structure with m membranes labelled in a one-to-one manner with $1, 2, \cdots, m$; $F_1, \cdots, F_m$ are finite sets (can be empty) of rectangular picture arrays over T with F_i being in the membrane or region labelled i for $1 \leq i \leq m$; $P_1, \cdots, P_m$ are finite sets of column tables or row tables of pure context-free rules over T (as in a $P2DCFG$) with P_i being in the membrane or region labelled i for $1 \leq i \leq m$. A region can contain both column tables of rules and row tables of rules. The application of a column or row table is as done in a $(l/u)IP2DCFG$. The tables have attached targets *here, out, in, in*$_j$ (in general, *here* is omitted) and i_o is the label of the output membrane which is an elementary membrane of μ.

As done in an array-rewriting P system [4], a computation in Π is done with the successful computations being the halting ones; each rectangular picture array in each region of the system, which can be rewritten by a column table of rules or a row table of rules, associated with that region, should be rewritten. This means that a region can contain column tables of rules and/or row tables of rules but one table (column or row) of rules is applied at a time to a picture array and the rewriting is done as in a $(l/u)IP2DCFG$. If the target associated with the table used is *here*, the picture array obtained by rewriting is retained in the same region while it is sent to an immediate outer region (respy. directly inner region, nondeterministically chosen), if the target is *out* (respy. *in*). If the target is *in*$_j$ then the array is immediately sent to a directly inner membrane with label j. If no internal membrane exists, then a table with the target indication *in* cannot be used. A computation is successful only if it stops, that is, a configuration is reached where no table of rules can be applied to the existing arrays in the regions.

The result of a halting computation consists of rectangular picture arrays over T collected in the output membrane with label i_o in the halting configuration. Note that all the picture arrays that stay at the output membrane in the halting configuration will belong to the picture language since there is only one kind of symbols, namely terminal symbols.

The set of all picture arrays generated by such a system Π is denoted by $(l/u)IAL(\Pi)$. The family of all picture array languages $(l/u)IAL(\Pi)$ generated by such systems Π as above, with at most m membranes, is denoted by $IAP_m((l/u)IP2DCFG)$.

Example 4.1. Consider the picture language L_3 consisting of square sized $n \times n, n \geq 4$, picture arrays p with $p(1,1) = a$, $p(1,j) = p(i,1) = p(n,k) = p(l,n) = b$, for $2 \leq j \leq n$, $2 \leq i \leq n$, $2 \leq k \leq n-1$ and $2 \leq l \leq n-1$, $p(i,2) = p(n-1,j) = x$, for $2 \leq i \leq n-2$ and $3 \leq j \leq n-1$, $p(n,n) = e$, $p(i,j) = y$, otherwise. We construct an array P system Π_3 with only one membrane that generates L_3. The membrane region contains tables having $(l/u)IP2DCFG$ kind of rules. Π_3 is given by

$$\Pi_3 = (T, \mu, F_1, P_1, 1),$$

where

i) $T = \{a, b, d_1, d_2, e, x, y\}$
ii) $\mu = [_1\,]_1$
iii) $F_1 = \left\{ M_0 = \begin{matrix} a & b & b & b \\ b & x & y & b \\ b & y & x & b \\ b & b & b & d_1 \end{matrix} \right\}$,
iv) P_1 consists of two column tables c_1 and c_2 and a row table r each having target *here*. The tables of rules are given as follows: $c_1 = \{a \to ab, d_1 \to bd_2, x \to xy, y \to yx\}$, $c_2 = \{a \to a, d_1 \to e, x \to x, y \to y\}$, $r = \{a \to \begin{matrix} a \\ b \end{matrix}, x \to \begin{matrix} x \\ x \end{matrix}, y \to \begin{matrix} y \\ y \end{matrix}, d_2 \to \begin{matrix} b \\ d_1 \end{matrix}\}$.

In the array P system Π_3, the membrane labelled 1 initially contains the array

$$\begin{matrix} a & b & b & b \\ b & x & y & b \\ b & y & x & b \\ b & b & b & d_1 \end{matrix}.$$

If the column table c_2 is applied, then the array generated is

$$\begin{matrix} a & b & b & b \\ b & x & y & b \\ b & y & x & b \\ b & b & b & e \end{matrix}.$$

This is collected in the language. Note that membrane 1 itself is the output membrane and no table of rules is applicable at this moment and the computation comes to a halt. If the column table c_1 (instead of c_2) is applied to the axiom array in region 1, then the array generated is

$$\begin{matrix} a & b & b & b & b \\ b & x & y & y & b \\ b & y & x & x & b \\ b & b & b & b & d_2 \end{matrix}.$$

The row table r can be applied now which generates the array

$$M = \begin{matrix} a & b & b & b & b \\ b & x & y & y & b \\ b & x & y & y & b \\ b & y & x & x & b \\ b & b & b & b & d_1 \end{matrix}.$$

Note that the process can be repeated. If the column table c_2 is applied during the process instead of c_1 in region 1, the symbol d_1 is changed to e in the array M and the generated array is

$$\begin{array}{ccccc} a & b & b & b & b \\ b & x & y & y & b \\ b & x & y & y & b \\ b & y & x & x & b \\ b & b & b & b & e \end{array}.$$

The computation comes to a halt and the array is collected in the language generated by Π_3. The generated array at the halting configuration will have an equal number of rows and columns and will be an element of L_3. Thus the system Π_3 generates the language L_3.

We now examine the generative power of the array-rewriting P systems with $IP2DCFG$ kind of rules in independent mode of derivation.

Theorem 4.1. $(l/u)IP2DCFL \subset IAP_1((l/u)IP2DCFG)$.

Proof. For proving the inclusion $(l/u)IP2DCFL \subseteq IAP_1((l/u)IP2DCFG)$, consider a picture array language $L \in (l/u)IP2DCFL$. Let $G = (T, P_1, P_2, I)$ be an $(l/u)IP2DCFG$ generating L. An array P system of degree 1, Π is now constructed with $(l/u)IP2DCFG$ kind of application of $P2DCFG$ type of rules. $\Pi = (T \cup \overline{T}, [_1\,]_1, F, P, 1)$ where $\overline{T} = \{\overline{a} \mid a \in T\}$. In other words Π contains all the symbols of T as well as the "barred" version of every symbol of T. Every symbol in each picture array of I is replaced by its barred version and F contains all these picture arrays. The array obtained from the array $M \in I$ by replacing each symbol of M by the corresponding barred symbol is denoted by $\overline{M}$. P contains all the column tables of P_1 and all the row tables of P_2 but each symbol in the right and left sides of every rule in the tables, is replaced by the corresponding barred symbol. In addition P contains a new column table $c = \{\overline{a} \to a \mid a \in T\}$. It can be seen that for every direct derivation $M_1 \Rightarrow M_2$ in G, there is a computation in Π with the array $\overline{M_1}$ generating $\overline{M_2}$ which then generates M_2 by the application of the rules of the column table c (until all the barred symbols are changed into their original symbols) and the computation halts. Hence every picture array in L generated in G from an axiom array in I is also computed by Π. Thus $L \in IAP_1((l/u)IP2DCFG)$.

The proper inclusion follows from the picture array language L_3 in Example 4.1 which shows that $L_3 \in IAP_1((l/u)IP2DCFG)$. On the other hand this picture language cannot be generated by any $(l/u)IP2DCFG$ since we need to have some control on the application of column tables of rules and row tables of rules to maintain square shape of the picture arrays generated by a $(l/u)IP2DCFG$. In fact if we assume that L_3 can be generated by a $(l/u)IP2DCFG$, then by the definition of $(l/u)IP2DCFG$ kind of rules and the mode of rewriting, a column table of rules cannot have any rule for the symbol b since such a rule will have to be applied only to the leftmost b in each row and clearly this will result in a picture array not in L_3. Thus the $(l/u)IP2DCFG$ should have a column table of rules with rules for a, x, y, e. Likewise a row table of rules also can have rules only for a, x, y, e. But then there is no restriction on the number of times this column or row table of rules can be applied. This means that the picture arrays of L_3 only cannot be generated as the arrays in L_3 are all square sized arrays. The argument that we can use some "intermediate" symbols in order to alternate the application of the column and row tables of rules is also not possible as this will result in picture arrays having these "intermediate" symbols, that are not in the language. □

Theorem 4.2. $IAP_1((l/u)IP2DCFG) \subset IAP_2((l/u)IP2DCFG)$.

Proof. The inclusion $IAP_1((l/u)IP2DCFG) \subseteq IAP_2((l/u)IP2DCFG)$ is simply a consequence of the definition of the family $(IAP_m((l/u)IP2DCFG)$.

For the proper inclusion we consider the language L consisting of picture arrays p of size $m \times (2n+5)$, $m \geq 4$, $n \geq 1$ over $\{a, b, d, e, p, q, x, y\}$. The array p has its first row in the form ab^{2n+4} $(n \geq 1)$ and the last two rows are respectively of the forms $bed^n eqp^n b$ and $b^{2n+4}y$. All other rows are of the form $bde^n dpq^n b$. The language L belongs to $IAP_2((l/u)IP2DCFG)$ generated by the P system with two membranes having the membrane structure $[_1\, [_2\,]_2\,]_1$. The only axiom array

$$\begin{array}{ccccccc} a & b & b & b & b & b & b \\ b & d & e & d & p & q & b \\ b & e & d & e & q & p & b \\ b & b & b & b & b & b & x \end{array}$$

is in membrane 1 initially. Membrane 1 has a row table r with target *here* and a column table c_1 with target *in*. Membrane 2 is the output membrane and has a column table c_2 with target *out* and another column table c_3 with target *here*.

The tables of rules are given below:

$r = \{a \to \begin{array}{c} a \\ b \end{array}, d \to \begin{array}{c} d \\ d \end{array}, e \to \begin{array}{c} e \\ e \end{array}, p \to \begin{array}{c} p \\ p \end{array}, q \to \begin{array}{c} q \\ q \end{array}, x \to \begin{array}{c} b \\ x \end{array}\}$, $c_1 = \{a \to ab, d \to de, e \to ed, x \to bx\}$, $c_2 = \{a \to ab, p \to pq, q \to qp, x \to bx\}$, $c_3 = \{a \to ab, p \to pq, q \to qp, x \to by\}$.

Application of the rules of the row table r will add in general, a row of the form $bde^n pq^n b$ for some $n \geq 1$, to a picture array in membrane 1 and the resulting picture array will remain in membrane 1. This row table can be applied any number of times. At any moment an application of the rules of the column table c_1 in membrane 1 can be done to such an array. If the computation starts applying the rules of the column table c_1 in membrane 1 to the axiom array, the generated array is

$$\begin{array}{cccccccc} a & b & b & b & b & b & b & b \\ b & d & e & e & d & p & q & b \\ b & e & d & d & e & q & p & b \\ b & b & b & b & b & b & b & x \end{array}$$

which will be sent to membrane 2. If the rules of the column table c_2 are applied, then the generated array is

$$\begin{array}{ccccccccc} a & b & b & b & b & b & b & b & b \\ b & d & e & e & d & p & q & q & b \\ b & e & d & d & e & q & p & p & b \\ b & b & b & b & b & b & b & b & x \end{array}$$

which is sent back to membrane 1. This process can repeat. If the rules of the column table c_3 are applied to an array in membrane 2 (instead of c_2), the computation halts and the generated array is collected in the language. Thus the picture language L is generated and hence L belongs to $IAP_2((l/u)IP2DCFG)$. This picture language cannot belong to $IAP_1((l/u)IP2DCFG)$. In fact when there is only one membrane, all the tables of rules of the system are in this membrane. In a picture array in L having $m(\geq 4)$ rows and $2n+5(n \geq 1)$ columns, row 2 to row $m-2$ have for the same $n \geq 1$, both e^n and q^n respectively to the left and right of dp while the row $m-2$ has d^n and p^n respectively to the left and right of eq. But we cannot generate picture arrays in L with this feature using tables of pure context-free rules in a single membrane as the rewriting is for the leftmost symbol in each row, having a rule to rewrite it.

We now show that the array P system with three membranes and $(l/u)IP2DCFG$ kind of rules in its regions can be constructed to generate a picture array language which is an element of the family of context-sensitive two-dimensional matrix languages ($CSML$) introduced and investigated in [17]. We briefly recall in an informal way, a context-sensitive matrix grammar ($CSMG$). A $CSMG$ has two phases of derivation. In the first phase, a context-sensitive string language is generated over an alphabet of "intermediate" symbols. In the second phase, "vertical derivations" take place as follows: A string of intermediates generated in the first phase is considered and each of the symbols in this string is rewritten in parallel in the vertical direction by regular rules of the form $A \to a$ or by regular rules of the form $A \to \begin{array}{c} a \\ B \end{array}$, a is a terminal symbol. □

Theorem 4.3. *$IAP_3((l/u)IP2DCFG) \cap CSML \neq \emptyset$.*

Proof. Consider the $CSMG$ generating a picture array language L_{abc} consisting of $m \times 3n$ arrays p $(m \geq 1, n \geq 1)$ such that $p(1, j) = a$ for $1 \leq j \leq n$, $p(1, j) = b$ for $n+1 \leq j \leq 2n$, $p(1, j) = c$ for $2n+1 \leq j \leq 3n$ and all other entries are d. A member of L_{abc} is given below:

$$\begin{array}{ccccccccc} a & \cdots & a & b & \cdots & b & c & \cdots & c \\ d & \cdots & d & d & \cdots & d & d & \cdots & d \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ d & \cdots & d & d & \cdots & d & d & \cdots & d \\ d & \cdots & d & d & \cdots & d & d & \cdots & d \end{array}.$$

Note that the first row has an equal number of $a's, b's$ and $c's$. This picture array language is generated by a CSMG which generates the string language $\{S_1^n S_2^n S_3^n \mid n \geq 1\}$ in the first phase. In the second phase vertical derivations are done using the regular nonterminal rules $S_1 \to aX_1, S_2 \to bX_2, S_3 \to cX_3$ initially applied in parallel and the derivation is continued as many times as needed using the regular nonterminal rules $X_1 \to dX_1, X_2 \to dX_2, X_3 \to dX_3$. Vertical derivations are terminated using the regular terminal rules $X_1 \to d, X_2 \to d, X_3 \to d$ yielding picture arrays in L_{abc}.

An $IAP_3((l/u)IP2DCFG)$ Π generating the language L_{abc}, is defined as follows: The membrane structure of Π is $[_1 [_2]_2 [_3]_3]_1$. The only axiom array

$$\begin{array}{ccc} a & b & c \\ d & d & d \end{array}$$

is in membrane 1 initially. Membrane 1 has a row table r with target *here*, a column table c_1 with target in_2 and another column table c_3 with target in_3. Membrane 2 has a column table c_2 with target *out*. Membrane 3 is the output membrane and has no tables of rules.

The tables of rules are given below:

$r = \{d \to \begin{array}{c} d \\ d \end{array}\}$, $c_1 = \{b \to abb, d \to ddd\}$, $c_2 = \{c \to cc, d \to dd\}$, $c_3 = \{a \to a, d \to d\}$.

Membrane 1 is the only region having an initial array. An application of the rules of the row table r initially will add a row of $d's$ of length $3n$ and this application can be repeated with a row of $d's$ being inserted after the second row, everytime the table of rules is used. The array remains in region 1. If the rules of the column table c_1 are applied, then the leftmost column of the form $(bd\cdots d)^t$ is replaced by

$$\begin{array}{ccc} a & b & b \\ d & d & d \\ \vdots & \vdots & \vdots \\ d & d & d \end{array}.$$

The array is then sent to region with label 2. The rules of the column table c_2 are applied, then the leftmost column of the form $(cd\cdots d)^t$ is replaced by

$$\begin{array}{cc} c & c \\ d & d \\ \vdots & \vdots \\ d & d \end{array}.$$

The array is then sent back to region with label 1. The process can repeat. When the rules of the column table c_3 are applied in region 1, then the generated array is of the required form and is sent to region 3, the computation halts as there is no table of rules in this region and the picture array is collected in the language. $\square$

5. Conclusions

For the generation of picture array languages, a variant is introduced in pure 2D context-free grammars with the independent mode of rewriting, motivated by (l/u) kind of rewriting. The resulting class of grammars is denoted by $(l/u)IP2DCFG$. Using P systems as a control mechanism for array rewriting with $(l/u)IP2DCFG$ kind of rules and rewriting, we have generated square pictures of a certain type (Example 4.1) using only one membrane and target agreement for rules. Also increase in the generative power is shown when using two membranes compared to the use of only one membrane. It remains to be seen whether the number of membranes used, especially in the Theorem 4.3 can be reduced. It remains open to characterize the classes of languages that can be generated by these P systems using m membranes, for each $m \geq 1$. Also, it will be of interest to compare the array models considered here with other types of array grammars [4–6].

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The authors thank the reviewers for their very relevant, useful and detailed comments. The paper has been revised suitably incorporating the comments correcting the errors pointed out and improving the presentation.

References

[1] S. Bera, R. Ceterchi, S. Sriram, K.G. Subramanian, Array P systems and pure 2D context-free grammars with independent mode of rewriting, J. Membr. Comput. 4 (1) (2022) 11–20.
[2] M.M. Bersani, A. Frigeri, A. Cherubini, On some classes of 2D languages and their relations, in: J.K. Aggarwal, et al. (Eds.), Combinatorial Image Analysis, in: Lecture Notes Comput. Sci., vol. 4958, 2011, pp. 222–234.
[3] M.M. Bersani, A. Frigeri, A. Cherubini, Expressiveness and complexity of regular pure two-dimensional context-free languages, Int. J. Comput. Math. 90 (2013) 1708–1733.
[4] R. Ceterchi, M. Mutyam, Gh. Păun, K.G. Subramanian, Array–rewriting P systems, Nat. Comput. 2 (2003) 229–249.
[5] R. Freund, Control mechanisms on #-context-free array grammars, in: Gh. Păun (Ed.), Mathematical Aspects of Natural and Formal Languages, World Scientific, Singapore, 1994, pp. 97–137.
[6] R. Freund, Array Grammars, Technical Rep. 15/00, Research Group on Mathematical Linguistics, Rovira i Virgili University, Tarragona, 2000, 164 pages.
[7] D. Giammarresi, A. Restivo, Two-dimensional languages, in: G. Rozenberg, A. Salomaa (Eds.), Handbook of Formal Languages, vol. 3, Springer Verlag, 1997, pp. 215–267.
[8] Z. Křivka, C. Martín-Vide, A. Meduna, K.G. Subramanian, A variant of pure two-dimensional context-free grammars generating picture languages, in: R.P. Barneva, V.E. Brimkov, J. Slapal (Eds.), Combinatorial Image Analysis, in: LNCS, vol. 8466, Springer, Heidelberg, 2014, pp. 123–133.
[9] H.A. Maurer, A. Salomaa, D. Wood, Pure grammars, Inf. Control 44 (1980) 47–72.
[10] Gh. Păun, Computing with membranes, J. Comput. Syst. Sci. 61 (2000) 108–143.
[11] Gh. Păun, Membrane Computing: An Introduction, Springer-Verlag, Berlin, Heidelbrg, 2000.
[12] Gh. Păun, G. Rozenberg, A. Salomaa, The Oxford Handbook of Membrane Computing, Oxford University Press, New York, USA, 2010.
[13] A. Rosenfeld, Picture Languages - Formal Models for Picture Recognition, Academic Press, New York, 1979.

[14] A. Rosenfeld, R. Siromoney, Picture languages - a survey, Lang. Des. 1 (1993) 229–245.
[15] G. Rozenberg, A. Salomaa (Eds.), Handbook of Formal Languages, vol. 1–3, Springer, Berlin, 1997.
[16] A. Salomaa, Formal Languages, Academic Press, London, 1973.
[17] G. Siromoney, R. Siromoney, K. Krithivasan, Abstract families of matrices and picture languages, Comput. Graph. Image Process. 1 (1972) 284–307.
[18] K.G. Subramanian, P systems and picture languages, Lect. Notes Comput. Sci. 4664 (2007) 99–109.
[19] K.G. Subramanian, R.M. Ali, M. Geethalakshmi, A.K. Nagar, Pure 2D picture grammars and languages, Discrete Appl. Math. 157 (16) (2009) 3401–3411.
[20] K.G. Subramanian, A.K. Nagar, M. Geethalakshmi, Pure 2D picture grammars (P2DPG) and P2DPG with regular control, in: Brimkov, et al. (Eds.), Combinatorial Image Analysis, in: LNCS, vol. 4958, 2008, pp. 330–341.
[21] K.G. Subramanian, S. Sriram, B. Song, L. Pan, An overview of 2D picture array generating models based on membrane computing, in: A. Adamatzky (Ed.), Reversibility and Universality. Emergence, Complexity and Computation, vol. 30, Springer, 2018, pp. 333–356.
[22] P.S.P. Wang, Array Grammars, Patterns and Recognizers, World Scientific, 1989.
[23] G. Zhang, M.J. Pérez-Jiménez, Gh. Păun, Real-life applications with membrane computing, in: Emergence, in: Complexity and Computation Series, 2017.