Mathematics

https://www.aimspress.com/journal/Math

AIMS Mathematics, 9(7): 17305–17318.
DOI: 10.3934/math.2024841
Received: 20 March 2024
Revised: 02 May 2024
Accepted: 10 May 2024
Published: 20 May 2024

Research article

Two-dimensional array grammars in palindromic languages

Hannah Blasiyus and D. K. Sheena Christy*

Department of Mathematics, SRM Institute of Science and Technology, Faculty of Engineering and Technology, Kattankulathur 603203, Chengalpattu District, Tamil Nadu, India

* **Correspondence:** Email: sheena.lesley@gmail.com.

Abstract: In this paper, we put forward models that generate two-dimensional palindromic languages with array-rewriting rules. The rewriting rules are of either regular or context-free type with terminals being arrays. The derivation lengths are managed by the array concatenation conditions. These grammars give rise to an extensive variety of palindromic pictures. Different hierarchies that exist between the classes defined are demonstrated. The closure properties have also been evaluated. Applications of these models have been explored by generating a few patterns of kolams.

Keywords: array grammars; palindromic arrays; picture languages; kolams; formal languages
Mathematics Subject Classification: 03D05, 68Q42, 68Q45

1. Introduction

The area of mathematics which concentrates on the combinatorial features used in computation theory is generally known as "combinatorics on words". The work of Axel Thue [1, 2] gave rise to the subject of study. Lothaire's *combinatorics on words* [3] touched on the algebraic aspects of combinatorics extensively. Combinatorics on words, on the other hand, is intimately related to the study of formal languages because both fields are concerned with words.

Recent research has focused on many combinatorial properties of words, such as tandem repeats, having a square-free property, primitive words, complexity, morphisms on words, borderness, periodicity, and so on. For example, Kari and Mahalingam investigated involutively bordered words in [4], Czeizler et al. concentrated on the primitivity property based on the action of molecules in DNA in [5], Yu studied the borderness property of words in [6], and so on. The study has now been expanded to include two-dimensional words and their combinatorial features.

The field of computer science that deals with the generation and analysis of pictures has been referred to as "picture processing". Rosenfeld [7] had examined the necessity for array-rewriting rules for picture languages in order to generate isometric arrays. Narasimhan [8] did groundbreaking work by proposing and implementing a language paradigm for the resolution of nontrivial picture processing problems. A linguistic model was presented by Siromoney et al. for the formation of matrices, or rectangular arrays of terminals, through the replacement of regular language families with well-known formal language families [9]. Later, by altering the type of rewriting rules, new language families were introduced [10]. The idea of rewriting rules in string grammars has been extended by authors to array-rewriting rules for matrix grammars; thus, at present, the rules are either regular (*Reg*), context-free (*CoFree*), or context-sensitive (*CoSen*). However, the row and column concatenation condition limits how the production rules can be used. Applications include the production of two-dimensional crystallography [11,12]. Two-dimensional *CoFree* grammars were introduced by Tomita in 1989 [13].

Amir and Benson [14] investigated periodicity in two-dimensional arrays. Various scholars explored the occurrence of palindromes in two-dimensional arrays [15–18]. Various formal models for creating or recognizing two-dimensional images have been published in the literature. These studies were driven by image processing and pattern identification problems [19,20], but two-dimensional patterns have also emerged in studies related to numerous parallel computing models, including combinatorics.

Many methods have been developed to study these features. For examples, Knuth et al. [21] studied pattern matching in words, Cole et al. [22] provided an algorithm for pattern matching in words and arrays, and Geizhals and Sokol [23] presented an approach for finding the maximal two-dimensional palindromes.

In this study, we aimed to generate palindromic picture languages by using array grammars and prove some of their properties.

This paper is organized as follows. Section 2 recalls several definitions from the literature that were required for our study. In Section 3, we define two-dimensional palindromic array languages (PALs) and discuss a few examples. In Section 4, we study some of the results from the perspective of the PALs and their closure properties. Finally, in Section 5, we discuss their application to a kolam pattern.

2. Preliminaries

Here, we recall a few definitions that were required for our study. For basic definitions, we refer the reader to [10,15,24].

2.1. One-dimensional words

Definition 2.1. ([24]) A finite and non-empty collection of symbols called letters is known as an *alphabet*, denoted by $\mathcal{A}$. $|\mathcal{A}|$ represents the total number of elements present in $\mathcal{A}$.

Definition 2.2. ([24]) A finite or infinite sequence of letters from the alphabet is known as a *string* or *word*.

If $\mathcal{A}$ is an alphabet, then $\mathcal{A}^*$ represents the collection of strings created by joining together zero or more letters from $\mathcal{A}$. A subset of $\mathcal{A}^*$ is usually known as *one-dimensional language*.

Definition 2.3. ([24]) If s and t are two strings from $\mathcal{A}^*$, then *word concatenation* of s and t is realized by appending letters of t to the right end of s. We denote this by $s \bullet t$, or simply, st; if

$$s = s_1 s_2 \cdots s_m$$

and

$$t = t_1 t_2 \cdots t_n,$$

where

$$s_1, s_2, \cdots s_m, t_1, t_2, \cdots t_n \in \mathcal{A},$$

then

$$st = s \bullet t = s_1 s_2 \cdots s_m t_1 t_2 \cdots t_n.$$

Definition 2.4. ([24]) *Length* of a string s is the total number of characters that it contains, denoted by $|s|$. If $|s| = 0$, we say that w is an *empty word*, usually denoted by λ.

Note that

$$\mathcal{A}^* \setminus \{\lambda\} = \mathcal{A}^+,$$

where $\mathcal{A}^+$ is the collection of all non-empty strings over $\mathcal{A}$.

Definition 2.5. ([24]) If s is a string over $\mathcal{A}$ which is given by

$$s = s_1 s_2 \cdots s_n,$$

then the *reversal* of the string s is defined as a rewriting of s from right to left, denoted by s^R:

$$s^R = s_n \cdots s_2 s_1,$$

where $s_i \in \mathcal{A}, i = 1, 2, \cdots, n$.

Definition 2.6. ([24]) String s is said to be *palindromic*, or we say s is a *palindrome* if $s = s^R$, i.e., if

$$s = s_1 s_2 \cdots s_n$$

and

$$s_i = s_{n-(i-1)}, \quad i = 1, 2, \cdots, n,$$

where $s_i \in \mathcal{A}$.

Definition 2.7. ([24]) If $s \in \mathcal{A}^*$, then s^k is obtained by appending s to the right end of s itself $k - 1$ times, i.e.,

$$s \bullet s \bullet s \bullet \cdots s \ \ (k - \text{times}),$$

for all $k \geq 1$. If $k = 0$, then

$$s^k = s^0 = \lambda,$$

which is an empty string.

2.2. *Two-dimensional words*

Definition 2.8. ([15]) A rectangular array of letters over a finite alphabet $\mathcal{A}$ is known as a *picture* or *matrix* over $\mathcal{A}$.

A collection of pictures is denoted by $\mathcal{A}^{**}$. A subset of $\mathcal{A}^{**}$ is usually known as *picture language* (or *two-dimensional language*) over $\mathcal{A}$.

Definition 2.9. ([10]) If

$$M_A = \begin{array}{ccc} u_{11} & \cdots & u_{1n} \\ \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots \\ u_{m1} & \cdots & u_{mn} \end{array}, \quad M_B = \begin{array}{ccc} v_{11} & \cdots & v_{1n'} \\ \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots \\ v_{m'1} & \cdots & v_{m'n'} \end{array},$$

then the vertical concatenation of M_A and M_B is defined as follows:

$$M_A \oplus M_B = \begin{array}{cccccc} u_{11} & \cdots & u_{1n} & v_{11} & \cdots & v_{1n'} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ u_{m1} & \cdots & u_{mn} & v_{m'1} & \cdots & v_{m'n'} \end{array}, \text{ provided } m = m',$$

and the horizontal concatenation of M_A and M_B is defined as follows:

$$M_A \ominus M_B = \begin{array}{ccc} u_{11} & \cdots & u_{1n} \\ \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots \\ u_{m1} & \cdots & u_{mn} \\ v_{11} & \cdots & v_{1n'} \\ \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots \\ v_{m'1} & \cdots & v_{m'n'} \end{array}, \text{ provided } n = n'.$$

We use $\oplus$ to denote that the concatenation is either $\oplus$ or $\ominus$.

Definition 2.10. ([10]) If

$$M_A = \begin{array}{ccc} u_{11} & \cdots & u_{1n} \\ \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots \\ u_{m1} & \cdots & u_{mn} \end{array},$$

then the *half-turn* of M_A (also called the reversal of M_A), denoted by M_A^R, is given by

$$M_A^R = \begin{array}{ccc} u_{mn} & \cdots & u_{m1} \\ \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots \\ u_{1n} & \cdots & u_{11} \end{array}.$$

Definition 2.11. ([15]) An array M_A is said to be *palindromic*, or, we say that M_A is a palindrome if $M_A = M_A^R$.

Definition 2.12. ([15]) The *order* or *size* of an array with r rows and s columns is denoted by $r \times s$, provided that $r, s \geq 1$. The arrays of order $r \times 0$ or $0 \times s$ are undefined. An array of order 0×0 is known as an *empty array*, denoted by Λ.

Note that

$$\mathcal{A}^{**} \setminus \{\Lambda\} = \mathcal{A}^{++},$$

where $\mathcal{A}^{++}$ represents the collection of all non-empty arrays over $\mathcal{A}$.

Definition 2.13. ([10]) The array-rewriting grammar G is defined as follows:

$$G = (Vr, I, Pr, S),$$

which denotes the array grammar, where

$$Vr = Vr_1 \cup Vr_2,$$

where Vr_1 is a finite collection of nonterminals, Vr_2 is a finite collection of intermediates, I is a finite collection of terminals;

$$Pr = Pr_1 \cup Pr_2 \cup Pr_3,$$

where Pr_1 is a finite collection of ordered pairs (u, v) (written $u \to v$), and u and v are contained within $(Vr_1 \cup Vr_2)_+$ or $(Vr_1 \cup Vr_2)^+$, classified as follows:

- Pr_1 is $CoSen$ if there is a (u, v) in Pr_1 such that $v = u_1\beta v_1$ and $u = u_1S_1v_1$, where $S_1 \in Vr_1$ and u_1, v_1, β are all contained within $(Vr_1 \cup Vr_2)_+$ or $(Vr_1 \cup Vr_2)^+$.
- Pr_1 is $CoFree$, if in Pr_1, for all (u, v) in Pr_1, we find that $v \in (Vr_1 \cup Vr_2)_+$ or $v \in (Vr_1 \cup Vr_2)^+$ and $u \in Vr_1$
- Pr_1 is Reg if v is of the form $U \oplus V$, given that $U \in Vr_1$ and $V \in Vr_2$ or $U \in Vr_2$ and $V \in Vr_1$ with $u \in Vr_1$.

Pr_2 is the finite collection of intermediate rules which are ordered pairs denoted by (u, v), where u and v in

$$\left(Vr_2 \cup \left\{x_1, \cdots, x_p\right\}\right)_+$$

or

$$\left(Vr_2 \cup \left\{x_1, \cdots, x_p\right\}\right)^+ : x_1, \ldots, x_p \in I^{++}$$

have the same number of columns and rows in the first and second cases, respectively, classified as Reg, $CoFree$, or $CoSen$ according to the intermediate matrix languages generated.

Pr_3, the finite collection of terminal rules, is represented by ordered pairs denoted by (u, v), where $u \in (Vr_1 \cup Vr_2)$ and $v \in I^{++}$; $S \in Vr_1$ is the start symbol.

Definition 2.14. ([10]) An array grammar is referred to as follows:

- $(Reg$: $Reg)AG$ if all intermediate languages are Reg and the nonterminal rules are Reg.
- $(Reg$: $CoFree)AG$ if none of the intermediate languages is Reg and the nonterminal rules are Reg.
- $(Reg$: $CoSen)AG$ if there exists an intermediate language which is not $CoFree$ and the nonterminal rules are Reg.
- The remaining six types, i.e., $(CoSen$: $CoSen)AG$, $(CoSen$: $CoFree)AG$, $(CoSen$: $Reg)AG$, $(CoFree$: $CoSen)AG$, $(CoFree$: $CoFree)AG$ and $(CoFree$: $Reg)AG$, can be defined similarly.

3. Two-dimensional PALs

In this section, we define two-dimensional palindromic array grammars (PAGs) and the PALs that are generated by them.

Definition 3.1. The PAG G is defined as follows:

$$G = (Vr, I, Pr, S),$$

where

$$V = Vr_1 \cup Vr_2,$$

where Vr_1 is a finite collection of nonterminals, Vr_2 is a finite collection of intermediates and I is a finite collection of terminals;

$$Pr = Pr_1 \cup Pr_2 \cup Pr_3,$$

where Pr_1 is a finite collection of ordered pairs denoted by (μ, ν) (written $\mu \rightarrow \nu$), given that μ and ν are contained within $(Vr_1 \cup Vr_2)_+$ or $(Vr_1 \cup Vr_2)^+$, classified as follows:

- Pr_1 is $CoFree$, if for all (μ, ν) in Pr_1, we find that $\mu \in Vr_1$ and ν is expressed as $\mu_1 \oplus \mu_2 \oplus \cdots \oplus \mu_k$, where $\mu_i \in (Vr_1 \cup Vr_2)_+$ or $\mu_i \in (Vr_1 \cup Vr_2)^+$ for all $i = 1, 2, \cdots, k$.
- Pr_1 is Reg, if for all (μ,ν) in Pr_1, we find that $\mu \in Vr_1$ and ν being expressed as $U \oplus V$, where $U \in Vr_1$ and $V \in Vr_2$ or $U \in Vr_2$ and $V \in Vr_1$.

Pr_2 is the finite collection of intermediate rules which are ordered pairs denoted by (μ, ν), where μ and ν in

$$\left(Vr_2 \cup \left\{x_1, \cdots, x_p\right\}\right)_+$$

or

$$\left(Vr_2 \cup \left\{x_1, \cdots, x_p\right\}\right)^+ : x_1, \ldots, x_p \in I^{++}$$

have the same number of columns and rows in the first and second cases, respectively, classified as *Reg* or $CoFree$ according to the intermediate matrix languages generated.

Pr_3, the finite collection of terminal rules, are ordered pairs denoted by (μ, ν), where $\mu \in (Vr_1 \cup Vr_2)$ and $\nu \in I^{++}$; $S \in Vr_1$ is the start symbol.

Definition 3.2. A PAG is referred to as follows:

- (*Reg*: *Reg*)*PAG* if all intermediate languages are *Reg* and the nonterminal rules are *Reg*.
- (*Reg*: $CoFree$)*PAG* if none of the intermediate languages is *Reg* and the nonterminal rules are *Reg*.
- The remaining two PAGs, i.e., ($CoFree$: $CoFree$)*PAG*, ($CoFree$: *Reg*)*PAG*, can be defined similarly.

Example 3.1. The PAL

$$L = \{(r)_+^+\}$$

is generated by the PAG

$$G = (Vr, I, Pr, S),$$

where

$$Vr = Vr_1 \cup Vr_2, \quad Vr_1 = \{S\}, Vr_2 = \{A_1, A_2\}, \quad I = \{r\},$$

$$Pr_1 = \{S \rightarrow S \ominus A_1, S \rightarrow S \oplus A_2\}, \quad Pr_3 = \{S \rightarrow r\}, \quad L_{A_1} = \{(r)^n : n \geq 1\}, \quad L_{A_2} = \{(r)_m : m \geq 1\}.$$

This grammar is of type (*Reg*: *Reg*)*PAG*.

For instance, one of the arrays generated by this grammar is as follows:

$$S \implies S \ominus A_1 \implies (S \oplus A_2) \ominus A_1 \implies ((S \oplus A_2) \oplus A_2) \ominus A_1 \implies \begin{matrix} r & r & r \\ r & r & r \end{matrix}$$

Example 3.2. The PAL given by

$$L = \left\{ \begin{pmatrix} r & s \\ s & r \end{pmatrix}_{+}^{+} \right\}$$

is generated by the PAG

$$G = (Vr, I, Pr, S),$$

where

$$\begin{aligned} & Vr = Vr_1 \cup Vr_2, \;\; Vr_1 = \{S\}, \;\; Vr_2 = \{A, B\}, \;\; I = \{r, s\}, \\ & Pr_1 = \{S \to S \ominus A, S \to S \oplus B\}, \\ & Pr_3 = \left\{ S \to \begin{matrix} r & s \\ s & r \end{matrix} \right\}, \;\; L_A = \left\{ \begin{pmatrix} r & s \\ s & r \end{pmatrix}^{n} : n \geq 1 \right\}, \\ & L_B = \left\{ \begin{pmatrix} r & s \\ s & r \end{pmatrix}_{m} : m \geq 1 \right\}. \end{aligned}$$

This grammar is of type (*Reg*: *Reg*)*PAG*.

For instance, one of the arrays generated by the above grammar is as follows:

$$S \implies S \ominus A \implies (S \oplus B) \ominus A \implies \begin{matrix} r & s & r & s \\ s & r & s & r \\ r & s & r & s \\ s & r & s & r \end{matrix}$$

Example 3.3. The PAL given by

$$L = \{\text{Collection of all hv-palindromes which are bordered by x's}\}$$

is generated by the PAG

$$G = (Vr, I, Pr, S),$$

where

$$Vr = Vr_1 \cup Vr_2, \;\; Vr_1 = \{S, S_1, S_2, S_3, S_4, C, D\}, \;\; Vr_2 = \{X, Y\}, \;\; I = \{x, y\}.$$

Pr_1 has the following production rules:

$$\begin{array}{cc} S \to X \oplus S_1 \oplus X, & S_1 \to X \oplus S_1 \oplus X, \\ S_1 \to S_3 \oplus S_1 \oplus S_3, & S_1 \to X \ominus D \ominus X, \\ S_1 \to (X \ominus D \ominus X) \oplus S_4, & S_4 \to X \ominus D \ominus X \\ S_3 \to C \ominus D \ominus C & S_3 \to D \ominus C \ominus D \\ C \to ((Y) \ominus (X)^m \ominus (Y)) : m = 0, 1, 2, \cdots, & D \to ((X) \ominus (Y)^m \ominus (X)) : m = 0, 1, 2, \cdots. \end{array}$$

Pr_3 has the following rules:

$$S \to (x \oplus x)^n, \ \ n \geq 1; \ \ S_1 \to (x \oplus x)^n, \ \ n \geq 1; \ \ S_1 \to (x)^n, \ \ n \geq 1.$$

It also has the following intermediate languages:

$$L_X = \{(x)^n : n \geq 1\}, \quad L_Y = \{(y)_m : m \geq 1\}.$$

This grammar is of type $(CF\colon CF)PAG$.

For instance, one of the arrays generated by the above grammar is as follows:

$$S \implies X \oplus S_1 \oplus X \implies X \oplus (X \ominus D \ominus X) \oplus X \implies \begin{matrix} x & x & x \\ x & Y & x \\ x & x & x \end{matrix} \implies \begin{matrix} x & x & x \\ x & y & x \\ x & x & x \end{matrix}.$$

Example 3.4. The PAL given by

$$L = \{\text{Collection of all hv- square palindromes of odd order}\}$$

is generated by the PAG

$$G = (Vr, I, Pr, S),$$

where

$$Vr = Vr_1 \cup Vr_2, \ \ Vr_1 = \{S, S_1, S_2, \cdots, S_n\}, \ \ I = \{x, y\},$$

$$Vr_2 = \{S_i^*, T_i^*, U_i^*, V_i^*, W_i^* : i = 1, 2, \cdots, n+1\}.$$

Pr_1 has the following production rules:

$$\begin{array}{rlllll} S \to & S_1^* \oplus S_1 \oplus S_1^* & /T_1^* \oplus S_1 \oplus T_1^* & /U_1^* \oplus S_1 \oplus U_1^* & /V_1^* \oplus S_1 \oplus V_1^* & /W_1^* \oplus S_1 \oplus W_1^* \\ S_1 \to & S_2^* \oplus S_2 \oplus S_2^* & /T_2^* \oplus S_2 \oplus T_2^* & /U_2^* \oplus S_2 \oplus U_2^* & /V_2^* \oplus S_2 \oplus V_2^* & /W_2^* \oplus S_2 \oplus W_2^* \\ \vdots & & & & & \\ S_{n-1} \to & S_n^* \oplus S_n \oplus S_n^* & /T_n^* \oplus S_n \oplus T_n^* & /U_n^* \oplus S_n \oplus U_n^* & /V_n^* \oplus S_n \oplus V_n^* & /W_n^* \oplus S_n \oplus W_n^* \end{array}$$

$$S_n \to S_{n+1}^* / T_{n+1}^* / U_{n+1}^* / V_{n+1}^* / W_{n+1}^*.$$

Pr_3 has the following rules:

$$S \to a, \quad S \to b.$$

It also has the following intermediate languages:

$$L_{S_i^*} = \{(a)_{2n+1} : n \geq 1\}, \quad L_{T_i^*} = \{(b)_{2n+1} : n \geq 1\},$$

$$L_{U_i^*} = \left\{ \begin{array}{l} (a)_k \\ (b)_j \\ (a)_k \end{array} : k, j \geq 0, 2k + j = 2n + 1, n \geq 1 \right\},$$

$$L_{V_i^*} = \left\{ \begin{array}{l} (b)_m \\ (a)_p \\ (b)_m \end{array} : m, p \geq 0, 2m + p = 2n + 1, n \geq 1 \right\},$$

$$L_{W_i^*} = \left\{ \begin{array}{l} (a)_q \\ (b)_r \\ (a)_s \\ (b)_t \\ \vdots \\ (b)_t \\ (a)_s \\ (b)_r \\ (a)_q \end{array} : q, r, s, t, \cdots \geq 0, 2q + 2r + 2s + 2t + \cdots = 2n + 1, n \geq 1 \right\}.$$

This grammar is of type $(CF\!: CF)PAG$.

4. Results on PAGs

In this section, we study some of the results on PAGs.

Theorem 4.1. Let

$$\{M_n : n \geq 1\}$$

be a sequence of palindromic matrices such that M_n has any one of the following forms:

$$M_n = (A \ominus M_{n-1}) \oplus B \;\text{ or }\; B \oplus (M_{n-1} \ominus A) \;\text{ or }\; (M_{n-1} \ominus A) \oplus B \;\text{ or }\; B \oplus (A \ominus M_{n-1}),$$

where A and B are taken from intermediate matrix language L_A and L_B for $n \geq 1$. Then, the sequence $\{M_n\}$ is generated by regular nonterminal rules. Further, if the recursive way of defining M_n is unique, then $\{M_n\}$ is a $(Reg\!: Reg)PAL$ or $(Reg\!: CoFree)PAL$.

Proof. If we use the nonterminal rules that are given in the form of $S \to (X \ominus S) \oplus Y$, terminal rules that are given in the form of $S \to M_1$ and the intermediate rules to generate L_X and L_Y will yield the following language: $(Reg\!: Reg)PAL$. We can similarly prove the other cases. □

Theorem 4.2. Let

$$\{M_n : n \geq 1\}$$

be a sequence of palindromic matrices such that M_n has any one of the following forms:

$$M_n = X_1 \oplus (Y_1 \ominus M_{n-1} \ominus Y_2) \oplus X_2 \;\text{ or }\; M_n = X_1 \ominus (Y_1 \oplus M_{n-1} \oplus Y_2) \ominus X_2,$$

where X_1, X_2, Y_1, and Y_2 are taken from intermediate matrix languages $L_{X_1}, L_{X_2}, L_{Y_1}$, and L_{Y_2} for $n \geq 1$ such that at least Y_1 and Y_2 or X_1 and X_2 are non-empty. Then, the sequence $\{M_n\}$ is generated by $CoFree$ nonterminal rules. Further, if the recursive way of defining M_n is unique, then $\{M_n\}$ is a ($CoFree$: *Reg*)*PAL* or ($CoFree$: $CoFree$)*PAL*.

Proof. We find that, in the grammars which generate the above-mentioned type of palindromic arrays, there exists at least one self-embedding nonterminal rule. Since the self-embedding nonterminals cannot generate *Reg* PALs, we say that the nonterminal rules are $CoFree$. Thus, the array language will be either ($CoFree$: *Reg*)*PAL* or ($CoFree$: $CoFree$)*PAL*. □

Next, we shall discuss the closure properties satisfied by PAGs.

Theorem 4.3. All families of palindromic matrix languages are closed under union, homomorphic, and reflect about the base.

Proof.
- If L_1 and L_2 are two palindromic matrix languages generated by the following (*Reg*: *Reg*)*PAG*:
$$H_1 = (Vr_1, I, Pr_1, S_1) \text{ and } H_2 = (Vr_2, I, Pr_2, S_2),$$
then
$$L = L_1 \cup L_2$$
generated by the grammar
$$H = (Vr, I, Pr, S),$$
where
$$V = Vr_1 \cup Vr_2, \quad P = Pr_1 \cup Pr_2, \quad S = S_1 \cup S_2$$
will also be (*Reg*: *Reg*)*PAL* because of the restrictions of the production rule, it is similar for the other families of PALs.
- Let H be the PAG that generates the PAL of type (*Reg*: *Reg*)*PAL*. Let H' be a PAG obtained by retaining the nonterminal rules of H, putting $h(a)$ in the place of a in the terminal rules and in the intermediate language of H, where $a \in I$ and h is a homomorphism. Then, we observe that H' itself is a (*Reg*: *Reg*)*PAG*. We can similarly prove this for the other families of PALs.
- Let H be the PAG (*Reg*: *Reg*)*PAG* and $M = M(H)$. Let H' be the grammar which generates the languages of arrays that are obtained by reflecting the language generated by H about the base. Then, H' can be derived as follows: all nonterminal rules in H that involve $\oplus$ only are to be retained in H'. Corresponding to the nonterminal rule $S_1 \rightarrow D \ominus E$ in H, we have that $S_1 \rightarrow E \ominus D$ in H'. If $S_1 \rightarrow X$ is a terminal rule in H, then $S_1 \rightarrow X^E$ is a terminal rule in H', where X^E is the reflection of X about its base. If L_D is an intermediate language generated by D in H, then L_D^E is the intermediate language generated by D in H'. Then, we find H' to be a (*Reg*: *Reg*)*PAG*. We can similarly prove this for the other families of PALs.

□

5. Palindromic kolams-an overview

Every morning, South Indian villagers draw kolams on the courtyard in front of each house. Young women use specks of rice flour to artistically decorate the floor with kolam. Celebrations require more intricate kolam patterns. A small design can be enlarged immediately to cover a wide floor space, but this is rarely done. Different designs have names, and larger variants of the same kolam use more sophisticated designs. For each kolam pattern, a grammar can build a two-dimensional language.

In this section, we will discuss a palindromic kolam pattern existing in the literature and generate it by using PAGs.

Vilvathalam kolam (see Figure 1):

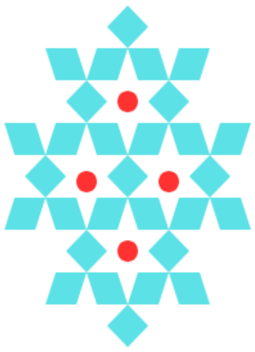

Figure 1. Vilvathalam kolam.

The kolam in Figure 1 can be generated by the grammar introduced in [25], which is a $(CoFree\colon Reg)AG$ given by

$$M = \begin{pmatrix} P \\ Q \\ {}^{\sim}P \end{pmatrix} R \begin{pmatrix} \tilde{P} \\ \tilde{Q} \\ {}^{\sim}\tilde{P} \end{pmatrix},$$

where

$$L_Q = \{\mathcal{T}\,\mathcal{T}\,(\textcolor{lightgray}{\blacktriangledown}\,\mathcal{T}\,\textcolor{lightgray}{\blacktriangledown}\,\mathcal{T}\ C)^n\} \cup \{\mathcal{T}\,\mathcal{T}\,(\textcolor{lightgray}{\blacktriangledown}\,\mathcal{T}\,\textcolor{lightgray}{\blacktriangledown}\,\mathcal{T}\,\textcolor{cyan}{\blacksquare}\,\mathcal{T})^n\,\textcolor{lightgray}{\blacktriangledown}\,\mathcal{T}\,\textcolor{lightgray}{\blacktriangledown}\,\mathcal{T} : n \geq 1\},$$

$$L_R = \left\{ \left(\begin{pmatrix} \mathcal{T} & \textcolor{cyan}{\blacktriangledown} & \mathcal{T} \\ \textcolor{lightgray}{\blacktriangledown} & \mathcal{T} & \textcolor{lightgray}{\blacktriangledown} \\ \mathcal{T} & \textcolor{cyan}{\blacktriangledown} & \mathcal{T} \\ \textcolor{lightgray}{\blacktriangledown} & \mathcal{T} & \textcolor{lightgray}{\blacktriangledown} \\ \mathcal{T} & \textcolor{cyan}{\blacksquare} & \mathcal{T} \\ \textcolor{lightgray}{\blacktriangledown} & \mathcal{T} & \textcolor{lightgray}{\blacktriangledown} \end{pmatrix}_n \ominus \begin{pmatrix} \mathcal{T} & \textcolor{cyan}{\blacktriangledown} & \mathcal{T} \\ \textcolor{lightgray}{\blacktriangledown} & \mathcal{T} & \textcolor{lightgray}{\blacktriangledown} \\ \mathcal{T} & \textcolor{cyan}{\blacktriangledown} & \mathcal{T} \end{pmatrix} : n \geq 1 \right) \right\},$$

and P is generated by the $(Reg\colon Reg)AG$ labeled as G, which is defined as follows:

$$G = (Vr, I, Pr, S),$$

where

$$V = Vr_1 \cup Vr_2,\quad Vr_1 = \{S\},\quad Vr_2 = \{A, B\},\quad I = \{\mathcal{T}, \textcolor{lightgray}{\blacktriangledown}, \textcolor{cyan}{\blacktriangledown}, \textcolor{cyan}{\blacksquare}\},\quad Pr_1 = \{S \to (B \oplus S) \ominus A\},$$

$$L_A = \left\{ \left(\begin{pmatrix} \mathcal{T} & \mathcal{T} & \mathcal{T} & \mathcal{T} \\ \textcolor{cyan}{\blacktriangledown} & \mathcal{T} & \textcolor{lightgray}{\blacktriangledown} & \mathcal{T} \\ \mathcal{T} & \textcolor{lightgray}{\blacktriangledown} & \mathcal{T} & \textcolor{cyan}{\blacktriangledown} \end{pmatrix} \begin{pmatrix} \mathcal{T} & \textcolor{lightgray}{\blacktriangledown} & \mathcal{T} & \textcolor{lightgray}{\blacktriangledown} & \mathcal{T} & \textcolor{cyan}{\blacksquare} \\ \textcolor{lightgray}{\blacktriangledown} & \mathcal{T} & \blacktriangle & \mathcal{T} & \textcolor{lightgray}{\blacktriangledown} & \mathcal{T} \\ \mathcal{T} & \textcolor{lightgray}{\blacktriangledown} & \mathcal{T} & \textcolor{lightgray}{\blacktriangledown} & \mathcal{T} & \textcolor{cyan}{\blacktriangledown} \end{pmatrix}^n \begin{matrix} \mathcal{T} \\ \textcolor{lightgray}{\blacktriangledown} \\ \mathcal{T} \end{matrix} : n \geq 0 \right) \right\}$$

$$\cup\left\{\left(\begin{matrix}\mathcal{T} & \mathcal{T} & \mathcal{T} & \mathcal{T}\\ \blacktriangledown & \mathcal{T} & \blacktriangledown & \mathcal{T}\\ \mathcal{T} & \blacktriangledown & \mathcal{T} & \blacktriangledown\end{matrix}\left(\begin{matrix}\mathcal{T} & \blacktriangledown & \mathcal{T} & \blacktriangledown & \mathcal{T} & \blacksquare\\ \blacktriangledown & \mathcal{T} & \blacktriangle & \mathcal{T} & \blacktriangledown & \mathcal{T}\\ \mathcal{T} & \blacktriangledown & \mathcal{T} & \blacktriangledown & \mathcal{T} & \blacktriangledown\end{matrix}\right)^n \begin{matrix}\mathcal{T} & \blacktriangledown & \mathcal{T} & \blacktriangledown\\ \blacktriangledown & \mathcal{T} & \blacktriangledown & \mathcal{T}\\ \mathcal{T} & \blacktriangledown & \mathcal{T} & \blacktriangledown\end{matrix} : n \geq 0\right)\right\},$$

$$L_B = \left\{\left(\begin{matrix}\mathcal{T} & \mathcal{T} & \mathcal{T}\end{matrix}\right)_n : n \geq 1\right\},$$

$$Pr_3 = \left\{S \to \begin{matrix}\mathcal{T} & \mathcal{T} & \mathcal{T} & \mathcal{T} & \mathcal{T}\\ \mathcal{T} & \mathcal{T} & \mathcal{T} & \mathcal{T} & \mathcal{T}\\ \blacktriangledown & \mathcal{T} & \blacktriangledown & \mathcal{T} & \blacktriangledown\\ \mathcal{T} & \blacktriangledown & \mathcal{T} & \blacktriangledown & \mathcal{T}\end{matrix}\right\}.$$

Clearly, the picture derived from M is palindromic and the production rules adhere to the nature of PALs. Hence, we found that G was a $(Reg : Reg)PAG$ and M was generated by $(CoFree\text{:}\ Reg)PAG$.

6. Conclusions

In this paper, we have defined PAGs as an extension of array-rewriting grammars and evaluate a few examples for various PAG. Further, we studied a few results on PAG and explored how they can facilitate the generation of kolam patterns. Future works will be focused on proving/disproving the other combinatorial properties of PAG and the study of their other applications.

Author contributions

Hannah Blasiyus: conceptualization, formal analysis, methodology, writing-original draft & editing; D. K. Sheena Christy: conceptualization, methodology, validation & writing-review. All authors have read and approved the final version of the manuscript for publication.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Conflict of interest

The authors declare no conflicts of interest.

References

1. A. Thue, Über unendliche zeichenreihen, *Norske Vid Selsk. Skr. I Mat-Nat Kl.*, **7** (1906), 1–22.
2. T. Nagell, A. Selberg, S. Selberg, K. Thalberg, *Selected mathematical papers of Axel Thue*, Universitetsforlaget, 1977.
3. M. Lothaire, *Combinatorics on words*, Addison-Wesley, 2 Eds., 1983. https://doi.org/10.1017/CBO9780511566097
4. L. Kari, K. Mahalingam, Involutively bordered words, *Int. J. Found. Comput. Sci.*, **18** (2022), 1089–1106. https://doi.org/10.1142/S0129054107005145

5. E. Czeizler, L. Kari, S. Seki, On a special class of primitive words, *Theor. Comput. Sci.*, **411** (2010), 617–630. https://doi.org/10.1016/j.tcs.2009.09.037
6. S. S. Yu, *Languages and codes*, Department of Computer Science, National Chung-Hsing University, Taichung, 2005.
7. A. Rosenfeld, *Picture languages, formal models for picture recognition*, Academic Press, 1979. https://doi.org/10.1016/C2013-0-11407-9
8. R. Narasimhan, Labeling schemata and syntactic description of pictures, *Inf. Control*, **7** (1964), 151–179. https://doi.org/10.1016/S0019-9958(64)90087-7
9. G. Siromoney, R. Siromoney, K. Krithivasan, Abstract families of matrices and picture languages, *Comput. Graph. Image Process.*, **1** (1972), 284–307. https://doi.org/10.1016/S0146-664X(72)80019-4
10. G. Siromoney, R. Siromoney, K. Krithivasan, Picture languages with array rewriting rules, *Inf. Control*, **20** (1973), 447–470. https://doi.org/10.1016/S0019-9958(73)90573-1
11. R. Siromoney, K. Krithivasan, G. Siromoney, N-dimensional array languages and description of crystal symmetry-I, *Proc. Indian Acad. Sci.*, **78** (1973), 72–88. https://doi.org/10.1007/BF03049472
12. R. Siromoney, K. Krithivasan, G. Siromoney, N-dimensional array languages and description of crystal symmetry-II, *Proc. Indian Acad. Sci.*, **78** (1973), 130–139. https://doi.org/10.1007/BF03049472
13. M. Tomita, *Parsing 2-dimensional language*, Carnegy Mellon University, 1989, 414–424.
14. A. Amir, G. Benson, Two-dimensional periodicity in rectangular arrays, *SIAM J. Comput.*, **27** (1998), 90–106. https://doi.org/10.1137/S0097539795298321
15. M. Kulkarni, K. Mahalingam, Two-dimensional palindromes and their properties, In: F. Drewes, C. Martín-Vide, B. Truthe, *Language and automata theory and applications*, Springer, 2017, 155–167. https://doi.org/10.1007/978-3-319-53733-7_11
16. K. Mahalingam, P. Pandoh, On the maximum number of distinct palindromic sub-arrays, In: C. Martín-Vide, A. Okhotin, D. Shapira, *Language and automata theory and applications*, Springer, 2019, 434–446. https://doi.org/10.1007/978-3-030-13435-8
17. K. Mahalingam, P. Pandoh, K. Krithivasan, On the least number of palindromes in two-dimensional words, *Theor. Comput. Sci.*, **807** (2020), 246–256. https://doi.org/10.1016/j.tcs.2019.06.030
18. K. Mahalingam, M. Sivasankar, K. Krithivasan, Palindromic properties of two-dimensional Fibonacci words, *Romanian J. Inf. Sci. Technol.*, **21** (2018), 267–277.
19. D. Giammarresi, A. Restivo, Recognizable picture languages, *Int. J. Pattern Recog. Artif. Intell.*, **6** (1992), 241–256. https://doi.org/10.1142/S021800149200014X
20. D. Giammarresi, A. Restivo, Two-dimensional languages, In: G. Rozenberg, A. Salomaa, *Handbook of formal languages*, Springer, 1997, 215–267. https://doi.org/10.1007/978-3-642-59126-6
21. D. E. Knuth, J. Morris, V. Pratt, Fast pattern matching in strings, *SIAM J. Comput.*, **6** (1977), 323–350. https://doi.org/10.1137/0206024

22. R. Cole, M. Crochemore, Z. Galil, L. Gasienec, R. Hariharan, S. Muthukrishnan, et al., Optically fast parallel algorithms for preprocessing and pattern matching in one and two dimensions, *Proceedings of 1993 IEEE 34th Annual Foundations of Computer Science*, 1993. https://doi.org/10.1109/SFCS.1993.366862

23. S. Geizhals, D. Sokol, Finding maximal 2-dimensional palindromes, *Inf. Comput.*, **266** (2019), 161–172. https://doi.org/10.1016/j.ic.2019.03.001

24. P. Linz, S. H. Rodger, *An introduction to formal languages and automata*, 7 Eds., Jones & Barlett, 2023.

25. G. Siromoney, R. Siromoney, K. Krithivasan, Array grammars and kolam, *Comput. Graph. Image Process.*, **3** (1974), 63–82. https://doi.org/10.1016/0146-664X(74)90011-2