Learning Very Simple Matrix Grammar

M. Iffath Mubeen
Department of Mathematics
Government Arts College for Men (Auto)
Chennai-35, Tamil Nadu, India

J. D. Emerald Princess Sheela
Department of Mathematics
Queen Mary's College (Auto)
Chennai-4, Tamil Nadu, India

D. G. Thomas
Department of Mathematics
Madras Christian College (Auto)
Chennai-59, Tamil Nadu, India

ABSTRACT

A linguistic model to generate matrices (arrays of terminals) to recognize pictures was introduced by Rani Siromoney [1]. Yokomori introduced very simple grammars and studied the problem of identifying the class in the limit from positive data [2]. Here a new grammar called very simple matrix grammar is introduced and shown that this class is polynomial time identifiable in the limit from positive data.

General Terms

Matrix grammar and languages, context-free grammar, Greibach Normal Form, simple deterministic.

Keywords

Very simple matrix grammar and language, a-handle rule, positive presentation, inference from positive data, characteristic sample, schema representation.

1. INTRODUCTION

The study of syntactic methods of describing pictures considered as connected, digitized finite arrays in a two dimensional plane have been of great interest. Picture languages generated by array grammars or recognized by array automata have been advocated since the 1970s for problems arising in the frame work of pattern recognition and image processing.

A digitized picture is a finite rectangular array of points or elements each of which is associated with it one of a discrete finite set of values. Thus a picture can be represented as a m x n matrix in which each entry a_{ij} $1\leq i\leq m$, $1\leq j\leq n$ has one of the values, say $v_1,v_2,,..v_k$ [3].

A linguistic model for the generation of matrices (rectangular arrays of terminals) by the substitution of regular sets into well known families of formal languages has been proposed in [4]. Some interesting classes of pictures including certain letters of the alphabet, kolam, (traditional picture patterns used to decorate the floor in south Indian homes) and wall paper designs (repetitive patterns) can be generated by certain grammars

In this paper, we define very simple matrix languages and study how they are consistent with positive data by identifying a ground interpretation for them and show how they are polynomial time identifiable in the limit just as the class of very simple grammars which includes only context free languages. Simple deterministic languages have been defined with respect to Automata for regular languages and learning has been done [5]. Here we are considering context-free matrix grammar and languages. And we introduce a very simple matrix grammar and language; study its properties and learning.

2. BASIC DEFINITIONS

Let $\sum$ be a finite alphabet and $\sum^*$ be the set of all finite length strings over $\sum$. Further, let $\sum^+ = \sum^* - \{\lambda\}$, where λ is the null string. By len (u) we denote the length of the string u. A language over $\sum$ is a subset of $\sum^*$. For a string w in $\sum^*$, alph (w) denotes the set of terminal symbols appearing in w. For a language L, alph(L) = $\bigcup_{w\in L}$ alph(w).

Definition 2.1 Let $\sum$ be an alphabet set-a finite non empty set of symbols. A *matrix* (or an image) over $\sum$ is an mxn rectangular array of symbols from $\sum$ where m, n ≥ 0. The set of all matrices over $\sum$ (including $\wedge$) is denoted by $\sum^{**}$ and $\sum^{++}=\sum^{**}-\{\wedge\}$, where $\wedge$ is the empty image.

Definition 2.2 R (M) and C (M) respectively denote the number of rows and columns of a given matrix M.

Definition 2.3 Let $\sum^*$denote the set of horizontal sequences of letters from $\sum$ and $\sum^+=\sum^*-\{\varepsilon\}$, where ε is the identity element (of length zero). $\sum_*$ denotes the set of all vertical sequences of letters over $\sum$, and $\sum_+=\sum_*-\{\varepsilon\}$. Length of the given string s is denoted by |s|. Precisely, if s $\in \sum^+$ then |s| =C(s) and if s $\in \sum_+$ then |s| =R(s).

Definition 2.4 We use the operators Θ for row concatenation and ϕ for column concatenation for arrays. If

$$X=\begin{array}{ccc} a_{11} & \ldots & a_{1n} \\ \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots \\ a_{m1} & \ldots & a_{mn} \end{array} \quad \text{and} \quad Y=\begin{array}{ccc} b_{11} & \ldots & b_{1n'} \\ \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots \\ b_{m'1} & \ldots & b_{m'n'} \end{array}$$

X Θ Y is defined only when at least one of them is $\wedge$ or n=n' and is given by

$$X\,\Theta\,Y=\begin{array}{ccc} a_{11} & \ldots & a_{1n} \\ \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots \\ a_{m1} & \ldots & a_{mn} \\ b_{11} & \ldots & b_{1n'} \\ \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots \\ b_{m'1} & \ldots & b_{m'n} \end{array}$$

X ϕ Y is defined only when at least one of them is $\wedge$ or m=m' and is given by

$$X\,\phi\,Y=\begin{array}{cccccc} a_{11} & \ldots & a_{1n} & b_{11} & \ldots & b_{1n'} \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ a_{m1} & \ldots & a_{mn} & b_{m'1} & \ldots & b_{mn'} \end{array}$$

Definition 2.5 Let x be a matrix (or an image) defined over $\sum$ then $(x)^{i+1} = (x)^i\,\phi\,x$ and $(x)_{i+1} = (x)_i\,\Theta\,x$, $i\geq 1$.

Definition 2.6 Let us define a mapping χ as follows: $\sum^+\rightarrow\sum_+$

For any string s= $a_1 a_2 \ldots a_n \in \Sigma^+$

$$\chi(s) = \begin{matrix} a_1 \\ a_2 \\ . \\ . \\ . \\ a_n \end{matrix}$$

i.e. $\chi(s) = a_1 \Theta a_2 \Theta \ldots \Theta a_n$

Definition 2.7 A matrix (or an image) is defined as follows:

Let $c_1, c_2, \ldots, c_n \in \Sigma^+$ be strings of same length.

We write I = $c_1 \Theta c_2 \Theta \ldots \Theta c_n$ is the matrix (or an image) represented by the image

$\chi(c_1) \Phi \chi(c_2) \Phi \chi(c_3) \ldots \Phi \chi(c_n)$

Example 2.1

If c_1= abc, c_2= efg, c_3= ijk then

I = $c_1 \Theta c_2 \Theta c_3 = \chi(c_1) \Phi \chi(c_2) \Phi \chi(c_3)$ is the image

$$\begin{matrix} a & e & i \\ b & f & j \\ c & g & k \end{matrix}$$

We now recall the notions of matrix grammar [1] and very simple grammar [2]

Definition 2.8 Let G = (V_N, ∑, P, S) be a context-free grammar (CFG) in Greibach Normal Form (GNF), i.e., each rule of P is of the form $A \rightarrow a\alpha$, where $A \in V_N$, $a \in \sum$, $\alpha \in V_N^*$.

For each terminal symbol $a \in \sum$, a rule whose right hand side is of the form aα, (where $\alpha \in V_N^*$) is called an ***a-handle rule***.

Then G is said to be ***Very Simple*** iff for each a in ∑, there exists exactly one a-handle rule in P.

A language L is said to be ***Very Simple*** iff there exists a ***Very Simple*** CFG G such that L = L(G) holds. (Note that since every simple grammar is λ - free, so is every simple language).

Example 2.2 Let Σ = {a, b, c, d}. Consider a CFG G = ({S, A, B}, Σ, P, S), where P consists of the following: S → aAB, A → aA, B → bB, A → c, B → d. The grammar G is ***Very Simple*** and L (G) = $\{a^m c\, b^n d \,/\, m, n \geq 0\}$.

Definition 2.9 (Matrix Grammars) A ***Phrase Structure Matrix Grammar(PSMG)***, ***Context Sensitive Matrix Grammar(CSMG), Context Free Matrix Grammar(CFMG), Right Linear Matrix Grammar(RLMG) is a two tuple G=(G,G´), where G=(V, I, P, S) is a Phrase Structure Grammar(PSG), Context Sensitive Grammar(CSG), Context Free Grammar(CFG), Right Linear Grammar(RLS)***, with V= finite set of horizontal non-terminals , I= a finite set of intermediates=($S_1, S_2, \ldots S_k$), P= a finite set of PSG(CSG, CFG, RLG) production rules called horizontal production rules and S is the start symbol. $S \in V$, and $V \bigcap I = \phi$.

$G_i' = \bigcup_{i=1}^{k} G_i'$ where G_i' =(V_i, T_i, P_i, S_i), i=1, 2…k are ***Right Linear Grammars*** with T_i= a finite set of terminals, V_i= finite set of vertical non-terminals, S_i the start symbol and P_i finite set of right linear production rules, $V_i \cap V_j = \phi$ if $i \neq j$.

Derivations are defined as follows: First a string $S_1S_2 \ldots S_n \in I$ is generated horizontally using the horizontal production rules P in G i.e. $S \Longrightarrow S_1S_2 \ldots S_n \in I$ and then vertical derivations proceed using the rules P_i of G_i' in G´

Definition 2.10 (Matrix Language*)* The set of all matrices generated by M is defined to be L(M) = {*m* x *n* arrays [a_{ij}], i=1…, *m*, j=1…, *n*, *m*, *n* ≥ 1/ $S_1 \ldots S_n \overset{*}{\Downarrow} [a_{ij}]$}

L(M) is called a **Phrase-Structure Matrix Language (PSML)** (**Context-Sensitive Matrix Language (CSML), Context-Free Matrix Language (CFML), Regular Matrix Language (RML)**) if G is a **(PSMG, CSMG, CFMG, RLMG)**.

Derivation trees for CFML and RML can be defined similar to derivation trees for a context -free language. Chomskian hierarchy can be extended to matrices and it can be established that the family of RML $\underset{\neq}{\subset}$ the family of CFML $\underset{\neq}{\subset}$ the family of CSML $\underset{\neq}{\subset}$ the family of PSML

3. VERY SIMPLE CONTEXT-FREE MATRIX GRAMMAR

Definition 3.1 A matrix grammar M= (G, G´) is said to be a context-free matrix grammar i.e. (CF: CF) matrix grammar if G is a context-free grammar G= (V, I, P, S) where I= {S_1, S_2,…S_n} and each G´ = (G_1', G_2',… G_k') where each G_i'= {V_i, Ti, Pi, Si} are length equivalent context- free grammars if there exists strings $\alpha_1\alpha_2\ldots\alpha_k$ such that $\alpha_i \in L(G_i')$, then $|\alpha_1| = |\alpha_2| = \ldots = |\alpha_k|$, $1 \leq i \leq k$

Let I = $c_1 \Theta c_2 \Theta \ldots \Theta c_n$ be an image defined over Σ. I ∈ M(G) iff there exists $S_1, S_2 \ldots S_n \in L(G)$ such that $c_j \in L(G_j)$, $1 \leq j \leq n$. The string $S_1S_2 \ldots S_n$ is said to be an intermediate string deriving I with respect to M. Note that there can be more than one intermediate string deriving I. The family of languages generated by (X:Y) MG is denoted as (X:Y) ML where X,Y ∈ {CF,R}.

Definition 3.2 A context-free matrix grammar M= (G, G´) is said to be a very simple matrix grammar if it satisfies the following properties

i) The context free grammars G and G_i´s in G´ are all in Greibach Normal Form in the strict sense, that is each rule in P and Pi's are of the form $A \rightarrow a\alpha$ where $A \in I$ or V_i, $a \in I$ or T_i and $\alpha \in V^*$ or V_i^* and no right hand side of the rules contains the starting non-terminal.

ii) For each intermediate symbol S_i in G there exists exactly one Si-handle rule in G.

iii) For each $a_i \in T_i$ in G_i', there exists exactly one a_i - handle rule in P_i.

iv) For each $S_i \rightarrow a_i\alpha_i$ or $A_i \rightarrow a\alpha_i$ in each G_i' where $S_i, A_i \in V_i$, $a \in T_i$ and $\alpha_i \in V_i^*$, all α_i's are of same length. That is if $A_1 \rightarrow a_1\alpha_1$, α_1 in G_1', $A_2 \rightarrow a_2\alpha_2$, α_2 in G_2', and $A_3 \rightarrow a_3\alpha_3$, α_3 in G_3' then $|\alpha_1| = |\alpha_2| = |\alpha_3|$

Example 3.1 Consider a very simple matrix grammar M= (G, G') where G=

$$\left(\{S, B, D\}, \{S_1, S_2, S_3, S_4\}, \left\{ \begin{matrix} S \rightarrow S_1B, B \rightarrow S_2BD, \\ D \rightarrow S_3B, B \rightarrow S_4 \end{matrix} \right\}, S \right)$$

$G' = G_1' \bigcup G_2' \bigcup G_3' \bigcup G_4'$ where

$G_i' = (V_i, T_i, P_i, S_i)$ where

$V_i = \{S_i, B_i, C_i, D_i, E_i, F_i, G_i\}$, $T_i = \{a_i, b_i, c_i, d_i, e_i, f_i, g_i\}$ $P_i = \begin{bmatrix} S_i \to a_i B_i C_i \\ B_i \to b_i \\ C_i \to c_i D_i \\ D_i \to d_i \\ E_i \to e_i C_i \\ C_i \to f_i G_i \\ G_i \to g_i \end{bmatrix}$

for i =1, 2, 3, 4

Then $S \to S_1B \Longrightarrow^{*}$ $\boxed{S_1\, S_2\, S_4\, S_3 S_4}$

$$\Downarrow$$

$$\boxed{\begin{array}{ccccc} a_1 & a_2 & a_4 & a_3 & a_4 \\ B_1 & B_2 & B_4 & B_3 & B_4 \\ C_1 & C_2 & C_4 & C_3 & C_4 \end{array}}$$

$$\Downarrow$$

$$\boxed{\begin{array}{ccccc} a_1 & a_2 & a_4 & a_3 & a_4 \\ b_1 & b_2 & b_4 & b_3 & b_4 \\ .. & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ G_1 & G_2 & G_4 & G_3 & G_4 \end{array}}$$

$$\Downarrow$$

$$\boxed{\begin{array}{ccccc} a_1 & a_2 & a_4 & a_3 & a_4 \\ b_1 & b_2 & b_4 & b_3 & b_4 \\ c_1 & c_2 & c_4 & c_3 & c_4 \\ d_1 & d_2 & d_4 & d_3 & d_4 \\ e_1 & e_2 & e_4 & e_3 & e_4 \\ f_1 & f_2 & f_4 & f_3 & f_4 \\ g_1 & g_2 & g_4 & g_3 & g_4 \end{array}}$$

In the above example the set of all Matrices generated by M is $L(M)=\{S_1[S_2^n(S_4S_3)^m]^kS_4 \,/ n,k \geq 0 \text{ and } 0 \leq m \leq n+1\}$

Definition 3.3 A positive presentation of a language L is an infinite sequence of strings $M_1, M_2, \ldots$ such that

$\{M \mid M=M_i \text{ for some } i\} = L$

Definition 3.4 A class of languages $L= \{L_1, L_2 \ldots\}$ is said to be inferable from positive data if there exists an Identification Algorithm IA such that M on input σ converges to L with

$L_j = L_i$ for any index i and any positive presentation σ on L_i

Lemma 3.1 Let L be a very simple matrix language. Then for each matrix $[a_{ij}]$ in L i=1, 2..m,j=1,2..n, n ≥2 the symbols of first and last columns must be different.

Example 3.2 i) For the rule in P: $S \to S_1\, AB$, $A \to S_2S_2$, $B \to S_1$, we get $\{S_1S_2S_2S_1\}$ as it is not a very simple matrix language.

ii) $\{S_1^n\}$ is not a very simple matrix language as $S \to S_1S$, $S \to S_1$ gives no unique S_1-handle rule

iii) $\{S_1^nS_2\, S_1^m / \, m,n \geq 0\}$ is not a very simple matrix language as $S \to S_1\, SA$, $S \to S_2$, $A \to S_1S_1$ gives no unique S_1-handle rule

Closure properties

Theorem 3.1 The class of very simple matrix languages is closed under none of the following: union, concatenation, intersection, complement, kleene closure (+,*), (λ-free) homomorphism, inverse homomorphism or reversal.

Proof (Union) Consider the very simple matrix languages

$L_1 = \begin{bmatrix} a_1 & a_2 \\ b_1 & b_2 \\ c_1 & c_2 \\ d_1 & d_2 \\ e_1 & e_2 \\ f_1 & f_2 \\ g_1 & g_2 \end{bmatrix}$ and $L_2 = \begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ d_1 \\ e_1 \\ f_1 \\ g_1 \end{bmatrix}$ then

$L_1 \bigcup L_2 = \left\{ \begin{bmatrix} a_1 & a_2 \\ b_1 & b_2 \\ c_1 & c_2 \\ d_1 & d_2 \\ e_1 & e_2 \\ f_1 & f_2 \\ g_1 & g_2 \end{bmatrix}, \begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ d_1 \\ e_1 \\ f_1 \\ g_1 \end{bmatrix} \right\}$ is not a very simple matrix language as $S \to S_1\, S_2$, $S \to S_1$ implies S_1–handle rule is not unique

Concatenation Consider a very simple matrix language

$L_3 = \begin{bmatrix} a_2 & a_1 \\ b_2 & b_1 \\ c_2 & c_1 \\ d_2 & d_1 \\ e_2 & e_1 \\ f_2 & f_1 \\ g_2 & g_1 \end{bmatrix}$ then L_1L_3 is not a very simple matrix language as the rules which generate L_1L_3

i.e., $S \to S_1Y\, S_2\, X$, $X \to S_1$, $Y \to S_2$ implies that the S_1–handle rule is not unique

Intersection Let $L_4 =$

$$\left\{ \begin{bmatrix} a_1 & a_2^m & a_3 & a_4^n & a_5 & a_6^n & a_7 \\ b_1 & b_2^m & b_3 & b_4^n & b_5 & b_6^n & b_7 \\ c_1 & c_2^m & c_3 & c_4^n & c_5 & c_6^n & c_7 \\ d_1 & d_2^m & d_3 & d_4^n & d_5 & d_6^n & d_7 \\ e_1 & e_2^m & e_3 & e_4^n & e_5 & e_6^n & e_7 \\ f_1 & f_2^m & f_3 & f_4^n & f_5 & f_6^n & f_7 \\ g_1 & g_2^m & g_3 & g_4^n & g_5 & g_6^n & g_7 \end{bmatrix} / \, m, n \geq 0 \right\}$$ and

$$L_5 = \left\{ \begin{bmatrix} a_1 & a_2^n & a_3 & a_4^m & a_5 & a_6^m & a_7 \\ b_1 & b_2^n & b_3 & b_4^m & b_5 & b_6^m & b_7 \\ c_1 & c_2^n & c_3 & c_4^m & c_5 & c_6^m & c_7 \\ d_1 & d_2^n & d_3 & d_4^m & d_5 & d_6^m & d_7 \\ e_1 & e_2^n & e_3 & e_4^m & e_5 & e_6^m & e_7 \\ f_1 & f_2^n & f_3 & f_4^m & f_5 & f_6^m & f_7 \\ g_1 & g_2^n & g_3 & g_4^m & g_5 & g_6^m & g_7 \end{bmatrix} / \, m, n \geq 0 \right\}$$

then

$$L_4 \bigcap L_5 = \left\{ \begin{bmatrix} a_1 & a_2^m & a_3 & a_4^m & a_5 & a_6^m & a_7 \\ b_1 & b_2^m & b_3 & b_4^m & b_5 & b_6^m & b_7 \\ c_1 & c_2^m & c_3 & c_4^m & c_5 & c_6^m & c_7 \\ d_1 & d_2^m & d_3 & d_4^m & d_5 & d_6^m & d_7 \\ e_1 & e_2^m & e_3 & e_4^m & e_5 & e_6^m & e_7 \\ f_1 & f_2^m & f_3 & f_4^m & f_5 & f_6^m & f_7 \\ g_1 & g_2^m & g_3 & g_4^m & g_5 & g_6^m & g_7 \end{bmatrix} / \, m \geq 0 \right\}$$

is not-context free.

Complement Let $L_6 = \begin{bmatrix} a_2 \\ b_2 \\ c_2 \\ d_2 \\ e_2 \\ f_2 \\ g_2 \end{bmatrix}$ a very simple matrix language over Σ, while its complement $L_6^c = \Sigma^* - L_6$ is not a very simple matrix language as the rules i.e., $S \rightarrow S_1 S$, $S \rightarrow S_1$ which generates L_6^c implies S_1 –handle rule is not unique

Kleene Closure Consider again L_6, then L_6^*(or L_6^+) is not a very simple matrix language as the rules $S \rightarrow S_1 S$, $S \rightarrow S_1$ which generate L_6^* implies S_1 –handle rule is not unique

Homomorphism Consider a very simple matrix language

$$L_7 = \left\{ \begin{bmatrix} a_1^n & a_2 \\ b_1^n & b_2 \\ c_1^n & c_2 \\ d_1^n & d_2 \\ e_1^n & e_2 \\ f_1^n & f_2 \\ g_1^n & g_2 \end{bmatrix} / \, n \geq 0 \right\}$$ and a homomorphism defined

by $h_1\left(\begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ d_1 \\ e_1 \\ f_1 \\ g_1 \end{bmatrix}\right) = \begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ d_1 \\ e_1 \\ f_1 \\ g_1 \end{bmatrix}$ and $h_1\left(\begin{bmatrix} a_2 \\ b_2 \\ c_2 \\ d_2 \\ e_2 \\ f_2 \\ g_2 \end{bmatrix}\right) = \Lambda$

Then $h_1(L_7) = \left\{ \begin{bmatrix} a_1^n \\ b_1^n \\ c_1^n \\ d_1^n \\ e_1^n \\ f_1^n \\ g_1^n \end{bmatrix} / \, n \geq 0 \right\}$ is not a very simple matrix language as $S \rightarrow S_1 S$, $S \rightarrow S_1$ implies S_1 –handle rule is not unique

Inverse Homomorphism For a very simple matrix language $L_6 = \begin{bmatrix} a_2 \\ b_2 \\ c_2 \\ d_2 \\ e_2 \\ f_2 \\ g_2 \end{bmatrix}$, consider a homomorphism h defined by

$h\left(\begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ d_1 \\ e_1 \\ f_1 \\ g_1 \end{bmatrix}\right) = \begin{bmatrix} a_2 \\ b_2 \\ c_2 \\ d_2 \\ e_2 \\ f_2 \\ g_2 \end{bmatrix}$ and $h\left(\begin{bmatrix} a_3 \\ b_3 \\ c_3 \\ d_3 \\ e_3 \\ f_3 \\ g_3 \end{bmatrix}\right) = \Lambda$.

Then $h^{-1}(L_6) = \left\{ \begin{bmatrix} a_3^m & a_1 & a_3^n \\ b_3^m & b_1 & b_3^n \\ c_3^m & c_1 & c_3^n \\ d_3^m & d_1 & d_3^n \\ e_3^m & e_1 & e_3^n \\ f_3^m & f_1 & f_3^n \\ g_3^m & g_1 & g_3^n \end{bmatrix} / \, m, n \geq 0 \right\}$

is not very simple matrix language as
$S \rightarrow X$, $X \rightarrow S_3 X$, $X \rightarrow S_1$, $X \rightarrow S_1 Y$, $Y \rightarrow S_3$ implies
S_1 –handle rule is not unique

Reversal Consider a very simple matrix language L_7 then its reversal L_7^R is not a very simple matrix language as the rules $S \rightarrow S_2 X$, $X \rightarrow S_1 X$, $X \rightarrow S_1$ which generates L_7^R

implies that S_1 –handle rule is not unique

4. LEARNING VERY SIMPLE MATRIX GRAMMAR AND LANGUAGE

We extend the algorithm given in [1] and use the schema representation method to learn the class of very simple matrix language. For the purpose of learning, we consider the following string representation of each column of the matrix using a mapping χ.

Definition 4.1 Let x be a matrix (or an image) defined over Σ, then $(x)^{i+1}=(x)^i \Phi x$ and $(x)_{i+1} = (x)_i \Theta x$, $i \geq 1$

A mapping χ is defined as follows:

$\chi: \Sigma^+ \rightarrow \Sigma_+$ such that for any string $s = a_1a_2\ldots.a_n \in \Sigma^+$

Let $I = c_1 \Theta c_2 \Theta \ldots. \Theta c_n$ be the image defined over Σ_+ $I \in L(M)$ if and only if there exists $S_1S_2\ldots S_n \in L(G)$ such that $c_j \in L(G_i')$, $1 \leq i \leq n$. The string $S_1S_2\ldots S_n$ is said to be an intermediate string deriving I with respect to M. Note that there can be more than one intermediate string deriving I.

In this section, we consider the following problem for very simple matrix grammars. Suppose that we are given a finite set of M arrays $[a_{ij}]$ $i=1\ldots m, j=1\ldots n$ from an unknown very simple matrix language L(M) for some very simple matrix grammar M=(G,G'), the algorithm identifies a ground interpretation I such that I(G) is consistent with M.

Identification algorithm for very simple matrix grammars

Algorithm VSMG

Input: A positive presentation of very simple matrix language $L(M)= G \bigcup G'$

Output: A sequence of a set of context -free grammars for the horizontal grammar G over intermediate symbols and the grammar G´ for the vertical columns.

Procedure

Initialize grammar $G= (\{p_0\}, \Phi, \Phi, p_0,\Phi)$

Initialize the set $H =\Phi$

Initialize the sets $T_1, T_2,\ldots T_k = \Phi$
/* Each positive presentation
$M_i \in M= c_1 \Theta c_2 \Theta \ldots. \Theta c_n$ $(1\leq n\leq k)$
$=\chi(c_1)\Phi\chi(c_2)\Phi\chi(c_3)\ldots.\Phi\chi(c_n)$
Where $c_1, c_2, \ldots. c_n \in \Sigma^+$ */

Step 1 For the first matrix of the sample i.e., M_1 do

1a. Assign a non terminal S_j $(1\leq j\leq k)$ to each different $\chi(c_n)$ $(1\leq n\leq k)$ and store the string of the sequence of S_j's corresponding to the sequence $\chi(c_1)\,\chi(c_2)\,\chi(c_3)\ldots.\,\chi(c_n)$ in a set H.

1b. If $S_j(1\leq j\leq k)$ is the non terminal associated to a column $\chi(c_n)$ $(1\leq n\leq k)$, put the string $c_n(1\leq n\leq k)$ in the set T_j $(1\leq j\leq k)$

Step 2 For the other matrices of the sample i.e.,
$Mi = c_1 \Theta c_2 \Theta \ldots. \Theta c_n$ $(1\leq n\leq k)$
$i \geq 2$, do

If the string c_1 corresponding to χ (c_1) has a common prefix and a common suffix with a string c_n $(1\leq n\leq k)$ in some Tj $(1\leq j\leq k)$, include the string c_1 in T_j ie $T_j = T_j \bigcup \{c_1\}$ and non terminal S_j $(1\leq j\leq k)$ to χ (c_1). Repeat the same for $c_2, c_3, \ldots, c_n$ and include the string of the sequence of Sj's corresponding to $\chi(c_1)\,\chi(c_2)\,\chi(c_3)\ldots.\,\chi(c_n)$ in the set H.

Step 3 Using the identification algorithm IA given below obtain the grammar G for the input set H, which accepts the context - free matrix grammar G

Step 4 Using the same identification algorithm IA repeatedly we obtain the grammar $G_1', G_2'\ldots G_k'$ corresponding to $T_1, T_2, \ldots, T_k$.

Lemma 4.1 For a finite subset R of L(G*), let $I= (fn,fp) =$ Consistent(R) and $G =I(G_0)$. Then either $L(G^*) = L(G)$ or $L(G^*)-L(G) \neq \phi$

Lemma 4.2 Let $G_{RO},G_{R1}\ldots,G_{RI},\ldots$be the intermediate (horizontal) sequence of conjectured grammars produced by IA, where $G_{RI}=I_1(G_0)$. Then there exists $r\geq 0$ such that for all $i\geq 0$, produced by IA, where $G_{Rr} =Ii(G_{0,\Sigma})$. Then there exists $r \geq 0$ such that for all $i \geq 0$, $G_{Rr}= G_{Rr+1}$ and $L(G_{Rr}) = L(G_*)$ and similarly let $G_{R0}', G_{R1}',\ldots G_{Ri}'\ldots$ be the vertical sequence of conjectured grammars produced by IA, where $G_{Rr}'=I_i(G_{0,\Sigma})$. Then there exists $r \geq 0$ su ch that for all $I \geq 0$, $G_{Rr}' = G_{Rr+1}'$ and $L(G_{Rr}') = L(G_*)$ where $G_{0,\Sigma} = (\{S\} \cup V_{N,\Sigma},\Sigma,P_\Sigma,S)$ where $V_{N,\Sigma} =\{ X_a /a \in \Sigma\}$ and $P_\Sigma = \{X_a \rightarrow ax_a /a \in \Sigma\}$

Proof From the property of IA, in particular, of the procedure consistent(R), there is an upper bound B (depending on the size of G*) for which each $i\geq 1$, the number of candidate interpretations Ii (=consistent (Ri)) for the i-th conjecture G_{Ri} is no more than B. (Note that for each $i\geq 1$, and interpretation I* for G* is potentially included in the set of those candidate interpretations Ii). Thus there exists $r \geq 0$ such that for all $i\geq 0$, $G_{Rr} = G_{Rr+1}$. Suppose that $L(G^*) \neq L(G_{Rr})$, then by the preceding lemma there exists a string $w \in L(G^*) - L(G_{Rr})$ such that w is not yet provided as a positive example. This implies that IA produces a conjecture distinct from G_{Rr}, a contradiction

Thus we have the following

Theorem 4.1 The class of very simple grammars is identifiable in the limit from positive data.

Identification Algorithm IA
Input: A positive presentation of a very simple matrix language L (G_*)
Output: A sequence of very simple grammars G_{R0}, G_{R1}……
Procedure

Initialize $R_0 =\acute{\emptyset}$;
Initialize the grammar schema $G_{0,\acute{\emptyset}}$;
Let $G_{R0} = (\{S\}, \acute{\emptyset}, \acute{\emptyset}, S)$;
Let i=1;
Repeat (forever)
Read the next positive example w_i;
Let $R_i = R_{i-1} \bigcup \{w_i\}$
Let $alph(R_1) = alph(R_{i-1}) \bigcup \{alph(w_i)\}$;
If $w_i \in L(G_{Ri-1})$, then let $G_{Ri} = G_{Ri-1}$;
Output G_{Ri};
Else
Augment $G_{0,\Sigma}$ using $\Sigma = alph(R_i)$;
Let I_i = Consistent (Ri);
Output $G_{Ri} = I_i (G_{0,\Sigma})$;

Lemma 4.3 Given any very simple matrix grammar G_*, the algorithm IA identifies in the limit a very simple matrix grammar G_R such that $L(G_*) = L(G_R)$, where R is the set of positive data provided

Thus we have the following

Theorem 4.2 The class of very simple matrix grammar is identifiable in the limit from positive data.

Constructing a characteristic sample

Let L be a very simple matrix language. A finite subset S_M of L is called a characteristic sample of L if and only if L is the smallest very simple matrix language containing S_M such that no rule can be applied more than twice.

Time complexity analysis

Time for updating a conjecture:

Let $N=\Sigma_{W\in R} len(W_j)$ and l be the maximum length of positive data in R. The time for updating a conjecture is obviously dominated by the time for the procedure consistent (R) where $I = (f_n, f_p)$ = consistent(R).

In performing consistent(R) determining f_n requires atmost O(N) times. It takes atmost O(N) times to construct Lg(R). Solving Lg(R) requires atmost $O(|\Sigma|^3)$ time, because it is reduced to the computation of an inverse matrix with atmost $|\Sigma|$ dimension. We have for any a Σ, $-1\le n_a \le B(a,w)$ (where $B(a,w) = len(w)/\#_a(w)$ and w is a string of minimum length in R_a). we see that the value of n_a is bounded by l, the number of all solution vectors of Lg(R) is bounded by $l^{|\Sigma|}$. Hence while loops are repeatedly performed atmost $l^{|\Sigma|}$ times. Each while loop requires O (N) times atmost. Thus the time for updating a conjecture is bounded by

$O(|\Sigma|^3) + O(l^{|\Sigma|}) * N \le O(Max\{N^{|\Sigma|+1}.|\Sigma|^3\})$

The above is repeated for each column. Therefore repeat n times if there are n columns

$O(|\Sigma|^3) + O(l^{|\Sigma|}) * N \le O(Max\{N^{|\Sigma|+1}.|\Sigma|^3\})^n$

Example Run

Consider the very simple matrix grammar given in Example 3.1.

Step1 Let $M_1 = \begin{bmatrix} a_1 & a_2 & a_4 & a_3 & a_4 \\ b_1 & b_2 & b_4 & b_3 & b_4 \\ c_1 & c_2 & c_4 & c_3 & c_4 \\ d_1 & d_2 & d_4 & d_3 & d_4 \\ e_1 & e_2 & e_4 & e_3 & e_4 \\ f_1 & f_2 & f_4 & f_3 & f_4 \\ g_1 & g_2 & g_4 & g_3 & g_4 \end{bmatrix}$

a. $\chi(c_1) = a_1 b_1 c_1 d_1 e_1 f_1 g_1 - S_1$
$\chi(c_2) = a_2 b_2 c_2 d_2 e_2 f_2 g_2 - S_2$
$\chi(c_3) = a_3 b_3 c_3 d_3 e_3 f_3 g_3 - S_3$
$\chi(c_4) = a_4 b_4 c_4 d_4 e_4 f_4 g_4 - S_4$
$H = \{S_1 S_2 S_4 S_3 S_4\}$

b. $T_1 = \{a_1 b_1 c_1 d_1 e_1 f_1 g_1\}$
$T_2 = \{a_2 b_2 c_2 d_2 e_2 f_2 g_2\}$
$T_3 = \{a_3 b_3 c_3 d_3 e_3 f_3 g_3\}$
$T_4 = \{a_4 b_4 c_4 d_4 e_4 f_4 g_4\}$

Step 2 $M_2 = \begin{bmatrix} a_1 & a_4 \\ b_1 & b_4 \\ f_1 & f_4 \\ g_1 & g_4 \end{bmatrix}$

Now $H = \{S_1 S_2 S_4 S_3 S_4, S_1 S_4\}$
$T_1 = \{a_1 b_1 c_1 d_1 e_1 f_1 g_1, a_1 b_1 f_1 g_1\}$
$T_2 = \{a_2 b_2 c_2 d_2 e_2 f_2 g_2, a_2 b_2 f_2 g_2\}$
$T_3 = \{a_3 b_3 c_3 d_3 e_3 f_3 g_3, a_3 b_3 f_3 g_3\}$
$T_4 = \{a_4 b_4 c_4 d_4 e_4 f_4 g_4, a_4 b_4 f_4 g_4\}$

$M_3 = \begin{bmatrix} a_1 & a_2 & a_2 & a_4 & a_3 & a_4 \\ b_1 & b_2 & b_2 & b_4 & b_3 & b_4 \\ c_1 & c_2 & c_2 & c_4 & c_3 & c_4 \\ d_1 & d_2 & d_2 & d_4 & d_3 & d_4 \\ e_1 & e_2 & e_2 & e_4 & e_3 & e_4 \\ f_1 & f_2 & f_2 & f_4 & f_3 & f_4 \\ g_1 & g_2 & g_2 & g_4 & g_3 & g_4 \end{bmatrix}$

$H = \{S_1 S_2 S_4 S_3 S_4, S_1 S_4, S_1 S_2 S_2 S_4 S_3 S_4\}$
$T_1 = \{a_1 b_1 c_1 d_1 e_1 f_1 g_1, a_1 b_1 f_1 g_1\}$
$T_2 = \{a_2 b_2 c_2 d_2 e_2 f_2 g_2, a_2 b_2 f_2 g_2\}$
$T_3 = \{a_3 b_3 c_3 d_3 e_3 f_3 g_3, a_3 b_3 f_3 g_3\}$
$T_4 = \{a_4 b_4 c_4 d_4 e_4 f_4 g_4, a_4 b_4 f_4 g_4\}$

$M_4 = \begin{bmatrix} a_1 & a_2 & a_4 & a_3 & a_4 \\ b_1 & b_2 & b_4 & b_3 & b_4 \\ c_1 & c_2 & c_4 & c_3 & c_4 \\ d_1 & d_2 & d_4 & d_3 & d_4 \\ e_1 & e_2 & e_4 & e_3 & e_4 \\ c_1 & c_2 & c_4 & c_3 & c_4 \\ d_1 & d_2 & d_4 & d_3 & d_4 \\ e_1 & e_2 & e_4 & e_3 & e_4 \\ f_1 & f_2 & f_4 & f_3 & f_4 \\ g_1 & g_2 & g_4 & g_3 & g_4 \end{bmatrix}$

Again $H = \{S_1 S_2 S_4 S_3 S_4, S_1 S_4, S_1 S_2 S_2 S_4 S_3 S_4\}$
$T_1 = \{a_1 b_1 c_1 d_1 e_1 f_1 g_1, a_1 b_1 f_1 g_1, a_1 b_1 c_1 d_1 e_1 c_1 d_1 e_1 f_1 g_1\}$
$T_2 = \{a_2 b_2 c_2 d_2 e_2 f_2 g_2, a_2 b_2 f_2 g_2, a_2 b_2 c_2 d_2 e_2 c_2 d_2 e_2 f_2 g_2\}$
$T_3 = \{a_3 b_3 c_3 d_3 e_3 f_3 g_3, a_3 b_3 f_3 g_3, a_3 b_3 c_3 d_3 e_3 c_3 d_3 e_3 f_3 g_3\}$
$T_4 = \{a_4 b_4 c_4 d_4 e_4 f_4 g_4, a_4 b_4 f_4 g_4, a_4 b_4 c_4 d_4 e_4 c_4 d_4 e_4 f_4 g_4\}$

$M_5 = \begin{bmatrix} a_1 & a_2 & a_2 & a_4 & a_3 & a_4 & a_3 & a_4 \\ b_1 & b_2 & b_2 & b_4 & b_3 & b_4 & b_3 & b_4 \\ c_1 & c_2 & c_2 & c_4 & c_3 & c_4 & c_3 & c_4 \\ d_1 & d_2 & d_2 & d_4 & d_3 & d_4 & d_3 & d_4 \\ e_1 & e_2 & e_2 & e_4 & e_3 & e_4 & e_3 & e_4 \\ f_1 & f_2 & f_2 & f_4 & f_3 & f_4 & f_3 & f_4 \\ g_1 & g_2 & g_2 & g_4 & g_3 & g_4 & g_3 & g_4 \end{bmatrix}$

Again
$H = \{S_1 S_2 S_4 S_3 S_4, S_1 S_4, S_1 S_2 S_2 S_4 S_3 S_4, S_1 S_2 S_2 S_4 S_3 S_4 S_3 S_4\}$
$T_1 = \{a_1 b_1 c_1 d_1 e_1 f_1 g_1, a_1 b_1 f_1 g_1, a_1 b_1 c_1 d_1 e_1 c_1 d_1 e_1 f_1 g_1\}$
$T_2 = \{a_2 b_2 c_2 d_2 e_2 f_2 g_2, a_2 b_2 f_2 g_2, a_2 b_2 c_2 d_2 e_2 c_2 d_2 e_2 f_2 g_2\}$
$T_3 = \{a_3 b_3 c_3 d_3 e_3 f_3 g_3, a_3 b_3 f_3 g_3, a_3 b_3 c_3 d_3 e_3 c_3 d_3 e_3 f_3 g_3\}$
$T_4 = \{a_4 b_4 c_4 d_4 e_4 f_4 g_4, a_4 b_4 f_4 g_4, a_4 b_4 c_4 d_4 e_4 c_4 d_4 e_4 f_4 g_4\}$

$M_6 =$

$$\begin{bmatrix} a_1 & a_2 & a_2 & a_4 & a_3 & a_4 & a_3 & a_2 & a_4 & a_3 & a_4 \\ b_1 & b_2 & b_2 & b_4 & b_3 & b_4 & b_3 & b_2 & b_4 & b_3 & b_4 \\ c_1 & c_2 & c_2 & c_4 & c_3 & c_4 & c_3 & c_2 & c_4 & c_3 & c_4 \\ d_1 & d_2 & d_2 & d_4 & d_3 & d_4 & d_3 & d_2 & d_4 & d_3 & d_4 \\ e_1 & e_2 & e_2 & e_4 & e_3 & e_4 & e_3 & e_2 & e_4 & e_3 & e_4 \\ f_1 & f_2 & f_2 & f_4 & f_3 & f_4 & f_3 & f_2 & f_4 & f_3 & f_4 \\ g_1 & g_2 & g_2 & g_4 & g_3 & g_4 & g_3 & g_2 & g_4 & g_3 & g_4 \end{bmatrix}$$

Again
$H = \{S_1 S_2 S_4 S_3 S_4, S_1 S_4, S_1 S_2 S_2 S_4 S_3 S_4, S_1 S_2 S_2 S_4 S_3 S_4 S_3 S_4,$
$S_1 S_2 S_2 S_4 S_3 S_4 S_3 S_2 S_4 S_3 S_4\}$

$T_1 = \{a_1 b_1 c_1 d_1 e_1 f_1 g_1, a_1 b_1 f_1 g_1, a_1 b_1 c_1 d_1 e_1 c_1 d_1 e_1 f_1 g_1\}$
$T_2 = \{a_2 b_2 c_2 d_2 e_2 f_2 g_2, a_2 b_2 f_2 g_2, a_2 b_2 c_2 d_2 e_2 c_2 d_2 e_2 f_2 g_2\}$
$T_3 = \{a_3 b_3 c_3 d_3 e_3 f_3 g_3, a_3 b_3 f_3 g_3, a_3 b_3 c_3 d_3 e_3 c_3 d_3 e_3 f_3 g_3\}$
$T_4 = \{a_4 b_4 c_4 d_4 e_4 f_4 g_4, a_4 b_4 f_4 g_4, a_4 b_4 c_4 d_4 e_4 c_4 d_4 e_4 f_4 g_4\}$

Identification of G:

The computation of f_p and f_n:

1. Let $w_1 = \{S_1S_4\}$ be the first word from H and $H_1 = \{w_1\}$, alph $(H_1) = \{S_1, S_4\}$. Since S_4 is final S_4 handle rule is

$X_{S4} \rightarrow S_4$ and $f_p(X_{S4}) = \lambda$ $(n_{S1} \geq 0)$ $f_n(X_{S1}) = S$ and

$f_p(X_{S1}) \neq \lambda$

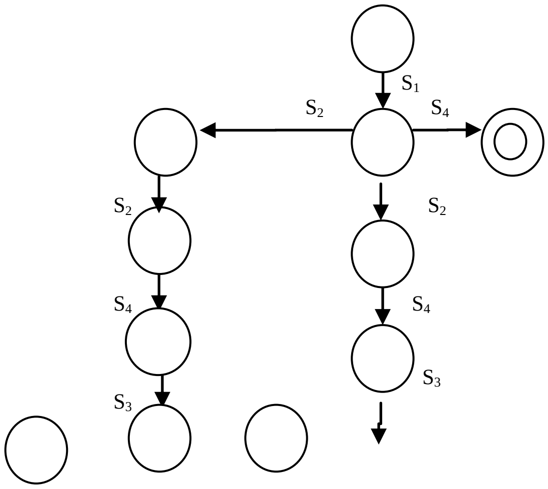
The length equation for w_1 is $n_{S1} + n_{S4} = -1$….. (l. w_1)
$n_{S1} = 0$ ….. (l. w_1)´
Lg $(H_1) = \{(l.\ w_1)\}$
Lg $(H_1´) = \{(l.\ w_1)´\}$
From the structure graph of H we get
$f_n(X_{S1}) = S$ and $f_n(X) = X$ for every other X
$\chi_1 = (0)$ $(=n_{S1})$
CR (χ_1) is
$S \rightarrow S_1 Z_{S1,1}$ and $X_{S4} \rightarrow S_4$
Simulating the derivation for w_1, via these rules we get
$Z_{S1,1} = X_{S4}$. As a result we see that CR (χ_1) is good and a ground interpretation. $I_1 = (f_n, f_p)$ admissible to
H_1 is obtained where
$f_n(X_{S1}) = S$ $\quad f_p(X_{S1}) = X_{S4}$
$f_n(X) = X$ otherwise $\quad f_p(X_{S4}) = \lambda$

The conjectured grammar $G_{H1} = I_1(G_{0,\Sigma})$ is
$(\{S, X_{S4}\}, \{X_{S1}, X_{S4}\}, P_1, S)$ where P_1 is
$S \rightarrow S_1 X_{S4}$, $X_{S4} \rightarrow S_4$

2. Let $w_2 = \{S_1 S_2 S_4 S_3 S_4\}$ be the second word from H and alph$(w_2) = \{S_1, S_2, S_3, S_4\}$. Then $H_2 = \{w_1, w_2\}$
Since S_4 is final S_4 handle rule is $X_{S4} \rightarrow S_4$ and
$f_p(X_{S4}) = \lambda$ $(n_{S1} \geq 0)$
From the structure graph of H we get
$f_n(X_{S1}) = S$ and
$f_n(X) = X$ for every other X
$f_p(X_{S1}) \neq \lambda$
and Lg$(H_2) = \{(l.w_1), (l.w_2)\}$ where

The Length equation for w_2 is
$n_{S1} + n_{S2} + n_{S3} + 2n_{S4} = -1$….. (l. w_2)
$n_{S1} + n_{S2} + n_{S3} = 0$ ….. (l.w_2)´

Using the set H, we construct a directed graph called the structure graph of H as follows



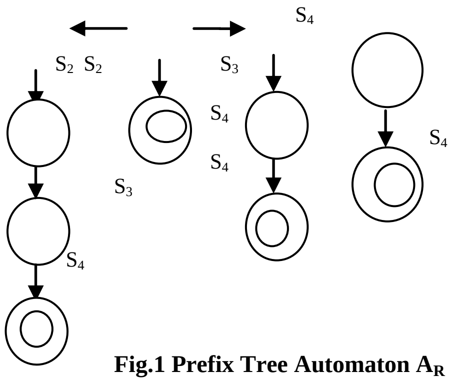


Fig.1 Prefix Tree Automaton A_R

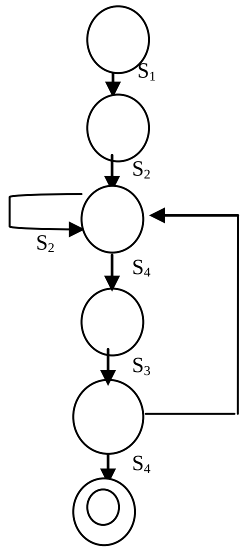


Fig.2 Structured graph H

To solve Lg$(H_2') = \{(l.w_1)´, (l.w_2)´\}$, we construct an associated matrix M_{H2} and a matrix equation $M_{H2} = \left(\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \end{array}\right)$

The matrix computation is done as follows

$$\left(\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \end{array}\right) \Rightarrow^* \left(\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{array}\right)$$

& solution of Lg $(H_2) = (n_{S1}=0) \cup (n_{S2} + n_{S3} = 0) \Longrightarrow n_{S1} = 0$
and we may choose a solution vector
$\chi_2 = (0, 2, -1)$ $(=(n_{S1}, n_{S3}, n_S))$
Hence the set of candidate rules
CR $(\chi_2)\{S \rightarrow S_1 Z_{S1,1}, X_{S2} \rightarrow S_2 Z_{S2,1} Z_{S2,2} Z_{S2,3}, X_{S3} \rightarrow S_3\}$ is obtained.
Simulating the derivation for w_2 via these rules we get
$Z_{S1,1} = X_{S2} = X_{S4}$, $Z_{S2,1} = X_{S4}$, $ZS_{2,2} = X_{S3}$, $Z_{S2,3} = X_{S4}$
$X_{S4} \rightarrow S_4$
We see that CR (χ_2) is good and a ground interpretation.
$I_2 = (f_n, f_p)$ obtained where
$f_n(X_{S1}) = S$ $\quad f_p(X_{S1}) = X_{S2} X_{S4}$
$f_n(X_{S4}) = X_{S2}$ $\quad f_p(X_{S2}) = X_{S3} X_{S4}$
$f_n(X) = X$ otherwise $\quad f_p(X_{S3}) = \lambda$
$f_p(X_{S4}) = \lambda$
Thus the conjectured grammar $G_{H2} = I_2(G_{0,\Sigma})$ is
$(\{S, S_2, S_3\}, \{S_1, S_2, S_3, S_4\}, P_2, S)$ where P_2 is
$S \rightarrow S_1 Z_{S1,1} \rightarrow S_1 X_{S4} \rightarrow S_1 S_4$
$S \rightarrow S_1 Z_{S1,1} \rightarrow S_1 X_{S2} \rightarrow S_1 S_2 Z_{S2,1} Z_{S2,2}$
$Z_{S2,3} \rightarrow S_1 S_2 X_{S4} X_{S3} X_{S4}$
$S \rightarrow S_1 S_2 S_4 S_3 S_4$
$S \rightarrow S_1 X_{S2}$ $\quad X_{S2} \rightarrow S_2 X_{S4} X_{S3} X_{S4}$, $\quad X_{S3} \rightarrow S_3$ $\quad X_{S4} \rightarrow S_4$

3. Let $w_3 = \{S_1\, S_2S_2S_4S_3S_4\, S_3S_4\}$ be the third word from H and $H_3 = \{w_1, w_2, w_3\}$ and alph $(H_3) = \{S_1, S_2, S_3, S_4\}$.
From the structure graph shown above we have $f_n(X_{S1}) = S$ and $f_p(X_{S1}) \neq \lambda$
Since S_4 is final, the S_4 handle rule is $X_{S4} \rightarrow S_4$ and $f_p(X_{S4}) = \lambda\ (n_{S1} \geq 0)$

The IA computes a length equation for w_3 and constructs

$Lg(H_3') = \{(l.w_1)', (l.w_2)', (l.w_3)'\}$ where
$n_{S1} + 2n_{S2} + 2n_{S3} + 3n_{S4} = -1$ ….. $(l.\ w_3)$
$n_{S1} + 2n_{S2} + 2n_{S3} = 2$ ….. $(l.\ w_3)'$
and the associated matrix M_{H3} is

$$\left(\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 2 & 2 & 2 \end{array}\right) \Rightarrow^* \left(\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{array}\right)$$

Thus the Sol $(Lg(H_3')) = (n_{S1} = 0) \cup (n_{S2} + n_{S3} = 1)$ is obtained from which we may choose a solution vector
$\chi_3 = (0,2,-1)(= (n_{S1}, n_{S2}, n_{S3}))$
Hence the set of candidate rules CR (χ_3) is
$S \rightarrow S_1 Z_{S1,1}$, $X_{S2} \rightarrow S_2 Z_{S2,1} Z_{S2,2} Z_{S2,3}$
$X_{S3} \rightarrow S_3$ $X_{S4} \rightarrow S_4$

Simulating the derivation for W_3, via these rules we get $Z_{S1,1} = X_{S2} = X_{S4}$, $Z_{S2,1} = X_{S2} = X_{S4}$, $ZS_{2,2} = X_{S3}$, $Z_{S2,3} = X_{S2} = X_{S4}$
We see that CR (χ_3) is good and a ground interpretation.
$I_3 = (f_n, f_p)$ is obtained where

$f_n(X_{S1}) = S$ $f_p(X_{S1}) = X_{S2} X_{S4}$
$f_n(X_{S4}) = X_{S2}$ $f_p(X_{S2}) = X_{S2} X_{S3} X_{S4}$
$f_n(X) = X$ otherwise $f_p(X_{S3}) = \lambda$
$f_p(X_{S4}) = \lambda$

Thus the conjectured grammar $G_{H3} = I_3(G_{0,\Sigma})$ is
$(\{S, S_2, S_3, S_4\}, \{S_1, S_2, S_3, S_4\}, P_3, S)$ where P_3 is
$S \rightarrow S_1 X_{S2}$ $X_{S2} \rightarrow S_2 X_{S4} X_{S3} X_{S4}$
$X_{S3} \rightarrow S_3$ $X_{S4} \rightarrow S_4$

4. Let $w_4 = \{S_1\, S_2S_2S_4S_3S_4S_3S_2S_4S_3S_4\}$ be the fourth word from H and $H_4 = \{w_1, w_2, w_3, w_4\}$ and alph $(H_4) = \{S_1, S_2, S_3, S_4\}$.
From the structure graph shown above we have $f_n(X_{S1}) = S$ and $f_p(X_{S1}) \neq \lambda$

Since S_4 is final, the S_4 handle rule is $X_{S4} \rightarrow S_4$ and $f_p(X_{S4}) = \lambda\ (n_{S1} \geq 0)$

The IA computes a length equation for w_4 and constructs $Lg(H_4') = \{(l.w_1)', (l.w_2)', (l.w_3)', (l.w_4)'\}$ where
$n_{S1} + 3n_{S2} + 3n_{S3} + 4n_{S4} = -1$ ….. $(l.\ w_4)$
$n_{S1} + 3n_{S2} + 3n_{S3} = 3$ ….. $(l.\ w_4)'$
and the associated matrix M_{H4} is

$$\left(\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 2 & 2 & 2 \\ 1 & 3 & 3 & 3 \end{array}\right) \Rightarrow^* \left(\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right)$$

Thus the $Sol(Lg(H_4')) = (n_{S1} = 0) \cup (n_{S2} + n_{S3} = 1)$ is obtained from which we may choose a solution vector
$\chi_4 = (0,2,-1)(= (n_{S1}, n_{S2}, n_{S3}))$
The set of candidate rules CR (χ_4) is
$S \rightarrow S_1 Z_{S1,1}$, $X_{S2} \rightarrow S_2 Z_{S2,1} Z_{S2,2} Z_{S2,3}$
$X_{S3} \rightarrow S_3$ $X_{S4} \rightarrow S_4$
Simulating the derivation for W_4, Via these rules we get
$Z_{S1,1} = X_{S2} = X_{S4}$, $Z_{S2,1} = X_{S2} = X_{S4}$, $ZS_{2,2} = X_{S3}$,
$Z_{S2,3} = X_{S2} = X_{S4}$ We see that CR (χ_4) is good and a ground interpretation. $I_3 = (f_n, f_p)$ is obtained where

$f_n(X_{S1}) = S$ $f_p(X_{S1}) = X_{S2} X_{S4}$
$f_n(X_{S4}) = X_{S2}$ $f_p(X_{S2}) = X_{S2} X_{S3} X_{S4}$
$f_n(X) = X$ otherwise $f_p(X_{S3}) = \lambda$
$f_p(X_{S4}) = \lambda$

Thus the conjectured grammar is
$G = G_{H4} = (\{S, S_2, S_3, S_4\}, \{S_1, S_2, S_3, S_4\}, P_4, S)$
where P_4 is
$S \rightarrow S_1 X_{S2}$ $X_{S2} \rightarrow S_2 X_{S4} X_{S3} X_{S4}$
$X_{S3} \rightarrow S_3$ $X_{S4} \rightarrow S_4$
The conjectured G_{H4} is equivalent to G_*. Hence G_{H4} is always an output as a conjecture for all input data afterwards.

Identification of G_i' using T_i:
The above procedure can be adapted to the sets T_1, T_2, T_3 and T_4 to conjecture the grammars G_1', G_2', G_3' and G_4' of G'

5. CONCLUSION

In this paper we have extended the notion of learning very simple grammars of Yokomori[1] to matrix grammars and adopted a modification of the algorithm to learn very simple matrix grammars which in turn can be used to study digitized pictures. The very simple matrix grammar considered here is (CF:CF) matrix grammar. Hence for further study, applications of these grammars to other domains can be considered.

6. REFERENCES

[1] Rani Siromoney: On equal matrix languages. Information and control. 14(2)(1969) 135–151

[2] Yokomori T. On polynomial - time identification of very simple grammars from positive data. Theoretical computer science 298 (2003)179-206

[3] A.Rosenfeld and J.L.Pfaltyz. Sequential operations in digital picture processing. J.Assoc.Comput.Mach.13, 1966, pp. 471-494.

[4] Gift Siromoney et al. Abstract families of matrices and picture languages. Computer graphics and image processing (1972) I, (284-307)

[5] Yokomori, T., On polynomial-time learnability in the limit of strictly deterministic automata. Machine learning 19(1995), 153-179.