数理解析研究所講究録
421 巻 1981 年 47-60

ON A RELATIONSHIP BETWEEN GRAPH L-SYSTEMS AND PICTURE LANGUAGES

Kunio Aizawa
Faculty of Integrated Arts and Sciences
Hiroshima University

Akira Nakamura
Department of Applied Mathematics
Hiroshima University

1. Introduction

In recent years, many models for graph L-systems or parallel graph grammars have been proposed. One of the reason why those models were proposed is to generalize the string L-systems to hyper-dimensional or more complex structures. Unlike the string L-systems, the basic primitive in a graph L-system is a directed graph which is a set of labelled nodes and labelled edges. In this paper, as the graph L-systems, we treat only one model which is called a *node-replacement graph L-system with explicit connection transformation* and its extensions. About other models for graph L-systems, see e.g. [3].

On the other hand, many grammars to generate two-dimensional patterns have been studied. Generally, in those two-dimensional grammars, to preserve topological relations between each point in generated arrays a restriction called "isometric" is inserted in each production rule. If we consider the application of graph L-systems to two-dimensional pictures then we can reduce this restriction and construct more natural two-dimensional pictures generating systems. From this point of view, the relation-

ships between graph L-systems and matrix grammars are discussed in Chapter 8 of Rajasethupathy [1]. In [1], it is shown the following facts:

The class of extended tabled node-replacement graph L-systems (especially without terminal set for alphabet of edge labels)

1. contains the class of matrix grammars properly,
2. generates several kolam patterns forming subclass of (CF:R) array grammars.

Furthermore this system without tables

3. generates "staircase of X's", "Kirsch's right triangles" etc.

From above facts, Rajasethupathy conjectured that for every array grammar there exists an equivalent extended tabled node-replacement graph L-system.

In this paper, we will show that the class of extended tabled node-replacement graph L-systems contains the class of isometric array grammars, by using isometric array L-systems which were introduced in Nakamura [2]. It seems to be a very interesting result to generate isometric array languages by graph L-systems which replace each node in context-free style.

2. Basic Definitions and Properties

In this section, we review definitions and properties used in the next section. In the following definitions of graph L-systems, we will use basic definitions and notations in [4].

Definition 2.1. A *node-replacement graph L-system with explicit connection transformation*, denoted by nGL-exp-system, is a 6-tuple $G = (V, \Sigma_V, \Sigma_E, RR, CR, d_0)$, where V is a nonempty finite set of node generators, Σ_V is a nonempty finite set of node labels, Σ_E is a nonempty finite set of

edge labels, $d_0 \in d(V, \Sigma_V, \Sigma_E) - \{d_\varepsilon\}$ is the *axiom graph*. RR is a finite set of replacement rules which is *complete* in the sense that for any one-node subgraph of the axiom graph or right hand sides of replacement rules, there is at least one replacement rule in RR having this one-node graph as left hand side. CR is a finite set of connection rules which is *complete* in the sense that for any triple $(v_1, a, v_2) \in \Sigma_V \times \Sigma_E \times \Sigma_V$ there is at least one connection rule such that v_1, v_2 are the labels of the source and target node and a is the label of the edge of the left hand side of the connection rule. The language of a nGL-exp-system G is defined as

$$\mathcal{L}(G) = \{ D \mid D \in d(V, \Sigma_V, \Sigma_E)/\equiv \text{ and } d_0/\equiv \xrightarrow[G]{pe}{}^{*} D \}.$$

Since, in this paper, we consider only explicit connection transformation systems, hereafter denote nGL-exp-system as nGL-system if no confusion occur. The extension of this nGL-system is as follows:

Definition 2.2. An *extended tabled nGL-system*, denoted by ETnGL-system, is a 7-tuple $G = (V, \Sigma_V, \Sigma_E, P, d_0, \Delta_V, \Delta_E)$, where V, Σ_V, Σ_E and d_0 are the same as defined for a nGL-system, $P = \{ (RR_i, CR_i) \mid 1 \leq i \leq n \}$ for some integer n, where RR_i is a finite set of replacement rules, and CR_i is a finite set of connection rules, $\Delta_V \subseteq \Sigma_V$ is a terminal alphabet of node labels, $\Delta_E \subseteq \Sigma_E$ is a terminal alphabet of edge labels. A derivation is obtained as in the case of nGL-systems except for the condition that for any single step of derivation all replacement rules used in this step must belong to the same set RR_i for some i; similarly all connection rules used at this step must belong to the corresponding CR_i. Then the language of an ETnGL-system G is defined as

$$\mathcal{L}(G) = \{ D \mid D \in d(V, \Delta_V, \Delta_E)/\equiv \text{ and } d_0/\equiv \xrightarrow[G]{pe}{}^{*} D \}.$$

Now, we review some definitions of isometric array grammars [6] and isometric array L-systems [2].

Defintion 2.3. An *isometric array grammar*, denoted by IAG, is a 5-tuple $G = (V_N, V_T, P, S, \#)$, where V_N is a finite nonempty set of non-terminals, V_T is a finite nonempty set of terminals, $V_N \cap V_T = \phi$, $S \in V_N$ is the start symbol, $\# \notin V_N \cup V_T$ is the blank symbol, and P is a set of production rules. Each member of P is a pair of arrays of finite size (α, β), for all of which

(a) α and β are geometrically identical,

(b) α does not consist entirely of #s, and

(c) β satisfies the following conditions:

(1) If the non-#s of α do not touch the border of α, then the non-#s of β must be connected (and nonempty).

(2) Otherwise,

(i) every connected component of non-#s in β must be contained in the intersection of some component of non-#s in α with the border of α;

(ii) conversely, every such intersection must be connected in some component of non-#s in β.

The array language generated by an IAG G is defined as

$$\mathcal{L}(G) = \{ A \mid S \overset{*}{\Longrightarrow} A, \text{ and A is a terminal array } \}.$$

Definition 2.4. Let k_1, k_2, k_3, k_4 be nonnegative integers. An *isometric array L-system with* $< k_3 \begin{matrix} k_2 \\ k_4 \end{matrix} k_1 >$ *array interactions*, denoted by IA$< k_3 \begin{matrix} k_2 \\ k_4 \end{matrix} k_1 >$L^2-system, is a 4-tuple $G = (\Sigma, \#, P, \omega)$, where Σ is a finite nonempty set of symbols (the alphabet of G), # is a symbol which

designates blank points of array (the blank symbol of G), ω is the start array (the *axiom* of G), P is a finite nonempty set of relations satisfying the following;

(1) each element of P has such a form as

$$\left\langle \begin{matrix} \xi_2 & \xi_1 \\ \xi_3 & \xi_4 \end{matrix} \right\rangle \quad \alpha \quad \rightarrow \quad \beta,$$

where α and β are arrays over $\Sigma \cup \{\#\}$, and are isometric. α contains exactly *one* element of Σ in context of #'s. $\alpha \rightarrow \beta$ is called *core production*. ξ_i, $1 \leq i \leq 4$, are rectangular arrays over $\Sigma \cup \{\#\}$, and they are called *array contexts*. ξ_1 consists of the first quadrant and positive part of horizontal axis, ξ_2 consists of the second quadrant and positive part of vertical axis, ξ_3 consists of the third quadrant and negative part of hrizontal axis, and ξ_4 consists of the fourth quadrant and negative part of vertical axis. Thus their sizes are $k_1 \times (k_2+1)$, $(k_3+1) \times k_2$, $k_3 \times (k_4+1)$, and $(k_1+1) \times k_4$ (width by height), respectively.

(2) P is *complete* in the sense that for every array contexts ξ_1, ξ_2, ξ_3, ξ_4, and a Σ which is contained in α, there exists at least one array β over $\Sigma \cup \{\#\}$ such that

$$\left\langle \begin{matrix} \xi_2 & \xi_1 \\ \xi_3 & \xi_4 \end{matrix} \right\rangle \quad \alpha \quad \rightarrow \quad \beta \quad \in \quad P,$$

(each element of P is called a *production rule*).

The array language of an isometric array L-system G is defined as

$$\mathcal{L}(g) = \{ \alpha \mid \alpha \text{ is an array over } \Sigma \cup \{\#\}, \text{ and } \omega \overset{*}{\underset{G}{\Rightarrow}} \alpha \}.$$

Definition 2.5. An *extended isometric array L-system*, denoted by

IEA$\langle k_3 \begin{matrix} k_2 \\ k_4 \end{matrix} k_1 \rangle$L^2-system, is a pair $G = (H, \Delta)$, where $H = (\Sigma, \#, P, \omega)$

is an underlying isometric array L-system, and $\Delta \subseteq \Sigma$ is the target alphabet of G. The language of an extended isometric array L-system G is defined as

$$\mathcal{L}(G) = \{ \alpha \mid \alpha \text{ is an array over } \Delta \cup \{\#\} \text{ and } \alpha \in \mathcal{L}(H) \}.$$

Note. While IAGs are the sequential grammars, isometric array L-systems are the parallel systems, as in the string case [5]. In a parallel derivation step, there exist some possibilities such that rewritten arrays are duplicated. In such a case, one of the duplicating subarrays are chosen as the rewritten subarray of duplicating area, in nondeterministically.

The following theorem which was shown in Nakamura [2] will clerify a relation between IAGs and isometric array L-systems.

Theorem 2.1. The class of languages generated by IEA$< 1 \begin{matrix} 0 \\ 1 \end{matrix} 0 >L^2$-systems is exactly the same one generated by IAGs.

3. Some Relations Between ETnGL-systems and Picture Languages

In this section, we will show some relationships between ETnGL-systems and isometric array languages. To show these relationships, we will use the following numbering technique for core productions of the isometric array L-systems.

Let k_1, k_2, k_3, and k_4 be nonnegative integers. Let $G = (\Sigma, \#, P, \omega, \Delta)$ be an IEA$< k_3 \begin{matrix} k_2 \\ k_4 \end{matrix} k_1 >L^2$-system. For some positive integer n, let $P = \{ p_1, p_2, \ldots, p_n \}$, where p_i is a production rule of G. As an example, let us suppose that the core production of i-th production rule p_i is of the following form;

```
  # #              a a
  #                c
# a #      →      c b c
    #                c,   where a, b, c∈Σ.
```

Then as the origin, we take the point of right hand side, which is corresponding to only one point of left hand side whose symbol is non-#. Each point of right hand side is numbered by a 3-tuple of integers as follows:

	(i,0,2)	(i,1,2)
	(i,0,1)	
(i,-1,0)	(i,0,0)	(i,1,0)
		(i,1,-1)

And we will use a notation $T(p_i) = \{ (i,0,2),(i,1,2),\ldots,(i,0,0),\ldots,(i,1,-1) \}$ to denote the above situation. Since the right hand side array of every production rule has finite size, the number of 3-tuple, i.e. the number of elements of the set $\bigcup_{i=1}^{n} T(p_i)$, is determined uniquely and only finite for any given system. Now we chose u and v as follows:

$$u = \mathrm{MAX}_{p_i \in P} \left(\mathrm{MAX}_{(i,j,k)\in T(p_i)} |j| \right),$$

$$v = \mathrm{MAX}_{p_i \in P} \left(\mathrm{MAX}_{(i,j,k)\in T(p_i)} |k| \right).$$

Then for $1 \leqq i \leqq n$, $-u \leqq j \leqq u$, and $-v \leqq k \leqq v$, we define two functions $\sigma: I\times I\times I \to \{0,1\}$ and $\tau: I\times I\times I \to \Sigma$ as follows:

$$\sigma(i,j,k) = \begin{cases} 1 & \text{if there exist point } (i,j,k) \\ 0 & \text{otherwise,} \end{cases}$$

$$\tau(i,j,k) = \begin{cases} \text{the symbol of the point } (i,j,k) & \text{if } (i,j,k)=1 \\ \text{undefined} & \text{otherwise.} \end{cases}$$

Here we stipulate two different way to regard an array as a ℓ-graph.

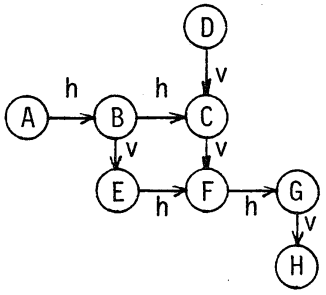
Stipulation 3.1.

(1) A symbol in an array is regarded as a label of the corresponding node of a ℓ-graph.

(2) A horizontal connection of two points is regarded as an edge labelled by "h". The direction of the edge is left to right.

(3) A vertical connection of two points is regarded as an edge labelled by "v". The direction of the edge is up to down.

Example 3.1. An array
```
  D
A B C
E F G
  H
```
is regarded as a ℓ-graph



in the sense of Stipulation 3.1.

Let $\mathcal{L}$ be a set of arrays. Let $\mathcal{L}'$ be a set of ℓ-graphs such that for each element ℓ of $\mathcal{L}'$ there exists an element of $\mathcal{L}$ which regarded as the ℓ-graph ℓ in the sense of Stipulation 3.1 and for each element of $\mathcal{L}$ there exists such a element of $\mathcal{L}'$. Then we will use a notation $\mathcal{L} \underset{1}{=} \mathcal{L}'$ to represent above relation.

Stipulation 3.2. An array is regarded as a ℓ-graph which takes the square pattern such that if all nodes which are labelled by #s are ignored, the array is regarded as the ℓ-graph in the sense of Stipulation 3.1.

As for the case of Stipulation 3.1, we will use a notation $\mathcal{L} \underset{2}{=} \mathcal{L}'$. Now we are ready to show that the class of ETnGL-systems contains the class of IAGs. Firstly we will show a relationship between ETnGL-systems and isometric array L-systems.

Lemma 3.1. For every IEA< 1 $\overset{0}{\underset{1}{}}$ 0 >L^2-system S there exists an ETnGL-system G such that $\mathcal{L}(G) \overset{=}{2} \mathcal{L}(S)$.

proof. Let S = (Σ,#,P,ω,Δ) and let P = { p_1,p_2,...,p_n }. Then we construct the following ETnGL-system.

$$G = (V,\Sigma_V,\Sigma_E,P,\textcircled{\omega},\Delta_V,\Delta_E),$$

where $\Sigma_V = \Sigma\cup\overline{\Sigma}\cup\Sigma_s\cup\Sigma_d\cup\overline{\Sigma}_d\cup\{\#,\#',\#'',\overline{\#}\}$, $\Sigma_E = \{h,v,D\}$, $P = \{(RR_i,CR_i)\,|\,1 \leq i \leq 5\}$ $\Delta_V = \Sigma\cup\{\#\}$, $\Delta_E = \{h,v\}$. Here $\overline{\Sigma} = \{\overline{A}\,|\,A\in\Sigma\}$, $\Sigma_s = \{A_{ijk}\,|\,A\in\Sigma$ and $\tau(i,j,k)=A\}$, $\Sigma_d = \{UL,UR,DL,DR,U,D,L,R\}$, $\overline{\Sigma}_d = \{\overline{A}\,|\,A\in\Sigma_d\}$. Since (RR_1,CR_1), (RR_2,CR_2), and (RR_5,CR_5) are easily constructed, we describe in detail only (RR_3,CR_3) and (RR_4,CR_4) in FIgure 3.1. Note that in the proof of Theorem 2.1, isometric array L-system which simulates IAG requires only two points as its contexts, i.e. left-context and lower-context. Without loss of generality, we can assume that the start array of extended isometric array L-system consists of only one symbol.

The following outline will be useful in order to understand the operations of G.

(1) Firstly, G constructs a square pattern as shown in Figure 3.2 by using tables (RR_1,CR_1) and (RR_2,CR_2). By m-time applications of (RR_1,CR_1), G can construct the pattern whose size is $(2m+1)\times(2m+1)$. Once G use table (RR_2, CR_2) then the square pattern remains in same size in the rest of derivation.

(2) Then, G simulates the derivation of S using tables (RR_3,CR_3) and (RR_4, CR_4). In general, each single step of S is simulated by two steps of G. For each node which has label from $\overline{\Sigma}$, G guesses what production of S is applied. Then G replaces allnodes having labels from $\overline{\Sigma}$ and necessary nodes having label $\overline{\#}$ by one-node graphs having labels from Σ_s, according to the guessed productions. Another nodes are replaced by one-node graphs having labels from $\overline{\Sigma}_d\cup\{\#''\}$. Next, to make certain of legality of its replacement opera-

tions, G checks subscripts of node labels of adjacent nodes using connection rules from CR_3. If adjacent nodes are replaced in regal way, they are connected as before. If not, they are connected by an edge having label D. Note that the edge labelled by D cannot be replaced by any other edges. Then G replaces all nodes having labels from $\Sigma_s \cup \{\#''\}$ by one-node graphs having labels from $\bar{\Sigma} \cup \{\bar{\#}\}$.

To simulate derivations of S, the step (2) are repeated.

(3) Finally, G guesses the end time of derivation of S and sets its -graph right by using table (RR_5, CR_5). G replaces all nodes having labels from $\bar{\Sigma} \cup \{\bar{\#}\}$ by one-node graphs whose labels are corresponding one from $\Sigma \cup \{\#\}$. Also G replaces all nodes having labels from $\bar{\Sigma}_d$ by one-node graphs labelled by a symbol #.

In this way, we obtain $\mathcal{L}(G) \underset{2}{=} \mathcal{L}(S)$.

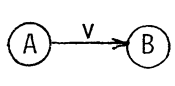
Figure 3.1. Constructions of table (RR_3, CR_3) and (RR_4, CR_4)

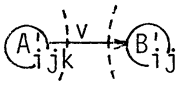
RR_3

$(\#) ::= (\#'')$	
$(\bar{A}) ::= (B_{i00})$	for every $A,B \in \Sigma$ such that there exists i-th production rule whose core production is $A \to \alpha$, and $\sigma(i,0,0)=1$, $\tau(i,0,0)=B$.
$(\bar{\#}) ::= (B_{ijk})$	for every B and i,j,k such that $\sigma(i,j,k)=1$ and $\tau(i,j,k)=B$.
$(A) ::= (A)$	for every $A \notin \Sigma \cup \{\bar{\#}\}$.

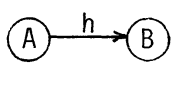
CR_3

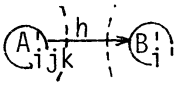
$(A) \xrightarrow{h} (B) ::= (A'_{ij-1k}) \xrightarrow{h} (B'_{ijk})$ for every $A,B \in \bar{\Sigma} \cup \{\bar{\#}\}$ and $A'_{ij-1k}, B'_{ijk} \in \Sigma_s$ such that $(A) ::= (A'_{ij-1k})$, $(B) ::= (B'_{ijk}) \in RR_3$. If $j=k=0$, A must be the left-context of B in S.



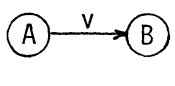
::= 

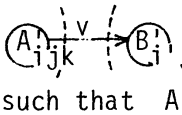
for every A,B$\in\Sigma\cup\{\overline{\#}\}$ and A'_{ijk},B'_{ijk-1} $\in\Sigma_s$ such that A ::= A'_{ijk}, B ::= B'_{ijk-1} $\in RR_3$. If j=k=0, B must be the lower-context of A in S.



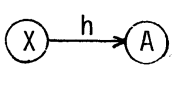
::= 

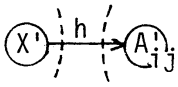
for every A,B$\in\Sigma\cup\{\overline{\#}\}$ and A'_{ijk},$B'_{i'j'k'}$ $\in\Sigma_s$ such that A ::= A'_{ijk}, B ::= $B'_{i'j'k'}\in RR_3$, and σ(i, j+1,k)=0 or σ(i',j'-1,k')=0. If j'=k'=0, A must be the left-context of B in S.



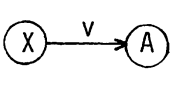
::= 

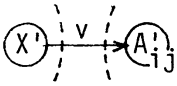
for every A,B$\in\Sigma\cup\{\overline{\#}\}$ and A'_{ijk},$B'_{i'j'k'}$ $\in\Sigma_s$ such that A ::= A'_{ijk}, B ::= $B'_{i'j'k'}\in RR_3$, and σ(i, j,k-1)=0 or σ(i',j',k'+1)=0. If j=k=0, B must be the lower-context of A in S.



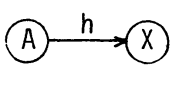
::= 

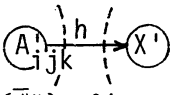
for every X$\in\Sigma_d\cup\{\overline{\#}\}$, A$\in\overline{\Sigma}\cup\{\overline{\#}\}$ and X' $\in\overline{\Sigma}_d\cup\{\overline{\#}"\}$, $A'_{ijk}\in\Sigma_s$ such that A ::= A'_{ijk}, X ::= X' $\in RR_3$ and σ(i,j-1,k)=0. If j=k=0, # must be the left-context of A in S.



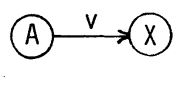
::= 

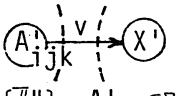
for every X$\in\overline{\Sigma}_d\cup\{\overline{\#}\}$, A$\in\overline{\Sigma}\cup\{\overline{\#}\}$ and X' $\in\overline{\Sigma}_d\cup\{\overline{\#}"\}$, $A'_{ijk}\in\Sigma_s$ such that A ::= A'_{ijk}, X ::= X' $\in RR_3$ and σ(i,j,k+1)=0.



::= 

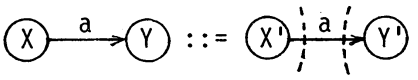
for every X$\in\overline{\Sigma}_d\cup\{\overline{\#}\}$, A$\in\overline{\Sigma}\cup\{\overline{\#}\}$ and X' $\in\overline{\Sigma}_d\cup\{\overline{\#}"\}$, $A'_{ijk}\in\Sigma_s$ such that A ::= A'_{ijk}, X ::= X' $\in RR_3$ and σ(i,j+1,k)=0.



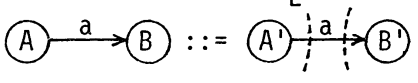
::= 

for every X$\in\overline{\Sigma}_d\cup\{\overline{\#}\}$, A$\in\overline{\Sigma}\cup\{\overline{\#}\}$ and X' $\in\overline{\Sigma}_d\cup\{\overline{\#}"\}$, $A'_{ijk}\in\Sigma_s$ such that A ::= A'_{ijk}, X ::= X' $\in RR_3$ and σ(i,j,k-1)=0. If j=k=0, # must be the lower-context of A in S.

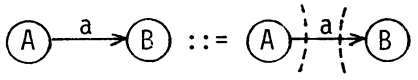
Figure 3.1 (continued).



for every $X,Y\in\bar{\Sigma}_d\cup\{\bar{\#}\}$, $X',Y'\in\bar{\Sigma}_d\cup\{\#''\}$, and $a\in\Sigma_E$.

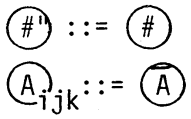


for every $A,B\in\bar{\Sigma}\cup\{\bar{\#}\}\cup\bar{\Sigma}_d$, $a\in\Sigma_E$ and $A',B'\in\Sigma_s\cup\bar{\Sigma}_d\cup\{\#''\}$ such that A' and B' don't satisfy the conditions which are stated in above connection rules.



for every $A,B\notin\bar{\Sigma}\cup\{\bar{\#}\}$, $a\in\Sigma_E$.

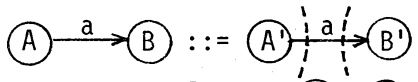
RR_4



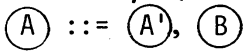
for every $A_{ijk}\in\Sigma_s$.

for every $A\notin\Sigma_s\cup\{\#''\}$.

CR_4



for every $A,B\in\Sigma_V$ and $a\in\Sigma_E$ such that



$\in RR_4$.

Figure 3.1 (continued).

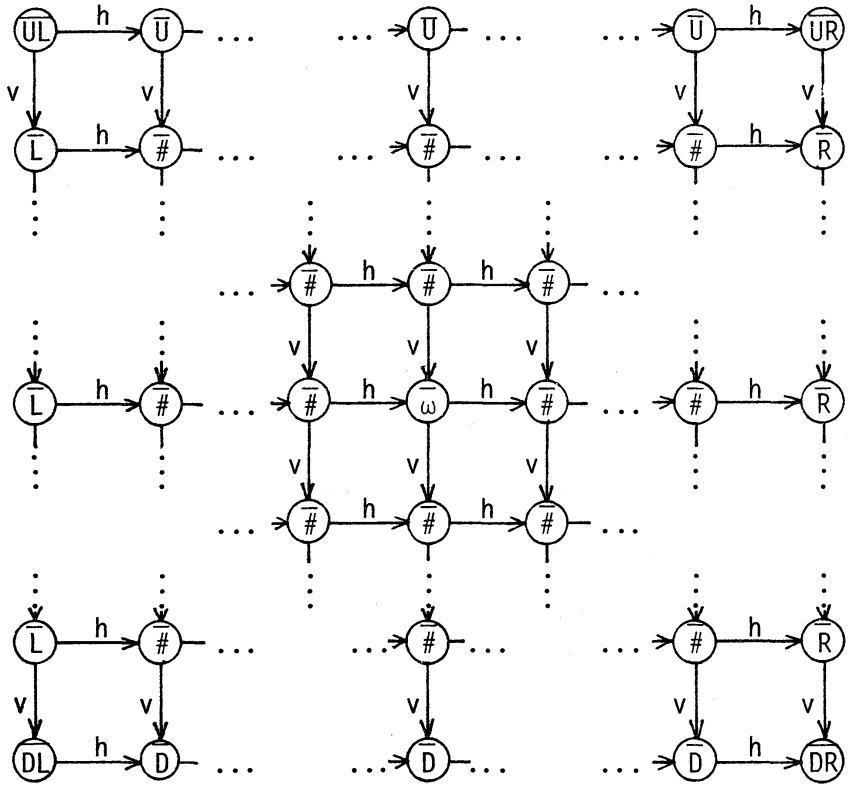
In above construction of G, if we construct table (RR_5,CR_5) such that G replaces all nodes having labels from $\bar{\Sigma}_d\cup\{\bar{\#}\}$ by empty graph, then we have G' which is equivalent to S in the sense of Stipulation 3.1. Therefore we get next lemma immediately.

Lemma 3.2. For every IEA$<\begin{matrix}1\\1\end{matrix}\ \begin{matrix}0\\0\end{matrix}>L^2$-system S there exists an ETnGL-system G' such that $\mathcal{L}(G')\underset{1}{=}\mathcal{L}(S)$.

From Lemmas 3.1, 3.2, and Theorem 2.1, we get the following theorem.

Theorem 3.1. The class of extended tabled node-replacement graph L-systems contains the class of isometric array grammars.

Figure 3.2. A square pattern constructed by G



4. Conclusion

In section 3, we have shown that the class of ETnGL-systems contains the class of isometric array grammars. In an ETnGL-system, each node is replaced in a context-free style, but the connection rules can bring some contexts into this replacement. This can be considered to be a main factor of our results.

In this paper, we considered only ETnGL-systems. However, as in the string case, we conjecture that the class of ETnGL-systems is the same as the class of EnGL-systems. So it seems that our results can be extended to the case of EnGL-systems.

References.

[1] Rajasethupathy,K.S.: Studies on L-systems, Ph.D. Thesis, Univ. of Bombay, 1980.

[2] Nakamura,A.: On Isometric Array L-systems, Technical Report No.59, Tata Institute of Fundamental Research, Bombay, 1980.

[3] Nagl,M.: A Tutorial and Bibliographical Survey on Graph Grammars, in Lecture Notes in Computer Science 73, Springer-Verlag, 1979.

[4] Grotsch,E. and Nagl,M : Explicit versus Implicit Parallel Rewriting on Graphs, in Lecture Notes in Computer Science 73, Springer-Verlag, 1979.

[5] Herman,G.T. and Rozenberg,G.: Developmental Systems and Languages, North-Holland, Amsterdam, 1975.

[6] Rosenfeld,A.: Picture Languages, Academic Press, New-York, 1979.