Pure 2D Eilenberg P Systems

Somnath Bera[1], Atulya K. Nagar[2], K.G. Subramanian[2,*], and Gexiang Zhang[3]

[1] School of Advanced Sciences-Mathematics, Vellore Institute of Technology, Chennai, Tamil Nadu 600 127 India
[2] School of Mathematics, Computer Science and Engineering, Liverpool Hope University, Hope Park, Liverpool L16 9JD, UK
(* Honorary Visiting Professor; Email: kgsmani1948@gmail.com)
[3] School of Automation, Chengdu University of Information Technology, Chengdu, Sichuan, China

Abstract. The computational power of Eilenberg P systems with string objects and rewriting rules in generating languages, has been studied. Extending this system to two dimensions, we introduce here an Eilenberg P system $P2DEPS$ with rectangular picture array objects and pure 2D context-free rules and examine the array language generating power of these systems. We show that with two membranes, any pure 2D context-free language can be generated. We also show that $P2DEPS$ has more generative power than pure 2D context-free grammar ($P2DCFG$). We also compare $P2DEPS$ with certain other picture array generating grammars.

Keywords: Eilenberg P system · pure context-free rules · pure 2D context-free grammar.

1 Introduction

In the area of membrane computing [7, 8], based on the biologically inspired computing model of P system introduced by Gh. Păun with string objects and evolution rules involving rewriting operation [6], a P system, called Eilenberg P system (EPS) was introduced in [1]. An EPS involves an Eilenberg machine which is similar to a finite state machine having associated with each transition, a specific set of evolution rules that are context-free rewriting rules belonging to a region of the EPS. The EPS starts in an initial state with an initial set of string objects. In the current state, with a current set of string objects, the EPS evolves by applying the rules associated with one of the transitions emerging out from the current state. The EPS continues its activity from the next state of the transition. Here we extend the concept of an EPS to two dimensions by introducing a P system, called pure 2D Eilenberg P system ($P2DEPS$) having picture array objects and tables of pure 2D context-free rules [12] as evolution rules in its regions. Here again as in the case of EPS, we have an Eilenberg machine with the evolution rules from the regions associated with the transitions. We examine the picture array generative power of $P2DEPS$ and show that any pure 2D context-free language [10] can be generated by a $P2DEPS$ with two

membranes and that $P2DEPS$ has more generative power than pure 2D context-free grammar ($P2DCFG$). We also compare the generative power of $P2DEPS$ with certain other picture array grammars.

2 Preliminaries

For formal language theory related notions, the reader can refer to [9] and to [5, 12] for two-dimensional array grammars and languages. For P systems and array P systems we refer to [6, 11]. For the definitions of Eilenberg P system and Eilenberg machine we refer to [1, 3]. We now recall certain basic notions on words and picture arrays as well as the definition of pure 2D context-free grammar [12].

A finite set T of symbols is called an alphabet. A word or a string $w = a_1a_2 \dots a_m$ $a_i \in T, 1 \leq i \leq m, (m \geq 1)$ of length m over an alphabet T is a finite sequence of symbols belonging to T. We denote by $|w|$, the length of the word w. The set of all words over T, including the empty word λ with no symbols, is denoted by T^*. For any word $w = a_1a_2 \dots a_n$, $t(w)$ is the vertical word with the word w written vertically. For example, if $w = aab$ over the alphabet $\{a, b\}$, then $t(w)$ is $\begin{matrix} a \\ a \\ b \end{matrix}$. A $p \times q$ array with p rows and q columns (also called a $p \times q$ picture array) M over an alphabet T is of the form

$$M = \begin{matrix} a_{11} & \cdots & a_{1q} \\ \vdots & \ddots & \vdots \\ a_{p1} & \cdots & a_{pq} \end{matrix}$$

where each $a_{ij} \in T, 1 \leq i \leq p$, $1 \leq j \leq q$. We denote by $|M|_r$, the number of rows of M and by $|M|_c$, the number of columns of M. The set of all picture arrays over T is denoted by T^{**}, which includes the empty array λ. $T^{++} = T^{**} - \{\lambda\}$. A picture array language is a subset of T^{**}.

Definition 1. *A pure 2D context-free grammar (P2DCFG) is given by* $G = (T, R_1, R_2, X)$ *where*

- T *is a finite set of symbols ;*
- $R_1 = \{c_i | 1 \leq i \leq m\}, R_2 = \{r_j | 1 \leq j \leq n\}$;

Each $c_i, (1 \leq i \leq m)$*, called a column table, is a set of pure context-free rules of the form* $a \to \alpha, a \in T, \alpha \in T^*$ *such that for any two rules* $a \to \alpha, b \to \beta$ *in* c_i*, we have* $|\alpha| = |\beta|$*; Each* $r_j, (1 \leq j \leq n)$*, called a row table, is a set of pure context-free rules of the form* $d \to t(\gamma), d \in T$ *and* $\gamma \in T^*$ *such that for any two rules* $d \to t(\gamma), e \to t(\delta)$ *in* r_j*, we have* $|\gamma| = |\delta|$;

- $X \subseteq T^{**} - \{\lambda\}$ *is a finite set of axiom arrays.*

A derivation in a P2DCFG is defined as follows: For any two arrays M_1, M_2*, we write* $M_1 \Rightarrow M_2$ *if* M_2 *is obtained from* M_1 *by either rewriting a column of* M_1 *by rules of some column table* c_i *in* R_1 *or a row of* M_1 *by rules of some row*

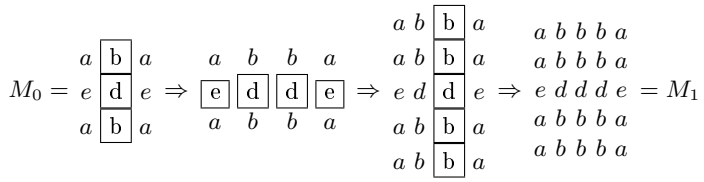

Fig. 1. Derivation $M_0 \Rightarrow^* M_1$

table r_j in R_2. The reflexive transitive closure of $\Rightarrow$ is denoted by $\Rightarrow^*$.
The picture array language $L(G)$ generated by G is the set of picture arrays $\{M | M_0 \Rightarrow^* M \in T^{**}$, for some $M_0 \in X\}$.

The family of picture array languages generated by pure 2D context-free grammars is denoted by $P2DCFL$.

We illustrate picture array generation in a $P2DCFG$ with an example.

Example 1. Consider the pure 2D context-free grammar $G_1 = (T, R_1, R_2, \{M_0\}\})$ where $T = \{a, b, d, e\}$, $R_1 = \{c\}, R_2 = \{r\}$. The column table c and the row table r are given by $c = \{b \to bb, d \to dd\}$, $r = \left\{ \begin{matrix} a & b \\ e \to e\,, & d \to d \\ a & b \end{matrix} \right\}$, $M_0 = \begin{matrix} a\ b\ a \\ e\ d\ e \\ a\ b\ a \end{matrix}$
On using c, r, c in this order, we have the derivation $M_0 \Rightarrow^* M_1$, which is shown in Fig. 1. The symbols rewritten in a column or a row are enclosed in boxes in the Figure.

3 Pure 2D Eilenberg P System

We now introduce a new array P system, called pure 2D Eilenberg P system ($P2DEPS$) extending the definition of EPS to two dimensions with the regions of the P system having rectangular picture array objects and pure 2D context-free grammar type of tables of rules but the picture arrays evolve on applying the rules of the tables in the regions that are associated with a transition emerging out of the current state. Thus the definition of the two-dimensional extension of EPS is very similar to the string case [1, 3] except that a deterministic version of the Eilenberg machine is used in the sense that the system from a given state moves to at most one state on reading the label of a transition. Some or all the states can be final states. The working of the two-dimensional extension of EPS is also similar to the string case except that the result, namely, the picture array generated, is collected in a designated output region of the system.

Definition 2. *A pure 2D Eilenberg P system (P2DEPS) is given by $\Pi = (\mu, T, Q, X_1, \cdots, X_n, \mathcal{C}, \delta, q_0, F, i_o)$ where*

(*i*) μ *is a membrane structure having* n *membranes;*
(*ii*) T *is the alphabet of the system;*
(*iii*) Q *is a finite set of states;*
(*iv*) $X_i, (1 \leq i \leq n)$ *is a finite set (which can be empty) of picture arrays representing the initial value in the region labelled* i*;*
(*iv*) $\mathcal{C} = (C_1, \cdots, C_p)$*, where for* $1 \leq i \leq p$*, the component* $C_i = (P_{i,1}, \cdots, P_{i,n})$ *with* $P_{i,j}$*, a finite set (which can be empty) of column and/or row tables of pure 2D context-free grammar rules (as in Definition 1) with a target (here, in or out) for each table of rules and the elements of* $P_{i,j}$ *belong to region* i*;*
(*v*) $\delta : Q \times \mathcal{C}; \rightarrow Q$ *is the next-state function;*
(*vi*) q_0 *is the initial state;*
(*vii*) F *is a set of final states;*
(*viii*) i_0 *is the label of the output region where the result of the system is collected.*

A computation in a $P2DEPS$ *takes place as follows: A computation starts with the initial state* q_0 *and the initial configuration of the system* $(M_1, \cdots, M_n)$*. A column or a row table of pure context-free rules in the component associated with the transition emerging out of* q_0*, is used to rewrite (with the rewriting done as in a pure 2D context-free grammar) the picture arrays in the corresponding region. At a time, only one table of rules can be applied to a picture array rewriting a column or a row of symbols and all the picture arrays which can be rewritten are to be rewritten The picture arrays evolved either remain in the same region or are sent to an immediate inner or outer region depending on the target here, in or out indicated with the table of rules. The process is continued using a table of rules in the component associated with a transition from the current state to which the system reaches from an immediate earlier step. At a given step of computation, a picture array in a region to which a non-deterministically chosen table of rules in the component used is applicable, is rewritten (as done in a pure 2D context-free grammar), thus evolving the picture array. The process repeats and the resulting configuration of the system in a computation step consists of picture arrays evolved at that moment. A computation halts when no table of rules associated with the current transition is applicable to the picture arrays in the regions. When the system reaches a final state and a halting computation, the picture arrays collected in the output region* i_0 *constitute the picture array language* $L(\Pi)$ *generated by the* $P2DEPS$*.*

The family of picture array languages generated by $P2DEPS$ *is denoted by* $\mathcal{L}(P2DEPS)$*. If we indicate the three parameters, namely, the number of membranes at most* m*, the number of states at most* s *and the number of components at most* p*, then we denote the system by* $P2DEPS(m, s, p)$ *and the corresponding family by* $\mathcal{L}(P2DEPS(m, s, p))$*.*

We illustrate with an example.

Example 2. Consider the $P2DEPS(1, 3, 3)$
$\Pi_1 = ([_1]_1, \{a, b, d, e\}, \{q_1, q_2, q_3\}, X_1, \mathcal{C}, \delta, q_1, 1, \{q_3\})$ where

$\mathcal{C} = (C_1, C_2, C_3)$, $\delta(q_1, C_1) = q_2, \delta(q_2, C_2) = q_1, \delta(q_1, C_3) = q_3$, with $C_1 = (\{c_1\}), C_2 = (\{r\}), C_3 = (\{c_2\})$. The column tables c_1, c_2 are given by $c_1 = \{e \to aeb, d \to ddd\}$, $c_2 = \{e \to ab, d \to dd\}$. The row table r is given by $r = \{a \to \begin{matrix} a \\ d \end{matrix}, e \to \begin{matrix} e \\ d \end{matrix}, b \to \begin{matrix} b \\ d \end{matrix}\}$. All the tables of rules have the target *here*.

$X_1 = \{M_1\}$, with the initial value in the region being $M_1 = \begin{matrix} a\ e\ b \\ d\ d\ d \end{matrix}$.

The picture array language $L(\Pi_1)$ generated by Π_1 consists of $(n+1) \times (2n+2)$, $(n \geq 1)$, picture arrays M where $M_{1j} = a$ for $1 \leq j \leq n+1$, $M_{1j} = b$, for $n+2 \leq j \leq 2n+2$ and all other entries are d. A picture array of $L(\Pi_1)$ is shown in Fig. 2. The $P2DEPS$ Π_1 has only one membrane with an initial value,

$$\begin{matrix} a \cdots a\ e\ b \cdots b \\ d \cdots d\ d\ d \cdots d \\ \vdots \ddots \vdots \vdots \vdots \ddots \vdots \\ d \cdots d\ d\ d \cdots d \end{matrix}$$

Fig. 2. A picture array of $L(\Pi_1)$

namely, the picture array M_1 in the membrane. The system Π_1 starts in the initial state q_1. If the emerging transition at q_1 given by $\delta(q_1, C_1) = q_2$ is used, then the rules of the column table c_1 are used to rewrite the column $t(ed)$ of the only initial picture array M_1 to yield a picture array

$$M_2 = \begin{matrix} a\ a\ e\ b\ b \\ d\ d\ d\ d\ d \end{matrix}.$$

The next state reached by the system is q_2. The system now uses the only emerging transition given by $\delta(q_2, C_2) = q_1$ and so the rules of the row table r are applied rewriting the row $aaebb$ of M_2 yielding the picture array

$$M_3 = \begin{matrix} a\ a\ e\ b\ b \\ d\ d\ d\ d\ d \\ d\ d\ d\ d\ d \end{matrix}$$

while the system reaches the next state q_1. The process can repeat as many times as needed until the emerging transition at q_1 given by $\delta(q_1, C_3) = q_3$ is used which makes the system reach the final state q_3 while the rules of the column table c_2 are applied to the picture array rewriting the symbols in the column $t(ed \cdots d)$ and yielding the picture array of the form shown in Fig. 2. The computation halts as there is no outgoing transition in the state q_3. Since the system reaches a halting computation and is also in the final state, the picture array generated is collected in the picture array language $L(\Pi_1)$. In fact $L(\Pi_1)$ belongs to the family $\mathcal{L}(P2DEPS(1,3,3))$.

4 Comparison results

We now compare the picture array generative power of $P2DEPS$ with $P2DCFG$. We show that a $P2DCFL$ can be generated by a $P2DEPS$ with two membranes, one state and two components. We also construct such a $P2DEPS$ generating a picture array language that cannot be generated by any $P2DCFG$.

Theorem 1. $P2DCFL \subseteq \mathcal{L}(P2DEPS(2,1,2))$

Proof. Consider a $P2DCFG$ $G = (T, R_1, R_2, X)$ generating a picture array language L. We construct a $P2DEPS$ $\Pi = ([_1[_2]_2]_1, T, \{q_1\}, X, \mathcal{C}, \delta, q_1, \{q_1\}, 2)$ where $\mathcal{C} = (C_1, C_2)$ with $C_1 = (R_1 \cup R_2, \emptyset)$ $C_2 = (\{c\}, \emptyset)$ where the column table $c = \{a \to a \mid a \in T\}$. $\delta(q_1, C_1) = \delta(q_1, C_2) = q_1$. The tables of rules that belong to $R_1 \cup R_2$ have the target *here* and the column table c has the target *in*. Corresponding to a derivation in the $P2DCFG$ G, a computation in the $P2DEPS$ Π is done as follows: The system Π starts in the initial state q_1 and an initial value which is an axiom picture array from X in the membrane with label 1. Since $\delta(q_1, C_1) = q_1$, rules of a column or row table in $R_1 \cup R_2$ can be applied simulating a corresponding step of derivation in G with the system remaining in the state q_1 itself. Thus a sequence of derivation steps in a derivation in G will be captured in the computation in Π. At any intermediate step of computation, the system Π can use the rules of the column table c rewriting any column in the picture array of that intermediate step. The application of the rules of the column table c does not change the picture array but the picture array is sent to the inner membrane with label 2. The computation halts as there are no rules (of the region 2) in the components that can be applied and the system is in the final state q_1 itself. Thus every picture array of L is collected in the output region 2. This proves the inclusion in the statement of the theorem.

Theorem 2. $\mathcal{L}(P2DEPS(2,1,2)) - P2DCFL \neq \phi$.

Proof. We now consider a $P2DEPS$
$\Pi_1 = ([_1[_2]_2]_1, \{a, b, e\}, \{q_1\}, \{M_0\}, \mathcal{C}, \delta, q_1, \{q_1\}, 2)$ where $M_0 = \begin{matrix} a\ e\ b \\ a\ e\ b \end{matrix}$, $\mathcal{C} = (C_1, C_2)$ with $C_1 = (\{c_1, r\}, \emptyset)$ $C_2 = (\{c_2\}, \emptyset)$ where the column tables are $c_1 = \{e \to aeb\}$, $c_2 = \{e \to ab\}$, the row table is $r = \{a \to \begin{matrix} a \\ a \end{matrix}, b \to \begin{matrix} b \\ b \end{matrix}, e \to \begin{matrix} e \\ e \end{matrix}\}$, $\delta(q_1, C_1) = \delta(q_1, C_2) = q_1$. The tables c_1 and r have the target *here* and the table c_2 has the target *in*. The system Π_1 generates a picture array language L_1 consisting of $m \times 2n$ $(m \geq 2, n \geq 2)$ picture arrays M such that $M_{ij} = a$ for $1 \leq i \leq m, 1 \leq j \leq n$, and $M_{ij} = b$ for $1 \leq i \leq m, n+1 \leq j \leq 2n$. A picture array of L_1 is shown below:

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

In fact the system Π_1 starts in the initial state q_1 with the initial picture array M_0 in the region with label 1. If the transition $\delta(q_1, C_2) = q_1$ is used, then the rules of the column table c_2 are applied to M_0 which evolves into the picture array $M_1 = \begin{matrix} a\ a\ b\ b \\ a\ a\ b\ b \end{matrix}$. M_1 is sent to the region with label 2. The computation halts with the system in the only state q_1 which is a final state. The picture array M_1 is collected in the language. If the transition $\delta(q_1, C_1) = q_1$ is used and the rules of the column table c_1 are applied to M_0, then it evolves into the picture array $\begin{matrix} a\ a\ e\ b\ b \\ a\ a\ e\ b\ b \end{matrix}$ that remains in the same membrane. The process can continue and at any step of computation, the rules of the row table r can be used if the transition selected is $\delta(q_1, C_1) = q_1$ and a row of the form $a^k e b^k$ (for some $k \geq 1$) is added to the picture array. Again if the transition $\delta(q_1, C_2) = q_1$ is selected with the system continuing in the state q_1, then the picture array generated is sent to region with label 2 and the computation halts. As the state q_1 is a final state, the picture array generated is collected in the language L_1. But this language cannot be generated by any $P2DCFG$. If such a $P2DCFG$ G_1 generates L_1, then starting from an axiom picture array, a column of $a's$ or a column of $b's$ are to be rewritten by rules of a column table of rules. But then any pure context-free rule for a or b will only yield pictures not in the language since every picture array in L_1 has a certain number of continuous columns (starting from the first column) of $a's$ followed by an equal number of columns of $b's$.

A two-dimensional picture array generating grammar, called regular matrix grammar (RMG), was introduced in [10] and extensively investigated in many studies. This two-dimensional grammar was later renamed as two-dimensional right-linear grammar ($2RLG$) [5]. Extensions of RMG to the context-free and context-sensitive cases have been considered in [10] and their properties have been studied. We refer to these two-dimensional grammars as two-dimensional context-free grammar ($2CFG$) and two-dimensional context-sensitive grammar ($2CSG$) in line with the naming of $2RLG$. We compare the generative power of $P2DEPS$ with those of $2RLG$, $2CFG$, and $2CSG$. In all these three grammars, namely, $2RLG, 2CFG, 2CSG$, there are two phases of derivations with the first phase involving respectively, regular, context-free and context-sensitive string grammars generating a string language over a set of symbols, called intermediates. Every string of intermediates obtained in the first phase is rewritten in the second phase in all these three grammars in the vertical direction using nonterminal regular grammar rules of the form $A \to aB$ applied together or terminal rules of the form $A \to a$ that are applied together, to generate the columns of the picture arrays over terminal symbols. The families of picture languages generated by $2RLG, 2CFG, 2CSG$ are respectively denoted by $2RLL, 2CFL, 2CSL$.

Theorem 3. $\mathcal{L}(P2DEPS(2,1,2)) - 2RLL \neq \emptyset$.

Proof. Consider the picture array language L_2 consisting of $m \times (2n+1)$ $(n \geq 1)$ picture arrays M such that $M_{1j} = a$, for $1 \leq j \leq n$, $M_{1(n+1)} = d$, $M_{1j} = b$, for $n+2 \leq j \leq 2n+1$, $M_{ij} = d$, otherwise. A $P2DEPS$ with two membranes, 1

state and 2 components, generating L_2 is given by $\Pi_2' = ([_1[_2]_2]_1, \{a,b,e,d\}, \{q_1\}, \{M_0\}, \mathcal{C}, \delta, q_1, \{q_1\}, 2)$ where $M_0 = \begin{matrix} a & e & b \\ d & d & d \end{matrix}$, $\mathcal{C} = (C_1, C_2)$ with $C_1 = (\{c_1, r\}, \emptyset)$, $C_2 = (\{c_2\}, \emptyset)$, where the column tables are $c_1 = \{e \to aeb, d \to ddd\}$, $c_2 = \{e \to d, d \to d\}$, the row table $r = \{a \to \begin{matrix} a \\ d \end{matrix}, b \to \begin{matrix} b \\ d \end{matrix}, e \to \begin{matrix} e \\ d \end{matrix}\}$. $\delta(q_1, C_1) = q_1$, $\delta(q_1, C_2) = q_1$. The tables of rules c_1, r have the target *here* and c_2 has target *in*. The system Π_2' starts in the initial state q_1 with the initial picture array M_0 in region with label 1. If the transition $\delta(q_1, C_2) = q_1$ is used, then the rules of the column table c_1 or the rules of the row table r are applied to M_0 yielding a picture array which remains in the same membrane and the next state is the same state q_1. The process can repeat as many times as needed generating a picture array of the form

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

With the system at the state q_1, if the transition $\delta(q_1, C_2) = q_1$ is used, then the rules of the column table c_2 are applied to the picture array evolved at that moment yielding a picture array belonging to L_2. This picture array is collected in the language as the computation halts with the system remaining in the final state q_1. But the picture array language L_2 cannot be generated by any $2RLG$ since the first rows of the picture arrays constitute a context-free language, namely $\{a^n d b^n \mid n \geq 1\}$. This would mean that a $2CFG$ will be required to generate L_2.

Theorem 4. $\mathcal{L}(P2DEPS(1,2,2)) - 2RLL \neq \emptyset$.

Proof. The picture array language L_2 considered in the proof of Theorem 3 cannot be generated by any $2RLG$ as seen in the proof of Theorem 3. A $P2DEPS$ with one membrane, 2 states and 2 components, generating L_2 is given by $\Pi_2 = ([_1]_1, \{a,b,d_1,d_2,e,d\}, \{q_1,q_2\}, \{M_0\}, \mathcal{C}, \delta, q_1, \{q_2\}, 1)$ where $M_0 = \begin{matrix} d_1 & e & d_2 \\ d & d & d \end{matrix}$, $\mathcal{C} = (C_1, C_2)$ with $C_1 = (\{c, r_1\})$, $C_2 = (\{r_2\})$, where the column table is $c = \{e \to d_1 e d_2, d \to ddd\}$, the row tables are $r_1 = \{d_1 \to \begin{matrix} d_1 \\ d \end{matrix}, d_2 \to \begin{matrix} d_2 \\ d \end{matrix}, e \to \begin{matrix} e \\ d \end{matrix}\}$, $r_2 = \{d_1 \to a, d_2 \to b, e \to d\}$. $\delta(q_1, C_1) = q_1$, $\delta(q_1, C_2) = q_2$. The tables of rules have the target *here*. The system Π_2 starts in the initial state q_1 with the initial picture array M_0 in region with label 1. If the transition $\delta(q_1, C_1) = q_1$ is used, then the rules of the column table c or the rules of the row table r_1 are applied to M_0 yielding a picture array which remains in the same membrane and the next state is the same state q_1. The process can repeat as many times as needed generating a picture array of the form

$$\begin{array}{c} d_1 \cdots d_1\ e\ d_2 \cdots d_2 \\ d \cdots d\ d\ d \cdots d \\ \vdots\ \ddots\ \vdots\ \vdots\ \vdots\ \ddots\ \vdots \\ d \cdots d\ d\ d \cdots d \end{array}.$$

With the system at the state q_1, if the transition $\delta(q_1, C_2) = q_2$ is used, then the rules of the row table r_2 are applied to the picture array evolved at that moment yielding a picture array of the form

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

This picture array is collected in the language as the computation halts with the system reaching a final state q_2.

Theorem 5. $\mathcal{L}(P2DEPS(2,2,4)) - 2CFL \neq \emptyset$.

Proof. Consider the picture array language L_3 consisting of $m \times (3n+2)$ $(m \geq 2, n \geq 1)$ picture arrays M such that $M_{1j} = a$, for $1 \leq j \leq n$, $M_{1j} = b$, for $n+2 \leq j \leq 2n+1$, $M_{1j} = e$, for $2n+3 \leq j \leq 3n+2$, $M_{ij} = d$, otherwise. A $P2DEPS$ with two membranes, 2 states and 4 components, generating L_3 is given by
$\Pi_3 = ([_1[_2]_2]_1, \{a, b, e, e_1, e_2\}, \{q_1, q_2\}, \{M_0\}, \mathcal{C}, \delta, q_1, q_1, 2)$ where
$M_0 = \begin{matrix} a & e_1 & b & e_2 & e \\ d & d & d & d & d \end{matrix}$, $\mathcal{C} = (C_1, C_2, C_3, C_4)$ with $C_1 = (\{r_1\}, \emptyset)$, $C_2 = (\{c_1\}, \emptyset)$, $C_3 = (\{c_2\}, \emptyset)$, $C_4 = (\{r_2\}, \emptyset)$ where the column tables are $c_1 = \{e_1 \to ae_1b, d \to ddd\}$, $c_2 = \{e_2 \to e_2e, d \to dd\}$, the row tables are $r_1 = \{a \to \begin{matrix} a \\ d \end{matrix}, e_1 \to \begin{matrix} e_1 \\ d \end{matrix}, b \to \begin{matrix} b \\ d \end{matrix}, e_2 \to \begin{matrix} e_2 \\ d \end{matrix}, e \to \begin{matrix} e \\ d \end{matrix}, \}$, $r_2 = \{a \to a, b \to b, e \to e, e_1 \to d, e_2 \to d\}$, $\delta(q_1, C_1) = q_1$, $\delta(q_1, C_2) = q_2$, $\delta(q_2, C_3) = q_1$, $\delta(q_1, C_4) = q_1$. The column tables of rules c_1, c_2 and the row table of rules r_1 have the target *here* and the table of rules r_2 has target *in*. The system Π_3 starts in the initial state q_1 with the initial picture array M_0 in the region with label 1. If the transition $\delta(q_1, C_4) = q_1$ is used, then the rules of the row table r_2 are applied to M_0 yielding a picture array of the form $\begin{matrix} a & d & b & d & e \\ d & d & d & d & d \end{matrix}$ which is sent to the output region 2 while the system If the transition $\delta(q_1, C_1) = q_1$ is used, then the rules of the row table r_1 are applied to M_0 yielding a picture array with a row of $d's$ added below the first row. This picture array remains in the same membrane and the next state is the same state q_1. The process can repeat as many times as needed generating a picture array of the form

$$\begin{array}{c} a\ e_1\ b\ e_2\ e \\ d\ d\ d\ d\ d \\ \vdots\ \vdots\ \vdots\ \vdots\ \vdots \\ d\ d\ d\ d\ d \end{array}$$

If the transition $\delta(q_1, C_2) = q_2$ followed by $\delta(q_2, C_3) = q_1$ are used, then the rules of the column table c_1 are applied followed by the application of the rules of the column table c_2 to the picture array remaining in region 1, generating a picture array of the form

$$\begin{array}{c} a \cdots a\ e_1\ b \cdots b\ e_2\ e \cdots e \\ d \cdots d\ d\ d \cdots d\ d\ d \cdots d \\ \vdots\ \ddots\ \vdots\ \vdots\ \vdots\ \ddots\ \vdots\ \vdots\ \vdots\ \ddots\ \vdots \\ d \cdots d\ d\ d \cdots d\ d\ d \cdots d \end{array}$$

Note that this process can be repeated with the state moving from q_1 to q_2 and returning back to q_1. When the system is at the state q_1, if the transition $\delta(q_1, C_4) = q_3$ is used, then the rules of the row table r_2 are applied to the picture array evolved at that moment yielding a picture array of the form

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

This picture array is collected in the language as the computation halts with the system reaching the final state q_1. But the picture array language L_3 cannot be generated by any $2CFG$ since the first row of the picture arrays constitute a context-sensitive language, namely $\{a^n db^n dc^n \mid n \geq 1\}$. This would mean that a $2CSG$ will be required to generate L_2.

Theorem 6. $\mathcal{L}(P2DEPS(1,3,4)) - 2CFL \neq \emptyset$.

Proof. The picture array language L_3 considered in the proof of Theorem 5 cannot be generated by any $2CFG$ as seen in the proof of Theorem 5. A $P2DEPS$ with one membrane, 3 states and 4 components, generating L_3 is given by $\Pi_3 = ([_1]_1, \{a, b, d_1, d_2, d_3, e, d\}, \{q_1, q_2, q_3\}, \{M_0\}, \mathcal{C}, \delta, q_1, \{q_3\}, 1)$ where $M_0 = \begin{array}{c} d_1\ e_1\ d_2\ e_2\ d_3 \\ d\ d\ d\ d\ d \end{array}$, $\mathcal{C} = (C_1, C_2, C_3, C_4)$ with $C_1 = (\{r_1\})$, $C_2 = (\{c_1, \})$ $C_3 = (\{c_2, \})$, $C_4 = (\{r_2\})$ where the column tables are $c_1 = \{e_1 \to d_1 e_1 d_2, d \to ddd\}$, $c_2 = \{e_2 \to e_2 d_3, d \to dd\}$, the row tables are $r_1 = \{d_1 \to \begin{array}{c} d_1 \\ d \end{array}, e_1 \to \begin{array}{c} e_1 \\ d \end{array}, d_2 \to \begin{array}{c} d_2 \\ d \end{array}, e_2 \to \begin{array}{c} e_2 \\ d \end{array}, d_3 \to \begin{array}{c} d_3 \\ d \end{array}, \}$, $r_2 = \{d_1 \to a, d_2 \to b, d_3 \to c, e_1 \to d, e_2 \to d\}$. $\delta(q_1, C_1) = q_1$, $\delta(q_1, C_2) = q_2$, $\delta(q_2, C_3) = q_1$, $\delta(q_1, C_4) = q_3$. All the tables of rules have the target *here*. The system Π_3 starts in the initial state q_1 with the initial picture array M_0 in the region with label 1. If the transition $\delta(q_1, C_1) = q_1$

is used, then the rules of the row table r_1 are applied to M_0 yielding a picture array which remains in the same membrane and the next state is the same state q_1. The process can repeat as many times as needed generating a picture array of the form

$$\begin{array}{ccccc} d_1 & e_1 & d_2 & e_2 & d_3 \\ d & d & d & d & d \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ d & d & d & d & d \end{array}.$$

If the transition $\delta(q_1, C_2) = q_2$ followed by $\delta(q_2, C_3) = q_1$ are used, then the rules of the column table c_1 are applied followed by the application of the rules of the table c_2 to the picture array remaining in region 1, generating a picture array of the form

$$\begin{array}{ccccccccccc} d_1 & \cdots & d_1 & e_1 & d_2 & \cdots & d_2 & e_2 & d_3 & \cdots & d_3 \\ d & \cdots & d & d & d & \cdots & d & d & d & \cdots & d \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ d & \cdots & d & d & d & \cdots & d & d & d & \cdots & d \end{array}.$$

When the system is at the state q_1, if the transition $\delta(q_1, C_4) = q_3$ is used, then the rules of the row table r_2 are applied to the picture array evolved at that moment yielding a picture array of the form

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

This picture array is collected in the language as the computation halts with the system reaching the final state q_3.

5 Conclusion

We have introduced here an extension of the string language generating Eilenberg P system (EPS) to two dimensions with sets of pure 2D context-free rules in the regions of the system. The resulting P system, namely, $P2DEPS$ generates picture array languages. It will be of interest to compare the generative power of $P2DEPS$ with other picture array grammar models such as [2, 4].

Acknowledgement

The authors thank the reviewers for their very useful and relevant comments which helped them to prepare this revised version. This work was supported by the National Natural Science Foundation of China (61972324), the Sichuan Science and Technology Program (23NSFTD0049, 23ZDYF0247, 2022YFG0181).

References

1. Balanescu, T., Gheorghe, M., Holcombe, M., Ipate, F.: Eilenberg P systems, In: Gh. Păun et al (eds): WMC – CdeA 2002, LNCS 2597, pp. 43-57, 2003.
2. Bera, S., Ceterchi, R., Sriram, S., Subramanian,K.G.: Array P systems and pure 2D context-free grammars with independent mode of rewriting. J. Membr. Comput. **4(1)**: 11-20 (2022)
3. Bernardini, Gheorghe, M., Holcombe, M. :PX Systems = P Systems + X Machines, Natural Computing, **2**, 201-213 (2003).
4. Fernau, H., Freund, R., Ivanov, S., Schmid, M.L., Subramanian, K.G. : Array Insertion and Deletion P Systems. In: Mauri, G., Dennunzio, A., Manzoni, L., Porreca, A.E. (eds) Unconventional Computation and Natural Computation. UCNC 2013. Lecture Notes in Computer Science, vol **7956**. Springer, Berlin, 67-78.
5. Giammarresi, D., Restivo, A.: Two-dimensional languages. In: Rozenberg, G., Salomaa, G. (eds.) Handbook of Formal Languages, pp. 215—267. Springer, Berlin (1997).
6. Păun, Gh. : Computing with membranes, J. Comp. System Sci., **61**, 108– 143 (2000).
7. Păun, Gh.: Membrane Computing: An Introduction. Springer-Verlag Berlin, Heidelbrg, 2000.
8. Păun, Gh., Rozenberg, G., Salomaa, A. : The Oxford Handbook of Membrane Computing. Oxford University Press, Inc., New York, NY, USA (2010)
9. Rozenberg, G., Salomaa, A. (Eds.): Handbook of Formal Languages. Vols. 1 - 3, Springer-Verlag, Berlin (1997).
10. Siromoney, G., Siromoney, R., Krithivasan, K.: Abstract families of matrices and picture languages. Computer Graphics and Image Proc. **1**, 284–307 (1972).
11. Subramanian, K.G. : P systems and picture languages. Lecture Notes in Computer Science, Springer Verlag, Vol. 4664, 99–109 (2007).
12. Subramanian, K.G., Ali, R.M., Geethalakshmi, M., Nagar, A.K.: Pure 2D picture grammars and languages. Discrete Appl. Math. **157**, 3401-–3411 (2009).