A GRAMMATICAL INFERENCE FOR CONTEXT-FREE LANGUAGES BASED ON SELF-EMBEDDING

Tanatsugu, Keisuke
Department of Mathematics, Kitakyushu University

https://doi.org/10.5109/13382

出版情報：Bulletin of informatics and cybernetics. 22 (3/4), pp.149-163, 1987-03. Research Association of Statistical Sciences
バージョン：
権利関係：

Bulletin of Informatics and Cybernetics Vol. 22, No. 3~4, 1987

A GRAMMATICAL INFERENCE FOR CONTEXT-FREE LANGUAGES BASED ON SELF-EMBEDDING

By

Keisuke TANATSUGU*

Abstract

Our purpose in this paper is to construct a practical algorithm to inductively infer context-free languages. A basic procedure is first introduced to take out the self-embedding structure from given strings. Secondly, based on this procedure an algorithm to infer linear grammars from given finite sample sets is presented and its completeness is proved. Finally, by using a method to compose linear grammars, we propose an algorithm which is also complete for context-free languages.

1. Introduction

For a language L over an alphabet Σ, a finite subset of $I(L)=\{+w\,;\,w\in L\}\cup\{-w\,;\,w\in\Sigma^*-L\}$ is said a sample of L. Given a family of languages $\mathcal{L}$, an algorithm f is called a complete grammatical inference algorithm for $\mathcal{L}$ if for any $L\in\mathcal{L}$ there exist some infinite sequence $x_1, x_2, \cdots \in I(L)$ and integer n_0 such that $n\geqq n_0$ implies $L=L[f(x_1, x_2, \cdots, x_n)]$. We may find such an algorithm by enumerating all grammars even for the whole of phrase structure languages [3]. However, we concern with an effective algorithm which enumerates only qualified grammars using structural information belonging to samples. Biermann [5] introduced such an interesting algorithm for regular languages and Tanatsugu [7] proposed for harmonic linear languages which is a superclass of regular languages. These method are based on the fact that, each variable in grammar to be inferred, may be characterized by a finite subset of the language generated from it.

In the present paper we construct an practical inference procedure for the family of context-free languages. We consider that the main structural feature of context-free languages is the self-embedding. So, we first give the procedure to take out redunduncy parts in both sides of strings by using the notion of derivative mapping [2], in section 3. Based on this procedure, we present an inference algorithm for linear languages. Then it is shown that this algorithm is complete for the family of linear languages in section 4. Futhermore, by adding a composition method of linear grammars, we propose an algorithm which is also complete for the family of context-free languages in section 5.

* Department of mathematics, Kitakyushu University, Kitakyushu 802, Japan.

2. Preliminaries

We shall give some basic notions and lemmas for the subsequent sections.

Let X be an alphabet, then the set of all strings over X is denoted by X^*. The empty string of zero symbols is denoted by ε.

A context-free grammar (CFG for short) is defined by a quadruple $G=(\Gamma, \Sigma, P, S)$ where Γ is a finite set of variable symbols, Σ called an alphabet is a finite set of terminal symbols, P is a finite set of production rules of the form $A\to x$ with $A\in\Gamma$ and $x\in(\Gamma\cup\Sigma)^*$, and $S\in\Gamma$ is the start symbol. We write $x\Rightarrow y$ if there exist $A\in\Gamma$, $x_1, x_2, z\in(\Gamma\cup\Sigma)^*$ such that $x=x_1Ax_2$, $y=x_1zx_2$ and the rule $A\to z$ is in P. The transitive closure of $\Rightarrow$ is written $\overset{*}{\Rightarrow}$. The language L_A generated from $A\in\Gamma$ is denoted by $\{w\,;\,A\overset{*}{\Rightarrow}w,\ w\in\Sigma^*\}$. Especially L_S is also represented as $L(G)$ and it is said the context-free language (CFL for short) generated by the grammar G. Now we add the following restrictions for every CFG $G=(\Gamma, \Sigma, P, S)$:

1. For each variable $A\in\Gamma$:

(1) There is a derivation $S\overset{*}{\Rightarrow}uAv$ for some $u, v\in(\Gamma\cup\Sigma)^*$.

(2) $L_A\neq\phi$.

(3) $(A\to A)\notin P$.

(4) If $A\neq S$, then there is at least one production rule of the form $A\to\alpha A\beta$ where $\alpha, \beta\in(\Gamma\cup\Sigma)^*$.

2. $L_A\neq L_B$ for any $A, B\in\Gamma$ such that $A\neq B$.

In spite of the above restrictions, for any CFL $L\subset\Sigma^*$ there exists a grammar G such that $L=L(G)$. A CFG G is called linear if every production rule is the form of $A\to uBv$ or $A\to w$ with $A, B\in\Gamma$ and $u, v, w\in\Sigma^*$, then $L(G)$ is said a linear language.

DEFINITION 1. Let L be a language over Σ, then we denote a derivative language of L by

$$\bar{u}L\bar{v}=\{x\,;\,uxv\in L\}.$$

Then (u, v) is called a cover of L for $\bar{u}L\bar{v}$. Futhermore, L is said to be embedding with respect to (u, v) if L is a subset of $\bar{u}L\bar{v}$.

The following is immediately obtained from the above definition.

LEMMA 1. 1. *Let L_1, L_2, L be languages over Σ and $u, u_1, u_2, v, v_1, v_2\in\Sigma^*$, then the following relations hold*:

(1) $\bar{u}_2(\bar{u}_1L\bar{v}_1)\bar{v}_2=\overline{u_1u_2}L\overline{v_2v_1}$.

(2) *If $L_1\subset L_2$, then $\bar{u}L_1\bar{v}\subset\bar{u}L_2\bar{v}$.*

(3) $\bar{u}(L_1\cup L_2)\bar{v}=\bar{u}L_1\bar{v}\cup\bar{u}L_2\bar{v}$.

(4) $\bar{u}(L_1\cap L_2)\bar{v}=\bar{u}L_1\bar{v}\cap\bar{u}L_2\bar{v}$.

2. *Let $G=(\Gamma, \Sigma, P, S)$ be a CFG. If $A\overset{*}{\Rightarrow}uBv$ then $L_B\subset\bar{u}L_A\bar{v}$ where $A, B\in\Gamma$ and $u, v\in\Sigma^*$.*

We now introduce an order relation on $\Sigma^*\times\Sigma^*$ based on the lexicographical order on Σ^* as follows:

$$(u_1, v_1)<(u_2, v_2)$$

if and only if

$$(|u_1v_1|<|u_2v_2|)\vee(|u_1v_1|=|u_2v_2|\wedge u_1v_1<u_2v_2)\vee(u_1v_1=u_2v_2\wedge v_1<v_2).$$

For example, for $\Sigma=\{a, b\}$

$$(\varepsilon, \varepsilon)<(a, \varepsilon)<(\varepsilon, a)<(b, \varepsilon)<(\varepsilon, b)<(aa, \varepsilon)<(a, a)<(\varepsilon, aa)<(ab, \varepsilon)<(a, b)<\cdots.$$

We represent the successor of (u, v) as $(u, v)'$.

DEFINITION 2. Let $X, Y\subset\Sigma^*$ and $\alpha, \beta\in\Sigma^*$.

(1) $\mathcal{E}_X(\alpha, \beta)=\{X'\neq\phi\,;\ X'\subset\bar{\alpha}X'\bar{\beta},\ X'\subset X\}$.

(2) $M_X(\alpha, \beta)=\cup\mathcal{E}_X(\alpha, \beta)$.

(3) $\mathcal{E}_X=\{M_X(\alpha, \beta)\,;\ (\alpha, \beta)\neq(\varepsilon, \varepsilon)\}$.

(4) $\hat{M}_{X;Y}(\alpha, \beta)=X\cap M_Y(\alpha, \beta)$. We simply write $\hat{M}_X(\alpha, \beta)$ in the case that Y is evident.

The following facts are easily derived from the above definition:

LEMMA 2. *For any $X\in\Sigma^*$ and $\alpha, \beta\in\Sigma^*$:*

(1) $M_X(\alpha, \beta)\subset X$. *Particularly,* $M_X(\varepsilon, \varepsilon)=X$.

(2) $M_X(\alpha, \beta)\subset\bar{\alpha}M_X(\alpha, \beta)\bar{\beta}$.

(3) $\mathcal{E}_X=\phi$ *if X is finite.*

EXAMPLE 1. Let $X=\{a^mb^m\,;\ m\geqq0\}\cup\{a^nb^{2n}\,;\ n\geqq0\}$, then

$$\mathcal{E}_X(a, b)=\{\{a^mb^m\,;\ m\geqq0\},\ \{a^mb^m\,;\ m\geqq1\},\ \cdots\},$$
$$\mathcal{E}_X(a, bb)=\{\{a^nb^{2n}\,;\ n\geqq0\},\ \{a^nb^{2n}\,;\ n\geqq1\},\ \cdots\},$$
$$M_X(a, b)=M_X(a^2, b^2)=\cdots=\{a^mb^m\,;\ m\geqq0\},$$
$$M_X(a, bb)=M_X(a^2, b^4)=\cdots=\{a^nb^{2n}\,;\ n\geqq0\},$$
$$\mathcal{E}_X=\{\{a^mb^m\,;\ m\geqq0\},\ \{a^nb^{2n}\,;\ n\geqq0\}\}.$$

LEMMA 3. *Let $G=(\Gamma, \Sigma, P, S)$ be a CFG and $A\in\Gamma$. If $S\overset{*}{\Rightarrow}uAv$ and $A\overset{*}{\Rightarrow}\alpha A\beta$, then*

$$L_A\subset M_{\bar{u}L(G)\bar{v}}(\alpha, \beta)$$

where $(u, v), (\alpha, \beta)\in\Sigma^*\times\Sigma^*$.

PROOF. By Lemma 1, $L_A\subset\bar{u}L(G)\bar{v}$, $L_A\subset\bar{\alpha}L_A\bar{\beta}$. Hence

$$L_A\in\mathcal{E}_{\bar{u}L(G)\bar{v}}(\alpha, \beta).$$

Since

$$M_{\bar{u}L(G)\bar{v}}(\alpha, \beta)=\cup\mathcal{E}_{\bar{u}L(G)\bar{v}}(\alpha, \beta),$$

we obtain

$$L_A\subset M_{\bar{u}L(G)\bar{v}}(\alpha, \beta).\quad\square$$

LEMMA 4. *Let $G=(\Gamma, \Sigma, P, S)$ be a CFG and $L=L(G)$. If*

$$S\overset{*}{\Rightarrow}uAv,\quad A\overset{*}{\Rightarrow}u'Bv',\quad A\overset{*}{\Rightarrow}\alpha A\beta\quad\text{and}\quad B\overset{*}{\Rightarrow}\alpha'B\beta',$$

then

$$L_B\subset\bar{u}'M_{\bar{u}L\bar{v}}(\alpha, \beta)\bar{v}'\cap M_{\overline{uu'}L\overline{v'v}}(\alpha', \beta')$$

where $(u, v), (u', v'), (\alpha, \beta), (\alpha', \beta')\in\Sigma^*\times\Sigma^*$.

PROOF. By Lemma 3

$$L_A\subset M_{\bar{u}L\bar{v}}(\alpha, \beta),\qquad L_B\subset M_{\overline{uu'}L\overline{v'v}}(\alpha', \beta').$$

And $L_B \subset \bar{u}' L_A \bar{v}'$ since $A \overset{*}{\Rightarrow} u'Bv'$. Therefore,

$$\bar{u}' M_{\bar{u} L \bar{v}}(\alpha, \beta) \bar{v}' \cap M_{\overline{uu'} L \overline{v'v}}(\alpha', \beta') \supset \bar{u}' L_A \bar{v}' \cap L_B = L_B. \quad \square$$

We now denote $X_k(\alpha, \beta)$ by $\bigcap_{i=0}^{k} \bar{\alpha}^i X \bar{\beta}^i$ where $X \subset \Sigma^*$, $\alpha, \beta \in \Sigma^*$, accordingly, $X_\infty(\alpha, \beta)$ indicates $\bigcap_{i=0}^{\infty} \bar{\alpha}^i X \bar{\beta}^i$.

LEMMA 5. *If $X \subset \bar{\alpha} X \bar{\beta}$, then $M_X(\alpha, \beta) = X_\infty(\alpha, \beta)$ where $X \subset \Sigma^*$, $\alpha, \beta \in \Sigma^*$.*

PROOF. Let $x \in M_X(\alpha, \beta)$, then there exists X' such that $X' \subset X$, $X' \subset \bar{\alpha} X' \bar{\beta}$ and $x \in X'$. Therefore, for any integer i

$$x \in X' \subset \bar{\alpha}^i X' \bar{\beta}^i \subset \bar{\alpha}^i X \bar{\beta}^i \qquad \text{i.e.,} \quad x \in X_\infty(\alpha, \beta).$$

Inversely let $x \in X_\infty(\alpha, \beta)$, then

$$x \in X, \ \bar{\alpha} X \bar{\beta}, \ \bar{\alpha}^2 X \bar{\beta}^2, \ \cdots$$

that is,

$$x, \ \alpha x \beta, \ \alpha^2 x \beta^2, \ \cdots \in X.$$

Now let $X' = \{\alpha^i x \beta^i ; i \geqq 0\}$, then $X' \subset X$ and $X' \subset \bar{\alpha} X' \bar{\beta}$, hence it follows that $x \in X' \subset M_X(\alpha, \beta)$. $\square$

3. Inference Algorithm

3.1. Generation of Variables

Let L be a CFL and any grammar generating L be unknown. Giving a sample I of L, we shall consider the method to infer a grammar $G = (\Gamma, \Sigma, P, S)$ of L. We start with the specification of variables Γ' corresponding to Γ in G. Since $S \overset{*}{\Rightarrow} uAv$ and $(A \to \alpha A \beta) \in P$ imyly

$$L_A \subset M_{\bar{u} L \bar{v}}(\alpha, \beta) \subset \bar{u} L \bar{v},$$

it is natural that we select a (u, v) satisfying the above precondition for $A \in \Gamma$ and regard $M_{\bar{u} L \bar{v}}(\alpha, \beta)$ as the corresponding variable to A. However, since $M_{\bar{u} L \bar{v}}(\alpha, \beta)$ is generally infinite set, we take the following finite set instead of it as a variable in Γ:

$$\hat{M}_{\bar{u} I_+ \bar{v}; \bar{u} L \bar{v}}(\alpha, \beta) = \bar{u} I_+ \bar{v} \cap M_{\bar{u} L \bar{v}}(\alpha, \beta)$$

So, first, we shall show an algorithm to construct the finite set $X \cap W_k(\alpha, \beta)$ $(X \subset W \subset \Sigma^*)$ by Procedure DER in Fig. 1.

Notice that the operation $Y \cap \bar{\alpha}^k W \bar{\beta}^k$ in Procedure DER may be surely excuted because Y is finite even though $\bar{\alpha}^k W \bar{\beta}^k$ is infinite, under the actual assumption that we can always know whether $x \in W$ or not for every $x \in \Sigma^*$.

PROPOSITION 1. *Procedure DER terminates in finite steps and its final output is $Y = X \cap W_k(\alpha, \beta)$.*

PROOF. First we shall prove that $X' = X \cap W_k(\alpha, \beta)$ in arbitrary step k.
This holds in the case of $k=0$ since $X \subset W = W_0(\alpha, \beta)$. Suppose that this holds in stage $k-1 (k \geqq 1)$ and consider the case of stage k. As Y in stage k is X' in stage $k-1$,

Input: Finite set $X \subset \Sigma^*$, $(\alpha, \beta) \in \Sigma^* \times \Sigma^*$.
Output: $Y = X \cap W_k(\alpha, \beta)$.
Procedure DER(X, (α, β)):
begin
  $k := 0$; $X' := X$; $Y := \phi$;
  while $X' \neq Y$ **do**
    begin
      $Y := X'$;
      $k := k+1$;
      $X' := Y \cap \bar{\alpha}^k W \bar{\beta}^k$
    end;
  return Y
end

Fig. 1. Procedure DER.

$$\begin{aligned} X' &= Y \cap \bar{\alpha}^k W \bar{\beta}^k \\ &= (X \cap W_{k-1}(\alpha, \beta)) \cap \bar{\alpha}^k W \bar{\beta}^k \\ &= X \cap W_k(\alpha, \beta) \end{aligned}$$

Therefore $X' = X \cap W_k(\alpha, \beta)$ for any stage k.

Thus if $X' = Y$ in stage k, then Procedure DER stops and $Y = X \cap W_k(\alpha, \beta)$. If $X' \subsetneqq Y$ in stage k, then X' get to Y in the next stage. Since Y strictly decreases as stage k increases, $X' = Y$ is resulted in finite steps. □

Now we shall show that the output of Procedure DER coincides to $\hat{M}_{X;W}(\alpha, \beta)$ for an appropriate input $X \subset W$. Such a X is obtained by extending some finite set $X_0 \subset W$ for a given $(\alpha_0, \beta_0) \in \Sigma^* \times \Sigma^*$ as follows:

Procedure EXT.

Step 1. $X \leftarrow X_0$.
Step 2. $(\alpha, \beta) \leftarrow (\varepsilon, \varepsilon)$.
Step 3. $(\alpha, \beta) \leftarrow (\alpha, \beta)'$, $k \leftarrow 1$.
Step 4. If $X \cap W_{k-1}(\alpha, \beta) \subset M_W(\alpha, \beta)$, then go to step 7.
Step 5. If $X \cap W_{k-1}(\alpha, \beta) \neq X \cap W_k(\alpha, \beta)$, then add 1 to k and go to step 4.
Step 6. There is an integer p such that $x, \alpha x \beta, \cdots \alpha^p \beta^p \in W$ and $\alpha^{p+1} x \beta^{p+1} \notin W$ for each member x of $X \cap W_{k-1}(\alpha, \beta) - M_W(\alpha, \beta)$. Let p_0 be the minimal integer of them and x_0 designate one of the strings corresponding to the intger p_0. Add $\{\alpha x_0 \beta, \cdots, \alpha^{p_0-k+1} x_0 \beta^{p_0-k+1}\}$ to X, and go to step 2.
Step 7. If $(\alpha, \beta) < (\alpha_0, \beta_0)$ then go to step 3.
Step 8. Stop.

We have the below propositions related to Procedure EXT.

PROPOSITION 2. *Integer p_0 is always well defined in step* 6 *and $p_0 \geqq k$.*

PROOF. First we show that there exists certainly integer p_0 in step 6. If there is no integer p such that $x, \alpha x \beta, \cdots, \alpha^p x \beta^p \in W$ and $\alpha^{p+1} x \beta^{p+1} \notin W$, then, since $x \in W$, $\{\alpha^n x \beta^n ; n \geqq 0\}$ is a subset of W, i.e., of $M_W(\alpha, \beta)$. This contradicts $x \notin M_W(\alpha, \beta)$.

Secondly we show that $p_0 \geqq k$. If $x \in X \cap W_{k-1}(\alpha, \beta)$, then $p_0 \geqq k-1$ because

$x, \alpha x\beta, \cdots, \alpha^{k-1}x\beta^{k-1}\in W$. In the case of $p_0=k-1$, x_0 is in $X\cap W_{k-1}(\alpha, \beta)$ but not $W_k(\alpha, \beta)$, in other words,

$$X\cap W_{k-1}(\alpha, \beta)\neq X\cap W_k(\alpha, \beta).$$

This derives a contradiction. Thus $p_0>k-1$. □

PROPOSITION 3. *Procedure EXT stops in finite steps.*

PROOF. Notice that for any (α, β) there is k such that $X\cap W_{k-1}(\alpha, \beta)=X\cap W_k(\alpha, \beta)$. Let us set k, X to k_1, X_1, respectively, at the time that step 6 has been excuted for (α, β), and set k, X to k_2, X_2, respectively, at some later time that step 6 has been again excuted for (α, β). Then we shall show that $k_1<k_2$.

$$X_1\cap W_{k-1}(\alpha, \beta)\not\subset M_W(\alpha, \beta) \qquad (k\leqq k_1),$$
$$X_1\cap W_{k-1}(\alpha, \beta)\neq X_1\cap W_k(\alpha, \beta) \quad (k<k_1)$$

and

$$X_1\cap W_{k-1}(\alpha, \beta)=X_1\cap W_k(\alpha, \beta) \quad (k=k_1).$$

Since $X_1\subset X_2$,

$$X_2\cap W_{k-1}(\alpha, \beta)\not\subset M_W(\alpha, \beta) \qquad (k\leqq k_1).$$

Since $X_1\subset X_2$ and $W_{k-1}(\alpha, \beta)\supset W_k(\alpha, \beta)$,

$$X_2\cap W_{k-1}(\alpha, \beta)\neq X_2\cap W_k(\alpha, \beta) \qquad (k\leqq k_1).$$

When $k=k_1$, since

$$\alpha^{p_0-k+1}x_0\beta^{p_0-k+1}\in X_2, \quad \alpha^{p_0}x_0\beta^{p_0}\in W, \quad \alpha^{p_0+1}x_0\beta^{p_0+1}\notin W,$$
$$X_2\cap W_{k-1}(\alpha, \beta)\ni\alpha^{p_0-k+1}x_0\beta^{p_0-k+1}\notin X_2\cap W_k(\alpha, \beta).$$

That is,

$$X_2\cap W_{k-1}(\alpha, \beta)\neq X_2\cap W_k(\alpha, \beta) \qquad (k\leqq k_1).$$

Thus we obtain $k_1<k_2$.

Let us set the maximal number of p_0's for every (α, β) no larger than (α_0, β_0), as p^*, then $p^*\geqq p_0\geqq k$ by Proposition 3. Therefore, it turns out that the step 6 is never excuted for any (α, β) after some stage. That is, for any (α, β), an integer k is found such that $X\cap W_{k-1}(\alpha, \beta)\subset M_W(\alpha, \beta)$. Thus Procedure EXT stops. □

THEOREM 1. *Let $(\alpha_0, \beta_0)\in\Sigma^*\times\Sigma^*$. In Procedure DER, for any $(\alpha, \beta)\leqq(\alpha_0, \beta_0)$, there exists a finite subset X of W such that:*

(1) *If $M_W(\alpha, \beta)=\phi$, then the output $DER(X, (\alpha, \beta))$ is ϕ.*

(2) *If $M_W(\alpha, \beta)\neq\phi$, then the output $DER(X, (\alpha, \beta))$ is nonempty set $\hat{M}_{X;W}(\alpha, \beta)$.*

PROOF. We may consider a finite subset X_0 of W such that $M_W(\alpha, \beta)\cap X_0\neq\phi$ for any $(\alpha, \beta)\leqq(\alpha_0, \beta_0)$ satisfying $M_W(\alpha, \beta)\neq\phi$. For example, such a X_0 is easily obtained by taking out, one by one, a element from every nonempty $M_W(\alpha, \beta)$. Let X be the set obtained extending X_0 by Procedure EXT. Then for any $(\alpha, \beta)\leqq(\alpha_0, \beta_0)$ there is an integer k_0 such that

$$X\cap W_{k-1}(\alpha, \beta)\neq X\cap W_k(\alpha, \beta) \quad \text{if} \quad k<k_0,$$
$$X\cap W_{k-1}(\alpha, \beta)\subset M_W(\alpha, \beta) \quad \text{if} \quad k=k_0.$$

Now if

Input: $(u, v), (\alpha, \beta) \in \Sigma^* \times \Sigma^*$, $Y \subset \Sigma^*$, $n_0, r, r_0 \in N$, linear grammar $\hat{G}=(\hat{\Gamma}, \Sigma, \hat{P}, A_1)$.
Output: New variable A if possible and all production rules related to A.
Voriables:

$\mu(i)$: Finite set corresponding to i-th variable $A_i \in \hat{\Gamma}$.
$\hat{\Gamma}(u, v) \subset \hat{\Gamma}$: Set of vriables generated under (u, v).
SUB$\subset \Sigma^* \times \Sigma^* \times N$: Construct the rule $A_x \to u' A_r v'$ if $((u', v')x) \in$SUB.
SUB1$\subset N$: Construct the rule $A_x \to A_r$ if $x \in$SUB1.
SUB2$\subset N$: Construct the rule $A_r \to A_x$ if $x \in$SUB2.
r_0: Number of variable generated at first under (u, v).
check: 0 if no variable is generated; 1 if one variable is generated; 2 if two variables or more is generated under (u, v).

procedure LG(Y, (u, v), (α, β), n_0, r, r_0, $\hat{G}$):
bcgin
  check:=0;
  if $Y \neq \phi$ **then**
    begin
      if $r < r_0$ **then** check:=1
      else if $Y \neq \mu(i)$ for all $i (r_0 \leqq i \leqq r)$ **then** check:=2;
      if check$\neq$0 **then**
        begin
          SUB:=ϕ; SUB1:=ϕ; SUB2:=ϕ;
          if $r \neq 1$ **then**
            for i=1 **until** r_0-1 **do**
              if $Y \cap \bar{u}' \mu(i) \bar{v}' \neq \phi$ for some (u', v') such that
                $(u, v)=(u''u', v'v'')$ and $A_i \in \hat{\Gamma}(u'', v'')$ **then**
                  SUB:=SUB$\cup\{(u', v'), i)\}$;
          if check=2 **then**
            for $i=r_0$ **until** r **do**
              if $Y \subset \mu(i)$ **then** SUB1:=SUB1$\cup\{i\}$
              else if $Y \supset \mu(i)$ **then** SUB2:=SUB2$\cup\{i\}$;
          $r:=r+1$;
          $\mu(r):=Y$; $\hat{\Gamma}(u, v):=\hat{\Gamma}(u, v) \cup \{A_r\}$; $\hat{\Gamma}:=\hat{\Gamma} \cup \{A_r\}$;
          $\hat{P}=\hat{P} \cup \{A_r \to \alpha A_r \beta\}$
                $\cup\{A_x \to u' A_r v'$; $((u', v'), x) \in$SUB$\}$
                $\cup\{A_x \to A_r$; $x \in$SUB1$\} \cup \{A_r \to A_x$; $x \in$SUB2$\}$
                $\cup\{A_r \to w$; $|w| \leqq n_0$, $w \in \mu(r)\}$
        end
      else if there is no derivation such that $A_i \overset{*}{\Rightarrow} \alpha A_i \beta$ **then** $\hat{P}:=\hat{P} \cup \{A_i \to \alpha A_i \beta\}$
    end;
  return $\hat{G}$
end

Fig. 2. Procedure LG.

$$X\cap W_{k_0-1}(\alpha,\ \beta)\subset M_W(\alpha,\ \beta),$$

then

$$X\cap W_{k_0-1}(\alpha,\ \beta)\subset X\cap M_W(\alpha,\ \beta).$$

On the other hand

$$X\cap W_{k_0-1}(\alpha,\ \beta)\supset X\cap M_W(\alpha,\ \beta)$$

because

$$W_{k_0-1}(\alpha,\ \beta)\supset W_\infty(\alpha,\ \beta)=M_W(\alpha,\ \beta).$$

Thus

$$X\cap W_{k_0-1}(\alpha,\ \beta)=X\cap M_W(\alpha,\ \beta)=\hat{M}_{X;W}(\alpha,\ \beta).$$

Since $W_k(\alpha,\ \beta)\subset W_{k_0-1}(\alpha,\ \beta)$ we have similarly

$$X\cap W_{k_0}(\alpha,\ \beta)=\hat{M}_{X;W}(\alpha,\ \beta).$$

In Procedure DER,

$$Y=X\cap W_{k-1}(\alpha,\ \beta)\quad\text{and}\quad X'=X\cap W_k(\alpha,\ \beta).$$

Input: Finite set $I_+\subset\Sigma^*$, $(u_0,\ v_0)$, $(\alpha_0,\ \beta_0)\in\Sigma^*\times\Sigma^*$, $n_0\in N$.
Output: Linear grammar $\hat{G}=(\hat{\Gamma},\ \Sigma,\ \hat{P},\ A_1)$.
procedure LG_INF(I_+, $(u_0,\ v_0)$, $(\alpha_0,\ \beta_0)$, n_0):
begin

$\hat{\Gamma}:=\{A_1\}$; $\hat{\Gamma}(\varepsilon,\ \varepsilon):=\{A_1\}$; $\hat{P}:=\{A_1\to w\ ;|w|\leqq n_0,\ w\in I^+\}$;
$(u,\ v):=(\varepsilon,\ \varepsilon)$; $r:=1$; $r_0:=1$; $\mu(1):=I_+$;

while $(u,\ v)\leqq(u_0,\ v_0)$ **do**
if $I_+\not\subset L(\hat{G})$ **then**
begin
$X:=\bar{u}I_+\bar{v}$;
if $X\neq\phi$ **then**
begin /* construction of production rules under $(u,\ v)$ */
$(\alpha,\ \beta):=(\varepsilon,\ \varepsilon)'$;
while $(\alpha,\ \beta)\leqq(\alpha_0,\ \beta_0)$ **do**
begin
$Y:=$DER$(X,\ (\alpha,\ \beta))$;
$\hat{G}:=$LG$(Y,\ (u,\ v),\ (\alpha,\ \beta),\ n_0,\ r,\ r_0,\ \hat{G})$;
$(\alpha,\ \beta):=(\alpha,\ \beta)'$
end
end;
$(u,\ v):=(u,\ v)'$;
$r_0:=r+1$
end
else $(u,\ v):=(u_0,\ v_0)'$; /* this algorithm stops when $I_+\subset L(\hat{G})$ */
return $\hat{G}$
end

Fig. 3. Procedure LG_INF.

Hence $Y \neq X'$ if $k < k_0$; $Y = X' = \hat{M}_{X;W}(\alpha, \beta)$ and Procedure DER stops if $k = k_0$. Therefore if $M_W(\alpha, \beta) = \phi$, then $Y = \hat{M}_{X;W}(\alpha, \beta) = \phi$; if $M_W(\alpha, \beta) \neq \phi$, then $Y = \hat{M}_{X;W}(\alpha, \beta) \supset X_0 \cap M_W(\alpha, \beta) \neq \phi$. □

EXAMPLE 2. Let $W = \{a^m b^m ; m \geqq 0\} \cup \{a^n b^{2n} ; n \geqq 0\}$ and $X = \{\varepsilon, ab, abb\} \subset W$, then the output of Procedure DER is as follows:

(1) If $(\alpha, \beta) = (a, b)$, then

$$Y = \{\varepsilon, ab\} = X \cap M_W(a, b).$$

(2) If $(\alpha, \beta) = (a, bb)$, then

$$Y = \{\varepsilon, abb\} = X \cap M_W(a, bb).$$

(3) Otherwise, $Y = \phi$.

3.2. Construction of Production Rules

The output of Procedure DER may be $Y = \hat{M}_{X;W}(\alpha, \beta)$ for some input $X \subset W$. So our next interests is how to indicate new variables and construct new production rules from these outputs. If $M_{\bar{u}L\bar{v}}(\alpha, \beta)$ is nonempty and we can not find any old variable corresponding to it, then we produce a new variable and construct all production rules related to its variable. Such an Algorithm is given by Procedure LG in Fig. 2.

Table 1. Inference process in Procedure LG_INF for $I_+ = \{aaa, ababa\}$.

(u, v)	(α, β)	Y	$\hat{\Gamma}$	$\hat{P}$	$L(\hat{G})$
$(\varepsilon, \varepsilon)$	$(\varepsilon, \varepsilon)$	I_+	A_1	ϕ	ϕ
$(\varepsilon, \varepsilon)$	(a, ε)	ϕ			
	(ε, a)	ϕ			
	(b, ε)	ϕ			
	(ε, b)	ϕ		ϕ	ϕ
(a, ε)	(a, ε)	ϕ			
	(ε, a)	ϕ			
	(b, ε)	$\{aa, baba\}$	A_2	$A_1 \to aA_2$, $A_2 \to bA_2$	
	(ε, b)	ϕ			ϕ
(ε, a)	(a, ε)	ϕ			
	(ε, a)	ϕ			
	(b, ε)	ϕ			
	(ε, b)	$\{aa, abab\}$	A_3	$A_1 \to A_3a$, $A_3 \to A_3b$	ϕ
•	•				
•	•				
(a, a)	(a, ε)	ϕ			
	(ε, a)	ϕ			
	(b, ε)	$\{a, bab\}$	A_4	$A_1 \to aA_4a$, $A_4 \to bA_4$, $A_4 \to a$	
	(ε, b)	$\{a, bab\}$		$A_4 \to A_4b$	L

3.3. Inference of Linear Grammars

In this subsection, we propose the algorithm to construct a linear grammar from a positive sample I_+ of L where (u_0, v_0) is the upper limit on covers for generations of variables, (α_0, β_0) is the upper limit on covers for constructions of self-embedding rules and n_0 is the upper limit on lengths of right-side strings in the case of generating production rules of which the right-side consists of only terminals. Such an algorithm is given by Procedure LG_INF in Fig. 3.

EXAMPLE 3. Let $L=\{ab^mab^na\,;m, n\geqq 0\}$, $I_+=\{aaa, ababa\}$, $(u_0, v_0)=(a, a)$, $(\alpha_0, \beta_0)=(\varepsilon, b)$ and $n_0=1$, then we can infer a linear grammar $\hat{G}=(\hat{\Gamma}, \Sigma, \hat{P}, A_1)$ such that $L=L(\hat{G})$ by using Procedure LG_INF as shown in Table 1.

4. Completeness for Linear Languages

In this section we show that for any linear language L, a grammar generating L can be inferred by using Procedure LG_INF.

LEMMA 6. *Let L be an arbitrary language over Σ and $G=(\Gamma, \Sigma, P, [L])$ be a linear grammar satisfying the following conditions. then $L(G)\subset L$:*

(1) *Γ is a set added the element $[L]$ to some finite subset of $\{[X]\,;$ there is $(u, v)\in \Sigma^*\times\Sigma^*$ such that $X\subset \bar{u}L\bar{v}$ and $X\neq\phi\}$.*

(2) *$X_2\subset \bar{u}X_1\bar{v}$ if $([X_1]\to u[X_2]v)\in P$; $w\in X$ if $([X]\to w)\in P$.*

PROOF. There is the following derivation for $x\in L(G)$:

$$[X_0]\Rightarrow u_1[X_1]v_1\Rightarrow u_1u_2[X_2]v_2v_1\Rightarrow\cdots$$
$$\Rightarrow u_1\cdots u_mX_mv_m\cdots v_1\Rightarrow u_1\cdots u_mwv_m\cdots v_1=x$$

where

$$X_0=L,\quad ([X_i]\to u_{i+1}[X_{i+1}]v_{i+1})\in P\quad (i=0, \cdots, m-1).$$

Since $([X_m]\to w)\in P$, $w\in X_m$, that is, $x\in u_1\cdots u_mX_mv_m\cdots v_1$. Futhermore $X_{i+1}\subset \bar{u}_{i+1}X_i\bar{v}_{i+1}$ by $([X_i]\to u_{i+1}[X_{i+1}]v_{i+1})\in P$, accordingly

$$u_{i+1}X_{i+1}v_{i+1}\subset u_{i+1}(\bar{u}_{i+1}X_i\bar{v}_{i+1})v_{i+1}\subset X_i\quad (i=0, \cdots, m-1)$$

Hence

$$x\in u_1\cdots u_mX_mv_m\cdots v_1\subset u_1\cdots u_{m-1}X_{m-1}v_{m-1}\cdots v_1\subset\cdots\subset u_1X_1v_1\subset L.\quad \square$$

LEMMA 7. *In Procedure LG_INF, if $M_L(\alpha, \beta)\in\mathcal{E}_{\bar{u}L\bar{v}}$ and $\bar{u}I_+\bar{v}\neq\phi$ $((u, v)\leqq(u_0, v)$, $(\alpha, \beta)\leqq(\alpha_0, \beta_0))$, then there is an integer m such that $\mu(m)=\hat{M}_{\bar{u}I_+\bar{v}}(\alpha, \beta)$.*

PROOF. Consider the case of $Y=\hat{M}_{\bar{u}I_+\bar{v}}(\alpha, \beta)$. If there is $i\,(1\leqq i\leqq r)$ such that $Y=\mu(i)$, this lemma holds. Otherwise, we have $\mu(r+1)=Y=\hat{M}_{\bar{u}I_+\bar{v}}(\alpha, \beta)$. $\square$

COROLLARY. *Let $G=(\Gamma, \Sigma, P, S)$ be a CFG and $I_+\subset L(G)$. In Procedure LG_INF under some (u_0, v_0), $(\alpha_0, \beta_0)\in\Sigma^*\times\Sigma^*$, for any variable $A\in\Gamma$ there is integer m such that $\mu(m)\subset\bar{u}I_+\bar{v}\cap L_A$ and $A_m\in\hat{\Gamma}(u, v)$.*

PROOF. Given large enough (u_0, v_0), (α_0, β_0) there exist derivations such that $S\overset{*}{\Rightarrow}uAv$ and $A\overset{*}{\Rightarrow}\alpha A\beta$ $((u, v)\leqq(u_0, v_0)$, $(\alpha, \beta)\leqq(\alpha_0, \beta_0)$, $(\alpha, \beta)\neq(\varepsilon, \varepsilon))$. By Lemma 1, $L_A\subset M_{\bar{u}L\bar{v}}(\alpha, \beta)$. By Lemma 7, there is an integer m such that $\mu(m)=\bar{u}I_+\bar{v}\cap M_{\bar{u}L\bar{v}}(\alpha, \beta)$. Hence $\mu(m)\supset\bar{u}I_+\bar{v}\cap L_A$. $\square$

The above Corollary shows that each variable A in CFG G corresponds to a variable

(written as $\hat{A}$) in the grammar $\hat{G}$ inferred from $I_+ \subset L(G)$ using Procedure LG-INF. Futhermore, the following Lemma asserts that the language generated from $\hat{A}$ includes the language generated from A if G is linear:

LEMMA 8. *Let $G=(\Gamma, \Sigma, P, S)$ be a linear grammer and $L=L(G)$. Then for some $I_+ \subset L$, (u_0, v_0), $(\alpha_0, \beta_0) \in \Sigma^* \times \Sigma^*$ and integer n_0, Procedure LG-INF infer the linear grammer $\hat{G}=(\hat{\Gamma}, \Sigma, \hat{P}, A_1)$ such that for any $A \in \Gamma$ there is $\hat{A} \in \hat{\Gamma}$ satisfying $L_A \subset L_{\hat{A}}$.*

PROOF. Let

$$(u_0, v_0)=\max\{(u, v);\ S \overset{*}{\Rightarrow} x \Rightarrow uyv \text{ and each variable in } \Gamma \text{ occurs at most once in the derivation process to } x \text{ where } x, y \in (\Gamma \cup \Sigma)^*\},$$

$$(\alpha_0, \beta_0)=\max\{(\alpha, \beta);\ (A \to \alpha A \beta) \in P,\ A \in \Gamma\},$$

$$n_0=\max\{|w|;\ (A \to w) \in P,\ w \in \Sigma^*,\ A \in \Gamma\},$$

$$I_+=\{x;\ x \leqq u_0 \alpha_0 \beta_0 v_0,\ x \in L\}.$$

Then for any $A \in \Gamma$, there exists a derivation such that

$$S \overset{*}{\Rightarrow} uAv\ ((u, v) \leqq (u_0, v_0)).$$

Now we consider the three forms of production rules from the variable

$$A:\quad A \to \alpha A \beta,\quad A \to u' B v' \quad (A \neq B) \quad \text{and} \quad A \to w \quad (w \in \Sigma^*).$$

(1) the case of form $A \to \alpha A \beta$

$L_A \subset M_{\bar{u} L \bar{v}}(\alpha, \beta)$ by Lemma 1 and $\mu(m)=\hat{M}_{\bar{u} I_+ \bar{v}}(\alpha, \beta)$ by Lemma 7. Since $\mu(1)=I_+$, $\mu(1) \cap \hat{M}_{\bar{u} I_+ \bar{v}}(\alpha, \beta) \neq \phi$. We now write $A_m \in \hat{\Gamma}$ as $\hat{A}$, then

$$(A_1 \to u \hat{A} v),\quad (\hat{A} \to \alpha \hat{A} \beta) \in \hat{P}.$$

(2) the case of form $A \to u' B v'$

Consider the case of $Y=\hat{M}_{\overline{uu'} I_+ \overline{v'v}}(\alpha', \beta')$, where

$$(B \to \alpha' B \beta') \in P,\quad (uu', v'v) \leqq (u_0, v_0) \quad \text{and} \quad (\alpha', \beta') \leqq (\alpha_0, \beta_0).$$

Then

$$\begin{aligned} \bar{u}' \mu(m) \bar{v}' &= \bar{u}' (\bar{u} I_+ \bar{v} \cap M_{\bar{u} L \bar{v}}(\alpha, \beta)) \bar{v}' \\ &= \overline{uu'} I_+ \overline{v'v} \cap \bar{u}' M_{\bar{u} L \bar{v}}(\alpha, \beta) \bar{v}'. \end{aligned}$$

Futhermore, $Y \cap \bar{u}' \mu(m) \bar{v}' \neq \phi$ by Lemma 4. Thus

$$(\hat{A} \to u' \hat{B} v'),\quad (\hat{B} \to \alpha' \hat{B} \beta') \in \hat{P}$$

where we write, as $\hat{B}$, $A_r \in \hat{\Gamma}$ in functional procedure LG.

(3) the case of form $A \to w$

Since we may assume that $|w| \leqq n_0$ and $w \in \mu(m)$, $(\hat{A} \to w) \in \hat{P}$. By the above analysis, we can understand that each production rule in $\hat{G}$ has its corresonding rule in G. Let w be one of the minimal length words in $L_A \triangle L_B$ for $B \in \hat{\Gamma}$ distinct with A, such as $w \in L_A - L_B$, then there exist (u_1, v_1), $(u_2, v_2) \leqq (u_0, v_0)$ such that

$$S \overset{*}{\Rightarrow} u_1 A v_1 \Rightarrow u_1 w v_1,\quad S \overset{*}{\Rightarrow} u_2 B v_2 \nRightarrow u_2 w v_2.$$

Thus

$$w \in \hat{M}_{\overline{u_1} I_+ \bar{v}_1}(\alpha, \beta) - \hat{M}_{\overline{u_2} I_+ \bar{v}_2}(\alpha', \beta')$$

where $(A\to\alpha A\beta)$, $(B\to\alpha' B\beta')\in\hat{P}$. Namely $\hat{A}\neq\hat{B}$.

Innversely, $L_A\neq L_B$ if $\hat{A}\neq\hat{B}$, here $A\neq B$ by the predescrived restriction on grammar. Therefore $L_A\subset L_{\hat{A}}$ by application of the induction. □

THEOREM 2. *For any linear language there are some sample* $I_+\subset L$, (u_0, v_0), $(\alpha_0, \beta_0)\in\Sigma^*\times\Sigma^*$ *and integer* n_0 *such that we can infer linear grammar* $\hat{G}$ *of* L *by using Pro-*

Input: $x, y\in(\hat{\Gamma}\cup\Sigma)^*$, $\hat{G}=(\hat{\Gamma}, \Sigma, \hat{P}, A_1)$.
Output: Set of all strings composed from x and y; H.
Variables:
m: Serial number of string analyzing currently.
n: Maximal number of string in the middle of composition.
$z(m)$: Composition string in the current.

```
procedure COMPO(x, y):
begin
  m:=1; n:=1; z(1):=ε; x(1):=x; y(1):=y; H:=φ;
  while m≦n do
    begin
      delete the maximal length common prefix of x(m) and y(m);
      if x(m)=Ax' for some A∈Γ̂ and x'∈(Γ̂∪Σ)*, and there exist w∈Σ* and
          y'∈(Γ̂∪Σ)* such that A⇒*w and y(m)=wy' then
        begin
          let A⇒*w_1, ···, A⇒*w_h and y(m)=w_1y'_1= ··· w_hy'_h;
          for j=n+1 until n+h do
            begin
              x(j):=x'; y(j):=y'_j; z(j):=z(m)A
            end;
          n:=n+h
        end
      else
        if y(m)=By' for some B∈Γ̂ and x'∈(Γ̂∪Σ)* then
          if there exist w∈Σ* and x'∈(Γ̂∪Σ)* such that A⇒*w and x(m)=wx' then
            begin
              let B⇒*w_1, ···, B⇒*w_h and x(m)=w_1x'_1= ··· =w_hx'_h;
              for j=n+1 until n+h do
                begin
                  x(j):=x'_j; y(j):=y'; z(j):=z(m)B
                end
            end
          else if x(m)=y(m)=ε then H:=H∪{z(m)};
        m:=m+1
      end
    return H
  end
```

Fig. 4. Procedure COMPO.

cedure LG_INF.

PROOF. Taking that $A=S$ in Lemma 8, $L\subset L(\hat{G})$. On the other hand, if we interpret A_r obtained when $\mu(r)=\hat{M}_{\bar{u}I_+\bar{v}}(\alpha, \beta)$ in Procedure LG_INF, as $[M_{\bar{u}L\bar{v}}(\alpha, \beta)]$, then the inferred grammar $\hat{G}$ satisfies the condition of Lemma 6 and thus $L(\hat{G})\subset L$.

Input: Finite set $I_+\subset\Sigma^*$, (u_0, v_0), $(\alpha_0, \beta_0)\in\Sigma^*\times\Sigma^*$, $n_0\in N$.
Output: CFG $\hat{G}=(\hat{I}, \Sigma, \hat{P}, A_1)$.
Variables:
check: 0 if it is impossible to compose the given strings for every variables; 1 otherwise.
$c(i)$: 0 if production rule from A_i is not composed at all; 1 otherwise.

```
procedupe CFG_INF (I+, (u0, v0), (α0, β0), n0);
begin
  check:=1;
  Ĝ:=LG_INF (I+, (u0, v0), (α0, β0), n0);
  for all i(1≦i≦r) do c(i):=1;
  while I+⊄L(Ĝ) do
    if check≠0 then
      begin
        check:=0;
        for i=1 until r do
          if c(i)=1 then
            begin
              check:=1; c(i):=0;
              let n(i) be the number of production rules from A and x1, ···, xn(i) be
                  right-side strings of these rules;
              for p=1 until n(i) do
                for q=p until n(i) do
                  if COMPO(xp, xq)≠φ then
                    begin
                      let COMPO(xp, xq)={y1, ···, yt};
                      for j=1 until t do
                        begin
                          P̂:=P̂∪{Ai→yj};
                          if I_∩L(Ĝ)≠φ then P̂:=P̂−{Ai→yj}
                          else c(i):=1
                        end
                    end
            end
      end
    else return φ;
  return Ĝ /* Ĝ is compatible to given sample (I+, I_) */
end
```

Fig. 5. Procedure CFG_INF.

Hence $L=L(\hat{G})$. □

5. Grammatical Inference for CFL's

We first give Procedure COMPO in Fig. 4 to compose production rules of linear grammars obtained by Procedure LG_INF. Then given a positive sample $I_+\subset L$ and a negative sample $I_-\subset\Sigma^*-L$ for a CFL L, we consider a method to adopt only composed rule compatible with these samples. In final, we propose Procedure CFG_INF to identify a CFG G such that $L=L(G)$ in Fig. 5.

THEOREM 3. *For any CFL, there are some sample* (I_+, I_-), (u_0, v_0), $(\alpha_0, \beta_0)\in\Sigma^*\times\Sigma^*$ *and integer* n_0 *such that we can infer a CFG* $\hat{G}=(\hat{\Gamma}, \Sigma, \hat{P}, A_1)$ *satisfying* $L=L(\hat{G})$ *in Procedure CFG_INF.*

PROOF. Let $L=L(G)$ and $G=(\Gamma, \Sigma, P, S)$. Let $A\to u_0A_1u_1\cdots u_{m-1}A_mu_m$ be a production rule of G, then by the similar method to Theorem 2, there are $(\hat{A}\to u_0w_1u_1\cdots u_{i-1}\hat{A}_iu_i\cdots w_mu_m)$ in $\hat{P}$ $(i=1, 2, \cdots, m)$ for some $I_+\subset L$, (u_0, v_0), $(\alpha_0, \beta_0)\in\Sigma^*\times\Sigma^*$ and $n_0\in N$, where w_j is a minimal length string in $L_A(j=1, \cdots, m)$. By the composition method of rules in Procedure COMPO, we have $(\hat{A}\to u_0\hat{A}_1u_1\cdots u_{m-1}\hat{A}_mu_m)\in\hat{P}$. We can check whether these composed rules are compatible for L, by giving a large enough negative sample I_-. □

EXAMPLE 4. Let $L=\{ab^ma^mba^nb^na\,;\ m, n\geqq 0\}$ be unknown. We apply Procedure CFG_INF for $I_+=\{aba, ababa, abababa\}\subset L$, $I_-=\{abaabbaba\}\not\subset L$, $(u_0, v_0)=(a, ba)$. $(\alpha_0, \beta_0)=(b, a)$ and $n_0=0$. First, by the function procedure LG_INF, a linear grammar $\hat{G}=(\{A_1, A_2, A_3\}, \{a, b\}, \hat{P}, A_1)$ is obtained as shown in Table 2, where $\hat{P}=\{A_1\to abA_2a/aA_3ba, A_2\to aA_2b/\varepsilon, A_3\to bA_3a/\varepsilon\}$.

Composing these production rules in ordering from A_1,

(1) COMPO$(abA_2a, abA_2a)=\{abA_2A_2a\}$:

If we add $(A_1\to abA_2A_2a)$ to $\hat{P}$, then it derives a contradiction because $abaabbaba$

Table 2. Inference process in Procedure CFG_INF for $I_+=\{aba, ababa, abababa\}$.

(u, v)	(α, β)	Y	$\hat{\Gamma}$	$\hat{P}$	$L(\hat{G})$
$(\varepsilon, \varepsilon)$	$(\varepsilon, \varepsilon)$	I_+	A_1	ϕ	ϕ
•	•				
•	•				
(ab, a)	•				
	•				
	(a, b)	$\{\varepsilon, ab\}$	A_2	$A_1\to abA_2a$, $A_2\to aA_2b$, $A_2\to\varepsilon$	
	•				$ababa$
	•				$\notin L(\hat{G})$
(a, ba)	•				
	•				$ababa$
	(b, a)	$\{\varepsilon, ba\}$	A_3	$A_1\to aA_3ba$, $A_3\to bA_3a$, $A_3\to\varepsilon$	$\notin L(\hat{G})$

$\in I_-$ is generated only by applying this rule.

(2) COMPO$(abA_2a, aA_3ba)=\{aA_3bA_2a\}$:

If we add $(A\to aA_3bA_2a)$ to $\hat{P}$, then Procedure CFG_INF terminates because $I_+\subset L(\hat{G})$ and $I_-\cap L(\hat{G})=\phi$. Indeed, $L=L(\hat{G})$.

References

[1] SOROMONOFF, R.: *A Formal Theory of Inductive Inference,* Information and Control, (1964), 1-22, 224-254.

[2] HUZINO, S.: *On Some Properties of Derivative-mappings, Structural Diagrams and Structural Equations: Part 1,* Memo. Fac. Sci. Kyushu Univ. Ser. **A20**, (1966), 179-265.

[3] GOLD, M.: *Language Identification in the Limit,* Information and Control, **10** (1967), 447-474.

[4] FELDMAN, J A., GIPS, J., HORNING, J. J. and READER, S.: *Grammatical Complexity and Inference,* Technical Report No. CS125, Computer Science Department, Stanford University (1969).

[5] BIERMANN, A. W.: *An Interactive Finite-state Language Learner,* Proc. 1-st USA-JAPAN Comp. Conf. (1972), 13-20.

[6] TANATSUGU, K. and ARIKAWA, S.: *On Characteristic Sets and Degrees of Finite Automata,* International Journal of Computer and Information Sciences, **6**, 1 (1977), 83-93.

[7] TANATSUGU, K.: *A Grammatical Inference for Harmonic Linear Languages,* International Journal of Computer and Information Sciences, **13**, 5 (1984), 413-423.

Communicated by S. Kanō
Received October 14 ,1986
Revised October 30, 1986