Journal of Universal Computer Science, vol. 18, no. 13 (2012), 1782-1801
submitted: 15/11/11, accepted: 30/3/12, appeared: 1/7/12 © J.UCS

P Systems with Shuffle Operation and Catalytic-Like Rules

Yunyun Niu, Jinbang Xu[1]
(Key Laboratory of Image Processing and Intelligent Control
Department of Control Science and Engineering
Huazhong University of Science and Technology
Wuhan 430074, China
niuyunyun1003@163.com, jbxuhust@gmail.com)

K.G. Subramanian, Rosni Abdullah
(School of Computer Sciences, Universiti Sains Malaysia
Penang 11800, Malaysia
kgsmani1948@yahoo.com, rosni@cs.usm.my)

Abstract: Shuffle operation on trajectories is useful in modeling parallel composition of words and languages. In this work, a new class of P systems with shuffle operation and catalytic-like rules is presented. Such a system has a membrane structure, where language-objects and shuffle-operation rules are placed in its regions. It can be used as a language generator. In this study, we propose a variant P system with shuffle operation on string-language objects. Some comparison results are obtained, which show that the power of shuffle operation is enlarged in the framework of P systems. Moreover, string-language objects are extended to array-language objects, and another variant P system with shuffle operation on picture-language objects is introduced. We also illustrate how to generate picture languages by using this kind of devices.

Key Words: membrane computing, P system, shuffle on trajectories, picture language
Category: F.1.1, F.4.3, I.7.0

1 Introduction

Membrane computing was introduced by Păun in 1998, which is inspired from the structure and the functioning of living cells, as well as the organization of cells in tissues, organs, and other higher order structures [Păun 2000]. Membrane system, also called P system, provides a non-deterministic distributed parallel model for dealing with mathematical or computational problems [Gutiérrez-Naranjo et al. 2006, Pan et al. 2011a]. As we know, most P systems are proved to be universal [Păun 2002], and computationally efficient [Alhazov et al. 2003, Pan and Alhazov 2006, Pan and Martín-Vide 2006, Pan and Pérez-Jiménez 2010, Pan and Martín-Vide 2005]. Nowadays, applications of P systems [Ciobanu et al. 2006], as well as exploring new models [Pan et al. 2011b, Pan et al. 2012a, Pan et al. 2012b, Zhang et al. 2012], are getting more and more attention. A most recent overview of the field of membrane computing can be found in [Păun et al. 2010].

[1] Corresponding author

Shuffle on trajectories is useful in modeling parallel composition of words and languages [Păun et al. 1995]. Informally, trajectory is a segment of a line in the plane, starting in the origin of axes and continuing in parallel with the axes Ox and Oy. The line can change its direction only in points of nonnegative integer coordinates. A trajectory defines how to skip from a word to another word during the shuffle operation. Shuffle on trajectories [Mateescu et al. 1998, Kadrie et al. 2001] provides a method of great flexibility to handle the operation of parallel composition of processes from the catenation to the usual shuffle of processes.

In [Annadura et al. 2008], trajectories are drawn into P system for the first time, and a so-called trajectory P system is proposed. In a trajectory P system, shuffle operation deals with words from the same language set in each region, and this operation is applied in a special kind of sequential way. Specifically, if i is the elementary membrane and L_i is the set of words in this membrane, then $L_i^{T_i} = \bigcup_{\alpha,\beta \in L_i} \alpha ⧢_{T_i} \beta$ is the language computed in the i-th membrane. Sequences are considered as objects in membranes. However, the number of each kind of objects is not considered in that kind of P systems. Here, we introduce a new kind of P system with shuffle operation, which uses languages as objects and contains catalytic-like rules. This kind of system has a structure of basic P systems. All the evolution rules are based on shuffle operation and applied in a maximally parallel way as usual in P systems. Results of the shuffle operation are the only new products after applying evolution rules. It deals with words from different languages in each region. For example, if i is the elementary membrane and L_{i1}, L_{i2} are two languages in this membrane, then $L_{i1} ⧢_{T_i} L_{i2} = \bigcup_{\alpha \in L_{i1}, \beta \in L_{i2}} \alpha ⧢_{T_i} \beta$ is the new product in some step of the computation. If $L_{i1} = L_{i2}$ in our system, $L_{i1} ⧢_{T_i} L_{i2}$ is equal to $L_{i1}^{T_i}$ or $L_{i2}^{T_i}$ in trajectory P systems. In some sense, trajectory P system [Annadura et al. 2008] can be considered as a special case of the system proposed here. Some comparison results are given, which show that the string-language generative power of shuffle operation is further extended in a membrane structure. Moreover, we extend string objects to picture objects and propose another variant, a P system with shuffle operation on picture languages. It generates languages which can be generated by Siromoney matrix grammars [Siromoney et al. 1972] or array-rewriting P systems [Ceterchi et al. 2003]. The picture-language generative power of P system with shuffle operation on picture languages is illustrated with several examples. The trajectory P system considered in [Venkatesan et al. 2010] which is an extension to arrays of trajectory P system [Annadura et al. 2008] is also in some sense a special case of the P system with shuffle operation on picture languages considered here.

2 Preliminaries

An alphabet V is a non empty set, whose elements are called symbols. An ordered sequence of symbols is a string. The number of symbols in a string u is the length of the string, and it is denoted by $|u|$. As usual, the empty string (with length 0) is denoted by λ. The set of strings of length n built with symbols from the alphabet V is denoted by V^n and $V^* = \cup_{n\geq 0} V^n$. A language over V is a subset of V^*. The families of languages, which are regular, context free, context sensitive and recursively enumerable, are denoted by REG, CF, CS and RE, respectively. A picture A over V is a rectangular $m \times n$ array of elements of the form

$$A = \begin{array}{ccc} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \dots & a_{mn} \end{array} = [a_{ij}]_{m\times n}.$$

The set of all pictures or arrays over V is denoted by V^{**}. A picture or an array language over V is a subset of V^{**}. The number of columns in an array A is denoted by $|A|_c$. The number of rows in an array A is denoted by $|A|_r$. The empty array is denoted by Λ, $|\Lambda|_c = |\Lambda|_r = 0$.

Let $A = [a_{ij}]_{m\times p}, B = [b_{ij}]_{n\times q}$. The column concatenation $A\Phi B$ of A and B is defined only when $m = n$, which is given by

$$A\Phi B = \begin{array}{cccccc} a_{11} & \dots & a_{1p} & b_{11} & \dots & b_{1q} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ a_{m1} & \dots & a_{mp} & b_{n1} & \dots & b_{nq}. \end{array}$$

Similarly, the row concatenation $A\Theta B$ of A and B is defined only when $p = q$, which is given by

$$A\Theta B = \begin{array}{ccc} a_{11} & \dots & a_{1p} \\ \vdots & \ddots & \vdots \\ a_{m1} & \dots & a_{mp} \\ b_{11} & \dots & a_{1q} \\ \vdots & \ddots & \vdots \\ b_{n1} & \dots & a_{nq}. \end{array}$$

$\Lambda\Phi P = P\Phi\Lambda = P$ and $\Lambda\Theta P = P\Theta\Lambda = P$ for any $P \in V^{**}$.

Definition 1 *A trajectory is an element t, $t \in V^*$, where $V = \{r, u\}$.*

Let Σ be an alphabet and let t be a trajectory, $t = t_1t_2\cdots t_n$, where $t_i \in V$, $1 \leq i \leq n$. We define $|t|_r$ the number of r in t, and $|t|_u$ is defined as the number of u in t. Let α, β be two words over Σ, $\alpha = a_1a_2\cdots a_p, \beta = b_1b_2\cdots b_q$, where $a_i, b_j \in \Sigma, 1 \leq i \leq p$ and $1 \leq j \leq q$.

Definition 2 *The shuffle of α with β on the trajectory t, denoted by $\alpha \sqcup\!\sqcup_t \beta$, is defined as follows: if $|\alpha| \neq |t|_r$ or $|\beta| \neq |t|_u$, then $\alpha \sqcup\!\sqcup_t \beta = \emptyset$; else $\alpha \sqcup\!\sqcup_t \beta = c_1c_2\cdots c_{p+q}$,*

$$c_i = \begin{cases} a_{k_1+1} & \text{if } t_i = r, \\ b_{k_2+1} & \text{if } t_i = u, \end{cases}$$

where k_1 is defined as the number of r in $t_1, \ldots, t_{i-1}$, and k_2 is defined as the number of u in $t_1, \ldots, t_{i-1}$.

If T is a set of trajectories, the shuffle of α with β on the set T of trajectories, denoted $\alpha \sqcup\!\sqcup_T \beta$, is $\alpha \sqcup\!\sqcup_T \beta = \bigcup_{t \in T} \alpha \sqcup\!\sqcup_t \beta$. If $L_1, L_2 \subseteq \Sigma^*$, then $L_1 \sqcup\!\sqcup_T L_2 = \bigcup_{\alpha \in L_1, \beta \in L_2} \alpha \sqcup\!\sqcup_T \beta$ [Mateescu et al. 1998].

Theorem 1. *[Mateescu et al. 1998] Let L_1, L_2 and $T, T \subseteq \{r, u\}^*$ be three languages.*

(i) If all three languages are regular, then $L_1 \sqcup\!\sqcup_T L_2$ is regular.

(ii) If two languages are regular languages and the third one is a context-free language, then $L_1 \sqcup\!\sqcup_T L_2$ is a context-free language.

Definition 3 *Let V be a finite alphabet, t a string over $\{r, u\}, v \in \{r, u\}$, $P, Q \in V^{**}$ having the same number of rows and $|P|_c = |t|_r, |Q|_c = |t|_u$. The column shuffle of P with Q on the trajectory vt, denoted by $P \sqcup\!\sqcup^c_{vt} Q$, is recursively defined as follows.*

- *If $v = r$, then $P \sqcup\!\sqcup^c_{vt} Q = (A \Phi X) \sqcup\!\sqcup^c_{vt} Q = A \Phi (X \sqcup\!\sqcup^c_t Q)$, where $A \in V^{**}$ is the first column of P.*
- *If $v = u$, then $P \sqcup\!\sqcup^c_{vt} Q = P \sqcup\!\sqcup^c_{vt} (B \Phi Y) = B \Phi (P \sqcup\!\sqcup^c_t Y)$, where $B \in V^{**}$ is the first column of Q.*
- *If $v = r$, then $\Lambda \sqcup\!\sqcup^c_{vt} P = \emptyset, P \sqcup\!\sqcup^c_{vt} \Lambda = (A \Phi X) \sqcup\!\sqcup^c_{vt} \Lambda = A \Phi (X \sqcup\!\sqcup^c_t \Lambda)$, where $A \in V^{**}$ is the first column of P.*
- *If $v = u$, then $\Lambda \sqcup\!\sqcup^c_{vt} P = \Lambda \sqcup\!\sqcup^c_{vt} (A \Phi X) = A \Phi (X \sqcup\!\sqcup^c_t \Lambda), P \sqcup\!\sqcup^c_{vt} \Lambda = \emptyset$, where $A \in V^{**}$ is the first column of P.*
- *If $t \neq \lambda$, then $\Lambda \sqcup\!\sqcup^c_t \Lambda = \emptyset$; otherwise, $\Lambda \sqcup\!\sqcup^c_t \Lambda = \Lambda$.*

Definition 4 *Let V be a finite alphabet, t a string over $\{l, d\}, v \in \{l, d\}$, $P, Q \in V^{**}$ having the same number of columns and $|P|_r = |t|_l, |Q|_r = |t|_d$. The row shuffle of P with Q on the trajectory vt, denoted by $P \sqcup\!\sqcup^r_{vt} Q$, is recursively defined as follows.*

- *If $v = l$, then $P \sqcup\!\sqcup^r_{vt} Q = (A \Theta X) \sqcup\!\sqcup^r_{vt} Q = A \Theta (X \sqcup\!\sqcup^r_t Q)$, where $A \in V^{**}$ is the first row of P.*
- *If $v = d$, then $P \sqcup\!\sqcup^r_{vt} Q = P \sqcup\!\sqcup^r_{vt} (B \Theta Y) = B \Theta (P \sqcup\!\sqcup^r_t Y)$, where $B \in V^{**}$ is the first row of Q.*
- *If $v = l$, then $\Lambda \sqcup\!\sqcup^r_{vt} P = \emptyset, P \sqcup\!\sqcup^r_{vt} \Lambda = (A \Theta X) \sqcup\!\sqcup^r_{vt} \Lambda = A \Theta (X \sqcup\!\sqcup^r_t \Lambda)$, where $A \in V^{**}$ is the first row of P.*
- *If $v = d$, then $\Lambda \sqcup\!\sqcup^r_{vt} P = \Lambda \sqcup\!\sqcup^r_{vt} (A \Theta X) = A \Theta (X \sqcup\!\sqcup^r_t \Lambda), P \sqcup\!\sqcup^r_{vt} \Lambda = \emptyset$, where $A \in V^{**}$ is the first row of P.*

– *If $t \neq \lambda$, then $\Lambda \sqcup\sqcup_t^r \Lambda = \emptyset$; otherwise, $\Lambda \sqcup\sqcup_t^r \Lambda = \Lambda$.*

Definition 5 *[Siromoney et al. 1972] A Siromoney matrix grammar is a 2-tuple (G_1, G_2), where $G_1 = (H_1, I_1, P_1, S)$ is a regular, a context-free or a context-sensitive grammar*

– *H_1 is a finite set of horizontal non-terminals,*

– *$I_1 = \{S_1, S_2, \ldots, S_k\}$, a finite set of intermediates, $H_1 \cap I_1 = \emptyset$,*

– *P_1 is a finite set of production rules called horizontal production rules,*

– *$S \in H_1$ is the start symbol.*

$G_2 = (G_{21}, G_{22}, \ldots, G_{2k})$, where $G_{2i} = (V_{2i}, T, P_{2i}, S_i), 1 \leq i \leq k$ are regular grammars,

– *V_{2i} is a finite set of vertical non-terminals, $V_{2i} \cap V_{2j} = \emptyset, i \neq j$,*

– *T is a finite set of terminals,*

– *P_{2i} is a finite set of right linear production rules of the form $X \to aY$ or $X \to a$, where $X, Y \in V_{2i}, a \in T$,*

– *$S_i \in V_{2i}$ is the start symbol of G_{2i}.*

The type of G_1 gives the type of G. G is called regular, context-free or context-sensitive Siromoney matrix grammars if G_1 is regular, context-free or context-sensitive, respectively.

Derivations are defined as follows. A string $S_{i1}S_{i2}\ldots S_{in} \in I_1^*$ is generated horizontally using the horizontal production rules of P_1 in G_1, i.e., $S \Rightarrow S_{i1}S_{i2}\ldots S_{in} \in I_1^*$. Vertical derivations proceed as follows. We write

$$\begin{array}{c} A_{i1} \ \ldots \ A_{in} \\ \Downarrow \\ a_{i1} \ \ldots \ a_{in} \\ B_{i1} \ \ldots \ B_{in} \end{array}$$

if $A_{ij} \to a_{ij}B_{ij}$ are rules in $P_{2j}, 1 \leq j \leq n$, where A_{ij}, B_{ij} are the j-th non-terminals of some strings. The derivation terminates if $A_{mj} \to a_{mj}$ are all terminal rules in G_2.

The set $L(G)$ of picture arrays generated by G consists of all $m \times n$ arrays $[a_{ij}]$ such that $1 \leq i \leq m, 1 \leq j \leq n$ and $S \Rightarrow_{G_1}^* S_{i1}S_{i2}\ldots S_{im} \Rightarrow_{G_2}^* [a_{ij}]$. We denote the picture language classes of regular, context-free Siromoney matrix grammars by RML, $CFML$, respectively.

3 P System with Shuffle Operation and Catalytic-Like Rules

3.1 P System with Shuffle Operation on String Languages

Definition 6 *A P system with shuffle operation on string languages is defined as:*

$$\Pi = (V, T, \mu, L_1, \cdots, L_m, T_1, \cdots, T_m, R_1, \cdots, R_m, i_o),$$

where:

- V *is an alphabet;*
- $T = \{r, u\}$ *is the control alphabet;*
- μ *is a membrane structure consisting of* m *membranes;*
- $L_1, \cdots, L_m$ *are initial sets of languages associated with regions* $1, \cdots, m$ *of* μ*, respectively;*
- $T_i \subseteq T^*, 1 \leq i \leq m$ *are sets of trajectories associated with regions* $1, \cdots, m$ *of* μ*, respectively;*
- R_i *is a finite set of evolution rules associated with regions* i *of* μ*; and evolution rules are of the forms:* $A_{h_1}B_{h_2} \rightarrow (C, tar)$*, or catalytic-like rules:* $A_{h_1}B_{h_2} \rightarrow (A, here)\ (C, tar)$ *or* $A_{h_1}B_{h_2} \rightarrow (C, tar)(B, here)$*, where* A, B, C *are languages defined over alphabet* V*,* $h_1, h_2 \in T$*,* $h_1 \neq h_2$*, and* $tar \subseteq \{here, out, in_j \mid 1 \leq j \leq m\}$*. Each language is assigned with either* r *or* u*, which means that it can appear in right or up direction in the shuffle operation. If there exist* $A', B' \in L_i, A' \subseteq A, B' \subseteq B, A' \sqcup\!\sqcup_{T_i} B' \neq \emptyset$*, then* A', B' *can be assigned to one rule of the three forms non-deterministically, and* $C = A' \sqcup\!\sqcup_{T_i} B'$ *is produced.* C *is the shuffle of* A' *with* B' *on the set* T_i *of trajectories.*
- i_o *is the output cell.*

The symbols $here, out, in_j, 1 \leq j \leq m$, are called target commands or target indications. The objects can be transported through membranes according to the targets. Specifically, suppose (C, tar) is present in the right-hand side of a rule in membrane i. If $here \in tar$, one copy of object C is placed in the same region i where the rule is applied. If $out \in tar$, one copy of object C is moved to the region immediately outside membrane i. If $in_j \in tar$, one copy of object C should be moved to the membrane with label j, provided that this membrane is immediately inside membrane i.

Rules from sets of $R_i, 1 \leq i \leq m$, are applied to sets of languages in corresponding region of μ synchronously, in a non-deterministic maximally parallel manner. The m-tuple of sets of languages present at any moment in the m regions of Π constitutes the configuration of the system at that moment. The m-tuple $(L_1, \cdots, L_m)$ is the initial configuration of Π. A transition between configurations is governed by application of evolution rules, all of which are based on shuffle operation as defined above. The results of shuffle operations are the only new products of each rule. A sequence of transitions between configurations of a given system Π is called a computation with respect to Π. A computation is successful if and only if it halts: there is no rule applicable to the objects present in the last configuration. The computational result defined here is the union set of objects (languages) which are present in cell i_o in the halting configuration. If there are some objects that exist in the output membrane in the initial configuration and remain unchanged during the whole computation, they are not included in the computational result.

The language generated by a P system with shuffle operation on string languages Π is denoted by $L(\Pi)$. The family of all languages $L(\Pi)$ generated by systems Π as above, with at most m membranes, is denoted by $LLP_m(FL_1, FL_2, \mathtt{type})$, where the second L indicates that the objects are languages, FL_1 is the family from which languages L_i are taken, FL_2 is the family from which the sets of trajectories are taken, and `type` indicates the type of rules, $\mathtt{type} \in \{\lambda, cat\}$; while the system is with catalytic-like rules, it is denoted by $LLP_m(FL_1, FL_2, cat)$.

3.1.1 The Language Generative Power of P System with Shuffle Operation on String Languages

In this subsection, the language generative power of P system with shuffle operation on string languages is investigated. When comparing with Theorem 1 in Section 2, the following theorems show that the shuffle operation has more power in the framework of P systems than in the standard set-up.

Theorem 2. $LLP_3(REG, REG, cat) - REG \neq \emptyset$.

Proof. Let us consider the system with catalytic-like rules

$$\Pi_1 = (V, T, \mu, L_1, L_2, L_3, T_1, T_2, T_3, R_1, R_2, R_3, i_o),$$

where:

- $V = \{a, b\}; T = \{r, u\}$;
- $\mu = [[\,]_2[\,]_3]_1$;
- $L_1 = \{\{a\}, \{b\}\},\ L_2 = \{\{b\}\},\ L_3 = \emptyset$;
- $T_1 = \{ru^n \mid n \geq 1\}, T_2 = \{r^n u \mid n \geq 1\}, T_3 = \emptyset$;
- R_1: $\{a\}_r\{a^{n-1}b^n \mid n \geq 1\}_u \rightarrow (C', in_j), j = 2, 3$,
  R_2: $\{a^n b^n \mid n \geq 1\}_r\{b\}_u \rightarrow (C'', out)\{b\}$,
  $R_3 = \emptyset$;
- $i_o = 3$.

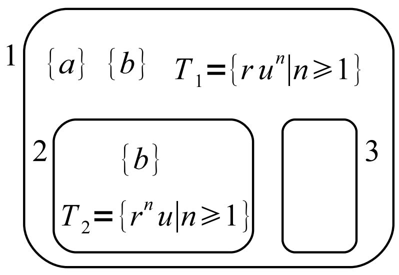

Figure 1: Initial structure of system Π_1

The initial configuration of this system is shown in Fig. 1.

Initially, there are objects $\{a\}$ and $\{b\}$ in membrane 1. According to T_1, one rule of R_1 is chosen non-deterministically, and object $\{ab\}$ is sent to membrane 2 or membrane 3 in a non-deterministic way. If it reaches membrane 3, there is no rule applicable in the next step, the computation halts, and object $\{ab\}$ is obtained in the output membrane. If $\{ab\}$ reaches membrane 2, by using the rule $\{a^nb^n \mid n \geq 1\}_r\{b\}_u \rightarrow (C'', out)\{b\}$, object $\{ab^2\}$ is introduced and sent to membrane 1. Object $\{a^2b^2\}$ is sent to membrane 2 or membrane 3 in a non-deterministic way by using rule $\{a\}_r\{ab^2\}_u \rightarrow (C', in_j), j = 2, 3$. If it reaches membrane 3, the computation halts and there is object $\{a^2b^2\}$ in the output membrane. If it reaches membrane 2, the computation continues in the same way as described above. According to the definition of the computational result, system Π_1 generates language $L = \{ab\} \cup \{a^2b^2\} \cup \cdots \cup \{a^nb^n\} \cup \cdots = \{a^nb^n \mid n \geq 1\}$. Language $\{a^nb^n \mid n \geq 1\}$ is context-free, but not regular. Therefore, $LLP_3(REG, REG, cat) - REG \neq \emptyset$, and this concludes the proof.

Theorem 3. $LLP_2(REG, CF) - CF \neq \emptyset$.

Proof. Let us consider a system without catalytic-like rules

$$\Pi_2 = (V, T, \mu, L_1, L_2, T_1, T_2, R_1, R_2, i_o),$$

where:

- $V = \{a, b, c\}; T = \{r, u\}$;
- $\mu = [[\,]_1]_2$;
- $L_1 = \{\{a^n \mid n \geq 1\}, \{b^n \mid n \geq 1\}\},\ L_2 = \{\{c^n \mid n \geq 1\}\}$;
- $T_1 = \{r^nu^n \mid n \geq 1\}, T_2 = \{r^{2n}u^n \mid n \geq 1\}$;
- R_1: $\{a^n \mid n \geq 1\}_r\{b^n \mid n \geq 1\}_u \rightarrow (C', out)$,
  R_2: $\{a^nb^n \mid n \geq 1\}_r\{c^n \mid n \geq 1\}_u \rightarrow (C'', in)$;

– $i_o = 1$.

The initial configuration of this system is shown in Fig. 2.

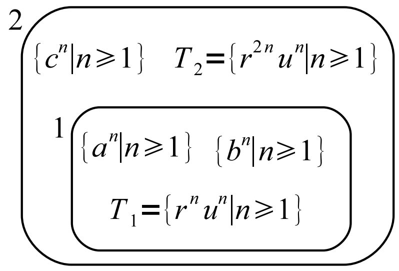

Figure 2: Initial structure of system Π_2

The computation starts from membrane 1. By using rule $\{a^n \mid n \geq 1\}_r\{b^n \mid n \geq 1\}_u \rightarrow (C', out)$ object $\{a^n b^n \mid n \geq 1\}$ is sent out. In the next step, by using rule $\{a^n b^n \mid n \geq 1\}_r\{c^n \mid n \geq 1\}_u \rightarrow (C'', in)$, language $\{a^n b^n c^n \mid n \geq 1\}$ is obtained as the computational result of Π_2. That is, $L(\Pi_2) = \{a^n b^n c^n \mid n \geq 1\}$. As we know, $\{a^n b^n c^n \mid n \geq 1\}$ is context-sensitive, but not context-free. Therefore, $LLP_2(REG, CF) - CF \neq \emptyset$, and this concludes the proof.

Theorem 4. *There are P systems Π with shuffle operation of degree two, with regular initial languages, one regular and one context-free sets of trajectories and catalytic-like rules such that $L(\Pi)$ is not context-free.*

Proof. Let us consider a system with catalytic-like rules

$$\Pi_3 = (V, T, \mu, L_1, L_2, T_1, T_2, R_1, R_2, i_o),$$

where:

– $V = \{a, b, c\}; T = \{r, u\}$;

– $\mu = [[\,]_2]_1$;

– $L_1 = \{\{a\}, \{b\}\},\ L_2 = \{\{b\}, \{c^n \mid n \geq 1\}\}$;

– $T_1 = \{ru^n \mid n \geq 1\}, T_2 = \{r^n u, r^{2n}u^n \mid n \geq 1\}$;

– R_1: $\{a\}_r\{a^{n-1}b^n \mid n \geq 1\}_u \rightarrow \{a\}(C', in)$,
R_2: $\{a^n b^n \mid n \geq 1\}_r\{b\}_u \rightarrow (C'', out)\{b\}$,
$\{a^n b^n \mid n \geq 1\}_r\{c^m \mid m \geq 1\}_u \rightarrow C'''\{c^m \mid m \geq 1\}$;

– $i_o = 2$.

The initial configuration of this system is shown in Fig. 3. Obviously, system Π_3 generates language $L = \{abc\} \cup \{a^2b^2c^2\} \cup \cdots \cup \{a^nb^nc^n\} \cup \cdots = \{a^nb^nc^n \mid n \geq 1\}$. We do not describe the process in details. That is, $L(\Pi_3) = \{a^nb^nc^n \mid n \geq 1\}$. Clearly, this language is not context-free. This concludes the proof.

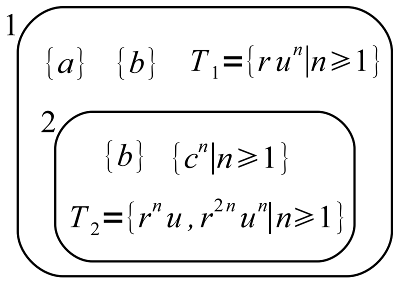
1
$\{a\}$ $\{b\}$ $T_1 = \{ru^n \mid n \geqslant 1\}$
2
$\{b\}$ $\{c^n \mid n \geqslant 1\}$
$T_2 = \{r^n u, r^{2n} u^n \mid n \geqslant 1\}$

Figure 3: Initial structure of system Π_3

3.2 P System with Shuffle Operation on Picture Languages

Definition 7 *A P system with shuffle operation on picture languages is defined as:*

$$\Pi = (V, T, \mu, L_1, \cdots, L_m, T_1, \cdots, T_m, R_1, \cdots, R_m, i_o),$$

where:

- V *is an alphabet;*
- $T = T_c \cup T_r$ *where* $T_c = \{r, u\}$, $T_r = \{l, d\}$, *is the control alphabet;*
- μ *is a membrane structure consisting of* m *membranes;*
- $L_1, \cdots, L_m$ *are initial sets of languages associated with regions* $1, \cdots, m$ *of* μ, *respectively;*
- $T_i = T_{ic} \cup T_{ir}, T_{ic} \subseteq T_c^*, T_{ir} \subseteq T_r^*, 1 \leq i \leq m$ *are sets of trajectories associated with regions* $1, \cdots, m$ *of* μ, *respectively;*
- R_i *is a finite set of evolution rules associated with regions* i *of* μ; *and evolution rules are of the forms:* $A_{h_1}B_{h_2} \xrightarrow{T_{ic}} (C, tar)$, $A_{h_1}B_{h_2} \xrightarrow{T_{ir}} (C, tar)$, *or catalytic-like rules:* $A_{h_1}B_{h_2} \xrightarrow{T_{ic}} (A, here)(C, tar)$, $A_{h_1}B_{h_2} \xrightarrow{T_{ir}} (A, here)(C, tar)$ *or* $A_{h_1}B_{h_2} \xrightarrow{T_{ic}} (C, tar)(B, here)$, $A_{h_1}B_{h_2} \xrightarrow{T_{ir}} (C, tar)(B, here)$, *where* A, B, C *are picture languages defined over alphabet* V, $h_1, h_2 \in T_c$ *or* $h_1, h_2 \in T_r$, $h_1 \neq h_2$, *and* $tar \subseteq \{here, out, in_j \mid 1 \leq j \leq m\}$. *If there exist* $A', B' \in L_i, A' \subseteq$

$A, B' \subseteq B, A' \sqcup\!\sqcup_{T_i} B' \neq \emptyset$, then A', B' can be assigned to one rule of the six forms non-deterministically, and $C = A' \sqcup\!\sqcup_{T_i} B'$ is produced. C is the shuffle of A' with B' on the set T_i of trajectories.

– *i_o is the output cell.*

The symbols $here, out, in_j, 1 \leq j \leq m$, are target commands which are the same as that defined in Definition 6. Rules from sets of $R_i, 1 \leq i \leq m$, are applied to sets of languages in corresponding region of μ synchronously, in a non-deterministic maximally parallel manner. The m-tuple of sets of languages present at any moment in the m regions of Π constitutes the configuration of the system at that moment. The m-tuple $(L_1, \cdots, L_m)$ is the initial configuration of Π. A transition between configurations is governed by application of evolution rules, all of which are based on shuffle operation as defined above. A sequence of transitions between configurations of a given system Π is called a computation with respect to Π. A computation is successful if and only if it halts: there is no rule applicable to the objects present in the last configuration. The computational result is defined as the union set of objects (languages) which are present in cell i_o in the halting configuration. If there are some objects that exist in the output membrane in the initial configuration and remain unchanged during the whole computation, they are not included in the computational result. The language generated by a P system with shuffle operation on picture languages Π is denoted by $L(\Pi)$. The family of all languages $L(\Pi)$ generated by systems Π as above, with at most m membranes, is denoted by $LL^pP_m(FL_1, FL_2, \texttt{type})$; while the system is with catalytic-like rules, it is denoted by $LL^pP_m(FL_1, FL_2, cat)$.

3.2.1 The Language Generative Power of P System with Shuffle Operation on Picture Languages

In this subsection, the language generative power of P system with shuffle operation on picture languages is investigated. Some examples are given to prove that this kind of P system can generate picture languages that can be generated by Siromoney matrix grammars [Siromoney et al. 1972] or array-rewriting P systems [Ceterchi et al. 2003].

Example 1 *Consider the system*

$$\Pi_4 = \{V, T, \mu, L_1, L_2, L_3, L_4, T_1, T_2, T_3, T_4, R_1, R_2, R_3, R_4, i_o\},$$

where:

– *$V = \{a, b\}$; $T = \{r, u, l, d\}$;*

– *$\mu = [[\,]_1[\,]_2[\,]_3]_4$;*

– *$L_1 = \{\{\begin{pmatrix} b \\ a \end{pmatrix}\}, \{b\}\},\ L_2 = \{\{(a, b)\}, \{b\}\},\ L_3 = \emptyset,\ L_4 = \{\{\begin{pmatrix} a \\ a \end{pmatrix}\}\}$;*

– $T_{1r} = \{ld^n \mid n \geq 1\}, T_{2c} = \{r^n u \mid n \geq 1\}, T_{4c} = \{r^n u \mid n \geq 1\},$
$T_{4r} = \{ld^n \mid n \geq 1\}, T_{1c} = T_{2r} = T_{3c} = T_{3r} = \emptyset;$

– R_1: $\{b\}_l\{\begin{pmatrix} b \\ \vdots \\ b \\ a \end{pmatrix}_{n\times 1} \mid n \geq 2\}_d \xrightarrow{T_{1r}} \{b\}(C', here, out),$

R_2: $\{(a, b, \ldots, b)_{1\times n} \mid n \geq 2\}_r\{b\}_u \xrightarrow{T_{2c}} (C'', here, out)\{b\},$
$R_3 = \emptyset,$
R_4: $\{\begin{pmatrix} a \\ a \end{pmatrix}\}_r\{\begin{pmatrix} b \\ a \end{pmatrix}\}_u \xrightarrow{T_{4c}} (C''', here)\{\begin{pmatrix} b \\ a \end{pmatrix}\},$

$\{\begin{pmatrix} a \\ a \end{pmatrix}\}_r\{\begin{pmatrix} b \\ a \end{pmatrix}\}_u \xrightarrow{T_{4c}} (C''', in_3)\{\begin{pmatrix} b \\ a \end{pmatrix}\},$

$$\{\begin{pmatrix} a\, b \ldots\ b \\ \vdots\ \vdots\ \ddots\ \vdots \\ a\, b \ldots\ b \\ a\, a\ \ a\ \ a \end{pmatrix}_{m\times n} \mid m, n \geq 2\}_r\{\begin{pmatrix} b \\ \vdots \\ b \\ a \end{pmatrix}_{m\times 1} \mid m \geq 2\}_u \xrightarrow{T_{4c}} (C^{(4)}, here)$$

$$\{\begin{pmatrix} b \\ \vdots \\ b \\ a \end{pmatrix}_{m\times 1} \mid m \geq 2\},$$

$$\{\begin{pmatrix} a\, b \ldots\ b \\ \vdots\ \vdots\ \ddots\ \vdots \\ a\, b \ldots\ b \\ a\, a\ \ a\ \ a \end{pmatrix}_{m\times n} \mid m, n \geq 2\}_r\{\begin{pmatrix} b \\ \vdots \\ b \\ a \end{pmatrix}_{m\times 1} \mid m \geq 2\}_u \xrightarrow{T_{4c}} (C^{(4)}, in_3)$$

$$\{\begin{pmatrix} b \\ \vdots \\ b \\ a \end{pmatrix}_{m\times 1} \mid m \geq 2\},$$

$$\{(a, b, \ldots, b)_{1\times n} \mid n \geq 2\}_l\{\begin{pmatrix} a\, b \ldots\ b \\ \vdots\ \vdots\ \ddots\ \vdots \\ a\, b \ldots\ b \\ a\, a\ \ a\ \ a \end{pmatrix}_{m\times n} \mid m, n \geq 2\}_d \xrightarrow{T_{4r}}$$

$\{(a, b, \ldots, b)_{1\times n} \mid n \geq 2\}(C^{(5)}, here),$

$$\{(a, b, \ldots, b)_{1\times n} \mid n \geq 2\}_l\{\begin{pmatrix} a\, b \ldots\ b \\ \vdots\ \vdots\ \ddots\ \vdots \\ a\, b \ldots\ b \\ a\, a\ \ a\ \ a \end{pmatrix}_{m\times n} \mid m, n \geq 2\}_d \xrightarrow{T_{4r}}$$

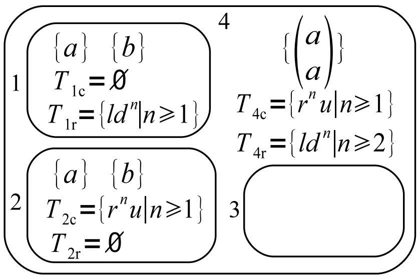
4
1
{a} {b}
$T_{1c}=\emptyset$
$T_{1r}=\{ld^n|n\geqslant 1\}$
2
{a} {b}
$T_{2c}=\{r^n u|n\geqslant 1\}$
$T_{2r}=\emptyset$
3
$T_{4c}=\{r^n u|n\geqslant 1\}$
$T_{4r}=\{ld^n|n\geqslant 2\}$

Figure 4: Initial structure of system Π_4

$$\{(a,b,\ldots,b)_{1\times n} \mid n \geq 2\}(C^{(5)}, in_3);$$

– $i_o = 3$.

The initial configuration of this system is shown in Fig. 4. In membrane 1, by applying rules in R_1, pictures of the form

$$\begin{pmatrix} b \\ \vdots \\ b \\ a \end{pmatrix}_{m\times 1}$$

are generated and sent out to membrane 4. At the same time, by applying rules in R_2 pictures of the form $(a,b,\ldots,b)_{1\times n}$ are generated in membrane 2 and also sent out to membrane 4. They shuffle in the skin membrane and all kinds of tokens L (Fig. 5) can be obtained. So the system can generate the family of all L-shaped angles, which is a RML.

$$\begin{matrix} a & b & b & b & b & b & b & b & b \\ a & b & b & b & b & b & b & b & b \\ a & b & b & b & b & b & b & b & b \\ a & b & b & b & b & b & b & b & b \\ a & b & b & b & b & b & b & b & b \\ a & a & a & a & a & a & a & a & a \end{matrix}$$

Figure 5: L in Example 1

$$\begin{matrix} a & a & a & a & a & a & a & a & a \\ b & b & b & b & a & b & b & b & b \\ b & b & b & b & a & b & b & b & b \\ b & b & b & b & a & b & b & b & b \\ b & b & b & b & a & b & b & b & b \\ b & b & b & b & a & b & b & b & b \end{matrix}$$

Figure 6: T in Example 3

Example 2 *Consider the system*

$$\Pi_5 = \{V, T, \mu, L_1, L_2, L_3, L_4, T_1, T_2, T_3, T_4, R_1, R_2, R_3, R_4, i_o\},$$

where $V, T, \mu, L_1, L_2, L_3, L_4, T_1, T_2, T_3, T_4, R_1, R_2, R_3, i_o$ are the same as those in Example 1.

R_4: $\{\begin{pmatrix} a \\ a \end{pmatrix}\}_r\{\begin{pmatrix} b \\ a \end{pmatrix}\}_u \xrightarrow{T_{4c}} (C''', here)$,

$\{\begin{pmatrix} a \\ a \end{pmatrix}\}_r\{\begin{pmatrix} b \\ a \end{pmatrix}\}_u \xrightarrow{T_{4c}} (C''', in_3)$,

$$\{(a, b, \dots, b)_{1\times n} \mid n \geq 2\}_l\{\begin{pmatrix} a\ b \dots\ b \\ \vdots\ \vdots\ \ddots\ \vdots \\ a\ b \dots\ b \\ a\ a\ \ a\ \ a \end{pmatrix}_{n\times n} \mid n \geq 2\}_d \xrightarrow{T_{4r}} (C^{(4)}, here),$$

$$\{\begin{pmatrix} a\ b \dots\ b \\ \vdots\ \vdots\ \ddots\ \vdots \\ a\ b \dots\ b \\ a\ a\ \ a\ \ a \end{pmatrix}_{(n+1)\times n} \mid n \geq 2\}_r\{\begin{pmatrix} b \\ \vdots \\ b \\ a \end{pmatrix}_{(n+1)\times 1} \mid n \geq 2\}_u \xrightarrow{T_{4c}} (C^{(5)}, here),$$

$$\{\begin{pmatrix} a\ b \dots\ b \\ \vdots\ \vdots\ \ddots\ \vdots \\ a\ b \dots\ b \\ a\ a\ \ a\ \ a \end{pmatrix}_{(n+1)\times n} \mid n \geq 2\}_r\{\begin{pmatrix} b \\ \vdots \\ b \\ a \end{pmatrix}_{(n+1)\times 1} \mid n \geq 2\}_u \xrightarrow{T_{4c}} (C^{(5)}, in_3).$$

The working of system Π_5 is the same as in Example 1 except for the working in membrane 4 by using rules in R_4. The system can generate all L-shaped angles with equal arms.

Example 3 *Consider the system*

$$\Pi_6 = \{V, T, \mu, L_1, \dots, L_6, T_1, \dots, T_6, R_1, \dots, R_6, i_o\},$$

where:

- $V = \{a, b\}$; $T = \{r, u, l, d\}$;
- $\mu = [[\ ]_2[\ ]_3[[\ ]_5[\ ]_6]_4]_1$;
- $L_1 = \{\{a\}, \{(b, b)\}, \{\begin{pmatrix} b\ b \\ b\ b \end{pmatrix}\}\}, L_2 = \{\{a\}, \{\begin{pmatrix} a \\ a \end{pmatrix}\}\}, L_3 = \{\{\begin{pmatrix} b \\ b \end{pmatrix}\}, \{\begin{pmatrix} b\ b \\ b\ b \end{pmatrix}\},$ $\{b\}, \{(b, b)\}\}$, $L_5 = \{\{a\}, \{(a, a)\}\}$, $L_4 = L_6 = \emptyset$;
- $T_{1c} = \{r^n u r^n \mid n \geq 1\}, T_{2r} = \{l^n d \mid n \geq 2\}, T_{3c} = \{r^n u \mid n \geq 2\},$ $T_{3r} = \{l^n d \mid n \geq 2\}, T_{4r} = \{l d^n \mid n \geq 1\}, T_{5c} = \{r^n u \mid n \geq 2\},$ $T_{1r} = T_{2c} = T_{4c} = T_{5r} = T_{6c} = T_{6r} = \emptyset$;
- R_1: $\{\begin{pmatrix} b \dots\ b \\ \vdots\ \ddots\ \vdots \\ b \dots\ b \end{pmatrix}_{m\times 2n} \mid m, n \geq 1\}_r\{\begin{pmatrix} a \\ \vdots \\ a \end{pmatrix}_{m\times 1} \mid m \geq 1\}_u \xrightarrow{T_{1c}} (C', here)$

$\{\begin{pmatrix} a \\ \vdots \\ a \end{pmatrix}_{m\times 1} \mid m \geq 1\}$,

- R_2: $\{\begin{pmatrix} a \\ \vdots \\ a \end{pmatrix}_{m\times 1} \mid m \geq 2\}_l\{a\}_d \xrightarrow{T_{2r}} (C'', here, out)\{a\}$,
- R_3: $\{\begin{pmatrix} b & \dots & b \\ \vdots & \ddots & \vdots \\ b & \dots & b \end{pmatrix}_{m\times n} \mid m, n \geq 2\}_r\{\begin{pmatrix} b \\ \vdots \\ b \end{pmatrix}_{m\times 1} \mid m \geq 2\}_u \xrightarrow{T_{3c}} (C''', here,$

  $out)\{\begin{pmatrix} b \\ \vdots \\ b \end{pmatrix}_{m\times 1} \mid m \geq 2\}$,

  $\{\begin{pmatrix} b & \dots & b \\ \vdots & \ddots & \vdots \\ b & \dots & b \end{pmatrix}_{m\times n} \mid m, n \geq 2\}_l\{(b, \dots, b)_{1\times n} \mid n \geq 2\}_d \xrightarrow{T_{3r}} (C^{(4)}, here,$

  $out)\{(b, \dots, b)_{1\times n} \mid n \geq 2\}$,

  R_4: $\{(a, \dots, a)_{1\times n} \mid n \geq 3\}_l\{\begin{pmatrix} b & \dots & b & a & b & \dots & b \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ b & \dots & b & a & b & \dots & b \end{pmatrix}_{m\times n} \mid m \geq 1, n \geq 3\}_d \xrightarrow{T_{4r}}$

  $\{(a, \dots, a)_{1\times n} \mid n \geq 3\}(C^{(5)}, in_6)$,

  R_5: $\{(a, \dots, a)_{1\times n} \mid n \geq 2\}_r\{a\}_u \xrightarrow{T_{5c}} (C^{(6)}, here, out)\{a\}$,

  $R_6 = \emptyset$;
- $i_o = 6$.

The initial configuration of system Π_6 is shown in Fig. 7. In membrane 3, $m\times n$ pictures of the form

$$\begin{pmatrix} b & \dots & b \\ \vdots & \ddots & \vdots \\ b & \dots & b \end{pmatrix}_{m\times n}$$

are generated and sent out to the skin membrane. At the same time, $m \times 1$ pictures

$$\begin{pmatrix} a \\ \vdots \\ a \end{pmatrix}_{m\times 1}$$

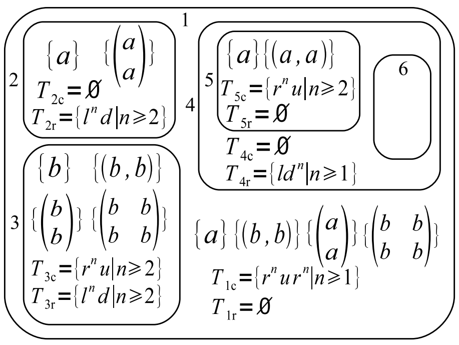
1
2
$\{a\}$ $\{\begin{pmatrix} a \\ a \end{pmatrix}\}$
$T_{2c}=\emptyset$
$T_{2r}=\{l^n d \mid n\geqslant 2\}$
3
$\{b\}$ $\{(b,b)\}$
$\{\begin{pmatrix} b \\ b \end{pmatrix}\}$ $\{\begin{pmatrix} b & b \\ b & b \end{pmatrix}\}$
$T_{3c}=\{r^n u \mid n\geqslant 2\}$
$T_{3r}=\{l^n d \mid n\geqslant 2\}$
4
5
$\{a\}\{(a,a)\}$
$T_{5c}=\{r^n u \mid n\geqslant 2\}$
$T_{5r}=\emptyset$
6
$T_{4c}=\emptyset$
$T_{4r}=\{ld^n \mid n\geqslant 1\}$
$\{a\}$ $\{(b,b)\}$ $\{\begin{pmatrix} a \\ a \end{pmatrix}\}$ $\{\begin{pmatrix} b & b \\ b & b \end{pmatrix}\}$
$T_{1c}=\{r^n u r^n \mid n\geqslant 1\}$
$T_{1r}=\emptyset$

Figure 7: Initial structure of system $\varPi_6$

are generated in membrane 2 and also sent out to the skin membrane. So they shuffle and produce pictures of the form

$$\begin{pmatrix} b & \dots & b & a & b & \dots & b \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ b & \dots & b & a & b & \dots & b \end{pmatrix}_{m\times n}$$

according to T_{1c}. These pictures are sent into membrane 4. Pictures of the form $(a,\dots,a)_{1\times n}$ are generated in membrane 5 and also sent out to membrane 4. According to T_{4r}, they are attached to the top of pictures

$$\begin{pmatrix} b & \dots & b & a & b & \dots & b \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ b & \dots & b & a & b & \dots & b \end{pmatrix}_{m\times n}.$$

Hence, the above system can generate the set of all digitalized pictures of the tokens T (Fig. 6), which is a $CFML$.

Example 4 *Consider the system*

$$\varPi_7 = \{V, T, \mu, L_1, \dots, L_6, T_1, \dots, T_6, R_1, \dots, R_6, i_o\},$$

where:

- $V = \{a, b\}$; $T = \{r, u, l, d\}$;
- $\mu = [[\,]_3[\,]_6[[\,]_4[\,]_5]_2]_1$;
- $L_2 = \{\{b, (b, b), \begin{pmatrix} b \\ b \end{pmatrix}, \begin{pmatrix} b\ b \\ b\ b \end{pmatrix}\}, \{(a, a), \begin{pmatrix} a\ a \\ a\ a \end{pmatrix}\}\}, L_3 = \{\{\begin{pmatrix} a \\ a \end{pmatrix}\},$

$\{\begin{pmatrix} a\ a \\ a\ a \end{pmatrix}\}\}, L_4 = \{\{b\}, \{(b,b)\}, \{\begin{pmatrix} b \\ b \end{pmatrix}\}, \{\begin{pmatrix} b\ b \\ b\ b \end{pmatrix}\}\}, L_5 = \{\{(a,a)\},$
$\{\begin{pmatrix} a\ a \\ a\ a \end{pmatrix}\}\}, L_1 = L_6 = \emptyset;$

- $T_{1r} = \{ld^n l \mid n \geq 1\}, T_{2c} = \{ru^n r \mid n \geq 1\}, T_{3c} = \{r^n u \mid n \geq 2\},$
  $T_{4c} = \{r^n u \mid n \geq 2\}, T_{4r} = \{l^n d \mid n \geq 2\}, T_{5r} = \{ld^n \mid n \geq 1\},$
  $T_{1c} = T_{2r} = T_{3r} = T_{5c} = T_{6c} = T_{6r} = \emptyset;$

- R_1: $\{\begin{pmatrix} a \dots a \\ a \dots a \end{pmatrix}_{2\times n} \mid n \geq 3\}_l \{\begin{pmatrix} a\ b \dots b\ a \\ \vdots\ \vdots\ \ddots\ \vdots\ \vdots \\ a\ b \dots b\ a \end{pmatrix}_{m\times n} \mid m \geq 1, n \geq 3\}_d \xrightarrow{T_{1r}}$
  $\{\begin{pmatrix} a \dots a \\ a \dots a \end{pmatrix}_{2\times n} \mid n \geq 3\}(C', in_6),$

  R_2: $\{\begin{pmatrix} a\ a \\ \vdots\ \vdots \\ a\ a \end{pmatrix}_{2\times n} \mid n \geq 1\}_r \{\begin{pmatrix} b \dots b \\ \vdots\ \ddots\ \vdots \\ b \dots b \end{pmatrix}_{m\times n} \mid m, n \geq 1\}_u \xrightarrow{T_{2c}}$
  $\{\begin{pmatrix} a\ a \\ \vdots\ \vdots \\ a\ a \end{pmatrix}_{2\times n} \mid n \geq 1\}(C'', out),$

  R_3: $\{\begin{pmatrix} a \dots a \\ a \dots a \end{pmatrix}_{2\times n} \mid n \geq 2\}_r \{\begin{pmatrix} a \\ a \end{pmatrix}\}_u \xrightarrow{T_{3c}} (C''', here, out)\{\begin{pmatrix} a \\ a \end{pmatrix}\},$

  R_4: $\{\begin{pmatrix} b \dots b \\ \vdots\ \ddots\ \vdots \\ b \dots b \end{pmatrix}_{m\times n} \mid m \geq 1, n \geq 2\}_r \{\begin{pmatrix} b \\ \vdots \\ b \end{pmatrix}_{m\times 1} \mid m \geq 1\}_u \xrightarrow{T_{4c}}$
  $(C^{(4)}, here, out)\{\begin{pmatrix} b \\ \vdots \\ b \end{pmatrix}_{m\times 1} \mid m \geq 1\},$
  $\{\begin{pmatrix} b \dots b \\ \vdots\ \ddots\ \vdots \\ b \dots b \end{pmatrix}_{m\times n} \mid m \geq 2, n \geq 1\}_l \{(b, \dots, b)_{1\times n} \mid n \geq 1\}_d \xrightarrow{T_{4r}}$
  $(C^{(5)}, here, out)\{(b, \dots, b)_{1\times n} \mid n \geq 1\},$

  R_5: $\{(a,a)\}_l \{\begin{pmatrix} a\ a \\ \vdots\ \vdots \\ a\ a \end{pmatrix}_{m\times 2} \mid m \geq 2\}_d \xrightarrow{T_{5r}} \{(a,a)\}(C^{(6)}, here, out),$

  $R_6 = \emptyset;$

- $i_o = 6.$

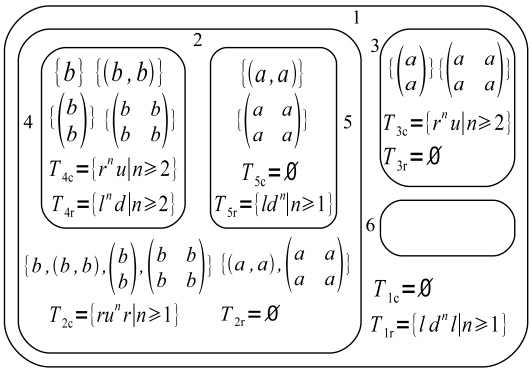
1
2
3
4
5
6
$\{b\}\ \{(b,b)\}$
$\{\begin{pmatrix} b \\ b \end{pmatrix}\}\ \{\begin{pmatrix} b & b \\ b & b \end{pmatrix}\}$
$T_{4c}=\{r^n u \mid n\geqslant 2\}$
$T_{4r}=\{l^n d \mid n\geqslant 2\}$
$\{(a,a)\}$
$\{\begin{pmatrix} a & a \\ a & a \end{pmatrix}\}$
$T_{5c}=\emptyset$
$T_{5r}=\{ld^n \mid n\geqslant 1\}$
$\{b,(b,b),\begin{pmatrix} b \\ b \end{pmatrix},\begin{pmatrix} b & b \\ b & b \end{pmatrix}\}\ \{(a,a),\begin{pmatrix} a & a \\ a & a \end{pmatrix}\}$
$T_{2c}=\{ru^n r \mid n\geqslant 1\}$
$T_{2r}=\emptyset$
$\{\begin{pmatrix} a \\ a \end{pmatrix}\}\{\begin{pmatrix} a & a \\ a & a \end{pmatrix}\}$
$T_{3c}=\{r^n u \mid n\geqslant 2\}$
$T_{3r}=\emptyset$
$T_{1c}=\emptyset$
$T_{1r}=\{l\,d^n l \mid n\geqslant 1\}$

Figure 8: Initial structure of system Π_7

The initial configuration of this system is shown in Fig. 8. $m \times n$ pictures of the form

$$\begin{pmatrix} b \dots b \\ \vdots \ddots \vdots \\ b \dots b \end{pmatrix}_{m\times n}$$

are generated in membrane 4 and sent out to membrane 2. At the same time, $m \times 2$ pictures of the form

$$\begin{pmatrix} a\ a \\ \vdots\ \vdots \\ a\ a \end{pmatrix}_{m\times 2}$$

are generated in membrane 5 and also sent out to membrane 2. So they shuffle and produce pictures

$$\begin{pmatrix} a\ b \dots b\ a \\ \vdots\ \vdots \ddots \vdots\ \vdots \\ a\ b \dots b\ a \end{pmatrix}_{m\times n}$$

according to T_{2c}. These pictures are sent out to the skin membrane. Moreover, $2 \times n$ pictures of the form

$$\begin{pmatrix} a \dots a \\ a \dots a \end{pmatrix}_{2\times n}$$

are generated in membrane 3 and also sent out to the skin membrane. They shuffle with pictures

$$\begin{pmatrix} a\ b \dots b\ a \\ \vdots\ \vdots \ddots \vdots\ \vdots \\ a\ b \dots b\ a \end{pmatrix}_{m\times n}$$

according to T_{1r}. So system Π_7 can generate all solid rectangles marked as shown in Fig. 9, which can also be generated by an array-rewriting P system [Ceterchi et al. 2003].

$$\begin{array}{ccccccccc}
a & a & a & a & a & a & a & a & a \\
a & b & b & b & b & b & b & b & a \\
a & b & b & b & b & b & b & b & a \\
a & b & b & b & b & b & b & b & a \\
a & b & b & b & b & b & b & b & a \\
a & a & a & a & a & a & a & a & a
\end{array}$$

Figure 9: Solid rectangle

4 Conclusions and Remarks

A new kind of P system with shuffle operation and catalytic-like rules has been proposed. It has a hierarchical membrane structure with language-objects in each compartment delimited by membranes. Evolution rules, including catalytic-like rules are based on shuffle operations and deal with different languages in each region. They are applied in a maximally parallel way as usual in P systems. The computational result is defined as the union set of languages which are present in the output membrane in halting configurations. In our study, string languages and picture languages are considered as objects of such P systems, respectively.

The language generative power of P systems with shuffle operation on string languages with or without catalytic-like rules is investigated. Comparing with Theorem 1 in section 2, the shuffle operation placed in a membrane structure is more powerful than in the standard set-up. It remains open whether Theorem 2 can be extended to the system without catalytic-like rules. We also illustrate the language generative power of P system with shuffle operation on picture languages with catalytic-like rules by comparing with Siromoney matrix grammars and array-rewriting P systems. This paper is an enhanced version of [Niu et al. 2010]. Another direction which is worth investigating is the way P system with shuffle operation on picture languages generates languages without catalytic-like rules.

Acknowledgment

The authors are grateful to the reviewers for very useful comments which helped in revising the paper. The work of Y. Niu is supported by National Natural Science Foundation of China (61033003, 91130034 and 30870826), Ph.D. Programs Foundation of Ministry of Education of China (20100142110072), Fundamental Research Funds for the Central Universities (2010ZD001 and 2010MS003), and National Science Foundation of Hubei Province (2008CDB113 and 2011CDA027).

References

[Alhazov et al. 2003] Alhazov, A., Martín-Vide, C., Pan, L.: “Solving a PSPACE-complete problem by P systems with restricted active membranes”; Fundamenta Informaticae, 58, 2(2003), 67–77.

[Annadura et al. 2008] Annadurai, S., Kalyani, T., Dare, V.R., Thomas, D.G.: “Trajectory P system”; Progress in Natural Science, 18(2008), 611–616.

[Ceterchi et al. 2003] Ceterchi, R., Mutyam, M., Păun, Gh., Subramanian, K.G.: “Array-rewriting P systems”; Natural Computing, 2(2003), 229–249.

[Ciobanu et al. 2006] Ciobanu, G., Păun, Gh., Pérez-Jiménéz, M.J. (eds): “Applications of Membrane Computing”; Springer–Verlag, 2006.

[Gutiérrez-Naranjo et al. 2006] Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A., Romero-Campero, F.J.: “Computational efficiency of dissolution rules in membrane systems”; International Journal of Computer Mathematics, 83(2006), 593–611.

[Kadrie et al. 2001] Kadrie, A., Dare, V.R., Thomas, D.G., Subramanian, K.G.: “Algebraic properties of the shuffle over ω-trajectories”; Information Processing Letters, 80(2001), 139–144.

[Mateescu et al. 1998] Mateescu, A., Rozenberg, G., Salomaa, A.: “Shuffle on trajectories: syntactic constraints”; Theoretical Computer Science, 197(1998), 1–56.

[Niu et al. 2010] Niu, Y., Xu, J., Subramanian, K.G.:“P systems with shuffle operation”; In Proceedings of BIC–TA 2010, IEEE, Liverpool, UK (Sep 2010), 1482–1486.

[Pan and Alhazov 2006] Pan, L., Alhazov, A.: “Solving HPP and SAT by P systems with active memabranes and separation rules”; Acta Informatica, 43(2006), 131–145.

[Pan et al. 2011a] Pan, L., Díaz-Pernil D., Pérez-Jiménez, M.J.: “Computation of Ramsey numbers by P systems with active membranes”; International Journal of Foundations of Computer Science, 22(1)(2011), 9–38.

[Pan and Martín-Vide 2005] Pan, L., Martín-Vide, C.: “Solving multidimensional 0-1 knapsack problem by P systems with input and active membranes”; Journal of Parallel and Distributed Computing, 65(2005), 1578–1584.

[Pan and Martín-Vide 2006] Pan, L., Martín-Vide, C.: “Further remark on P systems with active membranes and two polarizations”; Journal of Parallel and Distributed Computing, 66(2006), 867–872.

[Pan and Pérez-Jiménez 2010] Pan, L., Pérez-Jiménez, M.J.: “Computational complexity of tissue-like P systems”; Journal of Complexity, 26(3)(2010), 296–315.

[Pan et al. 2011b] Pan, L., Wang, J., Hoogeboom H.J.: “Limited Asynchronous Spiking Neural P Systems”; Fundamenta Informaticae, 110(2011), 271–293.

[Pan et al. 2012a] Pan, L., Wang, J., Hoogeboom H.J.: “Spiking Neural P Systems with Astrocytes”; Neural Computation, 24(3)(2012), 805–825.

[Pan et al. 2012b] Pan, L., Zeng, X., Zhang, X., Wang, J.: “Spiking Neural P Systems with Weighted Synapses”; Nenbcgxcnural Processing Letters, 35(1)(2012), 13–27.

[Păun 2000] Păun, Gh.: “Computing with membranes”; Journal of Computer and System Sciences, 61(2000), 108–143.

[Păun 2002] Păun, Gh.: “Membrane Computing. An Introduction”; Berlin, Germany: Springer–Verlag, 2002.

[Păun et al. 1995] Păun, Gh., Rozenberg, G. Salomaa, A.: “Grammars based on the shuffle operation”; Journal of Universal Computer Science, 1(1)(1995), 67–82.

[Păun et al. 2010] Păun, Gh., Rozenberg, G. Salomaa, A. (eds.): “The Oxford Handbook of Membrane Computing”; Oxford University Press, 2010.

[Siromoney et al. 1972] Siromoney, G., Siromoney, R., Krithivasan, K.: “Abstract families of matrices and picture languages”; Computer Graphics Image Processing, 1(1972), 234–307.

[Venkatesan et al. 2010] Venkatesan, A.P., Thomas, D.G., Nagar, A.K., Robinson, T.: “Trajectory array P system”; In Proceedings of BIC–TA 2010, IEEE, Liverpool, UK (Sep 2010), 1543–1549.

[Zhang et al. 2012] Zhang, X., Luo, B., Fang, X., Pan, L.: “Sequential spiking neural P systems with exhaustive use of rules”; BioSystems, 108(2012), 52–62.