P systems on iso-triangular arrays

K. Bhuvaneswari*
Department of Mathematics
Sathyabama Institute of Science and Technology
Chennai-600 119, Tamil Nadu
India
bhuvanamageshwaran@gmail.com

T. Kalyani
Department of Mathematics
St. Josesph's Institute of Technology
Chennai-600 119, Tamil Nadu
India
kalphd02@yahoo.com

D.G. Thomas
Institute of Science and Humanities
Saveetha School of Engineering, SIMATS
Chennai-602105
Tamil Nadu
India
dgthomasmcc@gmail.com

D. Lalitha
Department of Mathematics
Sathyabama Institute of Science and Technology
Chennai-600 119, Tamil Nadu
India
lalkrish2007@gmail.com

Abstract. Membrane computing is a branch of natural computing in theoretical computer science. This theoretical model was initiated by G.H. Paun in 1988 and it is called P system, inspired by the structure and functioning of the living cells. P system has become an appropriate platform for solving many computational problems arising in different fields. For certain DNA molecules, the behavior of the molecules are recombined by an operation called splicing. Splicing was introduced by Tom Head. Splicing operation on arrays inspired Helan Chandra et.al. introduced H-array splicing system , which generated 2D arrays. This Paper introduced iso-triangular domino array splicing system and iso-triangular domino array splicing P system. The computational complexity of introduced P systems is examined with few existing models.

Keywords: iso-triangular tiles, tile pasting P system, DNA computing, array splicing, two dimensional arrays.

∗. Corresponding author

1. Introduction

In the domain of theoretical computer science, P system [6,7] is a new computational model. The structure has membranes; each membrane is separated by the regions. Multi-sets of objects are placed inside the regions. The evaluation rules in the respective regions are associated with the target indicators. The target indicator 'in j ' communicates the object to the region j. The target indicator 'here' retains the picture pattern in the same region. The target symbol 'out' sends the resultant picture pattern to the outer region. P system can generate the string languages. P system for string languages motivated K.G. Subramanian et al. [10,11] and introduced tile pasting P system for 2D picture languages. Followed by this computational study on P system, Robinson et al. [3,8,9] proposed the models called tile pasting system and extended tile pasting system which are generated tiling and tessellations. In the survey it is found that the family of languages generated by tile pasting system and extended tile pasting system are incomparable. The family of languages generated by tile pasting P system is properly contained in the family of languages evaluated by extended tile pasting system has been noted. The research on P system induced kalyani et. al [5] and introduced a new tile pasting system called, triangular tile pasting system. It can generate iso-triangular picture languages. In this pasting system, triangular tile pasting rules are applied to the iso-triangular tiles to get glued or pasted along corresponding edges of tiles. Recently a new computational model, called triangular tile pasting P system and iso-array rewriting P system with iso-array grammar rules have been proposed in [1,2].

Splicing system is a model to DNA molecules; here the molecules are recombined by the splicing operation which cut a DNA sequence and pastes the fragment of another DNA under the action of restricted enzymes. This concept is applied theoretically to string of symbols and to 2D arrays. It was introduced by Tom Head [4]. Extending this study to images of pictures, Helen Chandra et al. [12,13] introduced H array splicing system. By applying splicing Operation on hexagonal arrays, hexagonal array splicing system and array splicing P system are introduced to generate some class of hexagonal array languages. Sheena Christy et al. introduced iso-array splicing grammar for iso-triangular tiles and discussed some results in [14]. It is motivated to introduce an iso-triangular domino array splicing system and iso-triangular domino array splicing P system on images of iso-triangular arrays. Here iso-triangular domino arrays are considered. An algorithm can be developed for the model which is defined in this paper. This algorithm can be used to generate two dimensional picture languages.

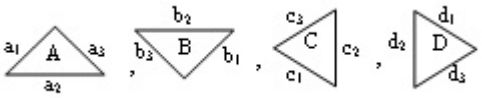
In this paper, section 2 gives the basic definition of iso-triangular tiles and pasting rules of iso triangular tiles have been recalled. In section 3 types of P systems namely triangular tile pasting P system, extended triangular tile pasting P system and Parametric triangular tile pasting P system are introduced and explained with suitable examples. The existing P system called iso-array

rewriting P system with iso-array rules [2] has been recalled. Also the computational power of the P systems is explained in this section. Section 4 denotes iso-triangular domino array splicing system and iso-triangular domino array splicing P system with examples. Section 5 explains the results of the types of tile pasting P systems and iso-triangular domino array splicing P system. It also discusses the computational powers of iso-array rewriting P system with iso-array grammar rules and iso triangular domino array splicing P system.

2. Preliminaries

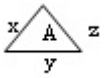
The art of tiling is a well-known theory in the applications of pattern generation. In this section we recollect the notion of iso triangular tiles and triangular tile pasting system. A tile is a topological disc with closed boundary in the XOY plane, whose edges are gluable. A tiling is a family of countable tiles with no gaps or overlaps that covers the Euclidean plane. We are concerned with labeled iso-triangular tiles, whose edges are also labeled.

Definition 2.1. *Consider the labeled iso-triangular tiles*

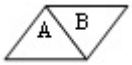


whose horizontal (vertical) and side edges are of length 1 unit and $1/\sqrt{2}$ *unit respectively.*

Definition 2.2. *Generally a pasting rule is a pair* (x, y) *of labeled tiles with distinct edges.*

For example, the iso triangular tile 

and tile 

are joined by the edges (z, t)*, which means that the edge* z *of tile* A *is glued with the edge* t *of tile* B*, we get the pattern* 

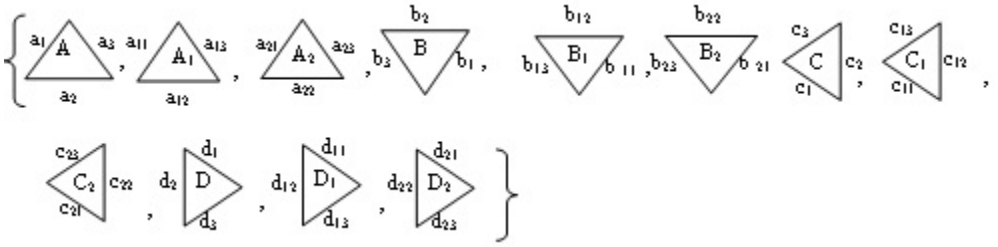
Note that the edges are of same length. The set of all edge labels is called an edge set denoted by E*. Tile pasting rules of the tiles* A, B, C, D *are given below:*

1. *Tile* A *can be glued with tile* B *by the pasting rules* $\{(a_1, b_1), (a_2, b_2), (a_3, b_3)\}$ *with tile* C *by the rule* $\{(a_3, c_1)\}$ *and with tile* D *by the rule* $\{(a_1, d_3)\}$
2. *Tile* B *can be glued with tile* A *by the pasting rules* $\{(b_1, a_1), (b_2, a_2), (b_3, a_3)\}$ *with tile* C *by the rule* $\{(b_1, c_3)\}$ *and with tile* D *by the rule* $\{(b_3, d_1)\}$
3. *Tile* C *can be glued with tile* A *by the pasting rule* $\{(c_1, a_3)\}$ *with tile* B *by* $\{(c_3, b_1)\}$ *and with the tile* D *by the pasting rules* $\{(c_1, d_1), (c_2, d_2), (c_3, d_3)\}$

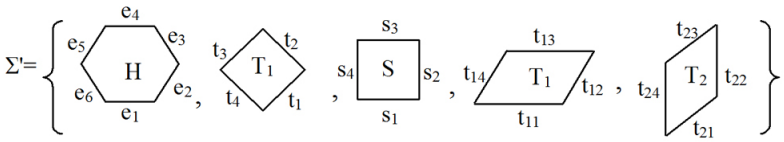
4. *Tile D can be glued with tile A by the pasting rule $\{(d_3, a_1)\}$ with tile B by $\{(d_1, b_3)\}$ and with the tile C by the pasting rules $\{(d_1, c_1)$, (d_2, c_2), $(d_3, c_3)\}$.*

Notations:

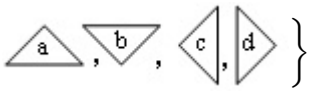
1. Iso-triangular tiles are



2. Extended Triangular tiles are



The set of labeled iso-triangular tiles

$N = \{A, A_1, B, B_1, C, C_1, D, D_1\}$ and $T =$ 

3. Types of triangular tile pasting P systems

In this section, types of triangular tile pasting P systems are defined and explained with suitable examples.

Definition 3.1. *A triangular tile pasting P system (TTPPS) is defined as 5 tuple $\Pi = (\Sigma, \mu, F_1, F_2, F_3, \ldots F_m, R_1, R_2, R_3, \ldots R_m, i_0)$, where Σ is a finite set of triangular tiles, μ is a membrane structure. In μ, the membranes are labeled in a one-one manner and the labels are $1, 2, \ldots, m$. $F_1, F_2, \ldots, F_m$ are finite sets of pictures over the iso-triangular tiles in Σ associated with m regions of the membranes. $R_1, R_2, \ldots, R_m$ are finite sets of pasting rules of type $((x_i, y_i), tar)$, $(1 \leq i \leq n)$ associated with m regions of μ and i_0 is the output membrane, which is an elementary membrane. An evaluation in TTPPS is defined in such a way that, to each picture pattern in each region of the system, a pasting rule could be applied and should be applied. The picture pattern is moved (retained) to another region (in the same region) with respect to the target indication associated with the pasting rules.*

A computation is successful only if all the pasting rules are applied. The computation is stopped, if there is no possibility of applying the pasting rules. The result of halting picture pattern is composed only by the pasting rules; the pattern is haltered in the output region of membrane i_0. The set of all picture patterns computed with the pasting rules of TTPPS is denoted by TTPPL(Π).

The set of all such languages $TTPPL(\Pi)$ is generated by the system Π is denoted by $TTPPL_m$.

Definition 3.2. *Extended triangular tile pasting P system is similar to the triangular tile pasting P system, but during the computation, to complete the picture pattern, the condition Δ over the pasting rule (a finite set of triangular pasting rules) is considered in the output region with target indications. Formally ETTPPS is defined as $\Pi = (\Sigma, \Sigma', E, \mu, F_i, R_i, \Delta, i_0)$, where*

Σ - *Finite set of labeled iso-triangular tiles.*

Σ' - *Finite set of regular or irregular polygons which are made by gluable iso-triangular tiles.*

E - *The edge set $E = \{E(\Sigma), E'(\Sigma')\}$.*

μ - *The membrane structure it has m membranes.*

R_i - *Finite sets of pasting rules associated with m regions of the membrane structure, $1 \leq i \leq m$.*

F_i - *Finite sets of pictures over the triangular tile set in $\Sigma \cup \Sigma'$ and $1 \leq i \leq m$.*

Δ - *The condition over the pasting rules to complete the picture pattern and it is defined in the output region.*

$\Delta = \{((x_i y_j, z_k), tar) \in R_i / x_i, y_j \in E(\Sigma'), z_k \in E(\Sigma)$ *and* $i, j, k \geq 1\}$. i_0 - *The output (skin) membrane. The computation is successful, if there is no possibility of applying the triangular tile pasting rules. Then the computation is stopped and the resultant picture is the halting one, which is collected in the output region. The set of all picture patterns generated by ETTPPS is called a language denoted by $ETTPPL(\Pi)$. The family of all such languages with at most m membranes is denoted by $PL_m(ETTPPS)$.*

Definition 3.3. *The geometric operations are defined in the following way:*
Translation: A translation of a tile or tiling pattern is changing the position of a tile or tiling pattern from (x, y) into (h, k) in Cartesian plane and due to target indication present in the region, the result picture moved into a region or retained in the same region. The labeled iso-triangular tile or tiling pattern T_0 is translated subject to the position (x, y) in the Cartesian plane.
Rotation: It is a mathematical operation, used to rotate a geometrical shape in the XOY plane. To rotate a tile or tiling pattern through an angle θ either in clockwise direction or anti clock wise direction with respect to an edge present in the tile / tiling pattern and generated picture enter from i^{th} region to j^{th} region. $[T_0]_i = [rot(t_i, \theta)]_j$
Replication: The operation replication is the process of multiplies the tiles in a region into k tiles and due to the target indicator in_j, the replicated tiles are

entered into the region j. An iso-triangular tile t_i or tiling pattern replicates copies which is denoted by $[(t_i)]_i \to [t_{i1}, t_{i2}t_{i3}, \ldots, t_{ik}]_j$.

Path rewriting: (vertex removal and merging of edges). The work of path rewriting a tile / tile pattern is rewritten (removing one or more vertices and merging edges of the path) by a new path of vertices without altering the shape of the tile. The sequence of vertices and edges of the form $v_i(e_iv_{i+1}e_{i+1})v_{i+2}\ldots v_{n-1}e_{i+k}v_n$ to the new sequence $v_ie_jv_{i+2}\ldots v_{n-1}e_{j+l}v_n$ for a finite sequence $k \geq l$ and maintained so for $k = l$.

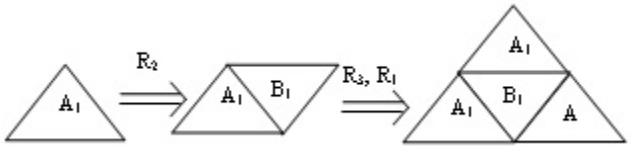
Definition 3.4. *Formally a parametric triangular tile pasting P system is defined as $\Pi = (\Sigma, \mu, F_i, R_i, i_0)$, where Σ is the finite set of labeled iso-triangular tiles, μ is the membrane structure has m membranes. R_i $(1 \leq i \leq m)$ is the finite sets of tile pasting rules associated with m regions of the membrane structure, F_i $(1 \leq i \leq m)$ is the finite sets of pictures over the tile set Σ and i_0 is the output membrane.*

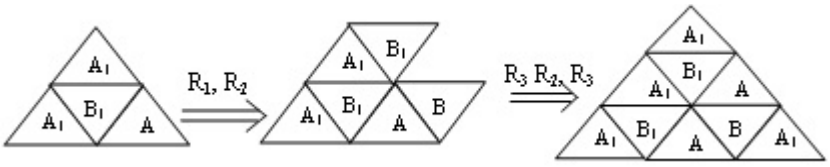
A computation in PTTPPS is defined as like as TTPPS but with geometrical operation. In this computation in each region of the system, the pasting rules of each region can be applied with the geometrical operations. Thus the picture pattern generated by the pasting rules can be placed in the region indicated by the target indicator. A computation is successful only if it stops; the halting computation is a picture such that no further rules can be applied to the existing picture pattern and which is the halting one in the membrane mentioned as output membrane. The set of all such picture pattern computed by PTTPPS is denoted by PTTPP(L) and the family of all such languages generated by Π is denoted by PL_m(PTTPPS).

Example 3.1. The triangular tile pasting P system $\Pi_1 = (\Sigma, [_1[_2]_2[_3]_3]_1, F_1, F_2, F_3, R_1, R_2, R_3, 1)$ generates a class of two dimensional iso-triangular picture language $\mathcal{L}_1$ with at most 4 distinct labeled iso- triangular tiles.
Here $\Sigma = \{A, A_1, B, B_1\}$, $F_1 = \emptyset$, $F_2 = A_1$, $F_3 = \emptyset$,
$R_1 = \{(A, (a_3, b_3), in_2), (B_1, (b_{11}, a_1), here)\}$,
$R_2 = \{(B, (b_2, a_{12}), in_3), (B, (b_1, a_{11}), in_3), (A_1, (a_{13}, b_{13}), in_3)\}$,
$R_3 = \{(B, (b_2, a_2), in_2), (B_1, (b_{12}, a_{12}), out)\}$ and 1 is the output region.

The rules of R_1, R_2 and R_3 are applied and $tar \in \{here, out\} \cup \{in_j\}$, $1 \leq j \leq m$ and then the resultant iso-triangular picture pattern is collected in the output region one finally. Here a copy of the resultant picture pattern is consider to generate the next member of the language.

Two members of the Picture language $\mathcal{L}_1$ is shown with derivation steps.





Example 3.2. The picture language $\mathcal{L}_2$ consisting of squares is constructed by the ETTPPS $\Pi_2 = (\Sigma, \Sigma', E, \mu, F_1, F_2, R_1, R_2, \Delta, 1)$, where
$\Sigma = \{A, B, C, D\}$, $\Sigma' = \{S, T_1, T_2\}$, $E = \{E(\Sigma), E(\Sigma')\}$,
$E(\Sigma) = \{a_1, a_2, a_3, b_1, b_2, b_3, c_1, c_2, c_3, d_1, d_2, d_3\}$,
$E(\Sigma') = \{s_1, s_2, s_3, s_4, t_{11}, t_{12}, t_{13}, t_{14}, t_{21}, t_{22}, t_{23}, t_{24}\}$, $\mu = [_1[_2]_2]_1$. $F_1 = S$, $F_2 = \emptyset$. $R_1 = \{((s_1, t_{13}), in), ((s_2, t_{24}), in)((a_2, t_{13}), here), ((t_{14}, t_{12}), in), ((d_2, t_{22}), here), ((t_{23}, t_{21}), in)$, $\Delta = \{((t_{12}t_{21}, a_1), here), ((t_{23}t_{12}, c_1), here), ((t_{23}t_{14}, b_3), in), ((t_{21}t_{14}, d_1), here)\}$, $R_2 = \{((s_3, t_{11}), out), ((s_4, t_{22}), out), ((b_2, t_{11}), here), ((t_{12}, t_{14}), out), ((c_2, t_{24}), here), ((t_{21}, t_{23}), out)\}$

The picture language $\mathcal{L}_2$ is shown in fig. 1

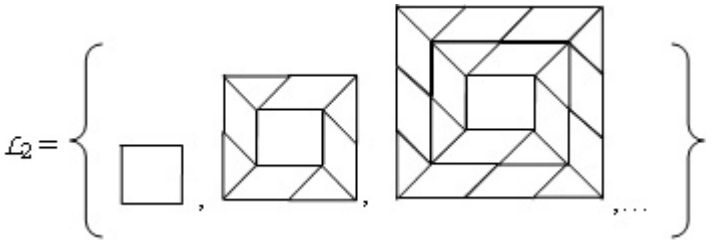
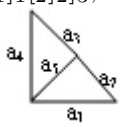


Figure 1: Language of squares

Example 3.3. The language consisting right angled triangles are generated by PTTPPS Π_3 and it is constructed with three membranes.
$\Pi_3 = (\Sigma, [_3[_1]_1[_2]_2]_3, F_1, F_2, F_3, R_1, R_2, R_3, 1)$, where $\Sigma = \{A, D\}$,

$F_1 = T_0 =$  , $F_2 = \emptyset$, $F_3 = \emptyset$.

In region one, the geometrical operation replication has been taken and the rules of R_1 is $R_1 = \{[T_0]_1 \rightarrow [T_0, T_1, T_2, T_3]_2\}$. The edges of the tile T_1, T_2, T_3 are b_i, c_i, d_i respectively, where $b_i = \phi_1(a_i)$, $c_i = \phi_2(a_i)$, $d_i = \phi_3(a_i)$. $R_2 = \{[T_0]_2 \rightarrow [T_0]_3, [T_1]_2 \rightarrow [T_1]_3, [T_2]_2 \rightarrow [T_2]_3, [T_3]_2 \rightarrow rot(T_3, 180^\circ]_3\}$, that is in R_2, the geometrical operation rotation is considered to generate the picture pattern. $R_3 = \{[T_0, T_1, T_2, rot(T_3, 180^\circ), (a_3, d_2), (a_2, d_3)(d_4, b_4), (d_1, c_1), |a_1|b_1| \rightarrow |a_1|, |a_4c_4| \rightarrow |a_4|, |b_2b_3c_2c_3| \rightarrow |a_2a_3|]in_1$. The first three members of the picture language $\mathcal{L}_3$ is shown in fig. 2.

Theorem 3.1. $\mathcal{L}(ETTPPS) \cap \mathcal{L}(TTPPS) \neq \phi$.

Proof. To explain the result, the language $\mathcal{L}_2$ generated by ETTPPS can also be generated by TTPPS. But the language $\mathcal{L}_4$ (refer L_3 in Theorem 8 of [9]).

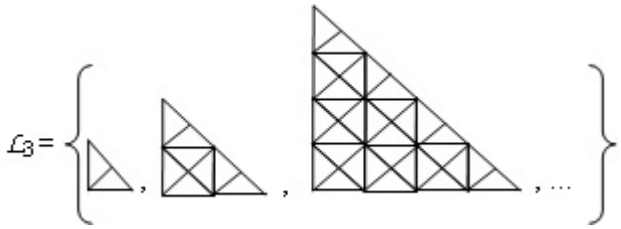


Figure 2: A set of digitized right angled triangles

The picture language $\mathcal{L}_3$ which can also be generated by the TTPPS $\Pi_4 = (\Sigma, [_1[_2]_2[_3]_3]_1, F_1, F_2, F_3, R_1, R_2, R_3, 3)$, where $\Sigma = \{A, B, C, D\}$,

$F_1 =$ 

, $F_2 = \emptyset$ and $F_3 = \emptyset$

$R_1 = \{(A, (a_1, d_1), here), (A, (a_3, c_1), here)), (C, (c_1, a_3), here), (D, (d_1, b_3), in_3)\}$,
$R_2 = \{(B, (b_1, c_3), here), (B, (b_3, c_1), here), (D, (d_1, b_3), here), (D, (d_3, a_1), in_1)\}$,
$R_3 = \{(A, (a_2, b_2), here), (B, (b_2, a_2), here), (C, (c_2, d_2), here), (D, (d_2, c_2), in_2)\}$
and 3 is the output region. But in ETTPPS, to complete the generation we must use the condition Δ over the pasting rule. So the triangular picture language generated by TTPPS cannot always be generated by ETTPPS. It proves the result. □

Theorem 3.2. *$PL_3(PTTPPS, \alpha) \subset TTPPL_3$, where α is the geometrical operation rotation on the iso-triangular tiles.*

Proof. To examine the result consider a class of language digitized right angled triangles generated by PTTPPS with the geometrical operation α. (refer example 3.3). But by parallel mechanism in TTPPS one can generate different classes of languages other than the language generated by PTTPPS. This result can be proved by an example.
The TTPPS $\Pi_5 = (\Sigma, [_1[_2]_2]_1, F_1, F_2, R_1, R_2, 1)$, where $\Sigma = \{A, B, C, D\}$,
$R_1 = \{(a_1, d_3), here), (a_3, c_1), in), (c_2, d_2), in), (c_1, a_3), out)\}$,
$R_2 = \{(d_1, b_3), here), (b_2, a_2), out), (d_3, a_1), out)(b_1, c_1), out)\}$.

$F_1 =$ 

, $F_2 = \emptyset$.

This system generates not only the language consists of right angled triangles but also the picture language of squares with output membranes one and two. □

Theorem 3.3. *The number of triangular tile types required by PTTPPS with rotation is less than the tiles required by TTPPS.*

Proof. Let the primary tiles be $P = \{t_1, t_2, t_3, t_4\}$. For each primary triangular tile $t_i \in P (1 \leq i \leq 4)$, let $\Sigma'' = \{t'_1, t'_2, t'_3, t'_4\}$ be the secondary tiles derived from the primary tiles by the gluing property as well as the geometrical operation rotation. Let $\Sigma = P \cup \Sigma''$ be the set of tiles or tiling patterns required to

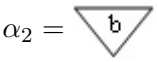
generate the picture language. Using triangular tile pasting P system we can generates the picture language by the derivations. $t_0 \Rightarrow T_1 \Rightarrow T_2 \Rightarrow \cdots \Rightarrow T_i$. With the help of geometrical operation rotation in parametric triangular tile pasting P system, the picture pattern T_i can be generated by Σ''. Here it is noticed that to generate the pattern T_i only less number of iso-triangular tiles are used. Thus the operation rotation reduces the number of tile types. □

4. Iso-triangular domino array splicing P system

In this section motivation of rectangular array splicing system, an iso-triangular domino array splicing system and iso-triangular domino array splicing P system are introduced with the examples.

Definition 4.1. *An iso-triangular domino array splicing system (ITDASS) is a splicing system, it is defined as* $\Gamma = (V = (\Sigma \cup \Lambda \cup \{\#, \$\}), A, A_i, (S_i, P_i))$, *where* Σ *is the finite set of labeled iso-triangular tiles,* Λ *is an empty symbol and* $\{\#, \$\}$ *is a set of special characters involved in splicing of two iso- triangular domino arrays.* A *is an axiom of triangular picture pattern,* A_i *is the finite set of intermediate iso-triangular arrays used to generate the iso-triangular arrays,* (S_i, P_i) *is a pair of finite set of splicing rules and pasting rules in which* S_i $(1 \leq i \leq n)$ *are the iso-triangular domino array splicing rules and* P_i $(1 \leq i \leq n)$ *are the associated pasting rules over the elements of* Σ. *The iso-triangular domino array splicing rules over* Σ *are given as a scheme* $S_i = \alpha_1 \# \alpha_2 \$ \alpha_3 \# \alpha_4$ *if*

(1) *The horizontal iso-triangular domino array splicing over* V *is*

$\alpha_1 =$  *or* Λ, $\alpha_2 =$  *or* Λ, $\alpha_3 =$  *or* Λ, $\alpha_4 =$  *or* Λ.

(2) *The vertical iso-triangular domino array splicing over* V *is*

$\alpha_1 =$ *or* Λ, $\alpha_2 =$  *or* Λ, $\alpha_3 =$  *or* Λ, $\alpha_4 =$  *or* Λ.

(3) *The right iso-triangular domino array splicing over* V *is any one of the following rule*

(*a*) $\alpha_1 =$  *or* Λ, $\alpha_2 =$  *or* Λ, $\alpha_3 =$  *or* Λ, $\alpha_4 =$  *or* Λ. *(right)*

(*b*) $\alpha_1 =$  *or* Λ, $\alpha_2 =$  *or* Λ, $\alpha_3 =$  *or* Λ, $\alpha_4 =$  *or* Λ. *(right down)*

(c) $\alpha_1 =$  *or* Λ, $\alpha_2 =$  *or* Λ, $\alpha_3 =$  *or* Λ, $\alpha_4 =$  *or* Λ. *(right down)*

(d) $\alpha_1 =$  *or* Λ, $\alpha_2 =$  *or* Λ, $\alpha_3 =$  *or* Λ, $\alpha_4 =$  *or* Λ. *(right down)*

(e) $\alpha_1 =$  *or* Λ, $\alpha_2 =$  *or* Λ, $\alpha_3 =$  *or* Λ, $\alpha_4 =$  *or* Λ. *(right up)*

(4) *The left iso-triangular domino array splicing over* V *is any one of the following rule*

(f) $\alpha_1 =$  *or* Λ, $\alpha_2 =$  *or* Λ, $\alpha_3 =$  *or* Λ, $\alpha_4 =$  *or* Λ. *(left)*

(g) $\alpha_1 =$  *or* Λ, $\alpha_2 =$  *or* Λ, $\alpha_3 =$  *or* Λ, $\alpha_4 =$  *or* Λ. *(left down)*

(h) $\alpha_1 =$  *or* Λ, $\alpha_2 =$  *or* Λ, $\alpha_3 =$  *or* Λ, $\alpha_4 =$  *or* Λ. *(left up)*

(i) $\alpha_1 =$  *or* Λ, $\alpha_2 =$  *or* Λ, $\alpha_3 =$  *or* Λ, $\alpha_4 =$  *or* Λ. *(Left down)*

(j) $\alpha_1 =$  *or* Λ, $\alpha_2 =$  *or* Λ, $\alpha_3 =$  *or* Λ, $\alpha_4 =$  *or* Λ. *(left up)*

(k) $\alpha_1 =$  *or* Λ, $\alpha_2 =$  *or* Λ, $\alpha_3 =$  *or* Λ, $\alpha_4 =$  *or* Λ. *(left down)*

The horizontal, vertical, right, right up, right down, left, left down, left up are denoted by $\$_H$, $\$_R$, $\$_{RU}$, $\$_{RD}$, $\$_L$, $\$_{LU}$, $\$_{LD}$ *respectively. Generally iso-triangular domino array splicing is denoted by* $\$_D$. *If* X *and* Y *are two different iso-arrays then iso-triangular domino array splicing over* X *and* Y *gives the new picture* Z, *is denoted by* $(X, Y) \underset{(S,P)}{\vdash} Z$, *pasting rules* $P = (x, y)$ *where* x *and* y *are the edges of the iso-triangular tiles presented in* X *and* Y *respectively. It*

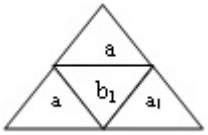
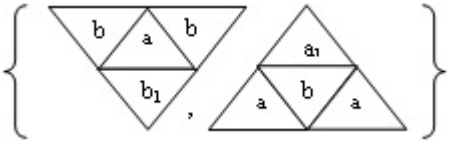
is noted that the computation is begin with the axiom of the system, then the iso-triangular domino array splicing rule is applied with the axiom and the intermediate iso-triangular arrays. The resultant iso-triangular picture has been collected and many copies have been taken as much as required to generate the next members of iso-triangular picture language.
The iso-triangular picture language generated by the system over V^{**} is denoted by $L(\Gamma) = \{M_i \in V^{**}/A \Rightarrow M_i (1 \leq i \leq n)\}$.

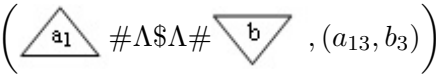
Definition 4.2. *An iso-triangular domino array splicing P system is defined as $\Pi = (V, \mu, L_i, ((S_i, P_i)_{tar}), i_0)$, where $1 \leq i \leq m$, $V = \{\Sigma \cup \Lambda \cup \{\#, \$\}\}$, Σ is the finite set of labeled iso-triangular tiles, Λ is the symbol used to denote the empty place, # and \$ are the special symbols used to splice iso-triangular dominos, which are not in Σ. μ is the membrane structure, it has m membranes. L_i is the initial iso-triangular array present in the m^{th} region. S_i is an iso-triangular domino array splicing rule over V associated with the region R_i $(1 \leq i \leq m)$. The splicing rules is of the form $S_i = \alpha_1 \# \alpha_2 \$ \alpha_3 \# \alpha_4$, where α_i's are defined in definition 4.1. P_i is the pasting rule along with the operation splicing over iso-triangular arrays and $target = \{here, out, in_j\}$ $(1 \leq j \leq m)$. Here means the resultant picture is retained in the same region, out means the picture pattern sent out from the membrane and in_j means the array can be sent into i^{th} region the membrane j. i_0 is the output membrane in which the resultant picture or iso-triangular arrays are collected.*

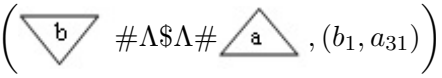
The computation starts with an initial iso-triangular array present in the first region. Note that an iso-triangular array present in the m^{th} region can be taken arbitrarily many copies. If X and Y are any two different iso-triangular arrays then the iso-triangular domino array splicing rule S_i is applied in parallel way between X and Y. The splicing rule generates a new picture W and is denoted by $(X, Y) \underset{\$_D}{\vdash} Z$. A finite sequence of iso-triangular arrays is said to be an iso-triangular array language. Here the sequence is computed by applying the iso-triangular domino splicing rules to the finite connected iso-triangular arrays L_i present in the corresponding regions. The computation of iso-triangular arrays are denoted like if $(T_1, T_2, \ldots, T_m)$ and $(T'_1, T'_2, \ldots, T'_m)$ are two sets of iso-triangular arrays over V. Then the P system gives $(T_i, T_j) \underset{\$_D}{\vdash} T'_i$. The computation stops if no splicing rule can be applied further. At last the resultant iso-triangular array is collected in the output region i_0 or sent out to the environment, it depends the target symbol associated with membrane region. $\mathcal{L}(\Pi) = \{T'_i/(T_i, T_j) \underset{\$_D}{\vdash} T'_i, 1 \leq j \leq m\}$. The family of all iso-triangular array languages is denoted $\cup \mathcal{L}_i$.

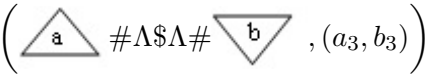
Example 4.1. The iso-triangular domino array splicing system ITDASS $= (V, A, A_i, (S_i, P_i))$, where $1 \leq i \leq 2$ generates rectangular picture language.

$V = \left\{ \text{a} , \text{a}_1 , \text{b} , \text{b}_1 \right\} \cup \Lambda \cup \{\#, \$\}$

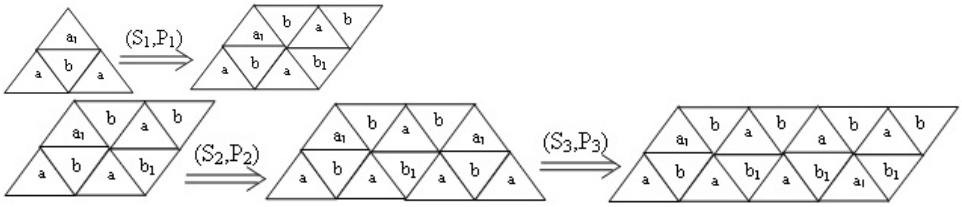
where $A =$  $A_i =$ 

$(S_1, P_1) =$ 

$(S_2, P_2) =$ 

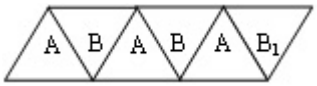
$(S_3, P_3) =$ 

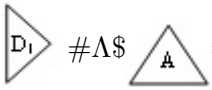
The derivation steps of two members of the picture language are shown below.

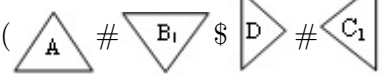


Example 4.2. A class of language consisting of staircase model of iso-triangular arrays is generated by an iso-triangular domino array splicing system Π_6.

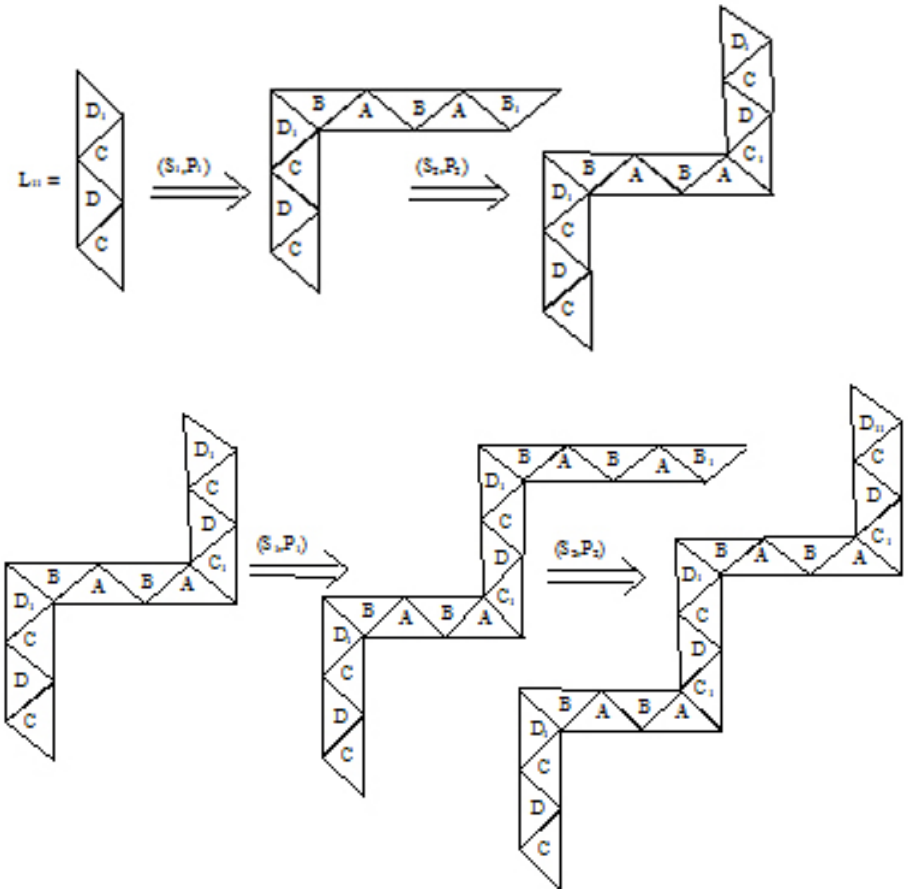
Consider an iso-triangular domino array splicing system
$\Pi_6 = (V, [_1[_2]_2]_1, L_1, L_2, S_1 = (S_i, P_i)_{tar}, 2)$ where $i = 1, 2$
$V = \{\Sigma \cup \{\Lambda\} \cup \{\#, \$\}\}$ and $\Sigma = \{A, B, C, D\}$, $L_1 = \{L_{11}, L_{12}\}$, L_2 are the initial arrays presented in the regions one and two respectively. S_i is the set of splicing rules in the regions R_i and 2 is the output membrane.

$L_{11} =$  , $L_{12} =$  , $L_2 =$ 

$(S_1, P_1) = ($  $, (d_1, b_{13}))_{in}$

$(S_2, P_2) = ($  $, (a_3, c_{11}))_{out}$

Two members of stair case model of iso-triangular arrays are generated by an iso-triangular domino triangular array splicing P system Π_6.

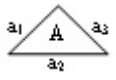


In region one, the triangular arrays L_{11} and L_{12} are spliced by the splicing rule (S_1, P_1) and the picture generated is sent to the region two and due to the target symbol 'in'. The splicing rule (S_2, P_2) is applied to the generated picture and the iso-triangular array L_2 present in the region two. The result is a picture of staircase model of iso-triangular arrays. Once the triangular array is generated, a copy has to be taken and the picture will go to the 1st membrane. Again the computation will start and the successive members of the language will be generated.

Now an algorithm can be written to generate triangular picture languages.

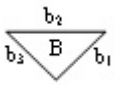
Algorithm

Step I Set the members of triangular picture language as M_i $(1 \leq i \leq n)$

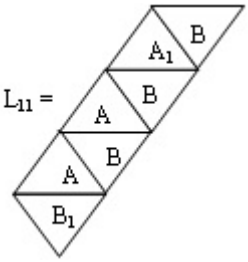
Step II Formation of iso-triangular tile $A =$ 

1. Set an edge $a_1 = x$ cm.
2. Set an edge $a_2 = 2x$ cm.
3. Set an edge $a_3 = x$ cm.
4. Join a_1 and a_2 through an angle 45°
5. Join a_2 and a_3 through an angle 45°

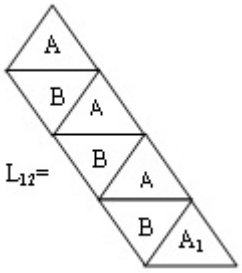
6. Label the tile as A.
7. Take a copy of the triangular tile A.
8. Rename the tile as A_1 and its edges are $|a_{11}| = |a_1|$, $|a_{12}| = |a_2|$, $|a_{13}| = |a_3|$

Step III Formation of iso-triangular tile $B =$ 

9. Rotate an iso- triangular tile A through an angle $180°$.
10. Label the tile as B and the edges of B are $|b_1| = |a_1|$, $|b_2| = |a_2|$, $|b_3| = |a_3|$
11. Take a copy of tile B and label it as B_1 and its edges are $|b_{11}| = |b_1|$, $|b_{12} = |b_2|$, $|b_{13}| = |b_3|$

Step IV Computation of $L_{11} =$ 

12. Join the edges b_{12} of iso-triangular tile B_1 and a_2 of iso-triangular tile A and label the generated triangular array as T_{11}
13. Then join the edge a_3 of tile A in T_{11} and the edge b_3 of tile B and assign a label to the generated triangular array as T_{12}
14. Then join the edge b_2 of tile B in T_{12} and the edge a_2 of tile A and assign a label to the generated triangular array as T_{13}
15. Repeat the step 13 to the new triangular array T_{13} and label the triangular array as T_{14}
16. Then join the edge b_2 of B in T_{14} and the edge a_{12} of tile A_1 and label the triangular array as T_{15}
17. Then join the edge a_{13} of tile A_1 in T_{15} and b_3 of tile B and label the triangular array as L_{11}.

Step V Computation of $L_{11}=$ 

18. Join the edges a_{11} of iso-triangular tile A_1 and b_1 of iso-triangular tile B and label the generated triangular array as T_{21}
19. Then join the edge b_2 of tile B in T_{21} and the edge a_2 of tile A and assign a label to the generated triangular array as T_{22}
20. Then join the edge a_1 of tile A in T_{22} and the edge b_1 of tile B and assign a label to the generated triangular array as T_{23}
21. Repeat the step 19 to the triangular array T_{23} and label the generated

triangular array as T_{24}.
22. Repeat the step 20 to the triangular array T_{24} and label the generated triangular array as T_{25}
23. Then repeat the step 19 to the triangular array T_{25} and label the generated triangular array as L_{12}

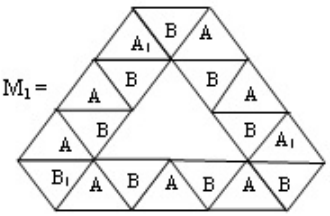
Step VI Computation of L_2 =

24. Join the edges b_1 of iso-triangular tile B and a_1 of iso-triangular tile A and label the generated triangular array as T_{31}
25. Then join the edge a_3 of tile A in T_{31} and the edge b_3 of tile A and assign a label to the generated triangular array as T_{32}
26. Repeat the step 24 to the triangular array T_{32} and label the generated triangular array as T_{33}
27. Repeat the step 25 to the triangular array T_{33} and label the generated triangular array as T_{34}
28. Repeat the step 24 to the triangular array T_{34} and label the generated triangular array as T_{35}
29. Repeat the step 25 to the triangular array T_{35} and label the generated triangular array as L_2

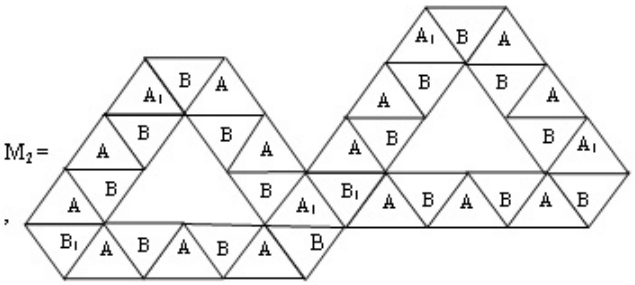
Step VII Computation of M_i $(i = 1)$
30. Remove the topmost tile B pasted with labeled tile A_1 in L_{11} and remove the topmost tile A pasted with tile B in L_{12}, then adjoin the triangular arrays and label the generated tile as T_1
31. Remove leftmost tile B in L_2 and adjoin the array with the left bottom tile B_1 of T_1 and then label the generated triangular array as M_i $(i = 1)$
32. Take a copy of M_1 and join the bottom tile B_1 of L_{11} with the right hand side tile A_1 of M_1 and label generated triangular tile as T_2
33. Repeat the steps 31 to the triangular array T_2 and set the label to the generated triangular array as M_2

Step VIII Computation of M_i $(1 \leq i \leq n)$
Repeat the steps 32 and 33 to get the successive members M_i $(1 \leq i \leq n)$ (Refer Example 4.3).

M_1 =



M_2 =

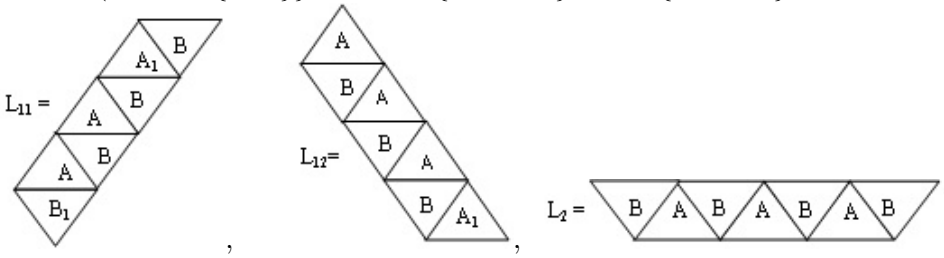


This algorithm can be used to generate the triangular picture language $\mathcal{L}_5$.

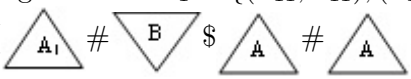
Example 4.3. An iso-triangular domino array splicing P system Π_7 generates a language consisting adjoin of hallow triangles by four iso-triangular tiles A, B, A_1, B_1

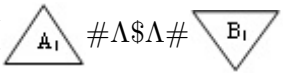
$$\Pi_7 = (V, [_1[_2]_2]_1, L_1, L_2, (S_i, P_i)_{tar}, 1)$$

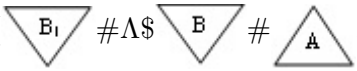
where $V = (\Sigma \cup \Lambda \cup \{\#, \$\}$ and $\Sigma = \{A, B, A_1\}$, $L_1 = \{L_{11}, L_{12}\}$,



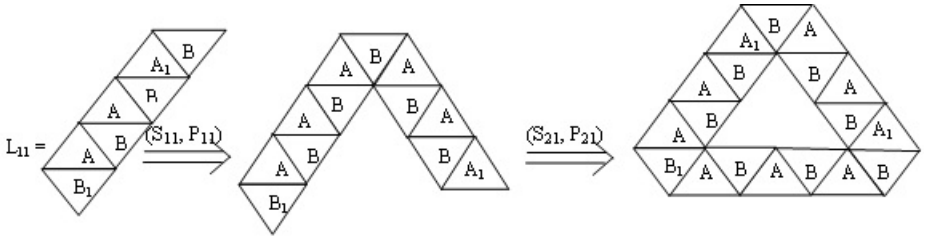
The splicing rules are $S_1 = \{(S_{11}, P_{11}), (S_{12}, P_{12})\}$, $S_2 = \{(S_{21}, P_{21})\}$ where

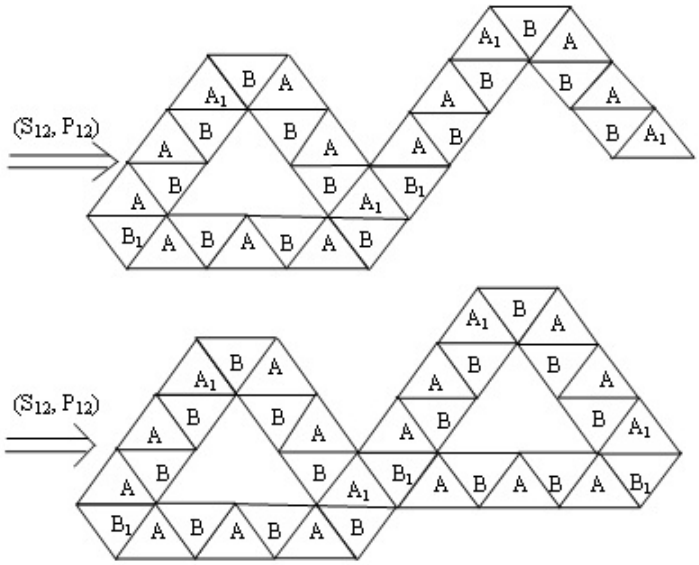
$(S_{11}, P_{11}) = ($  $, (a_{13}, b_3)_{in}$,

$(S_{12}, P_{12}) = ($  $\#\Lambda\$\Lambda\#$ $, (a_{13}, b_3))_{here}$,

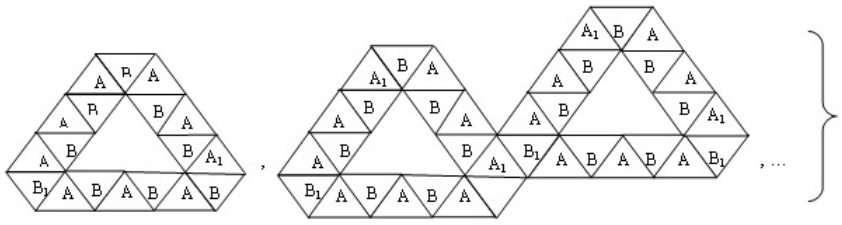
$(S_{21}, P_{21}) = ($  $, (b_1, a_1), (b_2, a_2))_{out}$

The computation begins with the triangular arrays L_{11}, L_{12} in R_1 and the iso-triangular domino array splicing rule (S_{11}, P_{11}) spliced the iso-triangulars L_{11} and L_{12}. The target symbol associated with the region one sends the picture into region two. In the region two, the iso-triangular domino array splicing rule (S_{21}, P_{21}) is applied to the result picture and L_2. The hallow iso-triangular picture is generated. Once a hallow triangular picture is generated copies of the member is taken. Again the computation starts with a copy of generated one to generate 2^{nd} member of triangular picture language. The first member is spliced by the triangular domino array splicing (S_{12}, P_{12}) in the region one. Then the splicing rule (S_{11}, P_{11}) is applied to the result triangular picture. Target symbol associated with (S_{11}, P_{11}) sends the picture into the region two. In region two, splicing is made between the resultant picture and the iso-array L_2. The the 2^{nd} member of a class of language consists of adjoin triangles made of iso-triangular tiles with hallows is generated in this way. Two members of hallow triangular picture language is derived by the following steps.



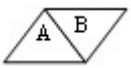
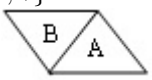


The hallow triangular picture language is shown below

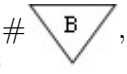
$\mathcal{L}_5 =$ 

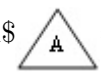
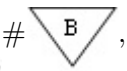
Result 4.1. $ITDASPS \cap TTPPS \neq \emptyset$

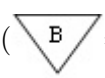
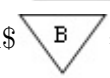
To explain the result the picture language consisting of iso-triangles generated by TTPPS Refer Example 3.1 can also be generated by $ITDASPS$ with three membranes.

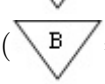
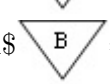
To explain the result consider the ITDASPS system $\Pi_7 = (V, [_1[_2]_2[_3]_3]_1, L_1, L_2, L_3, (S_i, P_i)_{tar}, 1)$ where $V = \Sigma \cup \{\Lambda\} \cup \{\#, \$\}$ and $\Sigma = \{A, A_1, B\}$, $L_1 =$ $\{L_{11} =$  $, L_{12} =$  $\}$, $L_2 =$, $L_3 =$  , the splicing rules are

$\{(S_1, P_1) = \{(S_{11}, P_{11}), (S_{12}, P_{12})\}, (S_2, P_2), (S_3, P_3)\}$ where

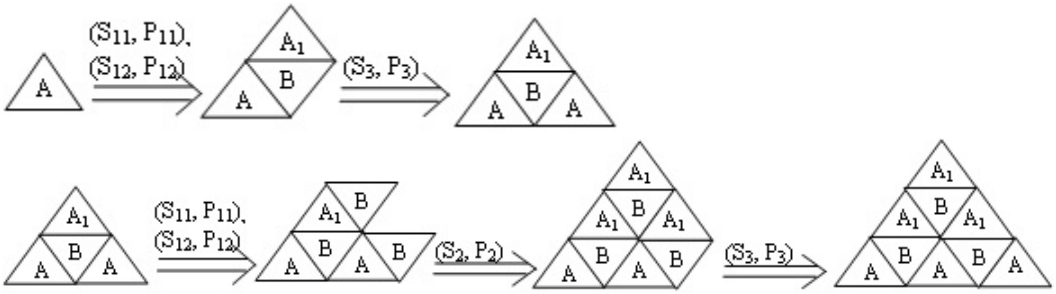
$(S_{11}, P_{11}) = ($  $\#\Lambda\$$  $\#$  $, (a_3, b_3))_{in2},$

$(S_{12}, P_{12}) = ($  $\#\Lambda\$$  $\#$  $, (a_{13}, b_3))_{here}$

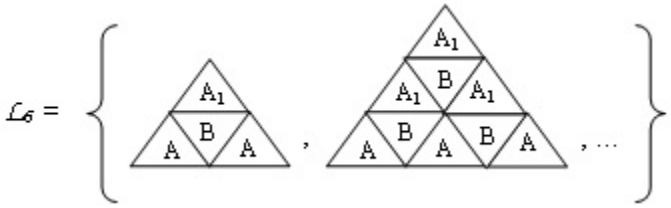
$(S_2, P_2) = ($  $\#\Lambda\$$  $\#$  $, (b_2, a_{12})_{in3}$

$(S_3, P_3) = ($  $\#\Lambda\$$  $\#$  $, (b_1, a_1)_{out}$

The derivation steps of two members of the iso-triangular picture language is shown below



The iso-triangular picture language is shown below



Once a member of the language is computed in the region 3, the picture pattern sent out from the region with many copies. Again a computation is started with a copy of the member to generate the next successive member of the iso-triangular picture language.

But the triangular picture language with hallows generated by $ITDASPS$ (Π_7) cannot be generated by any TTPPS. Due to parallel mechanism the tiles A and B are glued each other we cannot generate such hallow iso triangles. The picture language shown in example 4.2 can be generated by TTPPS. It means that not all the triangular picture language generated by ITDASPS can be computed by the rules of TTPPS. Hence it proved that intersection of these two P systems are non-empty.

Result 4.2. An iso triangular domino array splicing P system is not comparable with the extended triangular tile pasting P system.

Proof. It is clear from the defns 3.2 and 4.2 □

Result 4.3. An iso triangular domino array splicing P system cannot be compared with parametric triangular tile pasting P system.

Proof. It is clear from the defn 3.3 and 4.2 □

Result 4.4. $\mathcal{L}(ITDASPS) - \mathcal{L}(IARP_m(CFPIAG)) \neq \emptyset$.

To prove this result, it is considered that the language $\mathcal{L}_5$ generated by ITDASPS cannot be generated by any ITARPS with CFIA rules. Because in the pasting rules of context free iso array grammar, no terminal symbol can be rewritten once a member is generated. It means, a triangular picture or array with terminals cannot be used to generate or compute successive members of that particular triangular picture language. But the iso-picture language $\mathcal{L}_6$ consisting of iso-triangular arrays (refer Result 4.1) can be generated by ITDASPS system. Hence the two P systems are not disjoint.

5. Conclusion

This paper introduced iso-triangular domino array splicing system, iso-triangular domino array splicing P system and explained with suitable examples. The computational power of ITDASP system is compared with the existing P systems called triangular tile pasting P system, extended triangular tile pasting system, parametric triangular tile pasting system and iso array rewriting P system with context free iso array rules. The generative powers of types of P system also examined.

References

[1] K. Bhuvaneswari, T. Kalyani, *Triangular tile pasting P system for pattern generation*, in: Conference proceeding of CAMTCS-2013, 2013, 26-30.

[2] K. Bhuvaneswari, T. Kalyani, D. Gnanaraj Thomas, A.K. Nagar, T. Robinson, *Iso-array rewriting P systems with context-free iso-array rules*, International Journal of Mathematics for Applications, 3 (2014), 1-16.

[3] S. Jebasingh, T. Robinson, Atulya K. Nagar, *A variant of tile pasting P system for tiling patterns*, in: Proc. 2010 IEEE Fifth International Conference on Bio-Inspired Computing, Theories and P Applications, BIC-TA 2010, 2 (2010), 1568-1576.

[4] T. Head, Gh. Păun, D. Pixton, *Language theory and molecular P genetics: generative mechanisms suggested by DNA recomination*, in: G. Rozenberg, A. Salomaa (Eds.), Handbook of Formal Languages, Vol. 2, Springer-Verlag, 1997, 296-358.

[5] T. Kalyani, K. Sasikala, V.R. Dare, P.J. Abisha, T. Robinson, *Triangular pasting system, in: formal models*, Languages and Application, Series in Machine Perception and Artificial Intelligence, 66 (2006), 195-211.

[6] Gh. Păun, M.J. Perez-Jimenez, A. Riscos-Nunez, *P systems with tables of rules*, Lecture Notes in Computer Science, 31130 (2004), 235-249.

[7] Gh. Păun, G. Rozenberg, A. Salomaa, *The Oxford handbook of membrane computing*, Oxford Univ. Press, 2010.

[8] T. Robinson, *Extended pasting scheme for kolam pattern generation*, Forma, 22 (2007), 55-64.

[9] T. Robinson, S. Jebasingh, Atulya K. Nagar, K.G. Subramanian, *Tile pasting systems for tessellation and tiling patterns*, Lecture Notes in Computer Science, 6026 (2010), 72-84.

[10] K.G. Subramanian, T. Robinson, A.K. Nagar, *Tile pasting P system model for pattern generation*, in: Proc. Third Asia Internationa Conference on Modeling and Simulation, Indonesia, 2009.

[11] K.G. Subramanian, R. Saravanan, T. Robinson, *P system for array generation and application to kolam patterns*, Forma, 22 (2007), 47-54.

[12] P. Helen Chandra, K.G. Subramanian, D.G. Thomas, D.L. Van, *A note on parallel splicing on images*, Electronic Notes in Theoretical Computer Science, 46 (2001), 255-268.

[13] P. Helen Chandra, K.G. Subramanian, D.G. Thomas, *Parallel splicing on images*, International Journal of Pattern Recognition and Artificial Intelligence, 18 (2004), 1071-1091.

[14] D.K. Sheena Christy, V. Masilamani, D.G. Thomas, *Iso-array splicing grammar system*, Proceedings of 7th International Conference on Bio-Inspired Computing, Theories and Applications, BICTA 2012, Springer, (2012), 157-167.

Accepted: 4.03.2020