International Journal of Pure and Applied Mathematics
Volume 107 No. 1 2016, 111-128
ISSN: 1311-8080 (printed version); ISSN: 1314-3395 (on-line version)
url: http://www.ijpam.eu
doi: 10.12732/ijpam.v107i1.9
ijpam.eu

TRIANGULAR TILE PASTING *P* SYSTEM AND ARRAY GENERATING PETRI NETS

K. Bhuvaneswari[1 §], T. Kalyani[2], D. Lalitha[3]

[1,3]Department of Mathematics
Sathyabama University
Chennai, 600 119, INDIA

[2]Department of Mathematics
St. Joseph's Institute of Technology
Chennai, 600 119, INDIA

Abstract: In the area of membrane computing, a new computability model, called P system is a highly distributed and parallel theoretical computing model. Generating two dimensional picture languages is one of the applications of P systems. Ceterachi et al. began a study on connecting the two areas, membrane computing and array grammars. Triangular tile pasting system and the Triangular Tile pasting P system models have been introduced for pattern generation. This paper introduce the k-order TTPS and ETTPS. Also this paper has extended the theory of Triangular tile pasting P system to generate two dimensional picture languages. A comparative study between triangular tile pasting system and Triangular tile pasting P system is made. Comparison is also made with the array generating Petri nets.

AMS Subject Classification: 68Q05, 68Q10, 68Q42, 68Q45, 68Q55

Key Words: iso-arrays, pasting system, tile pasting P systems, iso-triangular tiles, extended tile pasting system, Petri net

1. Introduction

The art of tiling has been introduced early in human civilization. Motivated by the problem of tiling, puzzle grammars that fall under the category of syntactic

Received: January 25, 2016
Published: March 25, 2016

§Correspondence author

techniques have been proposed by Nivat et al. [9]. Syntactic methods have become one of the main areas of investigation in theoretical studies on digital pictures and image analysis. A two-dimensional pattern generating model called a Pasting System that generates tiling patterns by pasting square tiles at the edges, has been introduced and studied in [12]. Iso-Triangular picture languages were introduced in [1, 2, 3, 5]. Adopting this notion, Triangular Tile Pasting System that pastes edge-to-edge isosceles right angled triangular tiles in the two dimensional plane is considered here.

In the area of membrane computing a new computability model, called P system, which is a distributed, highly parallel theoretical computing model, was introduced by G.H. Păun [10]. This model has been found to be a powerful framework for studying many kinds of computational problems. The basic model processes multi-sets of objects in the regions that are defined by a hierarchical arrangement of membranes, by evolution rules associated with the regions. Ceterachi et al. [4] began a study on linking the two areas of membrane computing and picture grammars, which were not very much linked before, by relating P systems and array re-writing grammars generating picture languages, the author is proposed an array-rewriting P system. Subsequently an Extended pasting scheme with tile types have been introduced in [11].

Petri nets [6] are mathematical models introduced to model dynamic systems. Tokens are used to simulate the dynamic activity of the system. Tokens are represented by black dots, which move when the transition fires. Array token Petri nets [8] are models which can generate array languages. The tokens used are arrays over an alphabet and not just black dots. The transitions are associated with catenation rules. Firing of transitions, catenate arrays to build bigger arrays. Petri Nets models were also introduced in [7] to generate patterns over tiles. The tokens used in this model are arrays made up of tiles. The transitions are associated with catenation rules, which state how the tiles are joined. The firing rules in such models are different from the firing rules of ordinary Petri nets.

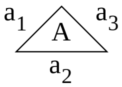
A theoretical model of a pasting system called a Triangular Tile Pasting P System (TTPPS) [1] is considered to generate two dimensional patterns that are formed by gluing triangular tiles and by iso-array grammars. This paper also compares the family of languages generated by Triangular Tile Pasting P System (TTPPS) with the family of languages generated by Petri nets (TATPN).

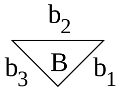
This paper is organized as follows. The second chapter gives all the preliminary definitions required. The third chapter introduce k-order TTPS and ETTPS with examples and the triangular tile pasting P system is considered and obtained certain results of comparison. The fourth chapter recalls the basic definitions of Petri Net, Array generating Petri net and explained them with examples and the comparison study of TTPPS, ETTPS with TATPN are made.

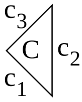
2. Preliminaries

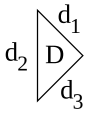
The art of tiling is a well-known theory in the applications of pattern generation. In this section we recollect the notion of triangular tiles and tile pasting system. A tile is a topological disc with closed boundary in the XOY plane, whose edges are gluable. A tiling is a family of countable tiles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

Definition 1. Consider the labeled triangular tiles



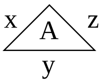
,


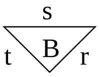
,


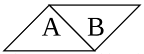
,


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

Definition 2. Generally a pasting rule is a pair (x, y) of labeled tiles with distinct edges (not necessarily distinct).

For example, the 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.

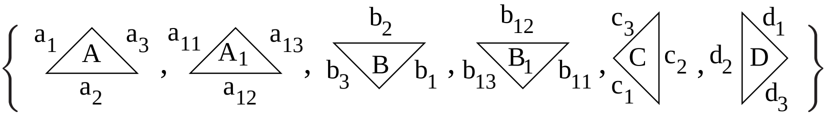
Tile pasting rules of the tiles A, B, C, D are given below:

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 $\{(d_1, c_1), (d_2, c_2), (d_3, c_3)\}$.

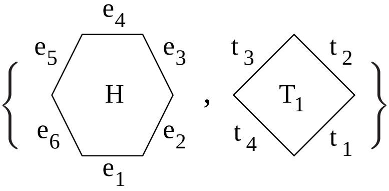
Notations:

1. Iso-triangular tiles

$\Sigma =$



2. Extended Triangular tiles $\Sigma' =$



Definition 3. A Triangular tile pasting system (TTPS) is $S = (\Sigma, P, t_0)$, where Σ is a finite set of labeled triangular tiles, P is a finite set of pasting rules and t_0 is the axiom of pattern. A pattern p_2 is generated from a pattern p_1 by applying the pasting rules in parallel manner to the edges of the pattern p_1, where pasting is possible. Note that the labels of pasted edges in a pattern are ignored once the tiles are pasted. The set of all patterns generated from the axiom t_0 constitutes the picture language $L(TTPS)$.

3. Types of Triangular Tile Pasting Systems, P System and Results

Definition 4. A k-order triangular tile pasting system is defined in such a way that k-TTPS $= (\Sigma, P, t_0)$ where Σ is the finite set of labeled iso-triangular tiles and P is the fine set of pasting subsets $(P = \{P_1, P_2, \ldots, P_k\}, k \geq 1)$ in which the pasting rules of each subset are applied one by one simultaneously. If the pasting rules of $P_1, P_2, \ldots, P_{k-1}$ of P are applied simultaneously and then the rules of P_k of P are applied completely for the completion of picture pattern. The language generated by the k-order TTPS is denoted by L(k-order TTPS) and the family of all languages generated by k-order TTPS is denoted $\mathcal{F}_{k-order}(\mathcal{L})$.

Definition 5. An extended triangular tile pasting system is a 6 tuple (ETTPS), ETTP$(S) = (\Sigma, \Sigma',\ E = \{E(\Sigma), E'(\Sigma')\}P, t_0, \Delta)$, where Σ is the finite set of labeled iso-triangular tiles and Σ' is the finite set of labeled regular or irregular polygons which are glued by iso-triangular tiles, E contain the edge sets of Σ, Σ', P is the finite set of pasting rules, t_0 is the axiom of pattern in $\Sigma \cup \Sigma'$ and Δ is the constraint over the edges of tiles, $\Delta = \{(x_i y_j, z_k)/x_i, y_j \in E(\Sigma')$ and $z_k \in E(\Sigma)$ where $i, j, k \geq 1\}$.

Example 6. Consider the extended triangular tile pasting system ETTP(S)

$= (\Sigma, \Sigma^{\square}, E, P, t_0, \Delta)$ where $\Sigma = \{C, D\}$, $\Sigma^{\square} = \{H, T_1\}$ and $t_0 = H$, $E = \{E(\Sigma) = \{c_1, c_2, c_3, d_1, d_2, d_3\}$, $E(\Sigma^{\square}) = \{e_1, e_2, e_3, e_4, e_5, e_6, t_1, t_2, t_3, t_4\}\}$, $P = \{(e_4, e_1), \{(e_3, t_3), \{(e_5, t_1)\}$, and $\Delta = \{(t_1t_2, c_1), (t_3t_4, d_1)\}$ which generates the language of octagons. Three members of the picture language are shown in Figure 1.

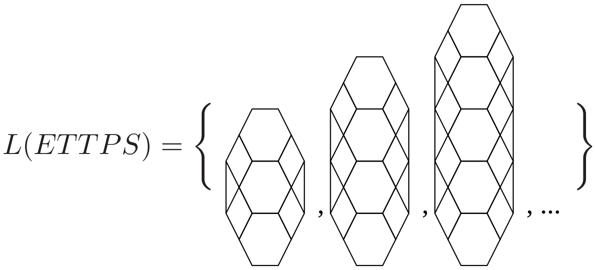


Figure 1: Language of Octants

Example 7. Consider the extended triangular tile pasting system ETTP(S_1) $= (\Sigma, \Sigma^{\square}, E, P, t_0, \Delta)$, where
$\Sigma = \{A, B\}$, $\Sigma^{\square} = \{H\}$, $t_0 = H$
$E(\Delta) = \{a_1, b_1, a_2, b_2, a_3, b_3\}$, $E^{\square}(\Sigma^{\square}) = \{e_1, e_2, e_3, e_4, e_5, e_6\}$ and $P = \{(e_4, a_2)$, (e_6, a_3), (e_2, a_1), (e_1, b_2), (e_3, b_3), (e_5, b_1), (a_2, e_4), (b_2, e_1), (a_3, e_6), $(b_3, e_3)\}$ and $\Delta = \{(e_2e_6, a_1)$, (e_2e_4, a_2), (e_4e_6, a_3), (e_1e_5, b_1), (e_3e_1, b_2), $(e_3e_5, b_3)\}$. The language of irregular hexagon pattern is generated in parallel manner, and the language is shown with three members in Figure 2.

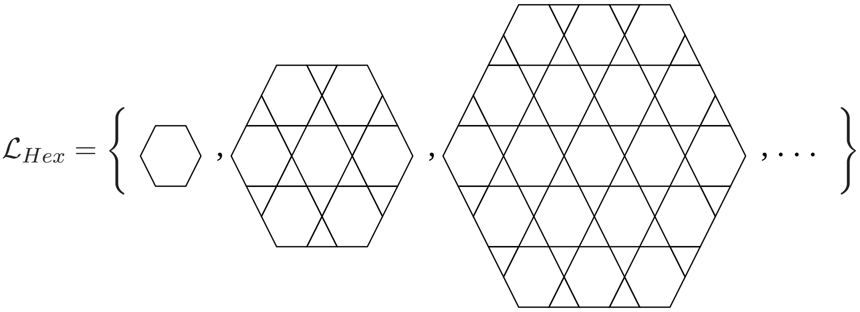
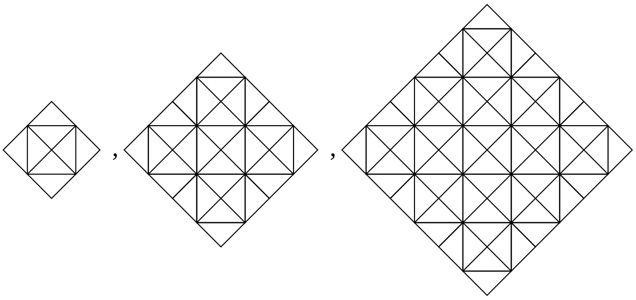


Figure 2: Hexagonal picture language

Theorem 8. *A class of language generated by k-order triangular tile pasting system cannot be generated by triangular tile pasting system.*

Proof. To explain the result, the 2-order TTPS $= (\Sigma, P = \{P_1, P_2\}, t_0)$ is considered, here $\Sigma = \{A, B, C, D\}$, $P_1 = \{(a_1, d_1), (a_3, c_1), (d_1, b_3), (b_1, c_3), (c_1, a_3)\}$, $P_2 = \{(a_2, b_2), (b_2, a_2), (c_2, d_2), (d_2, c_2)\}$ and $t_0 = A$. By parallel mechanism, the iso-triangular tiles A, B, C and D are glued each other by the pasting rules of P_1 and the resultant pattern is a square. In the next step, in each side of the square the iso-triangular tiles are pasted by the rules of P_2. But in TTPS all the rules are considered in a set P and due to parallel mechanism in TTPS, the picture language of rhombus cannot be generated. The picture language is shown below.

$L_3 = \Bigg\{$  $, \ldots \Bigg\}$ □

Triangular Tile Pasting P System

A triangular tile pasting P system (TTPPS) is defined as

$$\Pi = (\Sigma, \mu, F_1, F_2, \ldots, F_m, R_1, R_2, \ldots, R_m, i_0),$$

where Σ is a finite set of iso-triangular tiles, μ is a membrane structure. In μ, the membranes are labeled in a one-one manner with the labels $1, 2, \ldots, m$, $F_1, F_2, \ldots, F_m$ are finite sets of picture over the tiles of Σ associated with m regions of the membranes, $R_1, R_2, \ldots, R_m$ are finite sets of rules of the type rules $(t_i, (x_i, y_i), 1 \leq i \leq n)$ 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 from each region of the system, a pasting rule could be applied and should be applied. The picture pattern is moved (remains) in to another region (in the same region) by the target indication associated with the pasting rule.

A computation is successful only if the evaluation rules are applied and the computation is stops, if there is no rule can be applicable. The result of halting picture pattern is composed only by the pasting rules; the pattern is haltered in the membrane labeled by i_0. The set of all picture pattern computed by a TTPPS is denoted by $TTPPL(\Pi)$. The set of all such languages generated by the system Π is denoted by $TTPPL_m$.

Example 9. The family of language $\mathcal{L}_1$ is generated by TTPPS $\Pi_1 = (\Sigma, [_1[_2]_2[_3]_3]_1, F_1, F_2, F_3, R_1, R_2, R_3, 3)$, where $\Sigma = \{A, B, B_1\}$, $F_1 = A$, $F_2 = \phi$, $F_3 = \phi$, $R_1 = \{(A, (a_3, b_3), here),\ (B, (b_2, a_2), here),\ (D, (a_3, c_1), in_2)\}$, $R_2 = \{(C, (c_2, d_2), here),\ (D, (d_3, a_1), in_3)\}$, $R_3 = \{(A, (a_2, b_{12}), here),\ (B_1, (b_{11}, a_1), here)\}$ and 3 is the output region. The family of language $\mathcal{L}_1$ generated by the above rules are collected in the region three of the membrane 3. It is shown in the Figure 3.

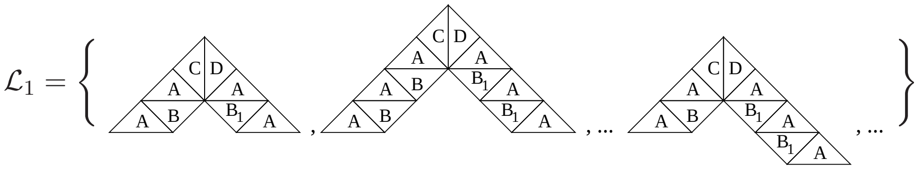


Figure 3: Language consists of arrow heads

Example 10. The triangular tile pasting P system

$$\Pi_2 = (\Sigma, [_1[_2]_2[_3]_3]_1, F_1, F_2, F_3, R_1, R_2, R_3, 1)$$

generates the family of two dimensional iso-triangular picture languages $\mathcal{L}_2$ and $\mathcal{L}_3$.
Here $\Sigma = \{A, A_1, B, B_1\}$, $F_2 = A_1$, $F_1 = \phi$, $F_3 = \phi$,
$R_1 = \{(A, (a_3, b_3), in_2), (B, (b_{11}, a_1), here)\}$,
$R_2 = \{(B, (b_2, a_{12}), in_3),\ (B, (b_1, a_{11}), in_3),\ (A_1, (a_{13}, b_{13}), in_3),$
$(A_1, (a_{13}, b_3), here)\}$,
$R_3 = \{(B, (b_2, a_2), in_2), (B_1, (b_{12}, a_{12}), out), (B, (b_1, a_1), out)\}$ and 1 is the output region.

The rules of R_2 and R_3 are applied with the target indications which belongs to the set $tar \in \{here, in, out\}$ and then the resultant iso-triangular picture pattern is collected in the output region one finally. Two members of the Picture language $\mathcal{L}_2$ is shown with derivation steps in Figure 4.

On the other hand, the computation starts from the region R_2. The first member of the language "adjoin of iso-triangles" is collected in the region 1. Again the generation is started in the same region one, the 2nd member of the language is generated and it is collected in the region one. The computation is continued in this way, the members of the language are collected in the region 1.

Theorem 11. $TTPPL_3 \cap L(TTPS) \neq \phi$

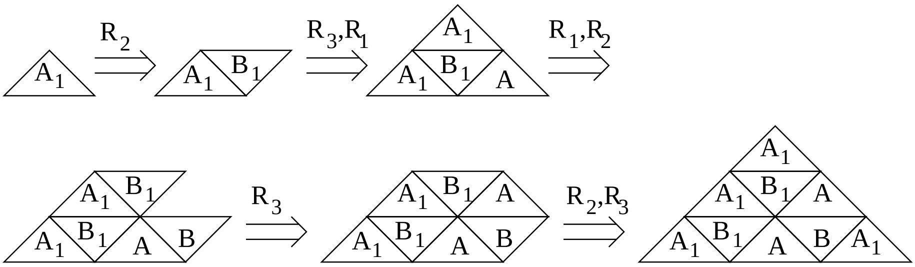


Figure 4: Two members of the language of iso-triangles

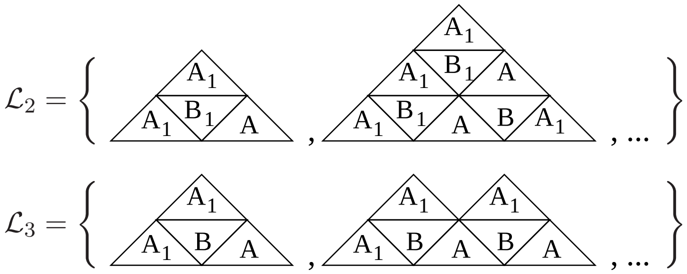


Figure 5: The family of languages of adjoin of iso-triangles and iso-triangles

Proof. The triangular tile pasting system $G = (\Sigma, P, t_0)$ generates the picture language $\mathcal{L}_1$, where $\Sigma = \{A, B, B_1\}$, $t_0 = A$ and

$$P = \{(a_3, b_3), (b_2, a_2), (a_3, c_1), (c_2, d_2), (d_3, a_1), (a_2, b_{12}), (b_{11}, a_1)\}.$$

The language generated starts with the rule (a_3, b_3) with the tile A. Then the rules given in P are applied one by one. The resultant language is $\mathcal{L}_1$. It is shown in Figure 3.

On the other hand if the computation starts with the rule (a_2, b_{12}), the language L_4 is the resultant array language which is given in the Figure 6. Thus the TTPS generate the families of languages $\mathcal{L}_1$ and L_4.

To explain the intersection is non empty it is considered that the TTPPS referred in Example 9 which generates the picture language $\mathcal{L}_1$ shown in the Figure 3 and the language L_4 cannot be generated by any TTPPS. Hence the result. □

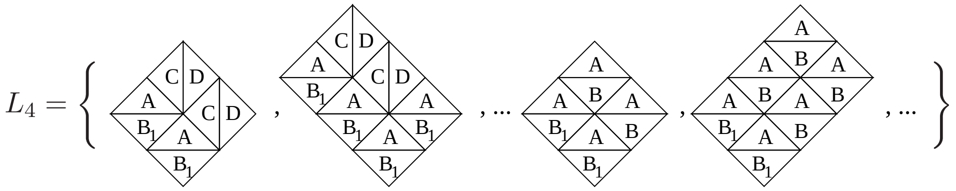


Figure 6: A language of overlaps of Rhombuses

4. Petri Nets

This chapter recalls the necessary definitions of Petri net, array generating Petri nets [7] and explains it with example. Some results of comparison are given.

Definition 12. A Petri net structure is a four tuple $C = (P, T, I, O)$ where $P = \{P_1, P_2, \ldots, P_n\}$ is a finite set of places, $n \geq 0$, $T = \{t_1, t_2, \ldots, t_m\}$ is a finite set of transitions $m \geq 0$, $P \cap T = \phi$, $I : T \to P^{\infty}$ is the input function from transitions to bags of places and $O : T \to P^{\infty}$ is the output function from transitions to bags of places.

Definition 13. A Petri Net marking is an assignment of tokens to the places of a Petri Net. The tokens are used to define the execution of a Petri Net. The number and position of tokens may change during the execution of a Petri Net. The marking can be defined as an n-vector $\mu = (\mu_1, \mu_2, \mu_3, \ldots, \mu_n)$ where μ_i is the number of tokens in the P_i, $i = 1, 2, \ldots, n$.

Definition 14. A Petri Net C with initial marking μ is called a marked Petri Net. A marked Petri Net $M = (C, \mu)$ can also be written as $M = (P, T, I, O, \mu)$.

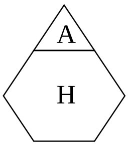
Catenation Rules

Now we introduce the catenation rule or rules that are used for joining various tiles. Tiles used in this paper are polygons. An array of tiles is formed when two or more tiles are joined. The hexagonal tile H can be catenated with the triangular tile A in three ways to form three different arrays. H can be joined with A in the upward, right down and left down direction. The same array is formed if the tile A is joined to H in the downward, left up and right up direction respectively. The notation used for these catenation rules are given below.

A in the upward direction of *H*
or
H in the downward direction of *A*

$H \textcircled{u} A$

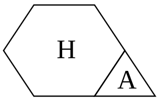
$A \textcircled{d} H$



A in the right down direction of *H*
or
H in the left up direction of *A*

$H \textcircled{rd} A$

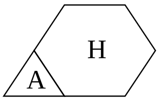
$A \textcircled{lu} H$



A in the left down direction of *H*
or
H in the right up direction of *A*

$H \textcircled{ld} A$

$A \textcircled{ru} H$

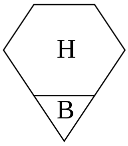


The hexagonal tile *H* and the triangular tile *B* can be joined in three different ways. *H* can be joined with *B* in the downward, right up and left up direction. The same array is formed if the tile *B* is joined in the upward, left down and right down direction respectively. The notation used for these catenation rules are given below.

B in the downward direction of *H*
or
H in the upward direction of *B*

$H \textcircled{d} B$

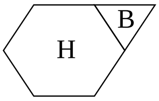
$B \textcircled{u} H$



B in the right up direction of *H*
or
H in the left down direction of *B*

$H \textcircled{ru} B$

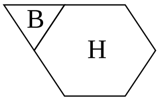
$B \textcircled{ld} H$



B in the left up direction of *H*
or
H in the right down direction of *B*

$H \textcircled{lu} B$

$B \textcircled{rd} H$

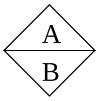


The triangular tiles A, B, C, D can be joined with each other in the following ways.

A in the upward direction of *B*
or
A in the downward direction of *A*

$B \textcircled{u} A$

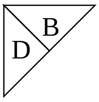
$A \textcircled{d} B$



D in the left down direction of B
or
B in the right up direction of D

$B \ \textcircled{ld} \ D$

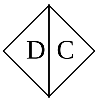
$D \ \textcircled{ru} \ B$



D in the right direction of C
or
C in the left direction of D

$C \ \textcircled{l} \ D$

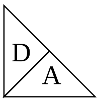
$D \ \textcircled{r} \ C$



D in the left up direction of A
or
A in the right down direction of D

$A \ \textcircled{lu} \ D$

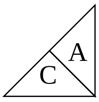
$D \ \textcircled{rd} \ A$



C in the right up direction of A
or
A in the left down direction of C

$A \ \textcircled{ld} \ C$

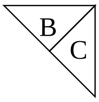
$C \ \textcircled{ru} \ A$



C in the right down direction of B
or
B in the left up direction of C

$B \ \textcircled{lu} \ C$

$C \ \textcircled{rd} \ B$



Firing rules

The transitions of the net can be associated with any of the above defined catenation rules. The transitions will be enabled to fire only when certain conditions are satisfied. The conditions that should be satisfied for a transition to be enabled and the conditions that will prevail after firing are given below.

1. All the input places of the transition should have the same array
2. If there is no catenation rule associated to the transition the same array will be moved into all the output places
3. If there is catenation rule associated to the transition then the catenation will take place in a parallel manner and the resulting array will be moved into all the output places

An example is given below to explain how the arrays are built and moved when an enabled transition fires. Figure 7 shows a transition t which is enabled as both the input places of the transition have the same array S. The transition is also associated with a catenation rule. If the array S has hexagonal tiles H in it, then all those hexagonal tiles will be joined with a B tile in the left upper direction. The array S is given in Figure 8.

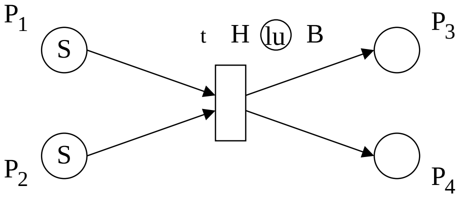


Figure 7

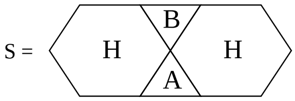


Figure 8

After firing the transition t, the array S is removed from both the input places. The tile B is joined in the left up direction to all the hexagonal tiles H of the array S. The resulting array T, given in Figure 10, is put in all the output places of the transition. The position of the arrays after firing the transition is given in Figure 9.

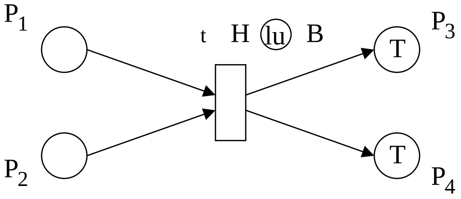


Figure 9

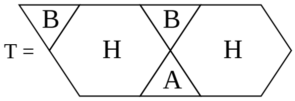


Figure 10

Now Triangular Tile array token Petri net is formally defined as follows.

Definition 15. A Triangular Tile Array Token Petri Net (TATPN) is a six tuple $N = (\Sigma, C, \mu, S, \sigma, F)$ where Σ is an alphabet of tiles, C is a Petri net structure, μ is an initial marking of arrays made up of tiles kept in some places of the net, S is a set of catenation rules, σ is a partial mapping from T to a set of catenation rules S, F is a sub-set of P, the set of places of the Petri net, which is the final set of places.

Definition 16. The Triangular array language generated by a TATPN is the collection of all arrays that reach the places of F the final set of places.

To explain the language generated: Starting with the initial marking of arrays in certain places of the net all possible firing sequences are considered. As the transitions fire the arrays grow in size and also move from one place to another. Only the arrays that reach the places belonging to F are collected as the language generated.

Example 17. Consider the TATPN which is the six tuple $N_1 = (\Sigma, C, \mu, S, \sigma, F)$ where $\Sigma = \{H, A, B\}$. $C = (P, T, I, O)$ where $P = \{P_1, P_2, \ldots, P_{10}\}$, $T = \{t_1, t_2, \ldots, t_9\}$. The initial marking μ is the hexagon H in the place P_1. The set S of catenation rules are given below
H $\textcircled{u}$ A, H $\textcircled{rd}$ A, H $\textcircled{ld}$ A, H $\textcircled{d}$ A, H $\textcircled{ru}$ B, H $\textcircled{lu}$ B, A $\textcircled{ru}$ H, A $\textcircled{lu}$ H, A $\textcircled{d}$ H

σ the mapping from the set of transitions to the set of rules is shown in the Figure 11 and $F = \{P_{10}\}$

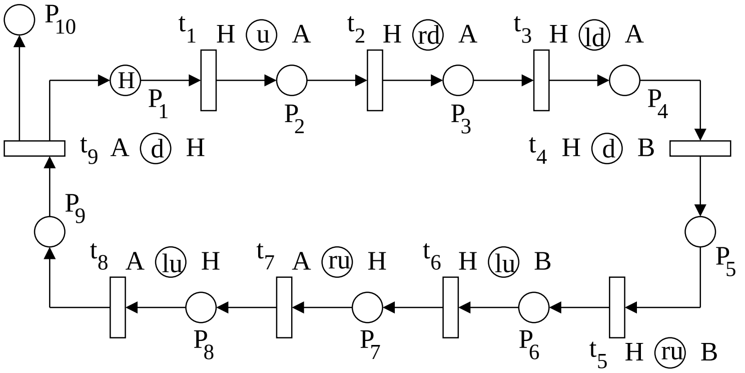


Figure 11: Tile array token Petrinet of the language

Starting with the array consisting of only the hexagon H, on firing the sequence $t_1t_2t_3$, the tile A is joined in the upward, right down and left down direction. This array is shown in Figure 12(a). On firing the sequence $t_4t_5t_6$, the tile B is joined in the downward, right up and left up direction of H. This array is shown in Figure 12(b). Lastly all the hexagons are joined when the sequence $t_7t_8t_9$ of transitions are fired. The resulting array is shown in Figure 12(c). This is the first array of the language generated by N_1.

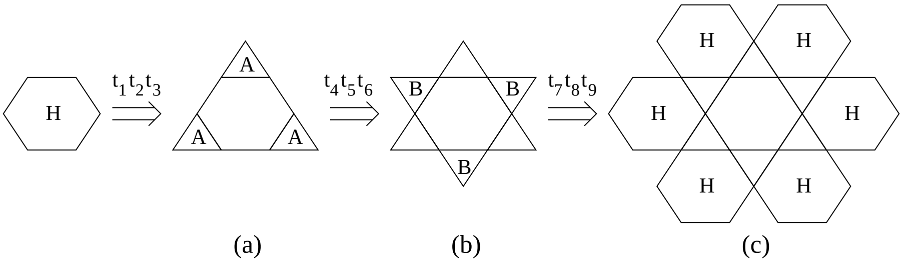


Figure 12: (a) Triangle with A tile, (b) Star with tiles A and B, (c) A member of triangular array language

Example 18. Consider the TATPN which is the six tuple $N_2 = (\Sigma, C, \mu, S, \sigma, F)$ where $\Sigma = \{A, B, C, D\}$. $C = (P, T, I, O)$ where $P = \{P_1, P_2, \ldots, P_{12}\}$, $T = \{t_1, t_2, \ldots, t_{13}\}$. The initial marking μ is the array R, given in Figure 14(a), in the place P_1. The set S of catenation rules are given below

$A \textcircled{ru} C$, $A \textcircled{lu} D$, $D \textcircled{rd} A$, $D \textcircled{ru} B$, $B \textcircled{rd} C$, $B \textcircled{ld} D$, $C \textcircled{lu} B$,

$C \textcircled{ld} A$, $B \textcircled{ld} D$, $B \textcircled{u} A$, $D \textcircled{l} C$, $C \textcircled{r} D$, $A \textcircled{d} B$

σ the mapping from the set of transitions to the set of rules is shown in the Figure 13 and $F = \{p_1\}$

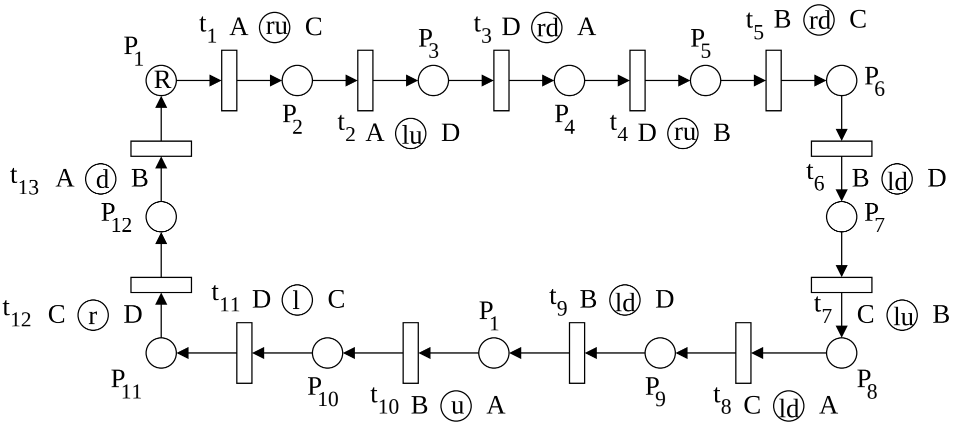


Figure 13: Triangular array Petri net of the language of Rhombuses

Starting with the array R, when every transition fires the corresponding tile is joined in parallel manner the given direction and the array is put in the output place. The result of firing the sequence of transitions $t_1 t_2 \ldots t_9$ is given in Figure 14(b). Firing the sequence $t_{10} t_{11} t_{12} t_{13}$ generates a Rhombus, which is shown in Figure 14(c). This array is the second pattern of the language generated by the net since it reaches the final place p_1.

Example 19. Consider the TATPN $N_3 = (\Sigma, C, \mu, S, \sigma, F)$ where $\Sigma = \{H, A, B, C, D\}$. $C = (P, T, I, O)$ where $P = \{P_1, P_2, \ldots, P_{11}\}$, $T = \{t_1, t_2, t_3, \ldots, t_{11}\}$. The initial marking μ is the array H, given in Figure 15, in the place P_1. The set S of catenation rules are given below

$H \textcircled{u} H$, $H \textcircled{ru} B$, $H \textcircled{lu} B$, $H \textcircled{rd} A$, $H \textcircled{ld} A$,

$B \textcircled{rd} C$, $B \textcircled{ld} D$, $B \textcircled{u} A$, $A \textcircled{d} B$, $H \textcircled{d} H$

σ the mapping from the set of transitions to the set of rules is shown in the Figure 13 and $F = \{p_{11}\}$.

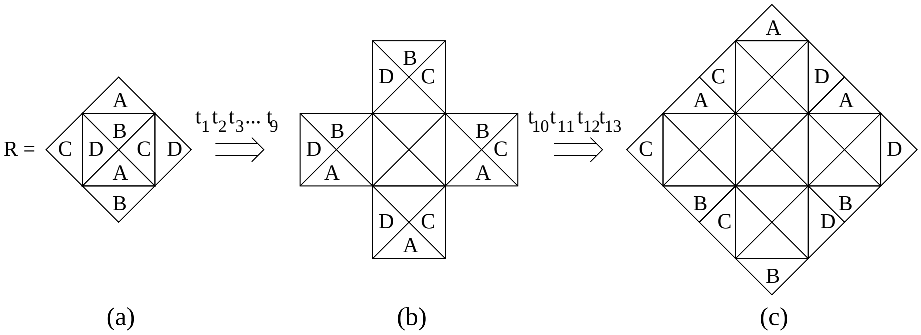


Figure 14: (a) Initial array (b) An array glued by squares (c) A rhombus

The transition t_1 on firing joins the hexagon H in the upper direction of a hexagon H. This transition may be fired any number of times. The number of hexagons joined in the final array depends on the number of times t_1 is fired. The array that is generated without firing t_1 is given in Figure 16. P_{11} is the final place of the language generated and the second member of the language is given in Figure 15.

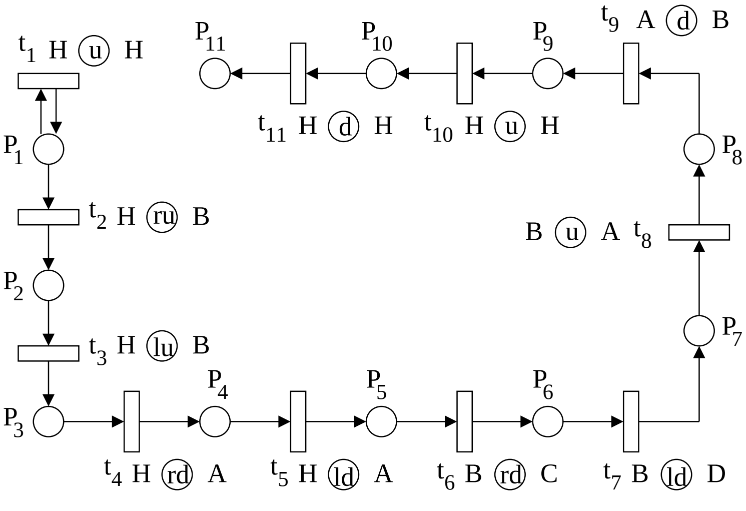


Figure 15: Triangular array Petri net of the language of Octagons

Example 20. Consider the TATPN $N_3 = (\Sigma, C, \mu, S, \sigma, F)$ where $\Sigma = \{A, B\}$. $C = (P, T, I, O)$ where $P = \{p_1, p_2, p_4\}$, $T = \{t_1, t_2, t_3\}$. The transition t_1 on firing joins the triangle B in right upper direction of all triangles A. This

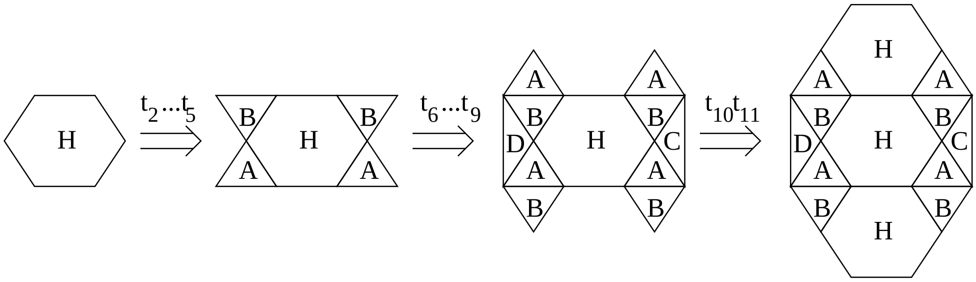


Figure 16: The Octagon generated without the transition t_1

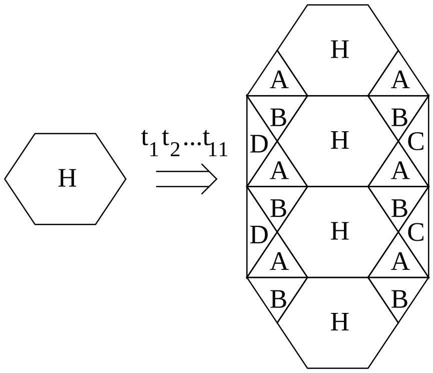
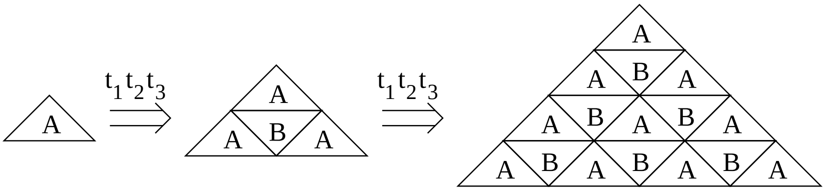


Figure 17: The Octagon generated with the transition t_1

array reaches the place p_2. The transition t_2 on firing joins the triangle A in the upward direction of all triangles B. This array reaches the place p_3. The transition t_3 on firing joins the iso-triangle A in right down direction of all iso-triangles A. This array reaches both the places p_4. Since p_4 is the final place the language generated is the set of all triangles given in Figure 18.



Theorem 21. *The family of languages generated by TATPN and ETTPS are not comparable.*

Proof. The family of languages generated by TATPN and ETTPS are by the parallel mechanism. The constraint in ETTPPS admits the language gen-

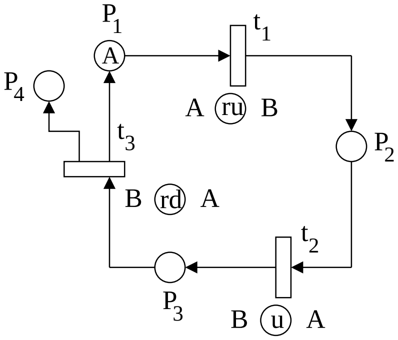


Figure 18: Triangular array Petri net of language of triangular array

erated using the extended triangular tiles. In TATPN the catenation rules can generate the family of languages. The languages generated by the extended triangular tile pasting system shown in Figure 2 cannot be generated by TATPN (refer Example 7). The presence of the constraint Δ in the extended triangular tile pasting system can do the computation of the patterns of language generated. But by the catenation rules of triangular array token petri net the language of hexagons cannot be generated. And the picture language of rhombuses explained in the Example 19 cannot be generated by any ETTPS. Because in the language generation in ETTPS, the constraint set is must. The language of octants generated by ETTPS, explained in Example 6 can also be generated by TATPN. It concludes the result. □

Theorem 22. *The family of languages generated by TATPN and TTPPS are not comparable.*

Proof. In TATPN, the catenation rules are applied in parallel manner and the same mechanism is admitted in TTPPS. But due to the presence of target indications in TTPPS, one can generate a family of languages which are the rotation of language generated by the rules and all of them cannot be generated by TATPN. The language of iso-triangles $\mathcal{L}_2$ computed in the Example 10 can also be generated by the triangular array token petri net. With the presence of target set, the language L_3 shown in Figure 5 cannot be generated by TATPN. And the language of rhombuses generated by triangular array token petri net cannot be generated by any TTPPS (refer Example 19) and which decides the result. □

5. Conclusion

In this paper k-order TTPS and ETTPS are defined with suitable example. To generate two dimensional picture languages TTPPS is considered and the comparision study between TTPPS and TTPS is made. Also the comparison study between the Extended triangular tile pasting system and Triangular array token petri net is made. In this paper it is decided that the Triangular tile pasting P system and Triangular array token petri net are not comparable.

References

[1] K. Bhuvaneswari, T. Kalyani, Triangular tile pasting P system for pattern generation, Conference Proceeding of ICAMTCS-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 rules, *International Journal of Application for Mathematics*, **3**, No. 1 (2014), 1-16.

[3] K. Bhuvaneswari, T. Kalyani, D. Gnanaraj Thomas, Contextual iso-triangular array P system and iso-triangular picture languages, *Asian Conference on Membrane Computing*, IEEE (2014), 1-5.

[4] R. Ceterchi, M. Mutyam, Gh. Păun, K.G. Subramanian, Array-rewriting P systems, *Natural Computing*, **2** (2003), 229-249.

[5] T. Kalyani, K. Sasikala, V.R. Dare, T. Robinson, Triangular pasting system, in: K.G. Subramanian, K. Rangarajan, M. Mukund (Eds.), *Formal Models, Languages and Applications, Series in Machine Perception and Artificial Intelligence*, **66** (2006), 195-211.

[6] D. Lalitha, Rectangular array languages generated by a petri net, computational vision and robotics, *Advances in Intelligent Systems and Computing*, **332** (2015), 17-27.

[7] D. Lalitha, K. Rangarajan, Characterization of pasting system using array token petri nets, *International Journal of Pure and Applied Mathematics*, **70**, No. 3 (2011), 275-284.

[8] D. Lalitha, K. Rangarajan, D.G. Thomas, Petri net generating hexagonal arrays, combinatorial image analysis, *LNCS*, **6636** (2011), 235-247.

[9] M. Nivat, A. Saoudi, K.G. Subramanian, R. Siromoney, V.R. Dare, Puzzle grammars and context-free array grammars, *Int. Journal of Pattern Recognition and Artificial Intelligence*, **5** (1991), 663-676.

[10] G.H. Păun, Membrane Computing: An Introduction, Springer-Verlag, Berlin, Heidelbrg (2002).

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

[12] K.G. Subramanian, T. Robinson, Atulya K. Nagar, Tile pasting P system model for pattern generation, in *Third Asian International Conference on Modelling & Simulation* (2009), 359-362.