Triangular and Octagonal Array Grammars

S. Kuberal*, K. Bhuvaneswari*, T. Kalyani*

*Department of Applied Mathematics, Pillai College of Engineering, New Panvel, Maharashtra
*Department of Mathematics, Sathyabama Institute of Science and Technology, Chennai
*Department of Mathematics, St. Joseph’s Institute of Technology, Chennai

kuberals@mes.ac.in

Abstract. Hexagonal Array Grammars (HAG) were introduced by K.G. Subramanian [4]. HAG increased the capacity of generating the hexagonal arrays on triangular grid. In this paper, using the Triangular Array Grammars (TAG) and Octagonal Array Grammars (OAG), we increasing the generative capacity for triangular and octagonal arrays. A system of the family of Triangular Array Language (TAL) and Octagonal Array Language (OAL) are presented.

1. Introduction

By using the array rewriting rules (kolam patterns) for generating the rectangular array were introduced by Siromoney et al.[2]. They extended this model to generate the hexagonal arrays by Siromoney et al[1]. Then introduced catenation of arrowhead to hexagonal arrays (HA) by using hexagonal kolam array grammars (HKAG)[1].

There are three steps in rewriting rules of a HKAG. The first step is non-terminal rules involving non-terminals and intermediaries but limited in extent to linear and right-linear rules suitable to constraints urged by arrowhead catenations. In this stage, a string of intermediaries and a only one non-terminal, catenated well adjusted with arrowhead catenations and parenthesis, is created. Using the rewrite rules convert a single terminal into a non-terminal as a hexagonal array of terminals are in second step. In the third step using the intermediary rules generate Regular (R), Context-free (CF) and Context-sensitive (CS) intermediary languages, printed in the appearance of arrowheads of permanent intensity.

Followed the same arrowhead catenations but altered the definition of Hexagonal Kolam Array Grammar (HKAG) to generate the hexagonal arrays. The outcome model is called a hexagonal array grammar (HAG)[4]. Here we retain the same concept of HAG, to generate a triangular arrays and octagonal arrays. These outcome models are called TAG and OAG.

In TAG, we have two stages: In the first stage, permitting primary rules which purpose any derivation from the initial sign to commence with an principal triangular array T’. Then we have Regular, Context-Free or Context-Sensitive non-terminal and terminal rules relating non-terminals and intermediaries. With the help of these rules, in successive derivation procedure of the first step just as in string grammars till all the non-terminals are restored. In the second stage, the intermediaries take place after T’ in the string created in the first stage are consecutively restored by arrowheads from the matching intermediary arrowhead languages and catenated to T’, one by one subject to conditions of
arrowhead catenations, thus creating a triangular array of terminals. The outcome model is triangular array grammar (TAG).

Hence in a HKAG, the innermost parentheses has the initial HA inside it and also has catenated the arrowheads starting from it and moving outwards. Whereas in TAG, the starting position of the triangular array T' is middle and it is also catenated using the arrowheads from up or left or right respectively. Likewise the initial position of Octagonal Array O' is left most location and arrowheads are catenated to it from left to right or kept in the position of right most and arrowheads are catenated to it from right to left or kept in the middle most and arrowhead are catenated to it form upper or lower direction. These rules are increasing the capacity of generating more pictures.

2. Triangular Array Grammars

Definition 1.
A Regular Triangular Array Grammar (RTAG) is G = (V, İ, C, R, Š, Ł). The notations V, İ and C are non-empty finite sets of symbols known as variables, intermediates and constants respectively. Š ∈ V is the start symbol. Ł = {Lb / B ∈ İ} where L is an intermediate regular language of caps. R = R₁ ∪ R₂ is a set of productions of a finite non-empty where R₁ subsists of starting rules are:

(1) Š → į U
(2) Š → į U
(3) Š → į į U

where U ∈ V, U ≠ Š and T denotes triangular array belongs to C. The rules of R₂ are of the forms:

(1) U₁ → B į U₂
(2) U₁ → B į U₂
(3) U₁ → B į U₂

where U₁, U₂ ∈ V, U₁ ≠ Š and B ∈ İ.

G is called CF-TAG (CS-TAG) if one of the intermediate languages in the above definition is context-free (context-sensitive).

Definition 2.
Let us consider X, Y ∈ {Regular, Context-Free, Context-Sensitive}, the (X, Y) triangular array language (X: Y) Triangular Array Language created by the (X: Y) Triangular Array Grammar. G is L(G) = { T / Š → į į T, T is a triangular array}.

Example 1.
Consider a TTRG G = (Σ, M, Š, R), where Σ = {*, a}, M = { S'}, the rules of R are:

\[
S' \rightarrow \begin{array}{c}
\text{a} \\
\text{*} \\
\text{Aa} \\
\text{Aa} \\
\text{Aa} \\
\text{Aa} \\
\text{Aa} \\
\text{Aa} \\
\end{array}
\]
The image of the language $L(G)$ is,

$\begin{align*}
a & \\
* & * & * & a & \\
* & a & * & * & a & \\
* & a & * & * & a & \\
* & a & * & * & a & \\
a & * & * & a & * & * & a & \\
* & * & a & * & * & a & \\
* & a & * & * & a & \\
* & a & * & * & a & \\
a & * & * & a & * & * & a & \\
* & * & a & * & * & a & \\
* & * & a & * & * & a & \\
* & * & a & * & * & a & \\
a & * & * & a & * & * & a & \\
* & a & * & * & a & \\
* & a & * & * & a & \\
* & a & * & * & a & \\
a & * & * & a & \\
* & * & a & \\
* & a & \\
\end{align*}$

Size 8

It is identify with the purpose of the language $L(G)$ is created by the grammar $(R : CF)$ TAG,

$G = \{M, \bar{I}, \{*, a\}, R_1 \cup R_2, U^1, U^2, T'\}$

$M = \{U^1, U^2\}$

$\bar{I} = \{B_1, B_2, B_3\}$
\[ T^* \rightarrow a \]

\[ R_1 = \{ U^1 \rightarrow T_U U^2 \} \]

\[ R_2 = \{ U^2 \rightarrow B_1 U^2, U^2 \rightarrow B_2 U^2, U^2 \rightarrow B_3, \text{ and } U^2 \rightarrow B_1 U^2, U^2 \rightarrow B_2 \} \]

\[ LB_1 = \{ a \} \times m < a > * m \times a >, m \geq 2 \}

\[ LB_2 = \{ m < a > * m < a > * m \times a >, m \geq 2 \} \]

\[ LB_3 = \{ m \times a > * m < a > * m < a > * m \times a > * m, m \geq 2 \} \]

3. Octagonal Array Grammars

Definition 3.

An octagonal array grammar \( G = (M, \bar{I}, \bar{T}, R, \bar{S}, \bar{L}) \) where \( M, \bar{I} \) and \( \bar{T} \) are finite collection of non-terminals, intermediates and terminals correspondingly. \( R \) is a finite collection of productions, \( R = R_1 \cup R_2 \) and \( \bar{S} \in V \) is the begin symbol. For every \( B \) in \( \bar{I} \), \( \bar{L}_B \) is an intermediate language which is a regular Context-free and Context-sensitive string language written in the suitable arrowhead form. An arrowhead is written in the form \( \{ \ldots<\gamma>\ldots \} \) where \( <\gamma> \) denotes the vertex and the arrowhead is written in the clockwise direction \( \bar{L} = \{ \bar{L}_B / B \in \bar{I} \} \).

\( R_1 \) subsists of an initial rule of one of the following form

\( (1) \bar{S} \rightarrow O \bar{S}_U \)

\( (2) \bar{S} \rightarrow O \bar{S}_U \)

\( (3) \bar{S} \rightarrow \bar{S}_U \)

\( (4) \bar{S} \rightarrow \bar{S}_U \)

\( (5) \bar{S} \rightarrow \bar{S}_U \)

\( (6) \bar{S} \rightarrow \bar{S}_U \)

\( (7) \bar{S} \rightarrow \bar{S}_U \)

\( (8) \bar{S} \rightarrow \bar{S}_U \)

where \( U \in M, U \neq \bar{S} \) and \( O' \) is an octagonal array over \( \bar{T} \).

\( G \) is regular if the rules of \( R_2 \) are of the forms

\( (1) U^1 \rightarrow B \bar{S}_U \)

\( (2) U^1 \rightarrow B \bar{S}_U \)

\( (3) U^1 \rightarrow B \bar{S}_U \)

\( (4) U^1 \rightarrow B \bar{S}_U \)

\( (5) U^1 \rightarrow B \bar{S}_U \)

\( (6) U^1 \rightarrow B \bar{S}_U \)

\( (7) U^1 \rightarrow B \bar{S}_U \)

\( (8) U^1 \rightarrow B \bar{S}_U \)

where \( U^1, U^2 \in M, U^1, U^2 \neq \bar{S} \) and \( B \in \bar{I} \).

Additionally, if an preliminary rule is \( R_1 \) is of the form \( (t) \), \( t = 1, 2, \ldots \) then \( R_2 \) does not contain any rule of the form \( (t+1) \) if \( t \) is odd, \( (t-1) \) if \( t \) is even. Also \( R_1 \) and \( R_2 \) do not encompass rules of the form \((t)\) and \((t+1)\), \( t = 1, 3, 5, 7 \).

\( G \) is Context-free if the rules are of the form \( U^1 \rightarrow \alpha_1 \bar{S} \ldots \bar{S} \bar{S}_m (m \geq 1) \) where \( U^1 \in M; U^1 \neq \bar{S} \) and \( \alpha_1 \in (M - \{ \bar{S} \}) \cup \bar{I} \) \((1 \leq i \leq m)\) and \( \bar{S}_j \) denotes any one of the eight arrowhead catenations \((1 \leq j \leq m-1)\).

\( G \) is Context-sensitive if the rules of \( R_2 \) are of the form \( \beta \bar{S}_1 \bar{S}_2 \ldots \bar{S}_m \) where \( U^1 \in M; U^1 \neq \bar{S} \) and \( \beta, \gamma \) \((1 \leq i \leq m)\) and \( \alpha_1 \in (M - \{ \bar{S} \}) \cup \bar{I} \) \((1 \leq j \leq m-1)\).

Particular, \( G \) is called (X: Regular) OAG or (X: Context-Free) OAG or (X: Context-Sensitive) OAG, for \( X \in \{R, CF, CS\} \) in the opinion of all the intermediate languages are regular or at least one of them is Context-free or Context-sensitive.

Steps of derivations:

1. A preliminary rule of \( R_1 \) is involved and then \( R_2 \) are involved in the sequel as in string grammars until all the non-terminals are restored, the string form of the final result is \( O \bar{S}_1 \bar{S}_2 \ldots \bar{S}_m \) where \( B_1 \in I \) \((1 \leq i \leq m)\).
2. In next step, \( B_1 \) restored by an arrowhead from \( \hat{L}_{B_1} \) and catenated to an octagonal array \( O' \) according to the arrowhead catenation symbol between \( O' \) and \( B_1 \). Continue until \( B_m \) is restored in the form of an octagonal array of terminals.

**Definition 4.**
Consider \( \dot{X}, \dot{Y} \in \{ \text{Regular, Context-Free, Context-Sensitive} \} \), the \((\dot{X}, \dot{Y})\) octagonal array language \((\dot{X},\dot{Y})\text{OAL}) generated by the \((\dot{X},\dot{Y})\text{OAG}) \( G \) is \( L(G) = \{ O'/ S \rightarrow O', O' \text{ is an Octagonal array} \} \).

**Example 2.**
Consider \( G = \{ \Sigma, M, S, R \} \) is an octagonal tile rewriting grammar, where \( \{ \cdot, x \} \), \( M = \{ S \} \) and \( R \) having the rules:

\[
\begin{align*}
S &\rightarrow \{ \text{x} \} \\
S &\rightarrow \{ SS, SS, SS \} \\
\end{align*}
\]

A picture \( L(G) \) becomes,

\[
\begin{array}{c}
\text{Size 5}
\end{array}
\]

A picture \( L(G) \) is got by frequent appertain of the variable size rule and fixed size rule.

For example, we give the sample derivation to yield the picture.
4. Comparison Results

Lemma 1: The Triangular Array Language creates by the (Regular : Context-Free) Triangular Array Grammar.

Proof: Let $G = \{M, \bar{I}, \{*, a\}, R_1 \cup R_2, U^1, U^2, T'\}$

Here $M = \{U^1, U^2\}$ and $\bar{I} = \{B_1, B_2, B_3\}$

\[ T' \rightarrow * \quad a \]

$R_1 = \{U^1 \rightarrow T' \quad 4 \uparrow U^2\}$

$R_2 = \{U^2 \rightarrow B_1 \quad B_2 \quad B_3 \uparrow \downarrow U^2, U^2 \rightarrow B_1 \quad B_2 \quad B_3\}$

$LB_1 = \{<a>^{*m<\alpha>*m<\alpha>}, m \geq 2\}$

$LB_2 = \{<m<\alpha>*m<\alpha>*m<\alpha>*m<\alpha>*, m \geq 2\}$

$LB_3 = \{<m<\alpha>*m<\alpha>*m<\alpha>*m<\alpha>*m<\alpha>*, m \geq 2\}.

This is a (R:CF)TAL, it is not possible to generate by any (R:R)TAG.

Lemma 2: The TAL creates by the (CF:R)TAG.

Proof: Let $G = \{M, \bar{I}, \{*, a\}, R_1 \cup R_2, U^1, U^2, T'\}$

Here $M = \{U^1, U^2\}$ and $\bar{I} = \{B_1, B_2, B_3\}$
This is a (CF:R)TAL, it is not possible to generate by any (R:R)TAG.

**Lemma 3:**
The OAL generates by the (R:R)OAG G defined as,

\[ \begin{array}{c}
\text{a} \\
\text{a} \\
\end{array} \]

\[ \begin{array}{c}
\text{O}_1 = \\
\text{a} \\
\text{a} \\
\end{array} \]

\[ \begin{array}{c}
\text{b} \\
\text{b} \\
\end{array} \]

\[ \begin{array}{c}
\text{a} \\
\text{a} \\
\end{array} \]

\[ \begin{array}{c}
\text{c} \\
\text{c} \\
\end{array} \]

\[ \begin{array}{c}
\text{a} \\
\text{a} \\
\end{array} \]

\[ \begin{array}{c}
\text{d} \\
\text{d} \\
\end{array} \]

\[ \begin{array}{c}
\text{e} \\
\text{e} \\
\end{array} \]

\[ \begin{array}{c}
\text{f} \\
\text{f} \\
\end{array} \]

\[ \begin{array}{c}
\text{g} \\
\text{g} \\
\end{array} \]

\[ \begin{array}{c}
\text{h} \\
\text{h} \\
\end{array} \]

\[ \begin{array}{c}
\text{i} \\
\text{i} \\
\end{array} \]

\[ \begin{array}{c}
\text{j} \\
\text{j} \\
\end{array} \]

\[ \begin{array}{c}
\text{k} \\
\text{k} \\
\end{array} \]

\[ \begin{array}{c}
\text{l} \\
\text{l} \\
\end{array} \]

\[ \begin{array}{c}
\text{m} \\
\text{m} \\
\end{array} \]

\[ \begin{array}{c}
\text{n} \\
\text{n} \\
\end{array} \]

\[ \begin{array}{c}
\text{o} \\
\text{o} \\
\end{array} \]

\[ \begin{array}{c}
\text{p} \\
\text{p} \\
\end{array} \]

\[ \begin{array}{c}
\text{q} \\
\text{q} \\
\end{array} \]

\[ \begin{array}{c}
\text{r} \\
\text{r} \\
\end{array} \]

\[ \begin{array}{c}
\text{s} \\
\text{s} \\
\end{array} \]

\[ \begin{array}{c}
\text{t} \\
\text{t} \\
\end{array} \]

\[ \begin{array}{c}
\text{u} \\
\text{u} \\
\end{array} \]

\[ \begin{array}{c}
\text{v} \\
\text{v} \\
\end{array} \]

\[ \begin{array}{c}
\text{w} \\
\text{w} \\
\end{array} \]

\[ \begin{array}{c}
\text{x} \\
\text{x} \\
\end{array} \]

\[ \begin{array}{c}
\text{y} \\
\text{y} \\
\end{array} \]

\[ \begin{array}{c}
\text{z} \\
\text{z} \\
\end{array} \]

This is a (R:R)OAL, which is not possible to generate by any (R:CF)OAG.

**Example 3.**
Consider the (R:R)OAG \( G = \{ S', U_1 \}, \{ A_1, A_2 \}, \{ a \}, R, S', L \}, \)
where \( R = R_1 \cup R_2, \)
\( R_1 = \{ S' \to O_1, U_1 \to A_1 \} \)
\( R_2 = \{ U_1 \to A_2 \} \)
\( L_{A_1} = \{ a^m < a^n \to a^{m+n} \} \)

we can derive the element \( O_2 \in L(G) = \{ O_m, m \geq 1 \} \)

\[ \begin{array}{c}
\text{a} \\
\text{a} \\
\end{array} \]

\[ \begin{array}{c}
\text{b} \\
\text{b} \\
\end{array} \]

\[ \begin{array}{c}
\text{c} \\
\text{c} \\
\end{array} \]

\[ \begin{array}{c}
\text{d} \\
\text{d} \\
\end{array} \]

\[ \begin{array}{c}
\text{e} \\
\text{e} \\
\end{array} \]

\[ \begin{array}{c}
\text{f} \\
\text{f} \\
\end{array} \]

\[ \begin{array}{c}
\text{g} \\
\text{g} \\
\end{array} \]

\[ \begin{array}{c}
\text{h} \\
\text{h} \\
\end{array} \]

\[ \begin{array}{c}
\text{i} \\
\text{i} \\
\end{array} \]

\[ \begin{array}{c}
\text{j} \\
\text{j} \\
\end{array} \]

\[ \begin{array}{c}
\text{k} \\
\text{k} \\
\end{array} \]

\[ \begin{array}{c}
\text{l} \\
\text{l} \\
\end{array} \]

\[ \begin{array}{c}
\text{m} \\
\text{m} \\
\end{array} \]

\[ \begin{array}{c}
\text{n} \\
\text{n} \\
\end{array} \]

\[ \begin{array}{c}
\text{o} \\
\text{o} \\
\end{array} \]

\[ \begin{array}{c}
\text{p} \\
\text{p} \\
\end{array} \]

\[ \begin{array}{c}
\text{q} \\
\text{q} \\
\end{array} \]

\[ \begin{array}{c}
\text{r} \\
\text{r} \\
\end{array} \]

\[ \begin{array}{c}
\text{s} \\
\text{s} \\
\end{array} \]

\[ \begin{array}{c}
\text{t} \\
\text{t} \\
\end{array} \]

\[ \begin{array}{c}
\text{u} \\
\text{u} \\
\end{array} \]

\[ \begin{array}{c}
\text{v} \\
\text{v} \\
\end{array} \]

\[ \begin{array}{c}
\text{w} \\
\text{w} \\
\end{array} \]

\[ \begin{array}{c}
\text{x} \\
\text{x} \\
\end{array} \]

\[ \begin{array}{c}
\text{y} \\
\text{y} \\
\end{array} \]

\[ \begin{array}{c}
\text{z} \\
\text{z} \\
\end{array} \]
We generates the following pictures:

5. Conclusion
In this paper, the triangular and octagonal array grammars are used to increase the generative capacity of triangular and octagonal arrays. A hierarchy among the family of triangular and octagonal array languages are attained.

References
[1] G. Siromoney and R. Siromoney, Hexagonal arrays and rectangular blocks, Comput. Graphics Image Processing 5 (1976) 353–381.
[2] G. Siromoney, R. Siromoney and K. Krithivasan, Picture languages with array rewriting rules, Inform. Contr. 22 (1973) 447–470.
[3] G. Siromoney, R. Siromoney and K. Krithivasan, Array grammars and kolam, Comput. Graphics Image Processing 3 (1974) 63–82.
[4] K.G. Subramanian, Hexagonal Array Grammars, Computer Graphics and Image Processing 10 (1979) 388–394.
[5] S. CrespiReghizyi and M. Pradella, Tile Rewriting Grammars and Picture Languages, Theoretical Comp. Science 340 (2005) 257–272.
[6] F. Sweety, T. Kalyani and D.G. Thomas, Hexagonal Tile Rewriting Grammars, 7th National
Conference on Emerging Trends in Automata (ETA’07) (2007) 101–115.

[7] S. Kuberal, T. Kalyani and D.G. Thomas, Triangular Tile Rewriting Grammars and Triangular Picture Languages, Global Journal of Pure and Applied Mathematics 12(3) (2016) 1965–1978.

[8] S. Kuberal, Dr. T. Kalyani, Dr. T. Kamaraj and K. Bhuvaneswari, Octagonal Tile Rewriting Grammars and Picture Languages, International Journal of Computer & Mathematical Sciences IJCMS, 11(6) (2017) 219–232.

[9] S. Kuberal, Dr. T. Kamaraj and Dr. T. Kalyani, Regional Triangular Tile Rewriting Grammars and Triangular Picture Languages, International Journal of Pure and Applied Mathematics, 119(15) (2018) 1325–1337.

[10] S. Kuberal, Dr. T. Kamaraj and Dr. T. Kalyani, Octagonal Arrays and Petri Nets, A Computational Ecology Approach (Web of Science), Article No: e107031 (2018).

[11] S. Kuberal, Dr. T. Kalyani, and Dr. D.G. Thomas, Petri Net Generating Triangular Arrays, Elsevier Procedia Computer Science, 57 (2015) 642–649.