WEB LANGUAGES AND INTERACTIVE LANGUAGES

FEBRUARY 1975

YOSHINORI EZAWA

WEB LANGUAGES AND INTERACTIVE LANGUAGES

by

YOSHINORI EZAWA

Submitted in partial fulfillment of the
requirement for the degree of
DOCTOR OF ENGINEERING
(Electrical Engineering)
at
OSAKA UNIVERSITY
TOYONAKA, OSAKA, 560, JAPAN
February 1975

ACKNOWLEDGEMENTS

With the indulgence of the thorough reader
I wish to express my appreciation to those
who have participated in the development
of this thesis.

To Prof. Kokichi Tanaka for his invaluable guidance, substantial advice and continuous encouragement during the conduct of this thesis. To the members of my reading committee for their helpful suggestions.

To Associate Prof. Junichi Toyoda who has been my supervisor and has always given eager discussions and constant encouragements.

To Profs. Tadao Kasami, Makoto Kizawa and Toshio Fujisawa of Dept. of Information and Computer Sciences, Profs. Toshio Makimoto Kazuo Fujisawa, Susumu Nanba and Tadashi Sueta of Dept. of Electrical Engineering, Profs. Yoshifumi Sakurai, Saburo Tsuji and Yoshiyuki Sakawa of Dept. of Control Engineering, for their continued encouragements to continue graduate studies.

To Associate Profs. Masamichi Shimura, Nobuki Tokura and Susumu Matoba for their several relevant comments and suggestions.

To Drs. Tadahiro Kitahashi, Masaharu Mizumoto Shinichi Tamura, Norihiro Abe, Koichiro Ochimizu (presently is with Sizuoka University), Hidekazu Tsuji (presently is with Mitsubishi Ltd.), for their enlightening discussions and criticisms.

To Prof. Teruo Fukumura and Associate Prof. Yasuyoshi Inagaki of Nagoya University, Prof. Shoichi Noguchi of Tohoku University, for their relevant conversations and suggestions.

To Prof. A. Rosenfeld of the University of Maryland, Prof. A. Salomaa of the University of Turuku, Prof. G. Rozenberg of the University of Utrecht and Prof. B. H. Mayoh of Aarhus University, for sending me many enlightening papers.

To the staffs and students of Prof. Tanaka's Laboratory and Prof. Kizawa's Laboratory, especially Messrs. Tetsuro Ito, Kazuyoshi Mikami, Hyo Heng Kim, Shoji Tominaga, Nobuhito Yamamoto, Tatsuo Tsuji, Yoshihide Sano, Kyota Aoki, Motohide Umano, Masato Urakami and Hiromichi Ogata for their helpful conversations, cooperation, comments and suggestions. To Mr. Hideo Kawai, Miss Taneyo Ukawa, Miss Kyoko Nakatsukasa, Mrs. Hiroko Kawato, Miss Kazuko Okuno who have disposed of routine business.

Finally, to my parents for their help and understanding without which I probably would not have finished this work.

PREFACE

Man exchanges mutually his own ideas by language. Thus, human beings have accumulated their experiences and have built up a variety of sciences and techniques.

In the development of our understanding of complex problems, the most powerful tool available to the human intellect is abstraction. When we have developed an abstract concept to cover the set of objects or situations in question, we will usually introduce a word or a picture to symbolize the abstract concept; and any particular spoken or written words and pictures may be used to represent a particular or a general instance of the corresponding situations. The last stage in the process of abstraction is very much more sophisticated; it is the attempt to symbolize the most general facts about objects and situations covered under an abstraction by brief but powerful axioms, and to prove rigorously that the results obtained by manipulating symbols can also successfuly be applied to the real world.

In another point of view, there exist a lot of problems that are easily solved through an interaction among some people. In fact, a well-known quotation "Two heads are better than one", suggests that such an interaction between two persons could induce more illuminating ideas.

In this thesis, standing upon the preceeding two viewpoints, formal models for the utilization of computers are discussed.

Therefore, this thesis is divided into two parts.

In Part I, the notion of a web grammar is generalized and discussed. This notion provides a general formalism for modeling a wide variety of data structures; in particular, relational structures such as those that arise in many problems.

In Part II, a formal definition of an interaction among some formal grammars is given, where the grammars can be considered as the formal models of various systems (for example, men, computers and their combinations, etc.).

February, 1975
Yoshinori Ezawa

TABLE OF CONTENTS

PART I. WEB LANGUAGES AND WEB AUTOMATA

PART II. INTERACTIVE LANGUAGES

LIST OF SYMBOLS

Symbol	Description
$A \supset B$	: A includes B
$A \supsetneq B$	: A properly includes B
$a \varepsilon A$	: a is an element of A
2^A	: power set of A
ϕ	: empty set
$\|A\|$	: number of elements of A
$\|w\|$	: length of a word w
w^R	: mirror image of a word w
$g_G(c)$	: a word derived by a control word c in grammar G
$ng_G(c)$	: nondeterministically derived words by a control word c in grammar G
$x \underset{G}{\Longrightarrow} y$	: direct derivation in grammar G
$w_1 \underset{G_1G_2}{\Longrightarrow} w_2$	: interactive derivation in an interactive system (G_1, G_2)
$w_1 \underset{G_1\ldots G_n}{\Longrightarrow} w_2$	: sequential n-cyclic interaction among $(G_1, \ldots, G_n)$
$w_1 \underset{\bar{G}_1\ldots G_n}{\Longrightarrow} w_2$	: parallel n-cyclic interaction among $(G_1, \ldots, G_n)$ with respect to G_1
$w_1 \underset{G_1G_2}{\Longmapsto} w_2$	: nondeterministic interaction in (G_1, G_2)

Part I

WEB LANGUAGES AND WEB AUTOMATA

CHAPTER 1

INTRODUCTION

Mankind can not live without any information. Everyone gathers all sorts of information, and decides his own purpose by analyzing and synthesizing them. With the progress of civilization, the amount of information to process would become too large. Therefore, it would be necessary for correct comprehension of various information to develop the effective methods of classification, extraction and storage of information.

Although the early electronic computers were designed simply as fast numerical calculators, it was soon recognized that they possessed a more general capability for processing many other types of suitably coded symbols, and the present computers are used for library cataloguing, reservation systems, literary analysis, pattern recognition and other tasks far removed from mathematics.

An important feature of many of such applications is that the items of information are inter-related in a complex way. For example, the main issue in scene analysis is to find a structural description by the picture primitives, and to compare it with all available data. In the computer programs, a well-constructed structure between the main program and several subroutine programs facilitates the proof of their correctness. In the case of natural language processing the thesauri play an important role.

The term data-structures may be used to cover the analysis of natural structures, the definition of suitable storage structures, and the execution of operations on those storage structures. In fact, since the amount of data is particularly large for every field of artificial intelligence, flexibility and inference capability in all storage structures are prerequisites for obtaining reasonable performances. It is interesting and essential to search for such a formal model.

In 1969 Pfaltz and Rosenfeld [1.23] introduced the notion of a web grammar, whose productions replace subwebs by subwebs. This notion provides a general formalism for modeling a wide variety of data structures, in particular, relational structures such as those that arise in artificial intelligence problems. Although research in this area is still somewhat tentative, it looks promising. Papers have been published on aspects of web grammars for various classes of graphs (Montanari [1.19], Rosenfeld and Strong [1.26], Pfaltz [1.22]), 'Chomsky hierarchies' for such grammars (Pavlidis [1.21], Abe, et al. [1.2-1.5], Ezawa et al. [1.10]), web (graph) acceptors (Shank [1.31], Milgram [1.18], Rosenfeld and Milgram [1.27], Ezawa et al. [1.8-1.9], Mylopoulos [1.20]), pattern analysis (Underwood and Kanal [1.37]), and data structure manipulation by web grammars (Torii et al. [1.35], Yamamoto et al. [1.38]).

Part I of this thesis treats several problems concerning with web grammars and web automata.

In Chapter 2 the generalized webs that are labelled both on their nodes and on their arcs are defined. Based on these webs, web grammars are redefined. In most papers the web rewriting rules have negative or positive contextual-conditions which are predicates. It is very difficult, however, to evaluate the predicates mechanically, because these conditions allow any type of controls depending on the types of predicates. Instead of these contextual-conditions, the well-known concept of programming method for the formal grammars (see [1.28]) is also introduced. Using this programming method, any normal monotone context-sensitive web grammar is shown to have a standard form. The concept of a standard form of a web grammar is developed independently by Milgram [1.18] for general embedding web grammars. Moreover, parallel web grammars and their languages are also discussed.

Chapter 3 treats the problems of a web automaton which can be formulated as a model of structural data-matching. It is shown that a set of webs is accepted by a web automaton if and only if it is generated by a normal monotone context-sensitive web grammar. Some well-known graphic operations are also extended on webs and their closure properties are investigated.

CHAPTER 2

WEB GRAMMARS

2.1 Introduction

The subject of web grammars has received considerable attention recently because of data processing and data matching. One basic difference between web and string grammar is the way in which elements can be juxtaposed in webs while there are only two for strings (right and left). Therefore, the definition of such relations is crucial in the description of web grammars and different models may result for different grammars that are simple enough to allow one to prove results about them and,at the same time, general enough to encompass models of earlier investigators.

One generalization about web grammars included in this model is to allow symbols which are not simply on nodes but on arcs.

In this chapter, the generalized webs which have symbols on all of their nodes and arcs are defined. Based on the concept of these extended webs, the web grammars are redefined and discussed. For any normal monotone context-sensitive web grammar, the standard form is established. Moreover, a concept of the parallel web grammar is introduced and its family of languages is discussed.

2.2 Web Grammars

In this thesis, a web grammar generates a set of directed graphs having symbols on all of their nodes and arcs.

Definition 2.1. A web W over a finite set of symbols V ($= V_a \cup V_f$) is denoted by a triplet (N_w, F_w, A_w), where V_f is a set of node symbols and V_a is a set of arc symbols.

(1) N_w is a finite set of nodes, where integers are used to distinguish some of them.

(2) F_w is a node labelling function form N_w into V_f.

(3) A_w is an arc labelling function from $N_w X N_w$ into 2^{V_a}. Specifically, when 2^{V_a} has two elements, this definition coinsides with that of previous papers [1.23, 1.19, 1.5].

A set of all webs on V ($=V_a \cup V_f$) is denoted as $(V_a \cup V_f)^*$.

Definition 2.2. A web grammar is a triplet (V, I, R), where:

(1) V is a finite set of symbols such that

$$V = V_a \cup V_f,\ V_a = V_{aN} \cup V_{aT} \text{ and } V_f = V_{fN} \cup V_{fT}.$$

(2) I is a set of initial webs, $I \subset (V_a \cup V_f)^*$.

(3) R is a finite set of rewriting rules. Every rewriting rule has a form $<\alpha, \beta, E>$. Both α and β are webs and E is a set of

predicates called the embedding method of the rewriting rule. The predicates of E specify the embedding of β in the host web W-α as follows. An image function from N_α into the subsets of $N_{\beta 1}$ is defined as

$$f : N_\alpha \longrightarrow 2^{N_{\beta 1}}, \quad \text{where } N_{\beta 1} \subset N_\beta .$$

The predicates of E specify whether or not each node of W-α is connected to any node of β by the image function, that is,

< i > for any p_1 in $N_{\beta 1}$ and p_3 in $N_{W-\alpha}$ there exists p_2 in $f^{-1}(p_1) \subset N_\alpha$ such that

$$A_W(p_2, p_3) \subset A_{W'}(p_1, p_3),$$

$$A_W(p_3, p_2) \subset A_{W'}(p_3, p_1).$$

< ii > for any p_4 in N_β - $N_{\beta 1}$ and p_3 in $N_{W-\alpha}$

$$A_{W'}(p_4, p_3) = A_{W'}(p_3, p_4) = 0,$$[1]

where W is a web to be rewritten and W' is a rewritten web.

[1] In an ordinary graph, 0 denotes that there exists no relation between the corresponding two verticies.

Definition 2.3. Given a web $W = (N_w, F_w, A_w)$, the web $S = (N_s, F_s, A_s)$ is said to be a <u>subweb</u> of W if:

(1) N_s is a subset of N_w,

(2) $F_s(p) = F_w(p)$, for any p in N_s,

(3) $A_s(p_1, p_2) = A_w(p_1, p_2)$, for any p_1 and p_2 in N_s.

Definition 2.4. Two webs $W_1 = (N_{w1}, F_{w1}, A_{w1})$ and $W_2 = (N_{w2}, F_{w2}, A_{w2})$ are <u>isomorphic</u> if:

(1) there exists a one to one mapping g from N_{w1} onto N_{w2},

(2) for any node p in N_{w1}, $F_{w1}(p) = F_{w2}(g(p))$,

(3) for any nodes p_1 and p_2 in N_{w1},

$$A_{w1}(p_1, p_2) = A_{w2}(g(p_1), g(p_2)).$$

Definition 2.5. A rewriting rule $<\alpha, \beta, E>$ is <u>applicable</u> to a web W whenever there exists a subweb of W which is isomorphic to the web α. When the result of the application of the rewriting rule to the web W is a web W', we use the notation as $W \underset{G}{\Rightarrow} W'$.

Definition 2.6. In each rewriting rule $<\alpha, \beta, E>$, if for each node p of N_α there exists exactly one image node f(p) in N_β and if f is a one to one mapping, then the grammar is said to be <u>normal</u>.

Definition 2.7. The web language L(G) characterized by a web grammar G consists of those webs over (V_{aT}, V_{fT}) that can be generated from the initial web by applying the rewriting rules. Formally,

$$L(G) = \{ W \mid W \in (V_{aT}, V_{fT})^{*}, W_0 \in I, W_0 \overset{*}{\underset{G}{\Rightarrow}} W \},$$

where derivation chain $\overset{*}{\underset{G}{\Rightarrow}}$ follow the same conventional definition as in the one-dimensional phrase-structure grammars [1.14].

Definition 2.8 A programmed web grammar G = (V, I, R, J) (p-wg) is a quadruplet, and has a set of production labels J. The production is written in the following format:

$$(j) \quad < \alpha_j, \beta_j, E_j > \quad S(T_j) \; F(U_j); \quad T_j, U_j \subset J.$$

In applying this production to an intermediate web W, W is scanned to see if the rewriting rule $<\alpha_j, \beta_j, E_j>$ is applicable. If so, the one occurrence of α_j in W is erased and β_j is embedded according to E_j, and then the next production to be applied to the ensuing web is selected from the success field T_j. If $<\alpha_j, \beta_j, E_j>$ is not applicable, then no change is made, and the next production is selected from the failure field U_j.

Especially if for every production the failure field U_j is empty, then the grammar is said to be a $\hat{p}$-web grammar.

Definition 2.9. A web grammar is said to be a monotone context-sensitive web grammar (nmcswg), if for any rewriting rule $<\alpha, \beta, E>$, the following two conditions are satisfied.

(a) For any node p of N_α, $f(p) \neq \phi$.

(b) For any node q of $N_{\beta 1}$, there exists a unique node $f^{-1}(q)$ in N_α.

Definition 2.10. The order of a web grammar is n, if the number of nodes of webs in any rewriting rules and that of initial webs are less than or equal to n.

Example 2.1. Fig. 2.1 shows an example of normal monotone context-sensitive web grammar which generates a set of all complete graphs $K_n(n = 1, 2, \ldots)$.[2] For instance, the complete graphs K_3, K_4 and K_5 are generated by this web grammar with the control words 345·13, 344568·12·13 and 344456789·11·68·12·13 respectively. In general, K_n $(n \geqq 5)$ is generated with a control word as follows:

[2] This web grammar is a counter example to Abe's Lemma 18 and Theorem 19 in [1.2] which say that no nmcswg can generate a set of all complete graphs.

V_N = {S, S_1, A, A_1, A_2, A_3, B, #, \$, \$'},

V_T = {a}

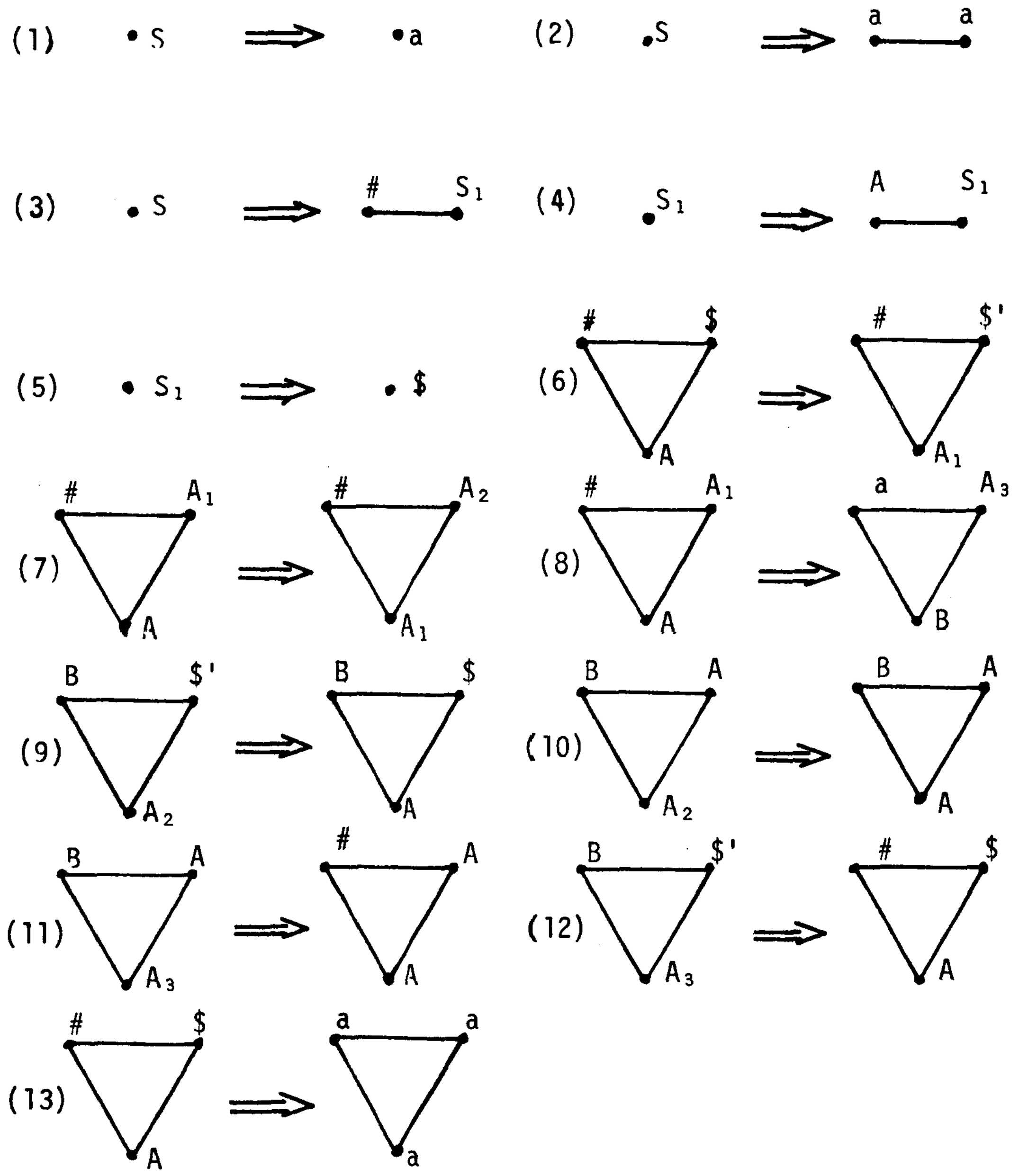


Figure 2.1 This nmcswg generates a set of all complete graphs {K_n}.

$$34^{n-2}5 \prod_{i=n-4}^{1} (67^{i}89\cdot 10^{i-1}\cdot 11)68\cdot 12\cdot 13 \quad ,$$

where $\prod$ denotes concatenation.

2.3 Standard Forms of Normal Monotone Context-Sensitive Web Grammars

In this section, it is shown that for any normal monotone context-sensitive web grammar (nmcswg) there exists a standard form. This standard form will be quite useful for constructing an automaton which can recognize a web language generated by an nmcswg.

Lemma 2.1. Given an nmcswg G, there exists a $\hat{p}$-nmcswg $G_{\hat{p}2}$ of order 2 which is equivalent to G.[3]

Proof. We assume that the initial web of G is a one-point web with no loss of generality (see[1.5]).

Since any embedding E is specified by the image function, hereafter we use the notation $\begin{bmatrix} n_1, & n_2,\dots \\ m_1, & m_2,\dots \end{bmatrix}$ instead of $f(n_1) = m_1$, $f(n_2) = m_2,\dots$ for E.

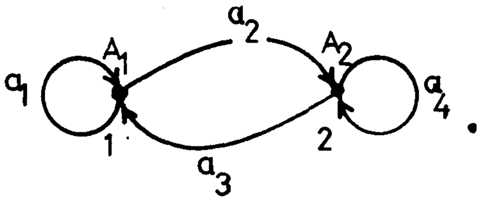
Moreover, any web $W = (N_w, F_w, A_w)$ such that

$$N_w = \{1, 2\},\ F_w(1) = A_1,\ F_w(2) = A_2,$$

[3] The two web grammars G_1 and G_2 are equivalent if and only if $L(G_1) = L(G_2)$.

$$A_W(1, 1) = a_1, A_W(1, 2) = a_2, A_W(2, 1) = a_3, A_W(2, 2) = a_4,$$

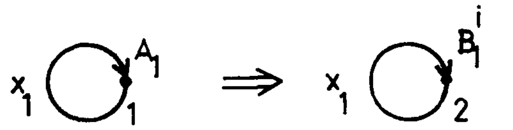
is abbreviated as follows:



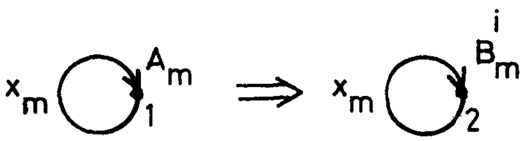
Now, we shall shown the proof which is constructed by two steps. Let a rewriting rule $< \alpha_i, \beta_i, E_i >$ be the i-th element of R of an arbitrary nmcswg G. We shall construct a set of order 2 $\hat{p}$-productions which is equivalent to the i-th rewriting rule.

(1) We examine whethere the web α_i is isomorphic to a certain subweb of W which will be rewritten.

<1.1> First, check up the node labelling function F_{α_i} with a following set of $\hat{p}$-productions.

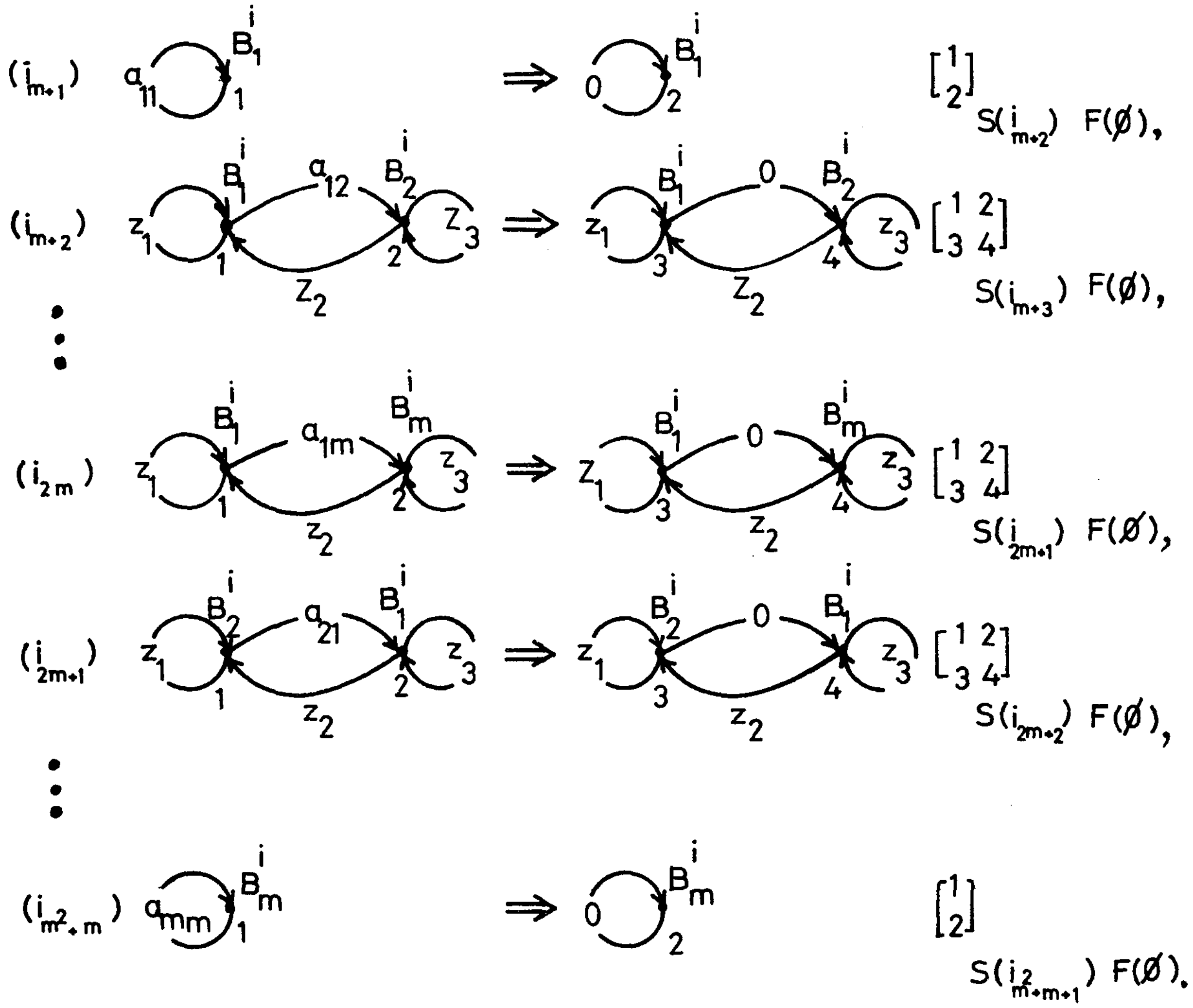
(i_1)  $\begin{bmatrix} 1 \\ 2 \end{bmatrix}$ $S(i_2)$ $F(\emptyset)$,

$\vdots$

(i_m)  $\begin{bmatrix} 1 \\ 2 \end{bmatrix}$ $S(i_{m+1})$ $F(\emptyset)$.

Here, $\{A_1, A_2, \ldots , A_m\}$ is a set of node symbols of α_i and m is the number of \ nodes :s of α_i. The set $\{B^i_1, B^i_2, \ldots , B^i_m\}$ is a set of non-terminal symbols newly added to V_{fN}. a_{jj} $(1 \leqq j \leqq m)$ is an element of 2^{V_a}.

<1.2> Next, check up the arc labelling function A_{α_i} sequentially via an arc, that is from a node p of N_{α_i} to any node q in N_{α_i}.



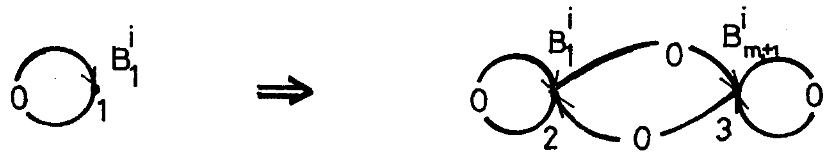
(i_{m+1})
a_{11} B^i_1 1 ⇒ 0 B^i_1 2
$\begin{bmatrix}1\\2\end{bmatrix}$ $S(i_{m+2})$ $F(\emptyset)$,
(i_{m+2})
z_1 B^i_1 a_{12} B^i_2 z_3 1 2 z_2 ⇒ z_1 B^i_1 0 B^i_2 z_3 3 4 z_2
$\begin{bmatrix}1&2\\3&4\end{bmatrix}$ $S(i_{m+3})$ $F(\emptyset)$,
(i_{2m})
z_1 B^i_1 a_{1m} B^i_m z_3 1 2 z_2 ⇒ Z_1 B^i_1 0 B^i_m z_3 3 4 z_2
$\begin{bmatrix}1&2\\3&4\end{bmatrix}$ $S(i_{2m+1})$ $F(\emptyset)$,
(i_{2m+1})
z_1 B^i_2 a_{21} B^i_1 z_3 1 2 z_2 ⇒ z_1 B^i_2 0 B^i_1 z_3 3 4 z_2
$\begin{bmatrix}1&2\\3&4\end{bmatrix}$ $S(i_{2m+2})$ $F(\emptyset)$,
(i_{m^2+m})
a_{mm} B^i_m 1 ⇒ 0 B^i_m 2
$\begin{bmatrix}1\\2\end{bmatrix}$ $S(i_{m^2+m+1})$ $F(\emptyset)$.

Here, $\{a_{11}, a_{12}, \ldots, a_{mm}\}$ is a set of arc symbols of α_i, and z_1, z_2 and z_3 are any arc symbols of the host web W. Therefore, these m^2+m $\hat{p}$-productions enable one to examine if the web W contains a subweb isomorphic to α_i or not.

(2) Now, the set of productions generating a web β_i is given.

<2.1> By the assumption, $|N_{\alpha_i}| \leqq |N_{\beta_i}|$ and G is normal. Therefore the image of any node of N_{α_i} must exist in N_{β_i}. The new nodes to be added are generated by the following productions:

(i_{m_1+1})

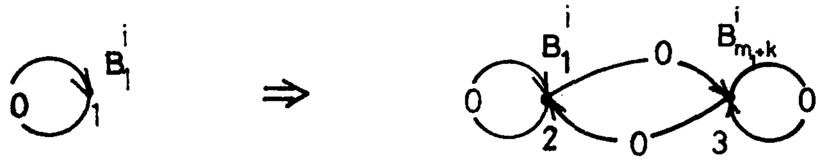


$\begin{bmatrix}1\\2\end{bmatrix}$

$S(i_{m+2})$ $F(\emptyset)$,

•
•
•

(i_{m_1+k})

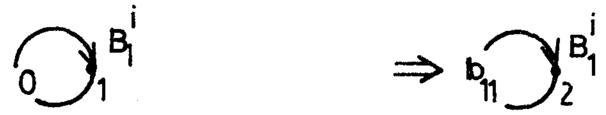


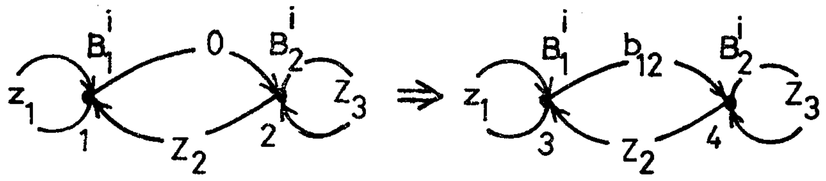
$\begin{bmatrix}1\\2\end{bmatrix}$

$S(i_{m_1+k+1})$ $F(\emptyset)$.

Here, $|N_{\beta_i}| = m + k = h$, $m_1 = m^2 + m$ and $m_2 = m_1 + k$. According to these productions, the node set $N_{\beta_i} - N_{\beta_i 1}$ is generated.

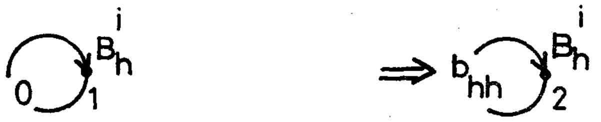
<2.2> Successively A_{β_i} is generated in order by the following h^2 $\hat{p}$-productions in the similar way as in step <1.2>.

(i_{m_2+1})  $\begin{bmatrix}1\\2\end{bmatrix}$ $S(i_{m_2+2})\ F(\emptyset)$,

(i_{m_2+2})  $\begin{bmatrix}1&2\\3&4\end{bmatrix}$

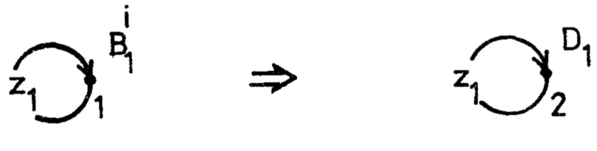
•
•
•

$S(i_{m_2+3})F(\emptyset)$,

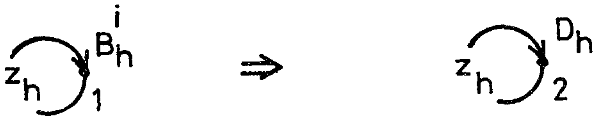
$(i_{m_2+h^2})$  $\begin{bmatrix}1\\2\end{bmatrix}$ $S(i_{m_2+h^2+1})\ F(\emptyset)$.

Here, $\{b_{11}, b_{12}, \ldots , b_{hh}\}$ is a set of arc symbols of the web β_i, and z_1, z_2 and z_3 are the elements of 2^{V_a}.

<2.3> Finally, by the following h $\hat{p}$-productions, F_{β_i} is constructed in the same manner as in step <1.1>.

(i_{m_3+1})  $\begin{bmatrix}1\\2\end{bmatrix}$ $S(i_{m_3+2})\ F(\emptyset)$,

•
•
•

(i_{m_3+h})  $\begin{bmatrix}1\\2\end{bmatrix}$ $S(\{j_1\})\ F(\emptyset)$.

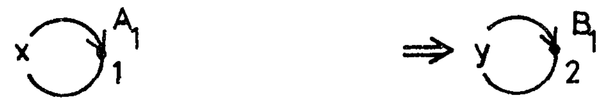
Here, $m_3 = m_2 + h^2$ and $\{D_1, D_2, \ldots, D_h\}$ is a set of node symbols of N_{β_i}. The success field of the last production, $\{ j_1 \}$, denotes a set of production labels, all having subscript 1. By assumption, the image function f is a one to one mapping from N_{α_i} onto $N_{\beta_i 1}$. The image specification in β_i is realized by the control of node symbols since every node of α_i is specified with B^i_j ($1 \leqq j \leqq m$) after the step <1.1>.

Finally, the rewriting rule $< \alpha_i, \beta_i, E_i >$ is simulated at most by the $|N_{\alpha_i}|^2 + |N_{\beta_i}| + 2|N_{\beta_i}|$ order-2 $\hat{p}$-productions. If a $\hat{p}$-programmaed web grammar $G_{\hat{p}2}$ has this set of $\hat{p}$-productions, then it is a $\hat{p}$-nmcswg such that $L(G_{\hat{p}2}) = L(G)$. Q. E. D.

From the proof of the previous lemma, it is reasonable to assume that any production of an arbitrary $\hat{p}$-nmcswg G with the order 2 has one of the following forms:

<I>

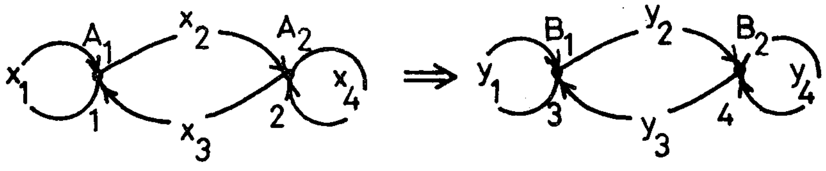
(j)



$\begin{bmatrix}1\\2\end{bmatrix}$ $S(T_j)$ $F(\emptyset)$,

<II>

(j)

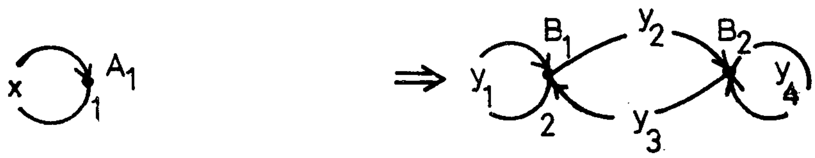


$\begin{bmatrix}1&2\\3&4\end{bmatrix}$

$S(T_j)$ $F(\emptyset)$,

<III>

(j)



$\begin{bmatrix}1\\2\end{bmatrix}$

$S(T_j)$ $F(\emptyset)$.

Definition 2.11. For any nmcswg G, an nmcswg G_2 which is equivalent to G and of order two is said to be a standard form of G.

Theorem 2.2. For every nmcswg G, there exists a standard form.

Proof. Given an arbitrary nmcswg G = (V, I, R), according to Lemma 1, there exists an equivalent p-nmcswg $G_{\hat{p}2} = (V_{\hat{p}}, I_{\hat{p}}, R_{\hat{p}}, J)$ of order two. Therefore, it is sufficient to show that for any $G_{\hat{p}2}$ there exists a standard form.

First, a set of productions with markers are constructed by repeating the procedures (a) and (b) below. Next, the control of the order of applying the productions is established according to the subscripts of markers appearing in the procedure 1 and 2 . Thus, an nmcswg G_2 of order two equivalent to the $\hat{p}$-nmcswg $G_{\hat{p}2}$ is obtained.

(a) Consider the case that, for a core rewriting rule $< \alpha_i, \beta_i, E_i >$ of the production in a set of productions of $G_{\hat{p}2}$ (P_1), there exists one node having a symbol S which is a node symbol of the initial web. If the image node of this vertex has a symbol v, then a $\hat{p}$-production of $P^m(1)$ whose core (i.e. $< \alpha_i', \beta_i', E_i' >$) is the same as $< \alpha_i, \beta_i, E_i >$ except for the symbols m_S and m_v used instead of S and v, respectively, has the same production label, the same success field and failure field as well. At this step, for instance, if the number of node with symbol are S is k, then $P^m(1)$ has k elements.

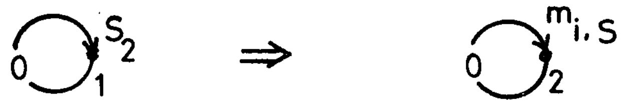
(b) Consider the case $k \geq 1$; (i) $F_\beta(N_\beta)$ of a certain production of $P^m(k)$ contains a marker symbol m_v; (ii) there exists a node whose label is v in α_j of the production (j) $< \alpha_j, \beta_j, E_j >$; (iii) the image of that node has a symbol u in β_j. If the preceding three conditions are all satisfied, then a $\hat{p}$-production of $P^+(k)$, whose core is the same as $< \alpha_j, \beta_j, E_j >$ except for replacing the symbols u and v with markers m_u and m_v, respectively, has the same production label and the same success fields. Therefore, let $P^m(k+1) = P^m(k) \cup P^+(k)$. To apply the procedures (a) and (b) repeatedly for any $i \geqq 1$, it is clear that

$$P^m(i) \subset P^m(n), \text{ where } n = |V_{\hat{p}}|.$$

Let a $\hat{p}$-web grammar $G_{\hat{p}m}$ have a set of productions $P^m = P^m(n) \cup P_1$. Then any production of P^m is one of the preceding forms (I), (II) and (III).

Now, we construct the rewriting rules R_2:

①



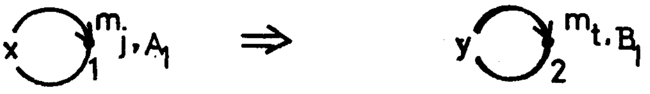
$\begin{bmatrix}1\\2\end{bmatrix} \in R_2(1)$.

Here, i denotes the label of the production whose left-hand side, α, is equal to $\{ \cdot S \}$ in P_1.

(2) Consider the case that $F_{\beta_i}(N_{\beta_i})$ contains a marker m_{j,A_1} ($A_1 \in V_f$) for any $k \geqq 1$.

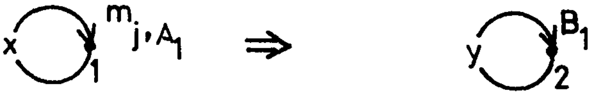
<2.1> When $F_{\alpha_i}(N_{\alpha_i})$ of production (j) in P^m contains a marker m_{A_1} :

(i) If the form of (j) is (I), then the rewriting rule



$\begin{bmatrix}1\\2\end{bmatrix}$; $t \in T_j$,

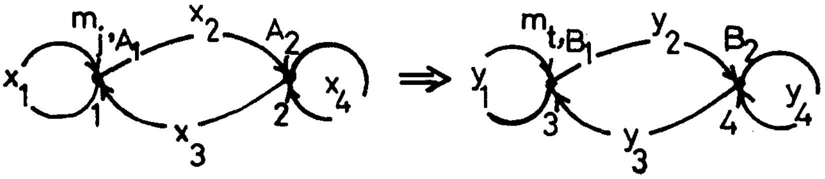
is contained in $R_2(k+1)$. But if T_j is empty, then the rule



$\begin{bmatrix}1\\2\end{bmatrix}$,

is contained in $R_2(k+1)$.

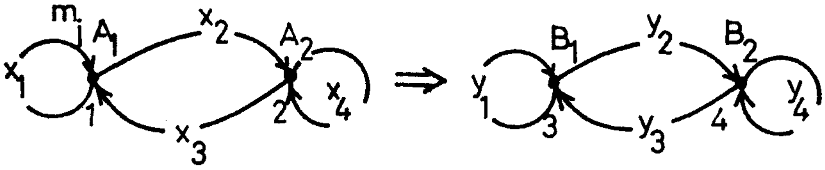
(ii) If the form of (j) is (II), then



$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}$

$\in R_2(k+1)$.

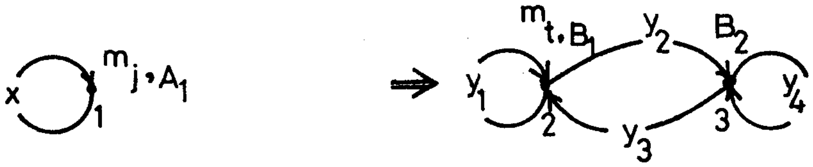
Here, $T_j(\ni t)$ is a success field of (j). But if T_j is empty, then



$\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}$

$\in R_2(k+1).$

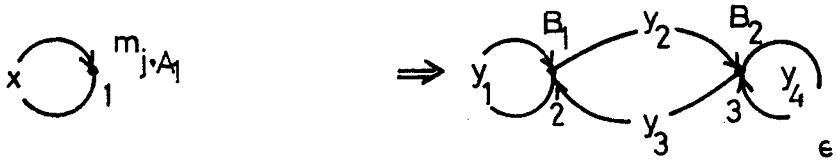
(iii) If the form of (j) is (III), then



$\begin{bmatrix}1\\ 2\end{bmatrix}$

$\in R_2(k+1).$

But if T_j is empty, then

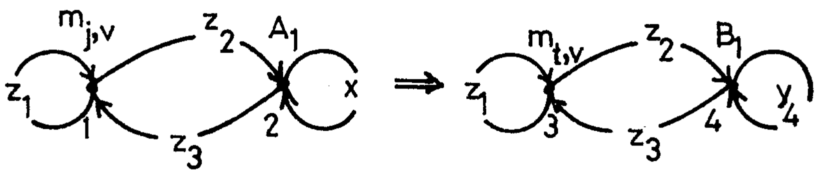


$\begin{bmatrix}1\\ 2\end{bmatrix}$

$\in R_2(k+1).$

<2.2 > When $F_{\alpha_i}(N_{\alpha_i})$ of production (j) in P^m does not contain a marker m_{A_1}:

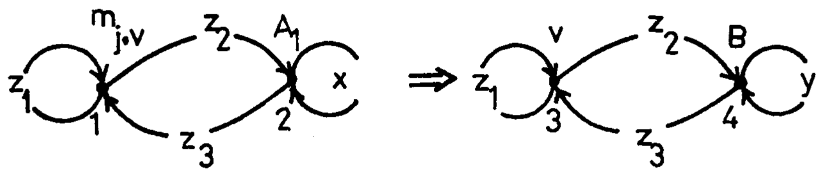
(i) If the form of (j) is (I), then



$\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}$

$\in R_2(k+1),$

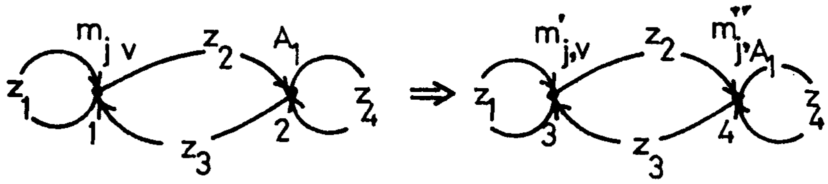
where v in V_f, t in T_j and z_1, z_2, z_3 in 2^{V_a}. But if T_j is empty, then



$\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}$

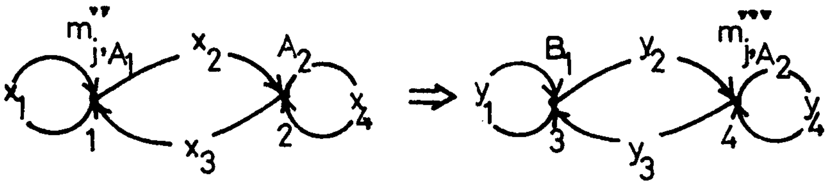
$\in R_2(k+1)$.

(ii) If the form of (j) is (II), then



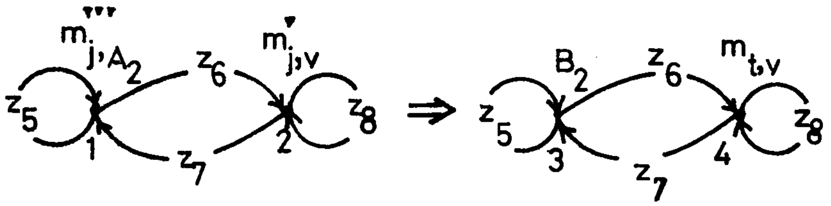
$\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}$

$\in R_2(k+1)$,



$\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}$

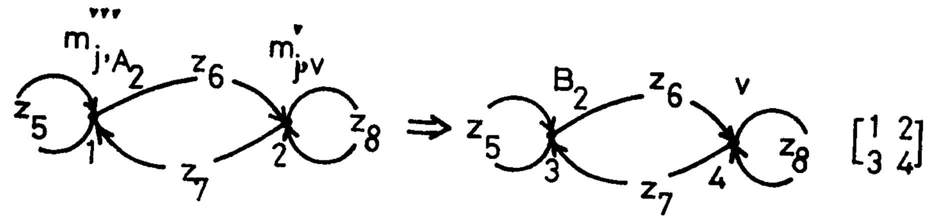
$\in R_2(k+1)$,



$\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}$

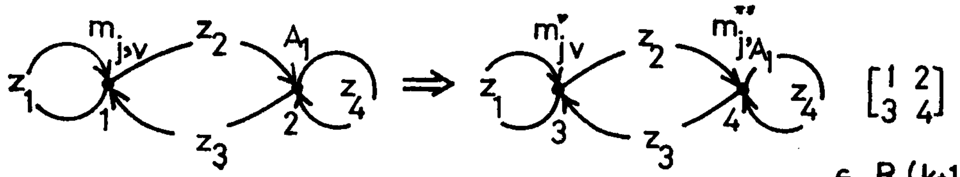
$\in R_2(k+1)$.

Here, z_1, z_2, ..., z_8 in 2^{V_a} and t in T_j. But if $T_j = \phi$, then the third rule above is replaced by the following rewriting rule:

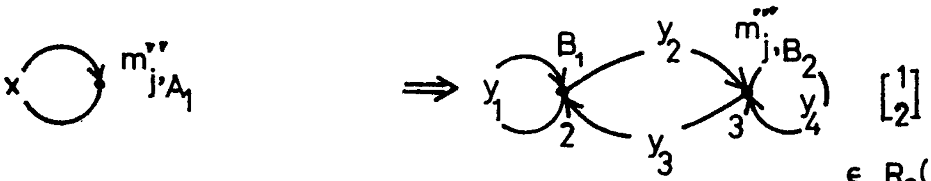


∈ $R_2(k+1)$.

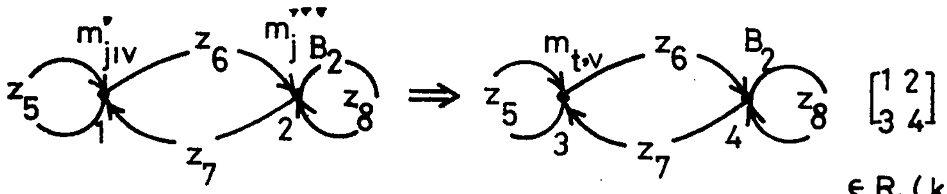
(iii) If the form of (j) is (III), then



∈ $R_2(k+1)$,

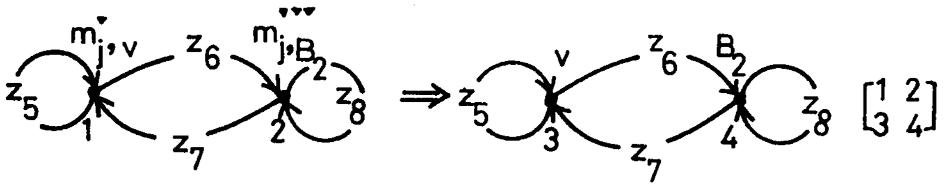


∈ $R_2(k+1)$,



∈ $R_2(k+1)$.

But if T_j is empty, then the third of these is replaced by the following rewriting rule:



$\in R_2(k+1)$.

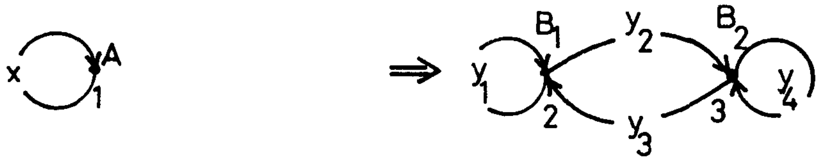
Now, applying the procedures ① and ② repeatedly it is clear that for any $i \geqq 1$, $R_2(K) \supset R_2(i)$ with $K = |V| \times |J|$. Thus, let $G_2 = (V_2, I_2, R_2)$ be a grammar such that (1) $V_2 = V \cup \{m_{i,v}, m'_{i,v}, m''_{i,v}, m'''_{i,v} \mid v$ in V_f, i in $J\} \cup \{S_2\}$, (2) $I_2 = \{\cdot S_2\}$, (3) $R_2 = R_2(K)$. An nmcswg G_2 thus obtained by the preceding procedures is of the order 2 and it is obvious that $L(G_2) = L(G)$. Q. E. D.

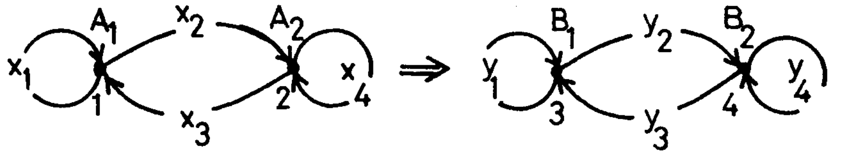
Lemma 2.3. The family of programmed normal monotone context-sensitive web languages (p-nmcsw$\mathcal{L}$) contains the family of non-normal monotone conext-sensitive web languages (anmcsw$\mathcal{L}$).

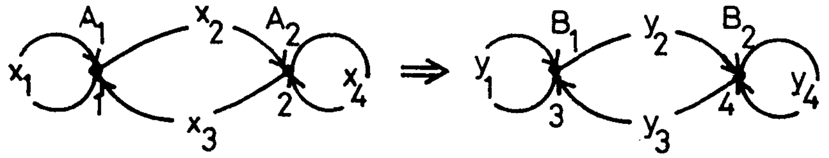
Proof. Similar to the normal case in previous Theorem 2.2, we can show that there exists a standard form for any non-normal monotone context-sensitive web grammar. Then we can suppose with no loss of generality that every monotone context-sensitive web grammar is of order 2.

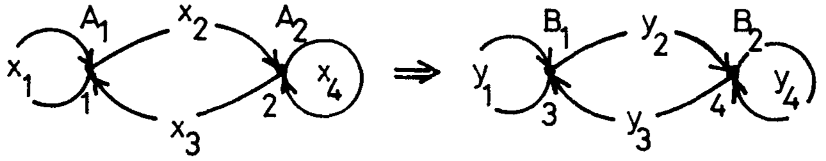
Now, let us show that for any order 2 monotone context-sensitive web grammar G there exists a programmed normal monotone context-sensitive web grammar PG which simulates that one. The main part of this proof is to show that the non-normal rewriting rules can be simulated by a set of programmed normal productions.

It is reasonable to assume that any non-normal rewriting rule has one of the following forms:

(A)  $\begin{bmatrix} 1 \\ \{2,3\} \end{bmatrix}$,

(B)  $\begin{bmatrix} 1 & 2 \\ 3 & 3 \end{bmatrix}$,

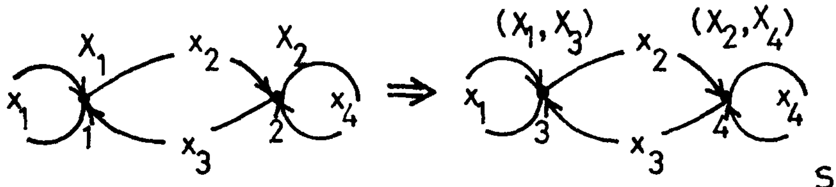
(C)  $\begin{bmatrix} 1 & 2 \\ \{3,4\} & 3 \end{bmatrix}$,

(D)  $\begin{bmatrix} 1 & 2 \\ \{3,4\} & \{3,4\} \end{bmatrix}$.

The cases (A), (B), (C) and (D) are discussed in a similar way, so we will show about the case (C).

Let the integer m be 2^{5a} X n (where a is $|V_A|$ and n is $|V_N|$). Consider the following programmed normal productions:

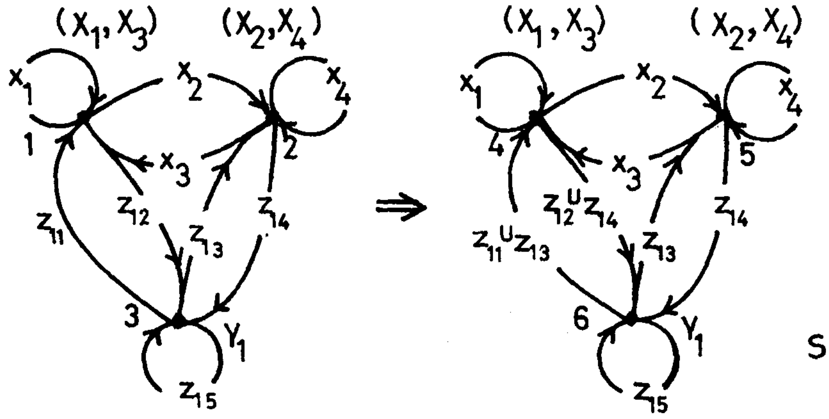
(C_0)



$\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}$

$S(C_1)\ F(C_1)$,

(C_1)

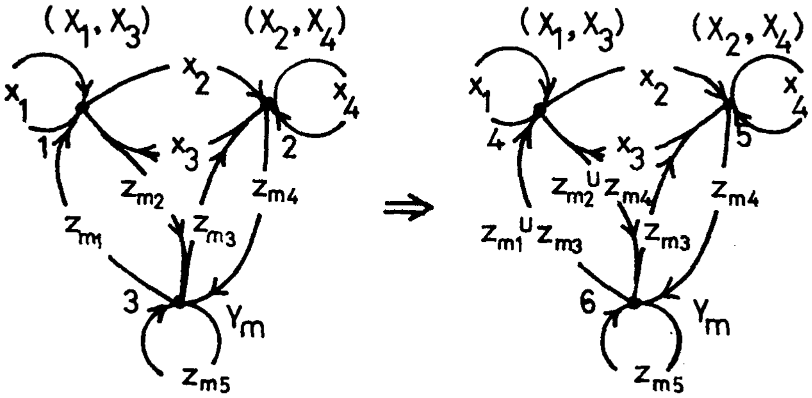


$\begin{bmatrix}123\\ 456\end{bmatrix}$

$S(C_2)\ F(C_2)$,

⋮

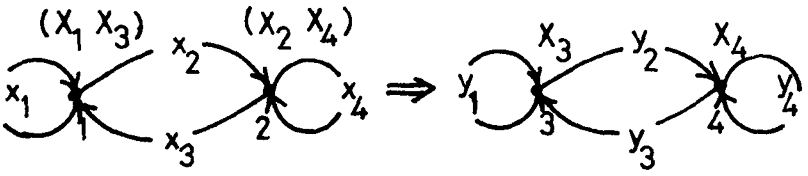
(C_m)



$\begin{bmatrix}123\\ 456\end{bmatrix}$

$S(C_{m+1})\ F(C_{m+1})$,

(C_{m+1})



$\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}$

$S(\text{all})\ F(\emptyset)$.

It is clear that the above productions just simulate the non-normal rewriting rule (C). Q. E. D

2.4 Web Languages

Web grammars for various classes of graphs and 'Chomsky hierarchies' for such grammars are under study by several investigators (Pavlidis [1.21], Abe et al.[1.5], Montanari [1.19]). In this section, the concept of a parallel web grammar is introduced. Next, the more general embedding method for webs (labeled directed graphs) is also proposed as the direction-sensitive embedding.

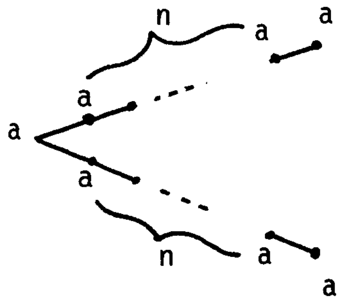
2.4.1 Parallel Web Languages

In data processing, a local operation is one which replaces a given data element by some data whose values depend on the value of the original element. This type of operation is analogous to a context-free web rewriting rule. On the other hand, local data processing operations are often applied to every element of the data in parallel, that is, using the original value of the element. Parallel languages for the string grammars have been investigated by many researchers [1.34], [1.24]. Especially in case of context-free type, L-systems are well developed field [1.17, 1.29, 1.30]. A lack of report on a parallel web grammar encouraged us to investigate it.

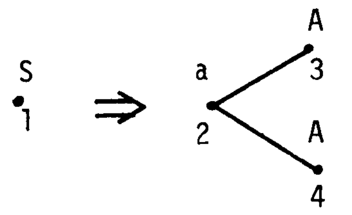
To start with, we shall define a parallel web language.

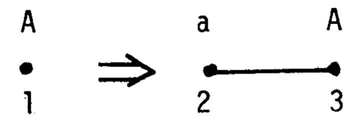
Definition 2.12· A parallel web language PL(G) is a web language which is a set of webs derived from the initial web by applying the rewriting rules of G in parallel: that is, when a rewriting rule is applied to an intermediate web every occurence of the left-hand side is replaced by the right-hand side. In this context G is said to be a parallel web grammar.

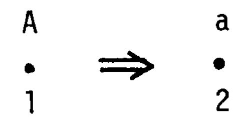
Example 2.2. The following is an example of a parallel normal context-free web language.

$PL(G_2) = \Big\{$  $\Big| \; n \geq 0 \Big\}.$

G_2: $V_N = \{S, A\}$, $V_T = \{a\}$,

(1)  f(1) = {2} ,

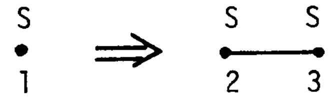
(2)  f(1) = {2} ,

(3)  f(1) = {2}.

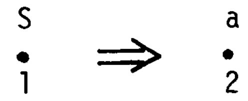
Example 2.3. The following is an example of a parallel non-normal context-free web language.

$PL(G_3) = \{K_{2^p} \mid p \geqq 0\}$[4].

G_3: $V_N = \{S\}$, $V_T = \{a\}$,

(1) 

$f(1) = \{2, 3\}$,

(2) 

$f(1) = \{2\}$.

Theorem 2.4. A family of parallel context-free web languages does not include a family of context-free web languages, and vice versa.

Proof Immediate from the following Lemma 2.5 and the above Examples. Q. E. D

Definition 2.13. An <u>n-star</u> S_n is a graph in which there exists a cut node v such that the graph S_n- v consists of a set of n trees whose each node's degree is at most 2.

[4] K_n denotes a complete graph which has n nodes [1.12].

Lemma 2.5. A set of stars SG = $\{S_n \mid n \geqq 2\}$ is not a parallel web language.

Proof. Assume that a parallel web grammar $G_p = (V_p, I_p, R_p)$ generates a set of stars SG. Consider the arbitrary integer m which is greater than the number of elements of the vocabulary V_p. Since the grammar G_p can generate the m-stars, there exists an intermediate web which contains a node p whose degree is m. There exist, however, at most $|V_p|$ distinct symbols on this intermediate web. As for all webs derived from this intermediate web in parallel by G_p, there exist at most $|V_p|$ distinct trees which are obtained by removing the cut node p from this web. This contradicts the assumption that $PL(G_p)$ equals $\{S_n \mid n \geqq 2\}$. Q. E. D

2.4.2 Direction-sensitive Web Grammars

In the original definition of a web grammar, the embeddings are defined only in terms of the image functions and they are independent of the arc direction. In this section, the direction-sensitive embedding for web grammars are defined: direction-sensitive embedding (abbreviated ds-embedding) can distinguish the two types of arcs (in and out) at each node.

Definition 2.14. Let G be a web grammar (V, I, R) where R is a set of rewriting rules and its element is formally described as a quadruplet $(\alpha, \beta, f_I, f_0)$, where α, β are webs, and two ds-embedding functions f_I and f_0 indicate the node to be connected as follows:

(1) $f_I : V_\alpha \longrightarrow V_\beta, f_0 : V_\alpha \longrightarrow V_\beta.$

(2) For any nodes p in W - α and q in α,

$$A_W(p, q) = A_{W'}(p, f_I(q)),$$
$$A_W(q, p) = A_{W'}(f_0(q), p),$$

where W is a web to be rewritten and W' is the rewritten web. Then G is termed a ds-web grammar.

It should be noticed that the string grammars are the special cases of the above ds-grammars. Therefore, in general we can make the following propositions:

Proposition 2.6. ds-lw$\mathcal{L}$ $\supsetneqq$ {regular set}.[5]

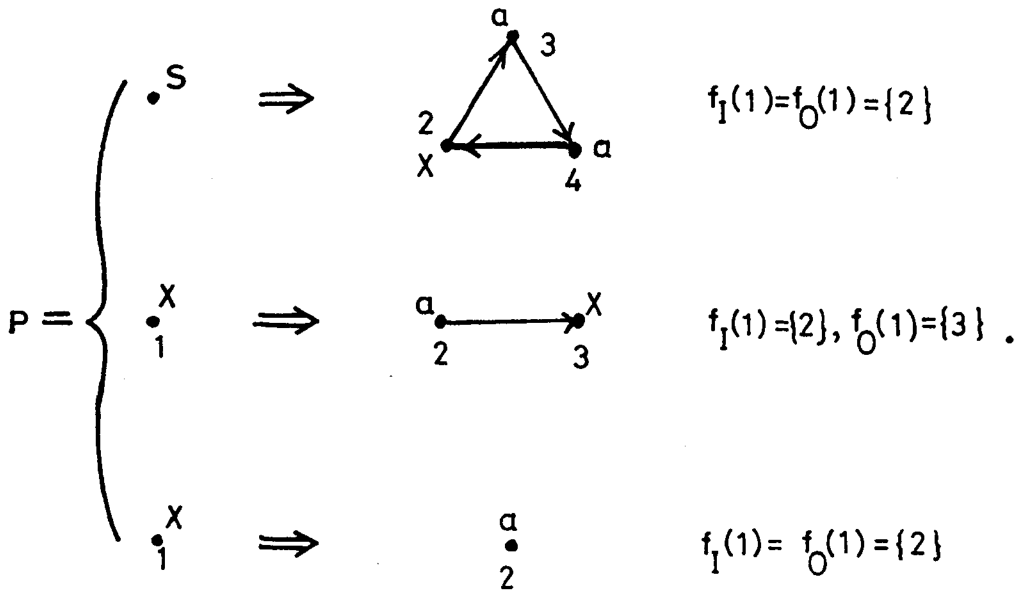
Proposition 2.7. ds-cfw$\mathcal{L}$ $\supsetneqq$ CF$\mathcal{L}$.

Proposition 2.8. ds-mcsw$\mathcal{L}$ $\supsetneqq$ CS$\mathcal{L}$.

[5] ds-lw$\mathcal{L}$ is a family of direction-sensitive linear web languages, the same holds for ds-cfw$\mathcal{L}$ and ds-mcsw$\mathcal{L}$.

Since the ds-embedding is an extention of the original embedding with respect to node mappings mentioned in Definition 2.2, generally the ds-web grammars are more powerful than the original web grammars. For example, the following ds-context-free web grammar generates a set of all circuits.

Example 2.4. Let G_4 = ({S, X} U{a}, { •S}, R_4),



$f_I(1)=f_O(1)=\{2\}$

$f_I(1)=\{2\}, f_O(1)=\{3\}$.

$f_I(1)= f_O(1)=\{2\}$

According to the above discussion, we have the following theorem immedistely.

Theorem 2.9. ds-cfw$\mathcal{L}$ $\supsetneqq$ n-cfw$\mathcal{L}$.

2.5 Conclusions

The generalized webs having symbols on all of their nodes and arcs are defined. An interesting example of a normal monotone context-sensitive web grammar which generates a set of all complete graphs is presented. The standard form of a web grammar is introduced, and it is shown that there exists a standard form for any normal monotone context-sensitive web grammar. Moreover, the concept of parallel web languages is proposed and it is shown that a family of parallel context-free web languages does not include a family of normal context-sensitive web languages, and vice versa. It is also shown that a family of direction-sensitive context-free web languages properly includes a family of normal context-free web languages.

CHAPTER 3

WEB AUTOMATA

3.1 Introduction

It is very attractive to consider machines which accept or recognize various types of graphs. The author believes that there is inherent merit in defining web acceptors (web automata). A web automaton in this thesis consists of a finite set of states, a head which traverses the web and a counter which checks up whethere the automaton's head has accessed all the nodes and arcs of the web before it halts. Milgram has defined a web automaton independently [1.18]. And Milgram's machine can simulate all the general functions such as splitting (σ) merging (μ) and negative checking up of the neighborhood's label (δ). The author thinks that it seems very hard for ordinary system to check-up whethere the neighboring nodes of the subject node have specified node labels, as for the function δ in Milgram's machine.

On the contrary, the web automaton in this thesis can only check up the labels of its subject nodes and arcs. And it is shown that a set of webs is accepted by a web automaton if and only if it is generated by a normal monotone context-sensitive web grammar. Moreover, the closure properties under some graphic operations of web languages are also discussed.

3.2 Web Automata

In this section, the representation of webs are defined, and it is shown that there exist automata which recognize the representation of normal monotone context-sensitive web languages.

Definition 3.1. The representation of any web W is denoted by an (n+1) X n matrix as follows:

$$R(W) = \begin{pmatrix} t_{1,1} & t_{1,2} & \cdots & t_{1,n} \\ t_{2,1} & t_{2,2} & \cdots & t_{2,n} \\ \cdot & & & \\ \cdot & & & \\ \cdot & & & \\ t_{n+1,1} & t_{n+1,2} & \cdots & t_{n+1,n} \end{pmatrix}$$

where

(1) $n = |N_W|$.

(2) F_W is a one-to-one mapping from N_W onto $\{t_{1,1}, \ldots, t_{1,n}\}$.

(3) $t_{i,j} = A_W(p, q)$, for any $n+1 \geq i \geq 2$ and $n \geq j \geq 1$, if $t_{1,i-1} = F_W(p)$ and if $t_{1,j} = F_W(q)$.

Generally speaking, it is useful to distinguish the input symbols from the other symbols of automata. Therefore, web automata are defined as follows.

Definition 3.2. A <u>web automaton</u> A is an 8-tuple such as $(Q, \Sigma, \Gamma, Z, M, \delta, q_0, F)$. Here,

(1) Q is a nonempty finite set of states.

(2) Γ is a nonempty finite set of symbols, $\# \in \Gamma$.

(3) Σ is a set of input symbols, $\Sigma \subset \Gamma$.

(4) Z is a set of nonnegative integers.

(5) $M = \{r, \ell\}$, where r denotes "relation" and ℓ denotes "label".

(6) $\delta : Q \times \Gamma \times Z \longrightarrow 2^{Q \times \Gamma \times Z \times M}$. If $\delta(q, \gamma, z)$ contains (q', γ', z', m), then the web automaton A replaces γ by γ', may enter state q' and moves the read-write head to "m", and z'=z or z-1. Moreover, if $\gamma = \#$, then $\gamma' = \#$ and $z = z'$.

(7) q_0 is an initial state of Q.

(8) F is a set of final states, $F \subset Q$.

Definition 3.3. The <u>configuration</u> C of a web automaton is represented as follows:

$$C = \begin{bmatrix} t_{1,1} & \cdots & t_{1,n} & \\ \vdots & qt_{i,j} & \vdots & ; z \\ t_{n+1,1} & \cdots & t_{n+1,n} & \end{bmatrix},$$

Here, $R(W) = (t_{i,j})$, q in Q and z in Z. As for the initial configuration C_0 , $qt_{i,j} = q_0t_{1,1}$ and $z = n^2 + n$.

Definition 3.4. The relation $\underset{A}{\vdash}$ is defined among all the configurations of any web automaton A as follows.

<1> If $\delta(q, t_{1,j}, z)$ contains $(q', t'_{1,j}, z', r)$, then

$$\begin{bmatrix} t_{1,1} \cdots qt_{1,j} \cdots t_{1,n} & \\ & ; \; z \\ t_{n+1,1} \quad \cdots \quad t_{n+1,n} & \end{bmatrix} \underset{A}{\vdash} \begin{bmatrix} t_{1,1} \cdots t'_{1,j} \cdots \quad t_{1,n} & \\ q't_{j+1,h} & ; \; z' \\ t_{n+1,1} \quad \cdots \quad t_{n+1,n} & \end{bmatrix}$$

where $1 \leq h \leq n$.

<2> If $\delta(q, t_{i,j}, z)$ contains $(q', t'_{i,j}, z', \ell)$, then for $i \neq 1$,

$$\begin{bmatrix} t_{1,1} \cdots\cdots t_{1,n} & \\ qt_{i,j} & ; \; z \\ t_{n+1,1} \cdots \; t_{n+1,n} & \end{bmatrix} \underset{A}{\vdash} \begin{bmatrix} t_{1,1} \cdots q't_{1,j} \cdots t_{1,n} & \\ t'_{i,j} & ; \; z' \\ t_{n+1,1} \quad \cdots \quad t_{n+1,n} & \end{bmatrix}$$

The movement of a read-write head of the web automaton can be determined according to the above definition. For example, consider a web automaton A with transition functions listed in Fig. 3.1-a. If A is in state q_1 reading A_1 on its input web, then it may go to state q_2, replace A_1 with B_1 and move to "r". At this step, there is

(a)

$$\delta(q_1, A_1, z) \ni (q_2, B_1, z, r)$$

$$\delta(q_2, a_1, z) \ni (q_3, a_{10}, z, \ell)$$

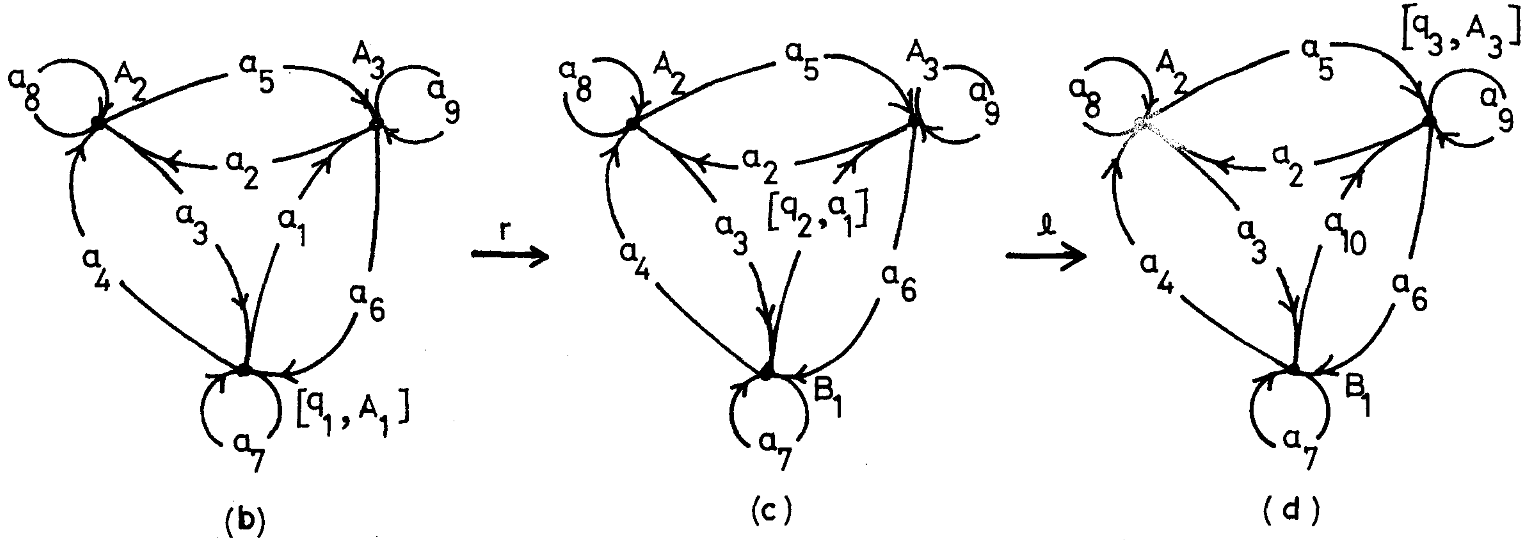


Figure 3.1. Head movements of a web automaton.

no way of specifying to which arc (e.g. a_1 or a_4) the head moves when δ is applied. That is, the head movement is non-deterministic. In the same manner, the web automaton A is in state q_3 (Fig. 3.1 - (d)) after the next step.

Between the two configurations C_1 and C_2 of a web automaton A, if a relation exists such that $C_1 = C_2$ or $C_1 = C_{10} \underset{A}{\vdash} C_{11} \underset{A}{\vdash} \cdots \underset{A}{\vdash} C_{1r} = C_2$, then the relation is denoted as $C_1 \overset{*}{\underset{A}{\vdash}} C_2$.

Definition 3.5. A web W is accepted by a web automaton A, if there exists a relation between its configurations such as

$$\begin{bmatrix} q_0 t_{1,1} \cdots t_{1,n} & \\ & ;\ n^2+n \\ t_{n+1,1} \cdots t_{n+1,n} & \end{bmatrix} \overset{*}{\underset{A}{\vdash}} \begin{bmatrix} u_{1,1} \cdots \quad u_{1,n} & \\ q_F u_{i,j} & ;\ 0 \\ u_{n+1,1} \quad u_{n+1,n} & \end{bmatrix} ,$$

where q_F is an element of F and the integer 0 at the final step ensures the perfect scanning of the given web W. A set of webs accepted by a web automaton A is denoted by T(A).

Definition 3.6. A web grammar is <u>bounded</u> if its initial web I is exactly a one-point web and if, for any rewriting rule $< \alpha, \beta, E >$, $|N_\alpha| = |N_\beta|$ except the case that α is isomorphic to I and $|N_\alpha| \leq |N_\beta|$.

This definition is a natural extension of a one-dimensional bounded grammar (see [1.16]). Therefore the following lemma is obtained immediately in the same way as in one-dimensional case.

Lemma 3.1. For any nmcswg G of order 2, there exists a bounded nmcswg G' of order 2 equivalent to G.

Theorem 3.2. If a web language L is generated by an nmcswg, then L is recognized by a web automaton.

Proof. Let G = (V, I, R) be an nmcswg such that L is equal to L(G). According to Lemma 3.1 and Theorem 2.2, there is no loss of generality in assuming that G is of order 2 and bounded. Moreover, the initial web of G is assumed to be { •S }. A web automaton A = $(Q, \Sigma, \Gamma, Z, M, \delta, q_0, F)$ which accepts L is constructed as follows.

(1) Q is a finite set of states given as follows:

<i> q_0, q_1, q_2 and q_F are the elements of Q.

<ii> For any v of V, q_v, q'_v, r_v and r'_v are the elements of Q.

(2) $\Sigma = 2^{V_{aT}} \cup V_{fT}$

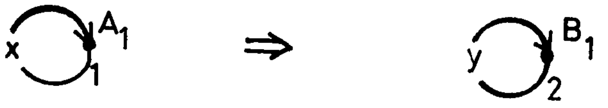
(3) $\Gamma = V \cup \{v', v'' \mid v \text{ in } V\} \cup \{\#\}$.

(4) q_0 is an initial state.

(5) q_F ia an element of F.

(6) Corresponding to the rewriting rules of R in the web grammar G, the transition function δ is defined as follows:

<6.1> If

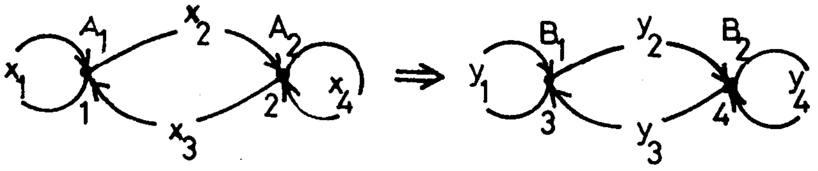


$\begin{bmatrix}1\\2\end{bmatrix} \in R$

is a rewriting rule of R, then

$\delta(q_0, B_1, z) \ni (q_{B_1}, B_1', z, r),$
$\delta(q_{B_1}, y, z) \ni (q_y, x, z, \ell),$
$\delta(q_y, B_1', z) \ni (q_0, A_1, z, r).$

<6.2> If



$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix} \in R$

is a rewriting rule of R, then

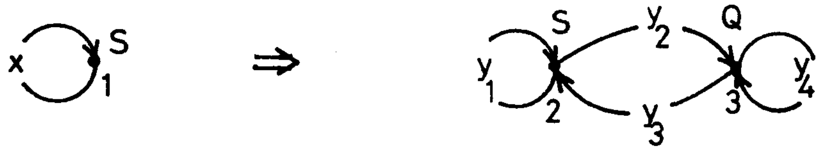
$\delta(q_0, B_1, z) \ni (q_{B_1}, B_1', z, r),$
$\delta(q_{B_1}, y_1, z) \ni (q_{y_1}, x_1, z, \ell),$
$\delta(q_{y_1}, B_1', z) \ni (q_{B_1'}, B_1'', z, r),$
$\delta(q_{B_1'}, y_2, z) \ni (q_{y_2}, x_2, z, \ell),$
$\delta(q_{y_2}, B_2, z) \ni (q_{B_2}, B_2', z, r),$
$\delta(q_{B_2}, y_4, z) \ni (q_{y_4}, x_4, z, \ell),$

$\delta(q_{y_4}, B_2', z) \ni (q_{B_2'}, A_2, z, r)$

$\delta(q_{B_2'}, y_3, z) \ni (q_{y_3}, x_3, z, \ell)$

$\delta(q_{y_3}, B_1'', z) \ni (q_0, A_1, z, r).$

<6.3> If



$\begin{bmatrix}1\\2\end{bmatrix} \in R$

is a rewriting rule of R, then

$\delta(q_0, Q, z) \ni (q_1, Q', z, r),$

$\delta(q_1, 0, z) \ni (q_1, \#, z-1, \ell),$

$\delta(q_1, Q, z) \ni (q_2, Q, z, r),$

$\delta(q_2, 0, z) \ni (q_2, \#, z-1, \ell),$

$\delta(q_2, Q', z) \ni (q_0, Q, z, r).$

Moreover,

$\delta(q_0, S, z) \ni (r_S, S', z, r),$

$\delta(r_S, y_1, z) \ni (r_{y_1}, x, z, \ell),$

$\delta(r_{y_1}, S', z) \ni (r_{S'}, S'', z, r),$

$\delta(r_{S'}, y_2, z) \ni (r_{y_2}, \#, z-1, \ell),$

$\delta(r_{y_2}, Q, z) \ni (r_Q, Q', z, r),$

$\delta(r_Q, y_4, z) \ni (r_{y_4}, \#, z\text{-}1, \ell)$,
$\delta(r_{y_4}, Q', z) \ni (r_{Q'}, \#, z\text{-}1, r)$,
$\delta(r_{Q'}, y_3, z) \ni (r_{y_3}, \#, z\text{-}1, \ell)$,
$\delta(r_{y_3}, S'', z) \ni (q_0, S, z, r)$,
$\delta(r_{y_3}, S, 1) \ni (q_F, \#, 0, r)$.

<6.4 > For all γ in Γ,

$\delta(q_0, \gamma, z) \ni (q_0, \gamma, z, m)$,

where m is equal to r or ℓ.

We shall sketch how this web automaton A accepts L(G). An automaton A in the state q_0 reads an arbitrary element of the representation R(W) of any input web W. Then in the state q_0, it reads an element of R(W') after several atomic actions $\overset{*}{\underset{A}{\vdash}}$. These actions correspond to the case such that

$$W' \overset{*}{\underset{G}{\Rightarrow}} W$$

for an nmcswg G. Therefore, the web language accepted by A is the set of those webs in $(V_{aT}, V_{fT})^*$ which causes A to enter a final state when placed in the representation of W with A in state q_0 and z = 0. That is;

$$\left[\begin{matrix} \# & \cdots & \# \\ \vdots & q_F\# & \vdots \\ \# & \cdots & \# \end{matrix} ; 0 \right].$$

Therefore L(G) includes T(A). Conversely, for every web W of L(G), there exists a derivation chain such as;

$$W_0 \underset{G}{\Rightarrow} W_1 \underset{G}{\Rightarrow} \cdots \underset{G}{\Rightarrow} W_n \underset{G}{\Rightarrow} W,$$

where W_0 is in I and W in L(G).

Accordingly, it is easy to show that T(A) includes L(G) because there exists a transition function δ of A which can simulate every rewriting rule conversely. Q. E. D.

Theorem 3.3. If a set of webs L is recognized by a web automaton A, then there exists an nmcswg G which generates L.

Proof. Let A =(Q, Σ, Γ, Z, M, δ, q_0, F) be a web automaton such that T(A) = L. Here,

$$Q = \{q_0, q_1, \ldots, q_m\}, \Gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_n\}.$$

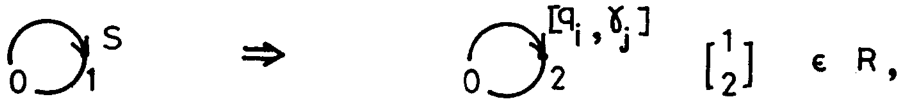
An nmcswg G = (V, I, R) which generates L is constructed as follows.

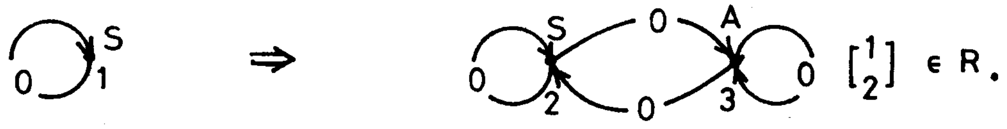
(1) $V = V_T \cup V_N$, $V_T = V_{aT} \cup V_{fT} = \Sigma$,
$V_N = V_{aN} \cup V_{fN} = (\Gamma - \Sigma) \cup [Q \times \Gamma] \cup \{A\}$.

(2) I = { •S }.

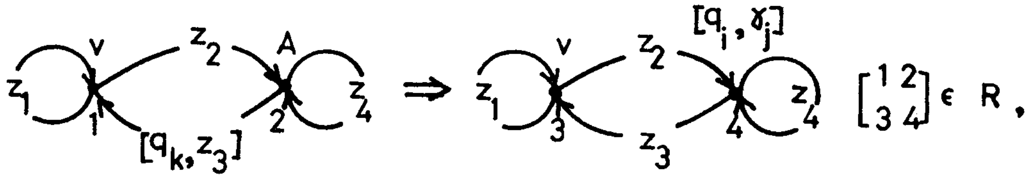
(3) R is constructed as follows.

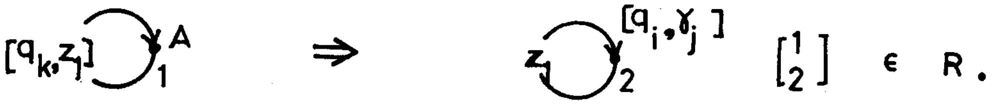
<3.1> If $\delta(q_i, \gamma_j, 1)$ includes $(q_F, \#, 0, m)$ where q_F is an element of F, then



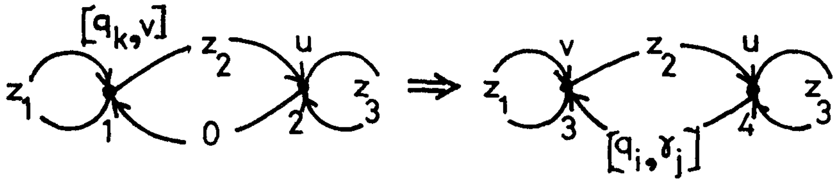


<3.2> If $\delta(q_i, \gamma_j, z)$ includes $(q_k, \#, z\text{-}1, r)$, then

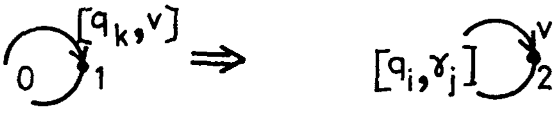




<3.3> If $\delta(q_i, \gamma_j, z)$ includes $(q_k, \#, z\text{-}1, \ell)$, then

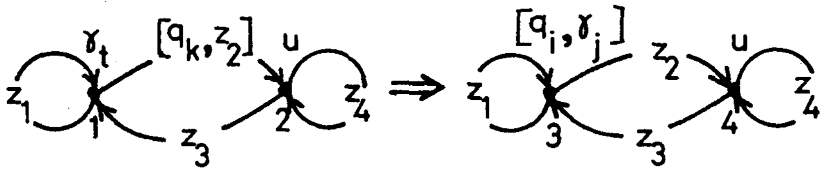


$\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix} \in R$,

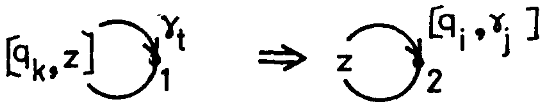


$\begin{bmatrix}1\\ 2\end{bmatrix} \in R$.

<3.4> If $\delta(q_i, \gamma_j, z)$ includes (q_k, γ_t, z, r), then

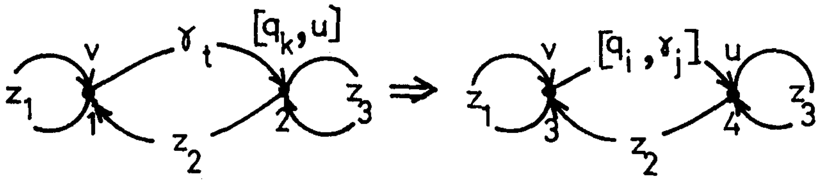


$\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix} \in R$,

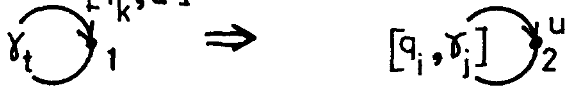


$\begin{bmatrix}1\\ 2\end{bmatrix} \in R$.

<3.5> If $\delta(q_i, \gamma_j, z)$ includes (q_k, γ_t, z, ℓ), then

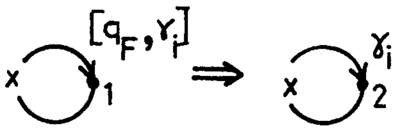


$\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix} \in R$,



$\begin{bmatrix}1\\ 2\end{bmatrix} \in R$.

<3.6> For any i (= 1, 2, ..., m),



$\begin{bmatrix}1\\2\end{bmatrix} \in R.$

For any web of T(A), there exists a chain of configurations as follows, where $R(W) = (t_{i,j})$,

$$\begin{bmatrix} q_0t_{1,1} & \cdots & t_{1,n} & \\ \vdots & \cdots & \vdots & ;\ n^2+n \\ t_{n+1,1} & \cdots & t_{n+1,n} & \end{bmatrix} \overset{*}{\underset{A}{\vdash}} \begin{bmatrix} u_{1,1} & \cdots & q_au_{1,j} & \cdots & u_{1,n} & \\ \vdots & & \cdots & & \vdots & ;\ z_1 \\ u_{n+1,1} & & \cdots & & u_{n+1,n} & \end{bmatrix}$$

$$\underset{A}{\vdash} \begin{bmatrix} u_{1,1} & \cdots & u'_{1,j} & \cdots & u_{1,n} & \\ \vdots & & q_bu_{j+1,i} & & \vdots & ;\ z_2 \\ u_{n+1,1} & & \cdots & & u_{n+1,n} & \end{bmatrix} \overset{*}{\underset{A}{\vdash}} \begin{bmatrix} v_{1,1} & \cdots & v_{1,n} & \\ \vdots & q_cv_{k,t} & \vdots & ;\ z_3 \\ v_{n+1,1} & \cdots & v_{n+1,n} & \end{bmatrix}$$

$$\underset{A}{\vdash} \begin{bmatrix} v_{1,1} & \cdots & q_dv_{1,t} & \cdots & v_{1,n} & \\ \vdots & & v'_{k,t} & & \vdots & ;\ z_4 \\ v_{n+1,1} & \cdots & & & v_{n+1,n} & \end{bmatrix} \overset{*}{\underset{A}{\vdash}} \begin{bmatrix} w_{1,1} & \cdots & q_ew_{1,g} & \cdots & w_{1,n} & \\ \vdots & & \cdots & & \vdots & ;\ 1 \\ w_{n+1.1} & & \cdots & & w_{n+1,n} & \end{bmatrix}$$

$$\underset{A}{\vdash} \left[\begin{array}{ccccccc} w_{1,1} & \cdots & w'_{1,g} & \cdots & w_{1,n} & & \\ & & q_F w_{g+1,h} & & & ; & 0 \\ w_{n+1,1} & & \cdots & & w_{n+1,n} & & \end{array}\right].$$

Corresponding to these configurations, there exists a derivation chain (shown as a sequence of representations of webs) of G as follows:

$$\begin{pmatrix} S \\ 0 \end{pmatrix} \overset{*}{\underset{G}{\Rightarrow}} \begin{pmatrix} A & \ldots & ASA & \ldots & A \\ 0 & & \cdots & & 0 \\ \vdots & & & & \vdots \\ 0 & & \cdots & & 0 \end{pmatrix} \overset{*}{\underset{G}{\Rightarrow}} \begin{pmatrix} w_{1,1} & \cdots & [q_e, w_{1,g}] & \cdots & w_{1,n} \\ \vdots & & & & \vdots \\ w_{n+1,1} & & \cdots & & w_{n+1,n} \end{pmatrix}$$

$$\overset{*}{\underset{G}{\Rightarrow}} \begin{pmatrix} v_{1,1} & \cdots & [q_d, v_{1;t}] & \cdots & v_{1,n} \\ \vdots & & v'_{k,t} & & \vdots \\ v_{n+1,1} & & \cdots & & v_{n+1,n} \end{pmatrix} \underset{G}{\Rightarrow} \begin{pmatrix} v_{1,1} & \cdots & v_{1,n} \\ \vdots & [q_c, v_{k,t}] & \vdots \\ v_{n+1,1} & \cdots & v_{n+1,n} \end{pmatrix}$$

$$\overset{*}{\underset{G}{\Rightarrow}} \begin{pmatrix} u_{1,1} & \cdots & u'_{1,j} & \cdots & u_{1,n} \\ & & [q_b, u_{j+1,j}] & & \\ u_{n+1,1} & & \cdots & & u_{n+1,n} \end{pmatrix} \underset{G}{\Rightarrow} \begin{pmatrix} u_{1,1} & \cdots & [q_a, u_{1,j}] & \cdots & u_{1,n} \\ & & \cdots & & \\ u_{n+1,1} & & \cdots & & u_{n+1,n} \end{pmatrix}$$

$$\overset{*}{\underset{G}{\Rightarrow}} \begin{pmatrix} [q_0,t_{1,1}] & \cdots & t_{1,n} \\ \vdots & & \vdots \\ t_{n+1,1} & \cdots & t_{n+1,n} \end{pmatrix} \underset{G}{\Rightarrow} \begin{pmatrix} t_{1,1} & \cdots & t_{1,n} \\ \vdots & & \vdots \\ t_{n+1,1} & \cdots & t_{n+1,n} \end{pmatrix} .$$

Therefore L(G) includes T(A). Since every rewriting rule of R corresponds to δ, it is clear that T(A) includes L(G). Q. E. D.

According to Theorem 3.2 and Theorem 3.3, there is a one to one correspondence between a set of nmcswgs and a set of web automata.

3.3 Closure Properties

We are interested in determining what operations preserve which classes of languages, the term 'preserve' means that the operations map languages in a class to languages in the same class. There are a number of reasons in the interest of this matter. First, knowing whether or not an operation preserves a given class of languages helps to characterize that class of languages. Second, it is often easier to determine that it is the result of various operations on other languages in the class, than by directly constructing a grammar for the language. Third, the knowledge obtained from a study of operations on languages can be used in proofs of theorems.

In this section we introduce operations such as union, converse, product, join and composition [1.12][1.10], to web languages of various types, and investigate the closure properties of web languages.

Definition 3.7. A converse web W^c of the web $W = (N, F, A)$ is a triplet (N, F, A^c) as follows:

$A^c(p, q) = A(q, p)$, for any nodes p and q in N.

The following definitions are natural extension of graphic binary operations. Let assume the operand webs to be $W_1 = (N_1, F_1, A_1)$ and $W_2 = (N_2, F_2, A_2)$, with the set of nodes N_1 and N_2 being mutually disjoint.

Definition 3.8. The union of the webs W_1 and W_2 is defined as $W_1 \text{ u } W_2 = (N_1 \cup N_2, F_1 \cup F_2, A^u)$, where $A^u = A_1 \cup A_2 \cup A_3$, and

$A_3(p, q) = A_3(q, p) = 0$, for any nodes p in N_1 and q in N_2.

Definition 3.9. The join of the webs W_1 and W_2 (Fig. 3.2 - c) is defined as $W_1 + W_2 = (N_1 \cup N_2, F_1 \cup F_2, A^+)$, where $A^+ = A_1 \cup A_2 \cup A_4$ such that for any nodes p in N_1 and q in N_2 there exist v_1 in V_1 and v_2 in V_2 as $A_4(p, q) = v_1$ and $A_4(q, p) = v_2$.

Definition 3.10. The product of the webs W_1 and W_2 is $W_1 \times W_2 = (N_1 \times N_2, F_1 \times F_2, A^x)$: (see Fig. 3.2 - d).

(1) For any $p = (p_1, p_2)$ and $q = (q_1, q_2)$ in $N_1 \times N_2$,

if $p_2 = q_2$ then $A^x(p, q) = A_1(p_1, q_1)$, else if $p_1 = q_1$, then $A^x(p, q) = A_2(p_2, q_2)$, otherwise $A^x(p, q) = A^x(q, p) = 0$.

(2) For any $p = (p_1, p_2)$ in $N_1 X N_2$, $F_1 X F_2(p) = (F_1(p_1), F_2(p_2))$.

Definition 3.11. The composition of the webs W_1 and W_2 (see Fig. 3.2 - e , f) is defined as $W_1 * W_2 = (N_1 X N_2, F_1 X F_2, A^*)$:

(1) $F_1 X F_2$ is the same as previous definition of product.

(2) For any $p = (p_1, p_2)$ and $q = (q_1, q_2)$ in $N_1 X N_2$, $A^*(p, q) = A_2(p_2, q_2)$ if $p_1 = q_1$, and $A^*(p, q) = A_1(p_1, q_1)$ otherwise.

It can be immediately known from the above definitions that the union is the special case of the join, and that the product is commutative if the vertex labelling function on $N_1 X N_2$ is commutative. Therefore, the extension of the above web operations to web languages is easy. Then the following theorems can be proved.

Theorem 3.4. The family of normal context-free web languages is closed under converse and union, but not closed under join, product and composition.

Theorem 3.5. The family of programmed normal monotone context-sensitive web languages is closed under converse, union, join, product and composition.

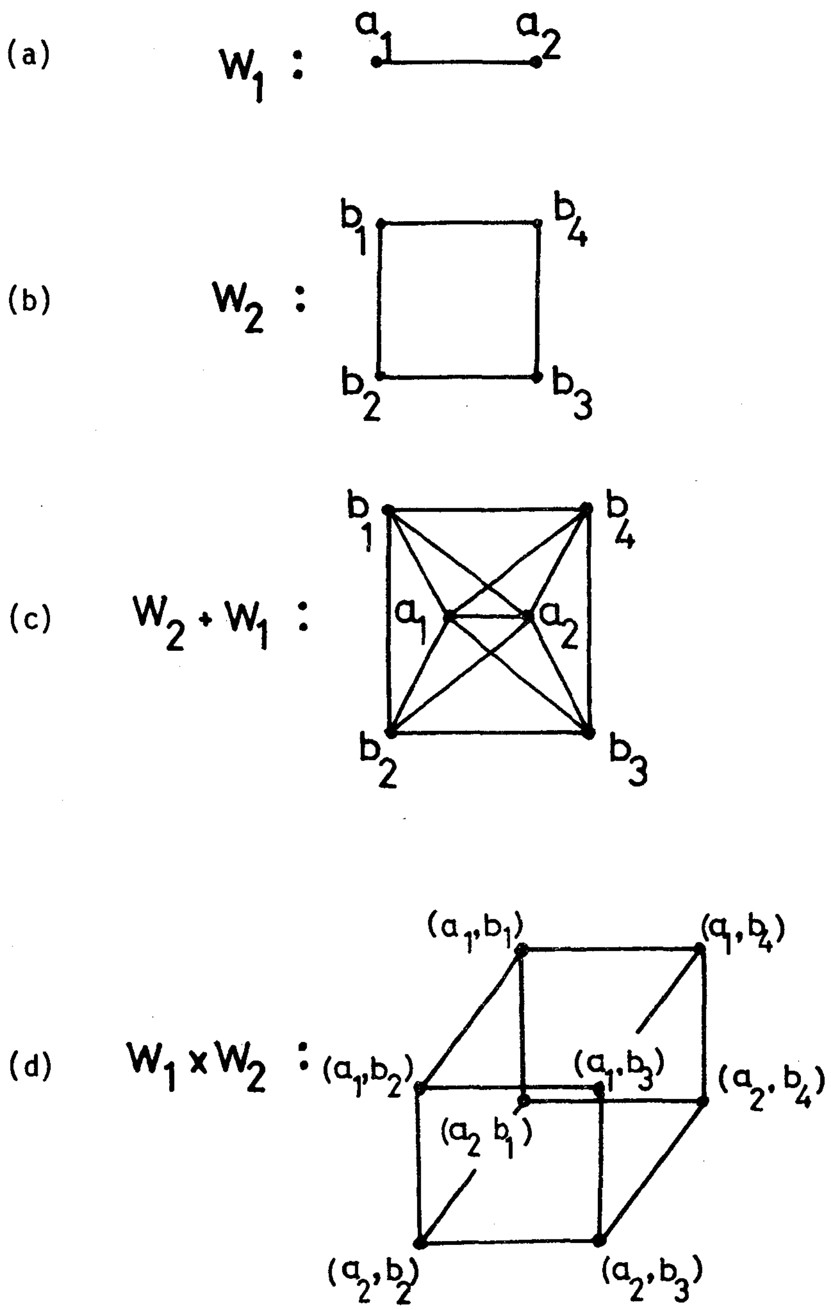


Figure 3.2. Operations on webs (continue).

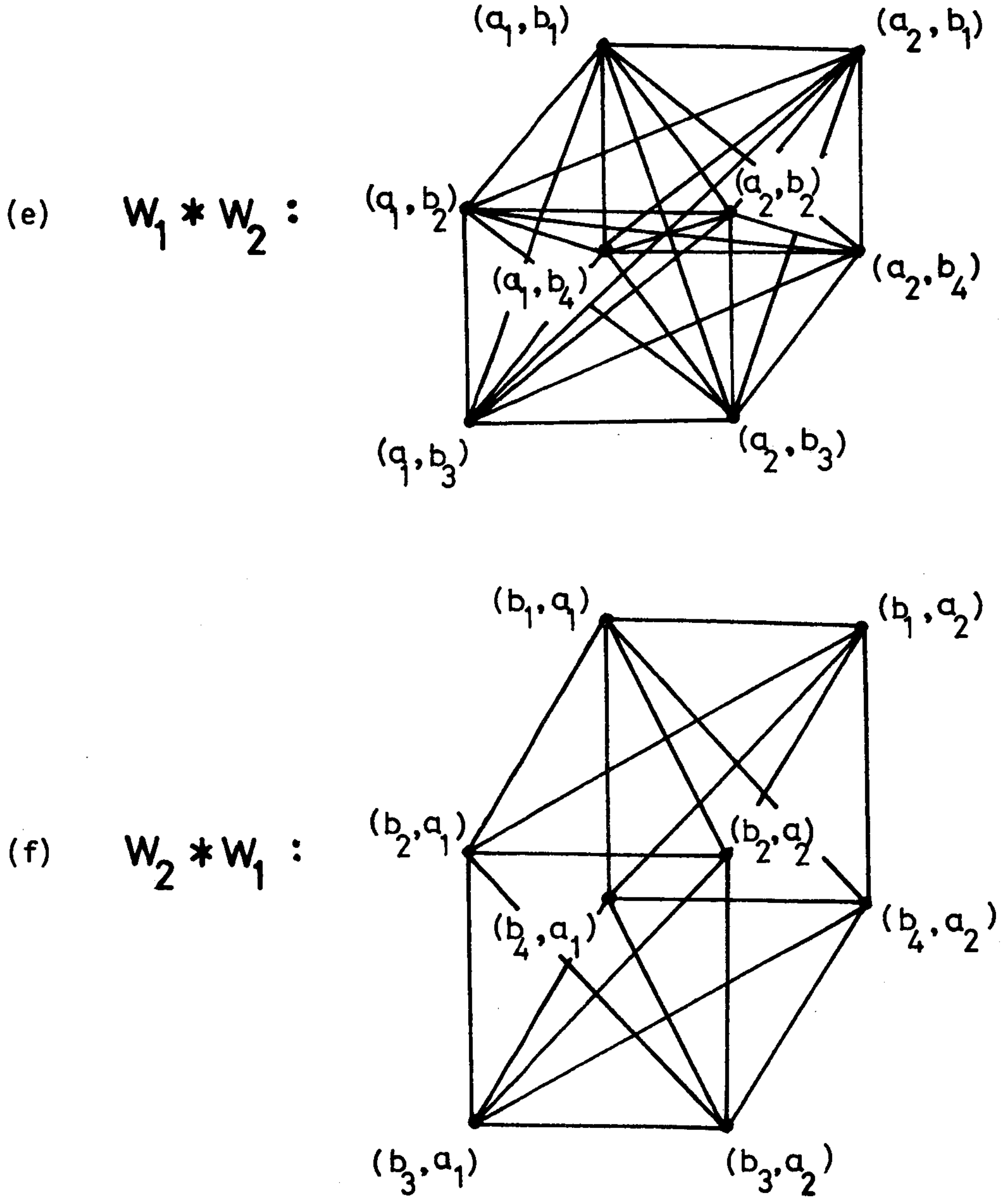


Figure 3.2. Operations on webs.

To begin with, we have Abe's lemma which follows.

Lemma 3.6 (Abe et al. [1.5]). The blocks of ncfwL L(G) generated by ncfwg whose initial webs consist of only one-point webs consist of only the right-hand side webs of the rules of G and the blocks of the right-hand side webs.

Proof of Theorem 3.4. Considering the Lemma 3.6 and the Examples in Fig. 3.2, it is clear that the family of normal context-free web languages is not closed under operations of join, product and composition. On the other hand, if the L(G) is a normal context-free web language, we can obtain a normal context-free web grammar G^c such as $L^c(G) = L(G^c)$ by conversing every right-hand side of the rewriting rule of G. With respect to union, it is trivial to show its closure property. Q. E. D.

3.4 Conclusions

A web automaton has been presented for a device to accept a set of webs. The equivalence relation between a set of webs accepted by a web automaton and a set of webs generated by a normal monotone context-sensitive web grammar is established.

And some well-known graphic operations are extended to webs. The closure properties of some web languages under these operations are also discussed. The family of normal context-free web languages is closed under union and converse, but not closed under join, product and composition.

CHAPTER 4

CONCLUSIONS

In the extended webs, it is proved that there exists a standard form for each normal monotone context-sensitive web grammar. Moreover, parallel web grammars and their languages are also investigated. It is shown that a family of parallel context-free web languages does not include a family of context-free web languages, and vice versa. Next, a web automaton has been presented as a device to accept a set of webs. And it is shown that a set of webs is accepted by a web automaton if and only if it is generated by a normal monotone context-sensitive web grammar.

The extended webs, in this thesis, are directed graphs having symbols on all of their nodes and arcs. Consequently, the web languages can be considered as a model of the languages which describe general data structures. The systems which recognize web languages can be considered as a model for explaining the recognition mechanism of data structures.

Numerous problems can be solved using either the present model or its modified version. These problems include, for example, the examination of the relationships between classes of programmed web languages, and the construction of the systems capable of recognizing nonnormal web languages.

Part II

INTERACTIVE LANGUAGES

CHAPTER 1

INTRODUCTION

The number of computers as well as their power, speed and memory size, and their applications are continuing to grow at an increasing rate. Along with this growth, a methodology of interactive systems is one of the most important problems in developing a system for the fields of applied computer science such as Artificial Intelligence, Information Retrieval, Pattern Recognition and so forth. With the development of computer techniques, the capabilities of machine problem solving in its general sense has so much increased that not only its problem solving methods, but also a certain amount of information which is necessary for problem solving can be gradually shifted from the human intelligence to the machine intelligence. This means that the solvable problem space is enlarged by means of an interactive system with its effective use of the advantageous characteristics of both sides.

As is hinted above, the object of an interactive system can be interpreted as the problem solving in any kind of application, e.g. information retrieval, computer-aided instruction, on-line process control, information inquiring and so on.

The problems of formalizing an interaction between two automata are under study by several investigators (Gabrielian [2.7], Kupka and Wilsing [2.12]). The new formalism of an interaction between two formal grammars are presented (Ezawa et al. [2.2-2.6]). The object of this work is to model such interactions among some formal grammars by using the well-known concept of control words [2.8].

In this thesis, the notions of interactive systems and interactive languages are proposed. An interactive language is a language defined by an interactive system, where an interactive system consists of two grammars which are controlled by each other with the words generated by them. Interactive systems may also be used to define interactions on webs. The problem of formalizing interactions between formal systems, even if they are purely formal grammars, is worthwhile to investigate and provides a new field.

Apart from the problem of an interaction itself between several grammars, another fact encourages the study of an interactive language. Recently, great efforts have been made to characterize the derivation chains of formal languages. Among them, L-languages [2.13], [2.17], [2.18], [2.20], [2.14], SF-languages [2.19] and Associate languages [2.15] are very interesting. Interactive languages may belong to that category. A peculiar feature of an interactive language is that it is generated by an interaction between two grammars in a simple way.

In Chapter 2 an interactive language is defined as a language which is generated in the process of a succeeding interaction between a pair of formal grammars, ana several characterizations of various classes of interactive languages are discussed. Some classes of interactive languages are compared with the so-called Chomsky's languages. It is shown that the well-known quotation from Homer's Iliad:

"Two heads are better than one."

is true for formal systems, too. For example, there exists an interactive system whose language is not a context-free language but a context-sensitive language, although it consists of two regular (or context-free) grammars.

It is also shown that the family of interactive languages is not closed under usual operations; it is an anti-AFL.

Chapter 3 treats an interaction among n grammars. First, a sequential n-cyclic interaction among n grammars is introduced. Next, a parallel n-cyclic interaction among n grammars is also discussed. It is shown that for any $n \geq 4$, there exists a parallel 3-cyclic interactive system which is equivalent to the parallel n-cyclic interactive system. This shows the fact that the following Japanese proverb:

"Three heads may induce the wisdom of Monju," (where Monju is the bodhisattva of wisdom and intellect.)

is true for formal systems, too.

In Chapter 4, a non-deterministic interactive language and an interactive web language are introduced. The concept of web interaction is a generalization of an interaction between two string grammars.

CHAPTER 2

INTERACTIVE LANGUAGES

2.1 Introduction

In this chapter, a type of formal system, called an interactive system, which may be used to define the interactions between two grammars is introduced.
interactive languages are shown. For example, a context-sensitive language $\{a^{2^n} b \mid b \geqq 0\}$ is generated by the interaction between two right-linear grammars. The family of interactive languages is denoted by $\mathscr{I}_{i-j}$ if the basis grammar is type i, in Chomsky's sense, and the type of the associate grammar is j. Then it is shown that a family $\mathscr{I}_{3-3}$ is a proper subfamily of $\mathscr{I}_{2-3}$ and $\mathscr{I}_{3-2}$. Moreover the families $\mathscr{I}_{2-3}$ and $\mathscr{I}_{3-2}$ are proper subfamilies of $\mathscr{I}_{2-2}$.

In the latter half of this chapter, the non-closure properties of the family of interactive languages under usual operations are presented: union, intersection, complement, the star operation, concatenation, homomorphism, inverse homomorphism, intersection with regular sets and mirror image. Therefore, every family of interactive languages is an anti-AFL [2.18].

2.2 Interactive Languages

First, a definition of an interactive language as a formalization of an interaction between several systems is introduced. For notations not explained in the sequel see [2.10].

Definition 2.1 A phrase-structure grammar is a 4-tuple $G = (V_N, V_T, P, S)$, where

(1) V_N is a finite set of nonterminal symbols.

(2) V_T is a finite set of terminal symbols.

(3) P is a finite set of productions $\alpha \longrightarrow \beta$, with α in $(V_N \cup V_T)^* V_N (V_N \cup V_T)^*$ and β in $(V_N \cup V_T)^*$.

(4) S is a start symbol in V_N.

Let $P_\ell = \{p_1, \ldots, p_r\}$ be a set of distinct labels for the productions in P and let

$$S = Q_0 \xrightarrow{p_{j(0)}} Q_1 \xrightarrow{p_{j(1)}} \cdots \xrightarrow{p_{j(m-1)}} Q_m, \quad m \geqq 1,$$

be a left-most derivation according to G, where for each i $(0 \leqq i \leqq m-1)$ the production labeled by $p_{j(i)}$ is $\alpha \longrightarrow \beta$ in P and there exist w_1 and w_2 such that $Q_i = w_1 \alpha w_2$ and $Q_{i+1} = w_1 \beta w_2$, where α occurs only once as a substring of $w_1\alpha$. Then the word $c = p_{j(0)} \cdots p_{j(m-1)}$ over the alphabet P_ℓ (label set) is termed a control word of the derivation, in symbols $g_G(c) = Q_m$. The mapping g_G from P_ℓ^* into $(V_N \cup V_T)^*$ is a function, since the word Q_m is uniquely defined by the control word c.

We now introduce the notion of interaction between a pair of phrase-structure grammars. The interaction is formulated as the process that two grammars mutually generate the control words each other for the derivation of another grammar.

Definition 2.2. Let $G_1 = (V_{N_1}, V_{T_1}, P_1, S_1)$ and $G_2 = (V_{N_2}, V_{T_2}, P_2, S_2)$ be two phrase-structure grammars, where $V_{T_1} \subset P_{\ell_2}$, $V_{T_2} \subset P_{\ell_1}$ (P_{ℓ_1} and P_{ℓ_2} are sets of labels of P_1 and P_2 respectively). Let

$$w_1 \underset{G_1G_2}{\Longrightarrow} w_2$$

be a relation between w_1 and w_2, where each of the following conditions is satisfied:

(1) w_1 and w_2 in $L(G_1) \subset V_{T_1}^* \subset P_{\ell_2}^*$

(2) There exists y in $L(G_2) \subset V_{T_2}^* \subset P_{\ell_1}^*$, such that
$y = g_{G_2}(w_1)$ and $w_2 = g_{G_1}(y)$.

In this context, the grammar G_1 is named a *basis grammar* and G_2 as an *associate grammar* and the word y over the alphabet V_{T_2} is called an *associate word* of w_2. The relation $\underset{G_1G_2}{\Longrightarrow}$ is termed an *interactive derivation* between G_1 and G_2. A pair of grammars (G_1, G_2) is named an *interactive system.*

If there is no cause for confusion, this relation will be abbreviated as $\Longrightarrow$, and its reflexive and transitive closure is denoted by $\overset{*}{\Longrightarrow}$.

Definition 2.3. The interactive language $L(G_1, G_2; w_0)$ generated by an interactive system (G_1, G_2) consists of those words in $V_{T_1}^*$ that can be interactively derived from the initial word w_0 in $L(G_1)$. Formally,

$$L(G_1, G_2; w_0) = \{w \in L(G_1) \mid w_0 \overset{*}{\underset{G_1G_2}{\Longrightarrow}} w\}.$$

A set of associate words generated by the associate grammar G_2 such that

$$A(G_1, G_2; w_0) = \{y \mid y \in L(G_2),\ g_{G_1}(y) \in L(G_1, G_2; w_0)\},$$

is called to be an associate language for the interactive language $L(G_1, G_2; w_0)$.

The family of interactive languages is denoted by $\mathscr{I}_{i-j}$ if the basis grammar G_1 is type i, in Chomsky's sense, and the type of the associate grammar is j ($i, j \in \{0, 1, 2, 3\}$).[6]

Example 2.1. The following is an example of an interactive language where $G_1 = (\{A\}, \{a, b\}, P_1, A)$ and $G_2 = (\{B\}, \{1, 2\}, P_2, B)$.

$$P_1 = \begin{cases} (1) \quad A \longrightarrow aA \\ (2) \quad A \longrightarrow b \end{cases}, \qquad P_2 = \begin{cases} (a) \quad B \longrightarrow 11B \\ (b) \quad B \longrightarrow 2 \end{cases}$$

[6] A right-linear (or left-linear) grammar is called to be type 3 in this thesis.

If ab is an initial word, then there exists an interactive derivation chain as follows:

$$\begin{tikzcd}
ab \arrow[rr, Rightarrow] \arrow[dr, dashed] & & aab \arrow[rr, Rightarrow] \arrow[dr, dashed] & & aaaab \arrow[rr, Rightarrow] \arrow[dr, dashed] & & \cdots \arrow[rr, Rightarrow] \arrow[dr, dashed] & & a^{2^n}b \arrow[rr, Rightarrow] \arrow[dr, dashed] & & \cdots \\
& 112 \arrow[ur, dashed] & & 11112 \arrow[ur, dashed] & & 1^8 2 & & 1^{2^n}2 \arrow[ur, dashed] & & 1^{2^{n+1}}2 &
\end{tikzcd}$$

The interactive language $L(G_1, G_2; ab)$ is $\{a^{2^n}b \mid n \geqq 0\}$, and this is a context-sensitive language.

Example 2.2. This example shows that various types of languages can be generated if the appropriate initial words are selected. Let $G_1 = (\{C\}, \{a, b, c\}, P_1, C)$ and $G_2 = (\{D\}, \{1, 2, 3\}, P_2, D)$, where

$$P_1 = \begin{cases} (1) & C \longrightarrow aC \\ (2) & C \longrightarrow bC, \\ (3) & C \longrightarrow c \end{cases} \qquad P_2 = \begin{cases} (a) & D \longrightarrow 12D \\ (b) & D \longrightarrow 2D. \\ (c) & D \longrightarrow 3 \end{cases}$$

Case 1. Let the initial word be bc, then there exists an interactive derivation chain as:

$$\begin{tikzcd}
bc \arrow[rr, Rightarrow] \arrow[dr, dashed] & & bc \arrow[rr, Rightarrow] \arrow[dr, dashed] & & \cdots \arrow[rr, Rightarrow] & & bc \arrow[rr, Rightarrow] \arrow[dr, dashed] & & \cdots \; . \\
& 23 \arrow[ur, dashed] & & 23 \arrow[ur, dashed] & & 23 \arrow[ur, dashed] & & 23 &
\end{tikzcd}$$

Thus, the interactive language is $L(G_1, G_2; bc) = \{bc\}$. This is a finite language.

Case 2. Let the initial word be ac, then there exists an interactive derivation chain as follows:

$$\begin{tikzcd}
ac \arrow[rr, Rightarrow] \arrow[dr, dashed] & & abc \arrow[rr, Rightarrow] \arrow[dr, dashed] & & abbc \arrow[rr, Rightarrow] \arrow[dr, dashed] & & \cdots \arrow[rr, Rightarrow] \arrow[dr, dashed] & & ab^i c \arrow[rr, Rightarrow] \arrow[dr, dashed] & & \cdots \\
& 123 \arrow[ur, dashed] & & 1223 \arrow[ur, dashed] & & 12^3 3 & & 12^i 3 \arrow[ur, dashed] & & 12^{i+1} 3 &
\end{tikzcd}$$

So the interactive language is $L(G_1, G_2; ac) = \{ab^n c \mid n \geqq 0\}$. This is a regular language, but not finite.

Case 3. Let the initial word be aac, then the interactive language $L(G_1, G_2; aac) = \{ab^n ab^n c \mid n \geqq 0\}$ is interactively derived in the same manner as in Case 2. This is a context-free language, but not a regular language.

Case 4. Let the initial word be aaac, then the interactive language $L(G_1, G_2; aaac) = \{ab^n ab^n ab^n c \mid n \geqq 0\}$ is interactively derived in the same manner as in Case 2. This is a context-sensitive language, but not a context-free language.

In each case, the type of the associate language can be shown to be the same as that of the interactive language.

The interactive language of Example 2.1 and Example 2.2 are the elements of a family $\mathcal{J}_{3-3}$, but the following Example 2.3 is an element of a family $\mathcal{J}_{2-3}$.

Example 2.3. The following interactive language is a well-known context-sensitive language and the associate language is a regular language. Let $G_1 = (\{S, X, Y, Z\}, \{a, b, c, \#, \$\}, P_1, S)$ and $G_2 = (\{T\}, \{1, 2, 3, 4, 5, 6, 7\}, P_2, T)$, where

$$P_1 = \left\{ \begin{array}{llll} (1) & S & \longrightarrow & \#XYZ\$ \\ (2) & X & \longrightarrow & aX \\ (3) & Y & \longrightarrow & bY \\ (4) & Z & \longrightarrow & cZ \\ (5) & X & \longrightarrow & a \\ (6) & Y & \longrightarrow & b \\ (7) & Z & \longrightarrow & c \end{array} \right. , \quad P_2 = \left\{ \begin{array}{llll} (\#) & T & \longrightarrow & 1T \\ (a) & T & \longrightarrow & T \\ (b) & T & \longrightarrow & T \\ (c) & T & \longrightarrow & 234T \\ (\$) & T & \longrightarrow & 567 \end{array} \right.$$

If the initial word is #abc$, then there exists an interactive derivation chain as:

$$\begin{tikzcd}
\#abc\$ \arrow[r, Rightarrow] \arrow[d, dashed] & \#aabbcc\$ \arrow[r, Rightarrow] \arrow[d, dashed] & \cdots \arrow[r, Rightarrow] \arrow[d, dashed] & \#a^nb^nc^n\$ \arrow[r, Rightarrow] & \cdots \\
1234567 \arrow[ur, dashed] & 1(234)^2567 & 1(234)^{n-1}567 \arrow[ur, dashed] & &
\end{tikzcd}$$

Consequently, the interactive language is $L(G_1, G_2; \#abc\$) = \{\#a^nb^nc^n\$ \mid n \geq 1\}$ and the associate language is $A(G_1, G_2; \#abc\$) = \{1(234)^n567 \mid n \geq 1\}$.

2.3 The Family of Interactive Languages and Chomsky's Hierarchy

In this section, the relationship between the classes of interactive languages and Chomsky's hierarchy is discussed.

The type of an interactive language $L(G_1, G_2; w_0)$ depends upon the type of its associate grammar G_2 as well as its basis grammar G_1. In particular, the interactive language whose associate grammar is a linear grammar has an interesting characterstic.

Lemma 2.1. If the language L contains two words w_1aa and w_2bb (where w_1, w_2 in V_T^* and a, b in V_T), it is not a member of a family of interactive languages whose associate grammars are linear grammars.

Proof. Consider the case that interactive language $L(G_b, G_a; w_0)$ contains two words w_1aa and w_2bb, and the associate grammar G_a is a linear grammar. There is no loss of generality in supposing that there exists an interactive derivation chain $w_1aa \overset{*}{\Longrightarrow} w_2bb$.

Since the associate grammar G_a is a linear grammar, its each production has the following form (i) or (ii).

$$\begin{array}{ll} \text{(i)}\ A \longrightarrow uBv & \\ & ,\quad A, B \text{ in } V_{N_a} \text{ and } u,v \text{ in } V_{T_a}^* . \\ \text{(ii)}\ A \longrightarrow u & \end{array}$$

If the form of production whose label is a is (i), the derivation in G_a controlled by w_1aa, should generate a sentential form which contains a non-terminal symbol. This contradicts with the hypothesis $w_1aa \overset{*}{\Longrightarrow} w_2bb$.

On the other case, if the production's form is (ii), there exists no derivation chain whose control word is w_1aa. The reason is that every sentential form derived by linear grammars has at most one non-terminal symbol. This is also a contradiction with the hypothesis. Q. E. D.

Corollary 2.2. The interactive language whose associate grammar is a linear grammar never contains two words w_1aw_2a and w_3bw_4b (where w_1, w_2, w_3, w_4 in $V_{T_1}^*$ and a, b in v_{T_1}).

Proof. It is clear from the proof of Lemma 2.1.

Q. E. D.

Theorem 2.3. The family of interactive languages $\mathcal{J}_{3-3}$ does not include the usual families $\mathcal{L}_2$, and $\mathcal{L}_3$, and vice versa.

Proof. It is easy to prove the former part from Lemma 2.1. The latter part is proved from the Example 2.1, and it is also shown that the interactive families $\mathcal{J}_{3-3}$ and $\mathcal{L}_3$ are not disjoint.

Q. E. D.

Lemma 2.4. $\mathcal{J}_{3-3} \subsetneq \mathcal{J}_{3-2}$.

Proof. It is known from Corollary 2.2 and the following Example 2.4. Q. E. D.

Example 2.4. Let $G_1 = (\{A\}, \{a, b, c, d\}, P_1, A)$ and $G_2 = (\{S, T\}, \{0, 1, 2, 3, 4\}, P_2, S)$, where

$$P_1 = \left\{ \begin{array}{llll} (0) & A & \longrightarrow & dA \\ (1) & A & \longrightarrow & aA \\ (2) & A & \longrightarrow & cA \\ (3) & A & \longrightarrow & bA \\ (4) & A & \longrightarrow & b \\ (5) & A & \longrightarrow & c \end{array} \right. , \qquad P_2 = \left\{ \begin{array}{llll} (a) & S & \longrightarrow & 1ST \\ (b) & T & \longrightarrow & 3 \\ (c) & S & \longrightarrow & 12 \\ (d) & S & \longrightarrow & 0S4 \end{array} \right.$$

Then the interactive language is $L(G_1, G_2; dc) = \{da^n cb^n \mid n \geqq 0\}$. This is a well-known context-free language and in a family $\mathcal{J}_{3-2}$.

Lemma 2.5. A language $L_2 = \{\#a_1^n a_2^n a_3^n a_4^n a_5^n\$ \mid n \geqq 0\}$ is not a member of the family of interactive languages $\mathcal{J}_{3-2}$.

Proof. Suppose that there exists two grammars G_1 and G_2, the former is type 2 and the latter is type 3, such that $L_2 = L(G_1, G_2; w_0)$. For an arbitrary positive integer i, the word $w_i = \#a_1^i a_2^i a_3^i a_4^i a_5^i\$$ is an element of L_2. If there exists the following relation:

$$w_i = \#a_1^i a_2^i a_3^i a_4^i a_5^i\$ \Longrightarrow \#a_1^j a_2^j a_3^j a_4^j a_5^j\$ = w_j,$$

then j equals i + 1. Therefore the variation rate in the number of occurence of the five letters $\{a_1, a_2, a_3, a_4, a_5\}$ is independent of the integer i. The rest of this proof is divided into two steps (I) and (II): one is the derivation step of the associate word $g_{G_2}(w_i)$ of w_i, and the other is the derivation step of the word w_j by the control word $g_{G_2}(w_i)$.

(I) Since $L(G_1)$ is the language on an alphabet which consists of seven letters, the set of rewriting rules of G_2 can be assumed to be as follows:

(#) $S_2 \longrightarrow s_1 Z s_2$,
(a_1) $A_1 \longrightarrow \alpha_1$,
(a_2) $A_2 \longrightarrow \alpha_2$,
(a_3) $A_3 \longrightarrow \alpha_3$,
(a_4) $A_4 \longrightarrow \alpha_4$,
(a_5) $A_5 \longrightarrow \alpha_5$,
($) $D \longrightarrow \delta$,

where the nonterminal symbols S_2, A_1, A_2, A_3, A_4, A_5, D are in V_{N_2}, the strings s_1, s_2, δ are in $V_{T_2}^*$, α_1, α_2, α_3, α_4, α_5 are in $(V_{N_2} \cup V_{T_2})^*$, and z is in $V_{N_2} \cup V_{N_2}(V_{N_2} \cup V_{T_2})^* V_{N_2}$.

Consequently, the associate word $g_{G_2}(w_i)$ is represented as $s_1\xi_1\xi_2\delta\xi_3\xi_4 s_2$, where ξ_1, ξ_2, ξ_3 and ξ_4 are in $(V_{N_2} \cup V_{T_2})^*$. The substrings, which are independent of the integer i, are s_1, δ

and s_2 in this associate word. The number of substrings whose length can be transformed from i to i + 1 by these three substrings s_1, s_2 and δ is at most four: such as ξ_1, ξ_2, ξ_3 and ξ_4. For this reason, it is impossible to control the variation in the length of the five strings independently of the integer i.

(II) According to the assumption, the basis grammar G_1 is type 3. So, let G_1 be a right-linear grammar, and its production set be as follows:

$$\begin{array}{lll}
(y_1) & S & \longrightarrow x_1S, \\
(y_2) & S & \longrightarrow x_2S, \\
\vdots & & \\
(y_m) & S & \longrightarrow x_mS, \\
(y_{m+1}) & S & \longrightarrow x_{m+1}, \\
\vdots & & \\
(y_k) & S & \longrightarrow x_k,
\end{array}$$

where the nonterminal symbol S is in V_{N_1} and the strings x_j are in $V_{T_1}^*$ for $1 \leq j \leq k$. If the associate word of the word w_i is $y = y_{f(1)}y_{f(2)}\cdots y_{f(q)}$, then $g_{G_1}(y)$ is $x_{f(1)}x_{f(2)}\cdots x_{f(q)}$. Thus the control word y should have been completed of variation in the length of five substrings.

The above discussion is similar if the basis grammar G_1 is left-linear grammar.

The discussions (I) and (II) contradict the assumption that $L_2$is $L(G_1, G_2; w_0)$. Q. E. D.

Theorem 2.6. The family of interactive languages $\mathcal{J}_{2-2}$ properly includes the family $\mathcal{J}_{3-2}$, and the family $\mathcal{J}_{2-3}$ properly includes the family $\mathcal{J}_{3-3}$.

Proof. This follows directly from Lemma 2.5 and next Example 2.5. Q. E. D.

Example 2.5. This example may complete the proof of Theorem 2.6. Let $G_1 = (\{S, X_1, X_2, X_3, X_4, X_5\}, \{\#, \$, a_1, a_2, a_3, a_4, a_5\}, P_1, S)$ and $G_2 = (\{Y\}, \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}, P_2, Y)$, where

$$P_1 = \left\{ \begin{array}{lll} (1) & S & \longrightarrow \#X_1X_2X_3X_4X_5\$ \\ (2) & X_1 & \longrightarrow a_1X_1 \\ (3) & X_2 & \longrightarrow a_2X_2 \\ (4) & X_3 & \longrightarrow a_3X_3 \\ (5) & X_4 & \longrightarrow a_4X_4 \\ (6) & X_5 & \longrightarrow a_5X_5 \\ (7) & X_1 & \longrightarrow a_1 \\ (8) & X_2 & \longrightarrow a_2 \\ (9) & X_3 & \longrightarrow a_3 \\ (10) & X_4 & \longrightarrow a_4 \\ (11) & X_5 & \longrightarrow a_5 \end{array} \right. , \quad P_2 = \left\{ \begin{array}{lll} (\#) & Y & \longrightarrow 1Y \\ (a_1) & Y & \longrightarrow Y \\ (a_2) & Y & \longrightarrow Y \\ (a_3) & Y & \longrightarrow Y \\ (a_4) & Y & \longrightarrow Y \\ (a_5) & Y & \longrightarrow 23456Y \\ (\$) & Y & \longrightarrow 789\cdot 10\cdot 11 \end{array} \right.$$

Thus, $L(G_1, G_2; \#a_1a_2a_3a_4a_5\$) = \{\#a_1^na_2^na_3^na_4^na_5^n\$ \mid n \geqq 1\}$.

Lemma 2.7. $\mathcal{I}_{2\text{-}3} \subsetneqq \mathcal{I}_{2\text{-}2}$.

Proof. It is obviously obtained from Lemma 2.1 and Example 2.4. Q. E. D.

Theorem 2.8. $\mathcal{I}_{3\text{-}3} \subsetneqq \mathcal{I}_{2\text{-}2}$.

Proof. The relations $\mathcal{I}_{3\text{-}3} \subset \mathcal{I}_{2\text{-}3} \subset \mathcal{I}_{2\text{-}2}$ are direct consequences of the definitions. Therefore, the proof is clear from Lemma 2.7. Q. E. D.

In our previous examples, the length of the control words increase monotonously. The next proposition shows that this is not true in general.

Proposition 2.9. There exists an interactive derivation chain in which the lengths of the control words do not increase monotonously.

Example 2.6. This example may complete the proof for the Proposition 2.9. Let $G_1 = (\{S\}, \{a_0, a_1, b, \alpha, \beta\}, P_1, S)$ and $G_2 = (\{A\}, \{0, 1, 2, 3, 4\}, P_2, A)$, where

$$P_1 = \begin{cases} (0) & S \longrightarrow S \\ (1) & S \longrightarrow bS \\ (2) & S \longrightarrow a_0a_1S \\ (3) & S \longrightarrow \alpha \\ (4) & S \longrightarrow b\beta \end{cases}, \qquad P_2 = \begin{cases} (a_0) & A \longrightarrow 0A \\ (a_1) & A \longrightarrow 1A \\ (b) & A \longrightarrow 2A \\ (\alpha) & A \longrightarrow 4 \\ (\beta) & A \longrightarrow 3 \end{cases}$$

If the initial word is bβ, there exists an interactive derivation chain as follows:

$$b\beta \Longrightarrow a_0a_1\alpha \Longrightarrow bb\beta \Longrightarrow (a_0a_1)^2\alpha \Longrightarrow b^3\beta \Longrightarrow \ldots$$

$$\ldots \Longrightarrow b^i\beta \Longrightarrow (a_0a_1)^i\alpha \Longrightarrow b^{i+1}\beta \Longrightarrow \ldots$$

The length of control words do not change monotonously as shown in Fig. 2.1.

Let us consider the properties of languages which are not members of interactive family to characterize indirectly the family of interactive languages.

Lemma 2.10. The empty set, ϕ, is not an interactive language.

Proof. Every interactive language contains an initial word. Q. E. D.

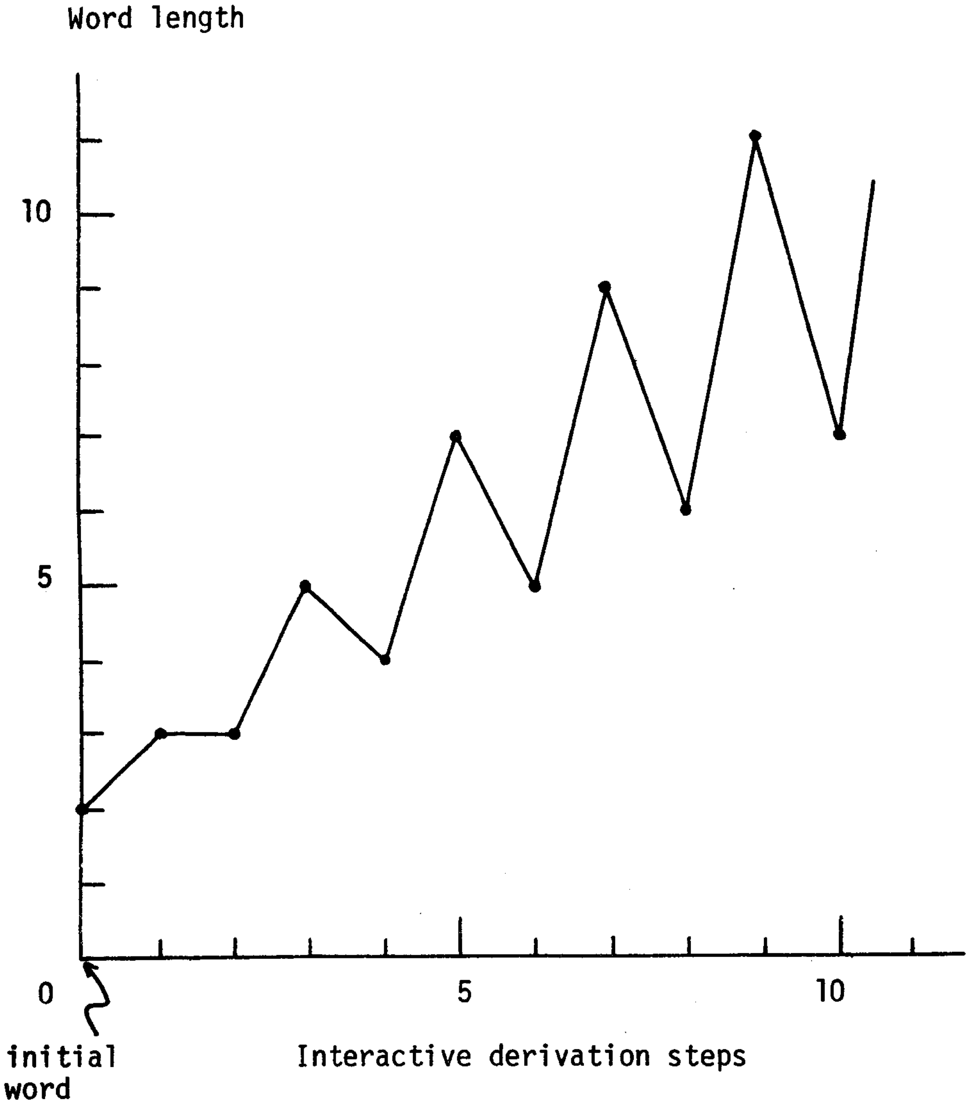


Figure 2.1. The relationships between the word length and interactive derivation steps in Example 2.6.

Lemma 2.11. The language which contains two words bw_1b and cw_2c (where b, c in V_T and w_1, w_2 in V_T^*), is not an interactive language.

Proof. If the interactive language $L(G_b, G_a; w_0)$ contains two words bw_1b and cw_2c, there exists an interactive derivation chain such that

$$bw_1b \overset{*}{\Longrightarrow} cw_2c.$$

So, we can suppose that there exists a word y in $L(G_a)$ such that $y = g_{G_a}(bw_1b)$. That is,

$$S_a \xrightarrow{b} z_1 \xrightarrow{w_1} z_2 \xrightarrow{b} y.$$

The production labelled b must have the following form,

$$\text{(b)} \quad S_a \longrightarrow z_1 \; \varepsilon \; P_a.$$

Since the sentential form z_1 contains some non-terminal symbols, y also contains some non-terminal symbols. This contradicts with the hypothesis that y is in $L(G_a)$. Q. E. D.

In the many characterizations of the interactive languages, the property mentioned in the next lemma is very interesting and inherent.

Lemma 2.12. If the interactive language $L(G_1, G_2; w_0)$ contains two words w and z where $w \Longrightarrow z$, then $|z| \leq K\cdot|w|$. Moreover the length of the derivation chain of z is less than $M\cdot|w|$ (K and M are constant numbers).

Proof. Suppose that the interactive language is generated by an interactive system (G_1, G_2). Let K_1 and K_2 be the maximum lengths of the strings on the right-hand sides of the production sets P_1 and P_2 respectively. From the assumption, there exists a relation $w \underset{G_1G_2}{\Longrightarrow} z$. Therefore,

$$|g_{G_2}(w)| \leqq K_2\cdot|w|,$$

and $$|z| = |g_{G_1}(g_{G_2}(w))| \leqq K_1\cdot|g_{G_2}(w)| \leqq K_1\cdot K_2\cdot|w|.$$

As the word $g_{G_2}(w)$ is a control word for z, if we set $K = K_1\cdot K_2$ and $M = K_2$, then the Lemma is clear. Q. E. D.

Corollary 2.13. The infinite language whose every word's length is represented by n! $(n \geqq 0)$ is not an interactive language.

Proof. Obvious from Lemma 2.12. Q. E. D.

Considering this Lemma 2.12, it is known that the following well-known context-sensitive grammar G_s can not generate the language $L(G_s)$ in the interactive way using any type 0 associate grammar.

EXample 2.7 [2.18]. Let $G_s = (\{X_0, X_1, X_2\}, \{a, b, c\}, P_s, X_0)$, where

$$P_s = \begin{cases} (1)\ X_0 \longrightarrow aX_0X_1X_2 \\ (2)\ X_0 \longrightarrow abX_2 \\ (3)\ X_2X_1 \longrightarrow X_1X_2 \\ (4)\ bX_1 \longrightarrow bb \\ (5)\ bX_2 \longrightarrow bc \\ (6)\ cX_2 \longrightarrow cc \end{cases}$$

The language generated by this grammar is $L(G_s) = \{a^nb^nc^n \mid n \geqq 1\}$. In general, a word $a^kb^kc^k$ has the following derivation chain;

$$X_o \xRightarrow{*} a^{k-1}X_0(X_1X_2)^{k-1} \Longrightarrow a^kb(X_2X_1)^{k-1}X_2$$

$$\xRightarrow{*} a^kbX_1^{k-1}X_2^k \xRightarrow{*} a^kb^kX_2^k \xRightarrow{*} a^kb^kc^k.$$

Thus, it requires a control word whose length is longer than $(k^2 + 5k - 2)/2$ in order to interactively generate the word $a^kb^kc^k$ by this grammar and a certain associate grammar. Accordingly it is impossible to generate interactively the words longer than $K\cdot|w_0|$, where K is a constant number and $|w_0|$ is a length of the initial word.

Lemma 2.14. An infinite language $\{ww \mid w \text{ in } V_T^*\}$ is not an interactive language.

Proof. Suppose that an infinite language $\{ww \mid w \text{ in } V_T^*\}$ is an interactive language. There must exist a derivation whose control word is $ww = a_1a_2\ldots a_na_1a_2\ldots a_n$ (a_i in V_T, $1 \leqq i \leqq n$). The sentential form derived from a start symbol S with a control word w must contain at least one non-terminal symbol S so as to apply a production labelled a_1. In that case, the sentential form derived from S with the control word ww still contains at least one non-terminal symbol S. This is a contradiction. Q. E. D.

Corollary 2.15. The language L on a one-letter alphabet {a} is an interactive language if and only if $|L| = 1$ or $|L| = 2$, in the latter case the initial word is a.

Theorem 2.16. $\mathcal{I}_{0-0} \subsetneqq \mathcal{L}_0$.

Proof. The proper part of the relation is obvious from Lemma 10. Q. E. D.

In conclusion of this section, we summarize the results mentioned above in Fig. 2.2.

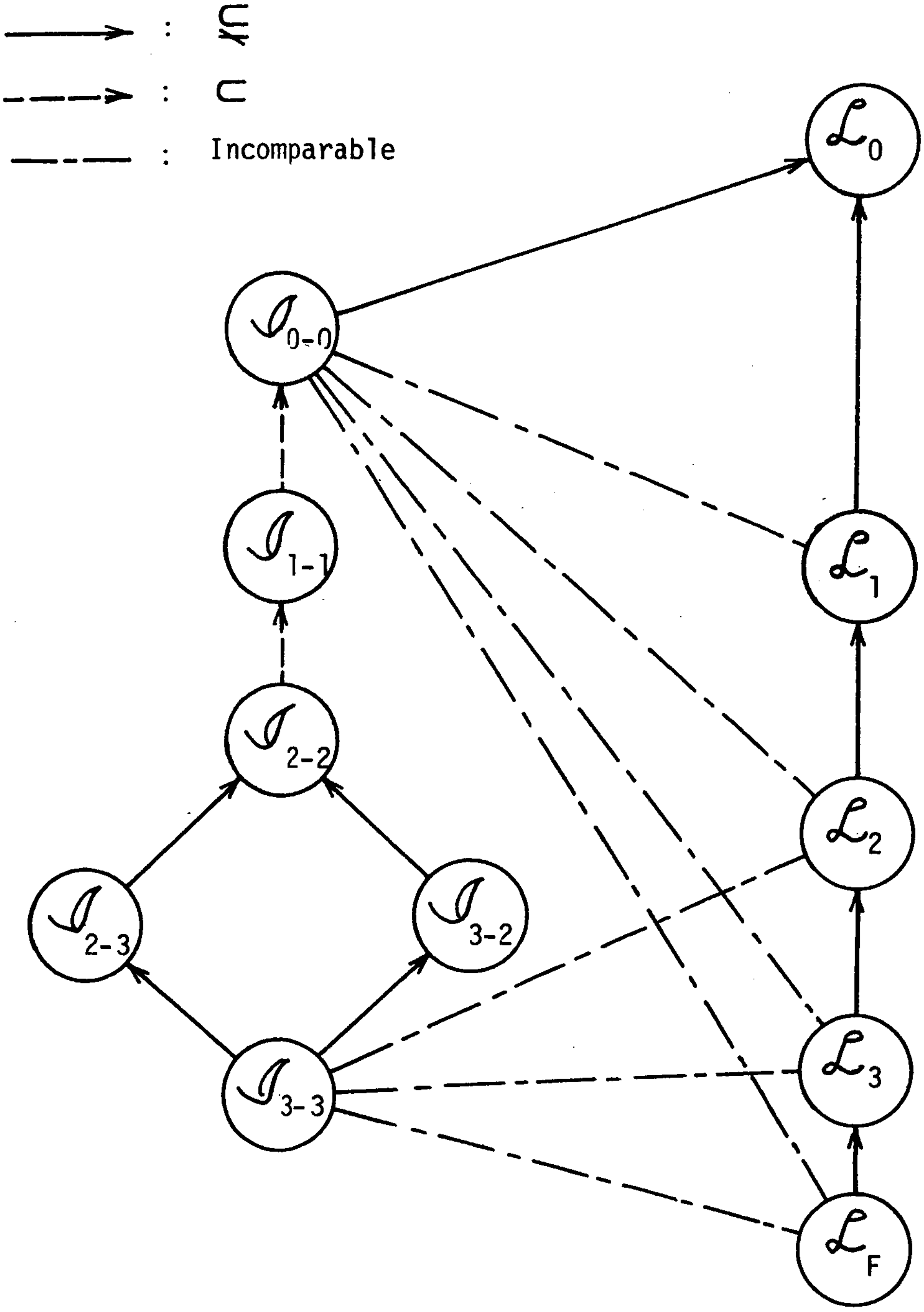


Figure 2.2. The hierarchy of interactive languages.

2.4 Closure Properties

Interactive languages are remarkable by their nearly complete lack of closure properties under the usually considered operations. Similar to the classes of L-languages [2.17] and SF-languages [2.19], it seems to be due to the fact that every string appeared in the interactive derivation processes is also an element of the language.

Theorem 2.17. The family of interactive languages $\mathcal{J}_{0-0}$ is not closed with respect to

(i) Union,

(ii) Complement,

(iii) Intersection,

(iv) The star operator (*),

(v) Concatenation,

(vi) Homomorphism,

(vii) Inverse homomorphism,

(viii) Intersection with regular sets,

(ix) Mirror image (Reverse).

Proof. We shall make use of the following interactive languages to provide counter example for each operation.

(I) $G_1 = (\{A\}, \{a\}, P_1, A)$, $G_2 = (\{B\}, \{0, 1\}, P_2, B)$

$$P_1 = \begin{cases} (0) \quad A \longrightarrow aA \\ (1) \quad A \longrightarrow a \end{cases}, \qquad P_2 = \{ (a) \quad B \longrightarrow 12.$$

Thus, $H_1 = L(G_1, G_2; a) = \{a, aa\}$ and $H_2 = L(G_1, G_2; aaa) = \{aaa\}$.

(II) $G_3 = (\{C\}, \{a, b, \alpha\}, P_3, C)$, $G_4 = (\{D\}, \{0, 1, 2\}, P_4, D)$, where

$$P_3 = \begin{cases} (0) \quad C \longrightarrow aC \\ (1) \quad C \longrightarrow \alpha C \\ (2) \quad C \longrightarrow \alpha b \end{cases}, \qquad P_4 = \begin{cases} (a) \quad D \longrightarrow 0D \\ (b) \quad D \longrightarrow 2 \\ (\alpha) \quad D \longrightarrow 1D \end{cases}.$$

Thus, $H_3 = L(G_3, G_4; a\alpha b) = \{a\alpha^n b \mid n \geqq 1\}$.

(i) A trivial counter example is $H_1 \cup H_2$; the component sets are in $\mathcal{J}_{0-0}$, but their union is not so (see Corollary 2.15).

(ii) Immediate from Corollary 2.15, the comlement of H_1, $V^*_{T_1} - H_1$, is not an interactive language.

(iii) The intersection of H_1 and H_2 is equal to ϕ; it follows directly from Lemma 2.10.

(iv) Again, considering Corollary 2.15, it is obvious that H_1^* is not in $\mathcal{J}_{0-0}$.

(v) Obviously, $H_1 \cdot H_2 = \{a^4, a^5\}$, but this is not in $\mathcal{J}_{0-0}$.

(vi) Consider a homomorphism defined by $h(a) = c$, $h(\alpha) = \alpha$ and $h(b) = c$; $h(H_3) = \{c\alpha^n c \mid n \geq 1\}$. This is not an interactive language.

(vii) Again ϕ can be used as a proof: if $h_2(b) = aa$, then $h_2^{-1}(H_2) = \phi$.

(viii) ϕ is a regular set: so intersection with it provides a counter example.

(ix) As shown in the Example 2.1, the language $\{a^{2^n} b \mid n \geq 0\}$ is in $\mathcal{J}_{0-0}$. Consider the mirror image of it, $\{ba^{2^n} \mid n \geq 0\}$. Is this an interactive language? If it is the case, there exists next interactive derivation chain for some positive integers, k, m, r.

$$ba^k \Longrightarrow ba^m \Longrightarrow ba^r \quad (k < m < r) \qquad (**)$$

It means that there are two derivation chains according to the associate grammar.

$$S_2 \xRightarrow{b} z \xRightarrow{a^k} y_k,$$

$$S_2 \xRightarrow{b} z \xRightarrow{a^k} y_k \xRightarrow{a^{m-k}} y_m.$$

Depending upon the former relation in (**), the sentential form y_k does not contain any non-terminal symbols. According to the latter one in (**), however, y_k must contain at least one non-terminal symbol.

Consequently, the desired relation (**) is not realized, and the family of interactive languages is not closed under mirror image operation.

Q. E. D.

2.5 Conclusions

Using the well-known concept of control words for formal grammars, we have obtained the notions of interactive systems and interactive languages. It is shown that a family of context-free languages does not properly include a family of interactive languages between two regular grammars, and vice versa. The family of interactive languages is not closed under any of the ordinary operations.

CHAPTER 3
CYCLIC INTERACTIVE LANGUAGES

3.1 Introduction

In the real world, many cases contain the problem of interactions among more than two systems. Lately, the problem concerned with the interactions for various systems, such as "Computer Network Systems" and "Time Sharing Systems", are proposed and discussed by many researchers. Generally speaking, the interactions among many systems are very complex.

In the present chapter, we shall try to define the cyclic interactions among n ($\geqq$ 3) grammars both in sequential case and in parallel case. In the sequential case, we have shown that there exists an (n-1)-cyclic interactive system which is equivalent to an n-cyclic interactive system if the n-cyclic interactive system contains a type 3 grammar and if it satisfies a non-blocked condition.
In the case of parallel interactions, for any parallel n-cyclic (n $\geqq$ 3) interactive system there exists a parallel 3-cyclic interactive system which is equivalent to that one.

3.2 Sequential Cyclic Interactive Languages

In the preceding sections, we discussed on the interactions between two grammars. But in the real systems, many cases contain the problem of interactions among more than two systems. In this section, sequential cyclic interactions among n grammars are defined and some characteristics of them are also discussed.

Definition 3.1. Let $G_1, G_2, \ldots, G_n$ $(n \geqq 3)$ be phrase-structure grammars, and extend the relation $\xRightarrow[G_1G_2]{}$ in Definition 2.2 to

$$w_1 \xRightarrow[G_1G_2\ldots G_n]{} w_2,$$

where the following conditions are satisfied:

$$\begin{aligned} w_1 &= x_1 \text{ in } L(G_1), \\ g_{G_2}(x_1) &= x_2 \text{ in } L(G_2), \\ &\vdots \\ g_{G_n}(x_{n-1}) &= x_n \text{ in } L(G_n), \\ g_{G_1}(x_n) &= w_2 \text{ in } L(G_1). \end{aligned}$$

Note that $V_{T_i} \subset P_{\ell_{i+1}}$ $(i < n)$ and $V_{T_n} \subset P_{\ell_1}$.

The sequential n-cyclic interactive language generated by an ordered n-tuple of grammars $(G_1, G_2, \ldots, G_n)$ is defined in the same manner as in Definition 2.3.

$$SCL^n(G_1, G_2, \ldots, G_n; w_0) = \{w \in L(G_1) \mid w_0 \xRightarrow[G_1G_2\ldots G_n]{*} w\}.$$

Then a sequential n-cyclic associate language is also defined as follows:

$$SCA^n(G_1, G_2, \ldots, G_n; w_0) = \{x_{2_i}, \ldots, x_{n_i} \mid w_0 \xRightarrow[G_1G_2\ldots G_n]{*} w_i,$$

$$w_i \xRightarrow[G_1G_2\ldots G_n]{} w_{i+1}, g_{G_2}(w_i) = x_{2_i}, \ldots, g_{G_n}(x_{n_i-1}) = x_{n_i},$$

$$g_{G_1}(x_{n_i}) = w_{i+1}\}.$$

In this context, the n-tuple of grammars is called to be a sequential n-cyclic interactive system, and the family of sequential n-cyclic interactive languages is denoted by $\mathcal{SCI}^n$.

Definition 3.2. A sequential n-cyclic interactive system $(G_1, G_2, \ldots, G_n)$ is called to be non-blocked if there exists an interactive derivation chain from an initial word w_0 to a word w in $L(G_1)$ then there exists a direct interactive derivation from the word w to some word w' in $L(G_1)$.

Theorem 3.1. Let $(G_1, \ldots, G_n)$ be a non-blocked sequential n-cyclic interactive system, where $n > 2$ and one grammar in $\{G_i \mid 1 \leq i \leq n,\ i \neq 2\}$ is type 3. Then there exists a non-blocked sequential (n-1)-cyclic interactive system $(G'_1, \ldots, G'_{n-1})$ such that $SCL^{n-1}(G'_1, \ldots, G'_{n-1};\ w'_0) = SCL^n(G_1, \ldots, G_n;\ w_0)$.

Proof. Let the sequential cyclic interaction among G_1, $\ldots$, G_n be shown as in Fig. 3.1-a and G_i $(i \neq 2)$ be type 3. It is sufficient to construct a new grammar G_{new} which can simulate the interaction from G_{i-1} to G_i (see Fig. 3.1-b). That is, if for any x in $L(G_{i-2})$, there exists a unique word y in $L(G_{i-1})$ and there exists a unique word z in $L(G_i)$ such that $g_{G_{i-1}}(x) = y$ and $g_{G_i}(y) = z$, then $g_{G_{new}}(x) = z$ (where $y = y_1 \ldots y_m$, y_j in P_{ℓ_i} for $1 \leq i \leq m$).

<i > If G_i is a right-linear grammar, a production labelled y_j in P_i has the form as follows:

$$(y_j)\quad A \longrightarrow \alpha_j B,$$

or

$$(y_j)\quad A \longrightarrow \alpha_j,$$

where α_j in $V^*_{T_{i-2}} \subset P^*_{\ell_{i-1}}$ and A, B in $V_{N_{i-2}}$. Therefore,

$$g_{G_i}(y) = g_{G_i}(y_1 \ldots y_m) = \alpha_1 \ldots \alpha_m = z.$$

In general, the production in P_{i-1} are supposed to be as

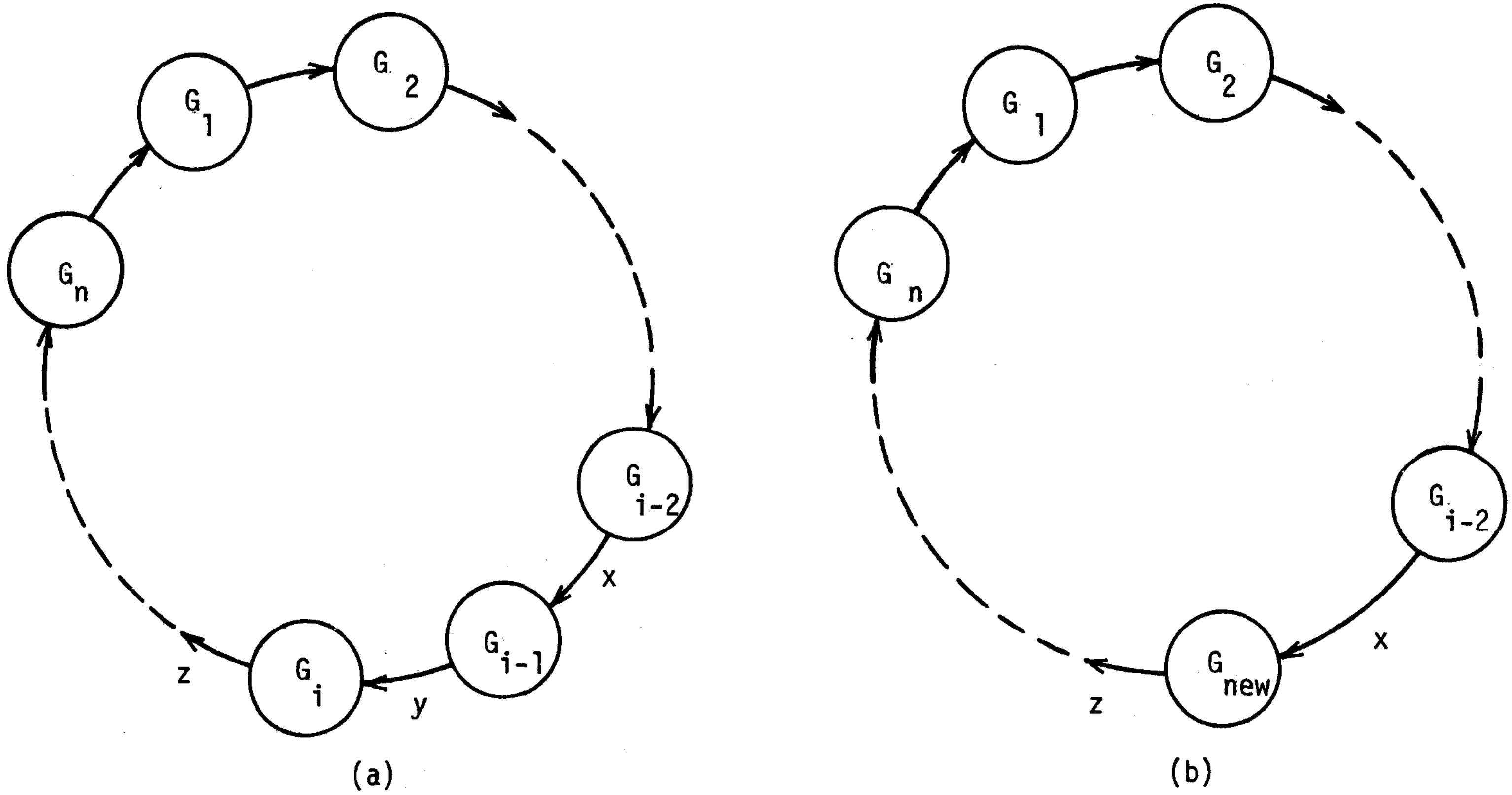


Figure 3.1. Sequential interaction among n grammars.

$$(x_k) \quad \beta \longrightarrow \delta,$$

where β and δ are in $(V_{T_{i-1}} \cup V_{N_{i-1}})^*$. The substituted grammar G_{new} has the following productions:

$$(x_k) \quad \tau(\beta) \longrightarrow \tau(\delta),$$

where $\tau(\beta)$ is a string that each terminal symbol y_j in β is substituted with α_j. In the case that $i = 1$, let $i-1 = n$ and $i-2 = n-1$. Furthermore, if $i = 1$, then the following production should be contained in P_{new} since the initial word w_0 must be in $L(G_{new})$.

$$(\$) \quad S_{i-1} \longrightarrow w_0, \text{ where } \{\$\} \cap P_{\ell_{i-2}} = \phi.$$

After all, we have a new grammar $G_{new} = (V_{N_{i-1}}, V_{T_i}, P_{new}, S_{i-1})$.

<ii> If the grammar G_i is a left-linear grammar, the above new grammar G_{new} should have the following productions

$$(x_k) \quad \tau(\beta^R) \longrightarrow \tau(\delta^R),$$

where β^R is a mirror image of the string β. Q. E. D.

Theorem 3.2. For any non-blocked sequential n-cyclic interactive system $(G_1, \ldots, G_n)$, where $n > 2$ and each of the grammar is type 3, there exists a non-blocked sequential (n-1)-cyclic interactive system $(G_1', \ldots, G_{n-1}')$ with

$$SCL^{n-1}(G_1', \ldots, G_{n-1}'; w_0') = SCL^n(G_1, \ldots, G_n; w_0),$$

where $G_1', \ldots, G_{n-1}'$ are also type 3.

Proof. This is really a corollary to Theorem 3.1. Since the substitution used in the former proof is a homomorphism, the type of new grammar G_{new} is just the same as that of the grammar G_{i-1}.

Q. E. D.

It is an open problem whether Theorems 3.1 and 3.2 hold or not in the case that non-blocked condition is not satisfied.

3.3 Parallel Cyclic Interactive Languages

In the previous section, we have discussed the sequential cyclic interactions among n phrase-structure grammars. In this section, its extension is considered: the parallel cyclic interaction among n phrase-structure grammars.

Definition 3.3 Let $G_1, \ldots, G_{n-1}$ and G_n $(n \geqq 3)$ be phrase-structure grammars, the following relation is called a parallel cyclic interaction among n grammars with respect to the grammar G_i.

$$w_1 \xRightarrow[G_1 \ldots \bar{G}_i \ldots G_n]{} w_2,$$

where the following condition (A) or (B) is satisfied:

(A) $w_1 = x_1$ in $L(G_i)$,

$g_{G_{i+1}}(x_1) = x_2$ in $L(G_{i+1})$,

$g_{G_i}(x_2) = w_2$ in $L(G_i)$,

where if i equals n then i+1 means 1.

(B) $w_1 = x_1$ in $L(G_i)$,

$g_{G_{i-1}}(X_1) = x_2$ in $L(G_{i-1})$,

$g_{G_i}(x_2) = w_2$ in $L(G_i)$,

where if i equals 1 then i-1 means n.

The parallel n-cyclic interactive language generated by an ordered n-tuple of grammars $(G_1, \ldots, G_n)$ is defined in the same manner as in Definition 3.1.

$$PCL^n(G_1, \ldots, G_n; w_0) = \{w \varepsilon L(G_1) \mid w_0 \overset{*}{\underset{\bar{G}_1 G_2 \ldots G_n}{\Longrightarrow}} w\}.$$

In this context, the n-tuple of grammars is called a parallel n-cyclic interactive system, and the family of parallel n-cyclic interactive languages is denoted by $\mathcal{PCI}^n$.

Example 3.1. Let $G_1 = (\{S\}, \{a, b\}, P_1, S)$, $G_2 = (\{A\}, \{a, b\}, P_2, A)$ and $G_3 = (\{B\}, \{a, b\}, P_3, B)$, where

$$P_1 = \begin{cases} \text{(a)} & S \longrightarrow aS \\ \text{(b)} & S \longrightarrow b \end{cases},$$

$$P_2 = \begin{cases} \text{(a)} & A \longrightarrow aA \\ \text{(b)} & A \longrightarrow ab \end{cases},$$

$$P_3 = \begin{cases} \text{(a)} & B \longrightarrow aaaB \\ \text{(b)} & B \longrightarrow b \end{cases}.$$

Then the parallel 3-cyclic interactions among G_1, G_2 and G_3 are shown in Table 3.1.

Example 3.2. Let $G_4 = (\{S\}, \{a, b, c\}, P_4, S)$, $G_5 = (\{X\}, \{a, b, c\}, P_5, X)$ and $G_6 = (\{Y\}, \{a, b, c\}, P_6, Y)$, where

$$P_4 = \begin{cases} \text{(a)} & S \longrightarrow aSa \\ \text{(b)} & S \longrightarrow bSb \\ \text{(c)} & S \longrightarrow c \end{cases},$$

Table 3.1 Parallel 3-cyclic interactions in Example 3.1.

G_1	G_2	G_3
ab	ϕ	ϕ
ϕ	aab	aaab
aab, aaab	aaaab	a^6b
a^4b, a^6b	a^3b, a^4b, a^7b	a^6b, a^9b, $a^{12}b$
a^3b, a^4b, a^6b a^7b, a^9b, $a^{12}b$	a^5b, a^7b, $a^{10}b$ $a^{13}b$	a^9b, $a^{12}b$, $a^{18}b$ $a^{21}b$, $a^{27}b$
a^5b, a^7b, a^9b $a^{10}b$, $a^{12}b$, $a^{13}b$ $a^{18}b$, $a^{21}b$, $a^{27}b$	a^4b, a^5b, a^7b a^8b, $a^{10}b$, $a^{13}b$ $a^{19}b$, $a^{22}b$, $a^{28}b$	a^9b, $a^{12}b$, $a^{15}b$ $a^{18}b$, $a^{21}b$, $a^{27}b$ $a^{30}b$, $a^{36}b$, $a^{39}b$
⋮	⋮	⋮

$$P_5 = \begin{cases} \text{(a)} & X \longrightarrow X \\ \text{(b)} & X \longrightarrow abX \ , \\ \text{(c)} & X \longrightarrow ac \end{cases}$$

$$P_6 = \begin{cases} \text{(a)} & Y \longrightarrow bY \\ \text{(b)} & Y \longrightarrow Y \quad , \\ \text{(c)} & Y \longrightarrow c \end{cases}$$

Then the parallel 3-cyclic interactions among G_4, G_5 and G_6 are shown in Table 3.2.

Thus the following relation holds:

$$PCL^3(G_4,\ G_5,\ G_6;\ c) \subsetneqq \{wcw^R \mid w \in \{a,\ b\}^*\}.$$

Theorem 3.3. $\mathcal{PCL}^n \supsetneqq \mathcal{SCL}^n$ $(n \geqq 3)$.

Proof. Consider the special case that the intersections of P_i and $V_{T_{i+1}}$, and of P_n and V_{T_1} are all empty, where i is an integer less than n. Then the parallel n-cyclic interactive language $PCL^n(G_1,\ \ldots,\ G_n;\ w_0)$ equals the sequential n-cyclic interactive language $SCL^n(G_1,\ \ldots,\ G_n;\ w_0)$. The combination of Lemma 2.11, Example 3.2 and the above discussion constitutes the proof.

Q. E. D.

Table 3.2. Parallel 3-cyclic interactions in Example 3.2.

G_4	G_5	G_6
c	ϕ	ϕ
ϕ	ac	c
aca, c	ac	bc
bcb, aca	abac, ac	c, bc
abacaba aca bcb c	ac, abac	bc, bbc
⋮	⋮	⋮

Theorem 3.4. Let $(G_0, \ldots, G_n)$ be a parallel (n+1)-cyclic interactive system with $n \geqq 4$. Then there exists a parallel (n-1)-cyclic interactive system $(G_0', \ldots, G_{n-2}')$ such that

$$PCL^{n-1}(G_0', \ldots, G_{n-2}'; w_0) = PCL^{n+1}(G_0, \ldots, G_n; w_0).$$

Proof. Let the parallel interaction among $G_0, \ldots, G_n$ be shown as in Fig. 3.2-a. It is sufficient to construct new grammars G_1'' and G_4'' which simulate the interactions among G_1, G_2, G_3 and G_4 (see Fig. 3.2-b). For each $G_i = (V_{T_i}, V_{N_i}, P_i, S_i)$ with $1 < i < n$, let G_i' be $(V_{T_i} \times \{i\}, V_{N_i}', P_i \times \{i-1\} \cup P_i \times \{i+1\}, S_i')$, where S_i' is in V_{N_i}' and if $i \neq j$ then $V_{N_i}' \cap V_{N_j}' = \phi$. $P \times \{k\}$ means that every production label (j) whose core rewriting rule is in P is changed to (j, k). And, let

$$G_1' = (V_{T_1} \times \{1\}, V_{N_1}', P_1 \cup P_1 \times \{2\}, S_1'),$$
$$G_n' = (V_{T_n} \times \{n\}, V_{N_n}', P_n \times \{n-1\} \cup P_n, S_n'),$$

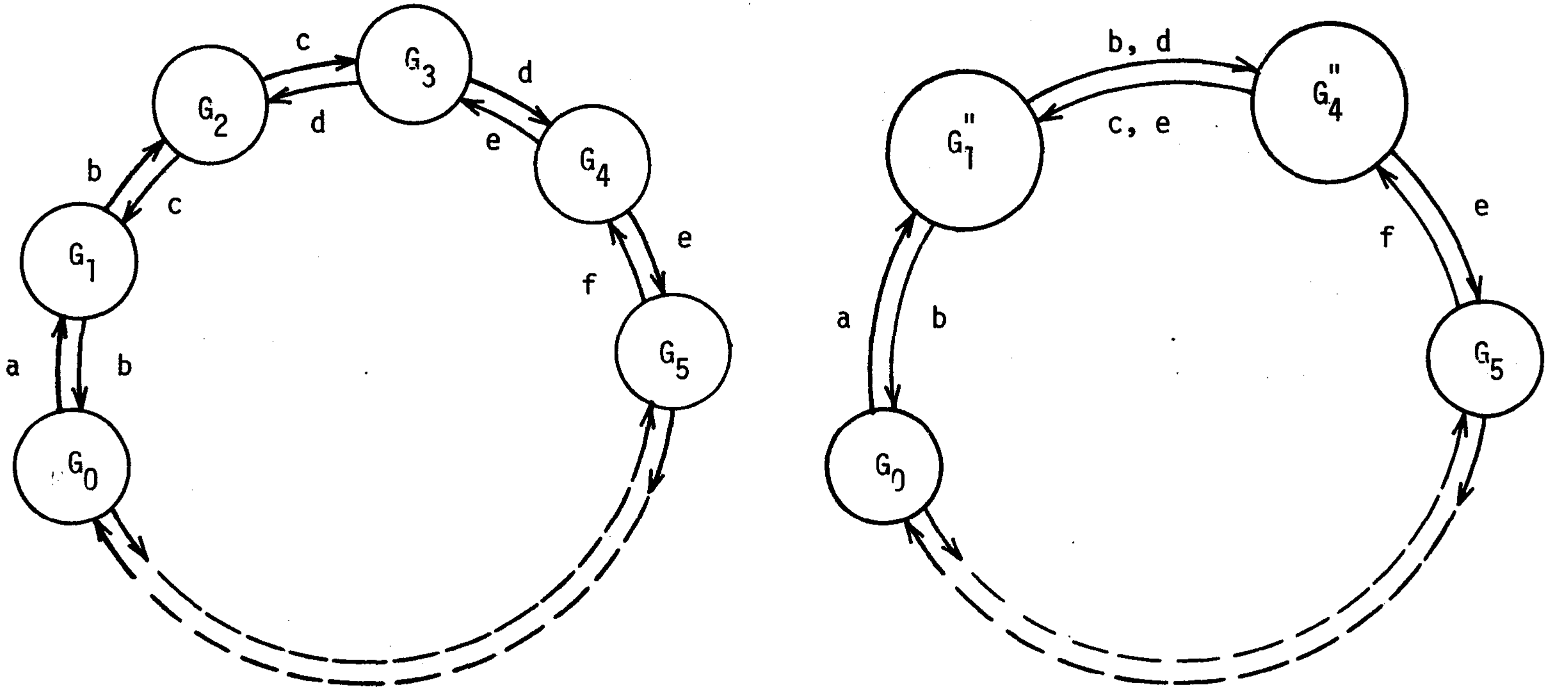
and
$$G_0' = (V_{T_0}, V_{N_0}', P_0 \times \{n\} \cup P_0 \times \{1\}, S_0').$$

Therefore,

$$PCL^{n+1}(G_0, \ldots, G_n; w_0) = PCL^{n+1}(G_0', \ldots, G_n'; w_0).$$

Let $G_1'' = G_1' \cup G_3'$
$= (V_{T_1} \times \{1\} \cup V_{T_3} \times \{3\}, V_{N_1}' \cup V_{N_3}', P_1 \cup P_1 \times \{2\} \cup P_3 \times \{2\} \cup P_3 \times \{4\}, \{S_1', S_3'\})$,

and $G_4'' = G_2' \cup G_4'$
$= (V_{T_2} \times \{2\} \cup V_{T_4} \times \{4\}, V_{N_2}' \cup V_{N_4}', P_2 \times \{1\} \cup P_2 \times \{3\} \cup P_4 \times \{3\} \cup P_4 \times \{5\}, \{S_2', S_4'\})$.



(a)

(b)

Figure 3.2. Parallel interactions among n grammars.

Although the number of start symbols of G''_1 and G''_4 is not one, the discussion does not lose generality since there exist the equivalent ordinary formal grammars with one start symbol to them. Consequently, it is clear that

$$PCL^{n-1}(G'_0, G''_1, G''_4, G'_5, \ldots, G'_n; w_0)$$
$$= PCL^{n+1}(G'_0, G'_1, G'_2, G'_3, G'_4, G'_5, \ldots, G'_n; w_0).$$

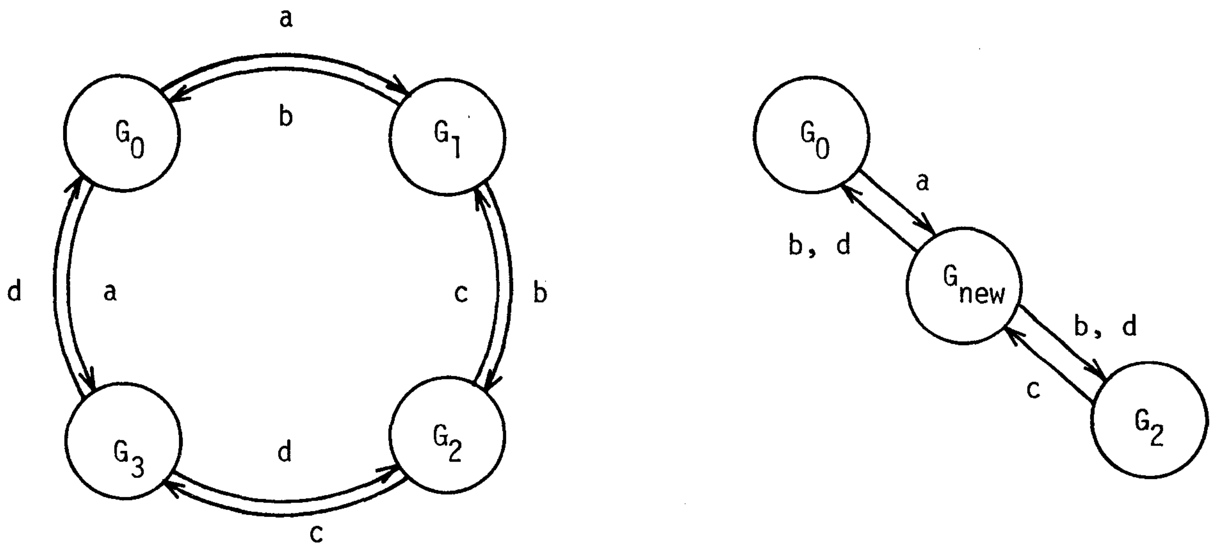
Q. E. D.

Remark. Generally, $PCL^{n-1}(G'_0, G''_1, G''_4, G'_5, \ldots, G'_n; w_0)$ given in the above proof is nondeterministic, even if the original system $PCL^{n+1}(G_0, \ldots, G_n; w_0)$ is deterministic.

Theorem 3.5. For any parallel n-cyclic interactive system with $n \geqq 3$, there exists a parallel 3-cyclic interactive system which is equivalent to that one.

Proof. In the case that n is odd, the theorem is the direct consequence of the previous Theorem 3.4. In the case that n is even, from the Theorem 3.4 it is easy to show that there exists a parallel 4-cyclic interactive system equivalent to the original one.

Next, we show that for any parallel 4-cyclic interactive system there exists a parallel 3-cyclic interactive system which is equivalent to that one. Let (G_0, G_1, G_2, G_4) be a parallel 4-cyclic



(a) (b)

Figure 3.3. Parallel 4-cyclic and 3-cyclic interactive systems.

interactive system shown in Fig. 3.3-a. If the new grammar G_{new} is $G_1 \cup G_3$ as shown in Fig. 3.3-b, then the following relation holds:

$$PCL^3(G_0, G_{new}, G_2 ; w_0) = PCL^4(G_0, G_1, G_2, G_3; w_0).$$

Note that in the case shown in Fig. 3.3-b, the interaction among three grammars is not cyclic: the information goes and returns from G_0 to G_2 via G_{new} i.e. no information is mutually transmitted directly between G_0 and G_2. But this is a parallel cyclic interactive system in a broad sense, too. Q. E. D.

3.4 Conclusions

The interaction among more than two grammars have been presented and discussed. It is shown that for any sequential n-cyclic ($n > 3$) interactive system, which consists of n regular grammars, there exists a sequential 2-cyclic interactive system which is equivalent to that one, if the original system satisfies the non-blocked condition. For the parallel interactions, it is shown that any parallel n-cyclic ($n \geqq 3$) interactive systems are not more powerful than parallel 3-cyclic interactive systems with respect to the generative power of the languages.

CHAPTER 4

NONDETERMINISTIC INTERACTIVE LANGUAGES AND INTERACTIVE WEB LANGUAGES

4.1 Introduction

As far as the author knows, there are reported few researches which explore the property of interactions using graphs. In fact, the concept of interaction using webs may seem to be new. Therefore, it will be appropriate to develop the interactve web languages.

Although the primary purpose of these results will be to provide a basis for solving problems using graphs, they are of independent interest and make it possible to derive some significant results concerning the interactive web languages as well.

Firstly, the nondeterministic interactive systems are proposed and investigated. Then, we shall try to define the interaction using webs and strings. And it is pointed out that for any nondeterministic interactive system which consists of only context-free grammars, there exists an interactive web system which generates the web representation of that languages as directed series-parallel networks.

4.2 Nondeterministic Interactive Languages

From the preceding concept of parallel cyclic interaction among many grammars, naturally we can have the notion of nondeterministic interaction between two grammars. In general, the concept of nondeterministics is familiar and important for usual formal systems.

Definition 4.1. Let G be a phrase-structure grammar such as (V_N, V_T, P, S) where each subset of the rewriting rules of P is distinctly labeled with a set of labels $P_\ell = \{p_1, \ldots, p_r\}$ using a labeling function $f : 2^P \longrightarrow P_\ell$. Let

$$S = Q_1 \xrightarrow{p_{j(1)}} \cdots \xrightarrow{p_{j(i-1)}} Q_i \xrightarrow{p_{j(i)}} \cdots \xrightarrow{p_{j(m)}} Q_{m+1}, \ m \geqq 0$$

be a left-most derivation according to G, where for each i $(0 < i \leqq m)$ at least one production labeled by $p_{j(i)}$ is $\alpha \longrightarrow \beta$ in P and there exist w_1 and w_2 such that $Q_i = w_1\alpha w_2$ and $Q_{i+1} = w_1\beta w_2$, where α occurs only once as a substring of $w_1\alpha$. Then the word $c = p_{j(1)}\cdots p_{j(m)}$ over the alphabet P_ℓ is termed as a <u>nondeterministic control word</u> of the derivation, in symbol $ng_G(c) \ni Q_{m+1}$. When the grammar G is deterministic every $ng_G(c)$ consists of at most one element.

Definition 4.2. Let $G_1 = (V_{N_1}, V_{T_1}, P_1, S_1)$ and $G_2 = (V_{N_2}, V_{T_2}, P_2, S_2)$ be two grammars, where $V_{T_1} \subset P_{\ell_2}$, $V_{T_2} \subset P_{\ell_1}$ (P_{ℓ_1} and P_{ℓ_2} are sets of labels of P_1 and P_2 respectively). Let

$$w_1 \underset{G_1G_2}{\Longmapsto} w_2 ,$$

be a relation between w_1 and w_2, where each of the following conditions is satisfied:

(1) w_1 and w_2 are in $L(G_1)$.

(2) There exists a word y in $L(G_2)$ such that

$$y \in ng_{G_2}(w_1) \text{ and } w_2 \in ng_{G_1}(y).$$

In this context, the relation $\mathrel{|}\!\!\!\Longrightarrow$ is termed a *nondeterministic interactive derivation* between G_1 and G_2.

Definition 4.3. The *nondeterministic interactive language* $NL(G_1, G_2; w_0)$ generated by a nondeterministic interactive system (G_1, G_2) consists of those words in $V_{T_1}^*$ that can be nondeterministically and interactively derived from the initial word w_0 in $L(G_1)$. Formally, we have

$$NL(G_1, G_2; w_0) = \{w \in L(G_1) \mid w_0 \overset{*}{\underset{G_1G_2}{\mathrel{|}\!\!\!\Longrightarrow}} w\}.$$

The family of nondeterministic interactive languages is denoted by $\mathcal{NI}$.

Definition 4.4. Let G be a phrase-structure grammar. Suppose that every word in L(G) is arranged in order of length, i.e., $L(G) = \{w_1, w_2, \ldots\}$ with $|w_i| \leqq |w_{i+1}|$ $(1 \leqq i)$. If there exists a positive integer K and for any word w_i and w_{i+1} the following relation holds,

$$|ng_{G_2}^{-1}(w_{i+1})| \leqq K\cdot|w_i|,$$ [7]

then G is called to be <u>simple.</u>

Example 4.1. The following is an example of a nondeterministic interactive language, where $G_1 = (\{S\}, \{a, b\}, P_1, S)$ and $G_2 = (\{A\}, \{1, 2\}, P_2, A)$.

$$P_1 = \begin{cases} (1) & S \longrightarrow aSb \\ (2) & S \longrightarrow ab \end{cases}, \qquad P_2 = \begin{cases} (a) & A \longrightarrow 1A \\ (b) & \left(\begin{array}{l} A \longrightarrow A \\ A \longrightarrow 2 \end{array}\right. \end{cases}$$

If the initial word is ab, then there exists the following nondeterministic interactive derivation chain:

$$ab \underset{12}{\mathrel{\vert\!\!\Longrightarrow}} aabb \underset{112}{\mathrel{\vert\!\!\Longrightarrow}} \cdots \underset{1^{i-1}2}{\mathrel{\vert\!\!\Longrightarrow}} a^ib^i \underset{1^i2}{\mathrel{\vert\!\!\Longrightarrow}} \cdots .$$

[7] $|ng_{G_2}^{-1}(w_{i+1})|$ denotes the maximum length of the derivation chain for w_{i+1} with respect to G_2.

Thus, the nondeterministic interactive language NL(G_1, G_2; ab) is $\{a^n b^n \mid n \geqq 1\}$ and this is a context-free language, but not a regular language. Furthermore we see that this is not a deterministic interactive language.

Example 4.2. This example shows that the well-known context-sensitive language can be a nondeterministic interactive language. Let $G_3 = (\{B\}, \{a, b, c\}, P_3, B)$ and $G_4 = (\{D\}, \{\alpha, \beta, \gamma\}, P_4, D)$, where

$$P_3 = \left\{ \begin{array}{lll} (\alpha) & B \longrightarrow & aB \\ (\beta) & B \longrightarrow & bB \\ (\gamma) & \left(\begin{array}{l} B \longrightarrow \\ B \longrightarrow \end{array} \right. & \begin{array}{l} ccB \\ cc \end{array} \end{array} \right. , \qquad P_4 = \left\{ \begin{array}{lll} (a) & D \longrightarrow & \alpha\alpha D \\ (b) & D \longrightarrow & \beta\beta\beta D \\ (c) & \left(\begin{array}{l} D \longrightarrow \\ D \longrightarrow \end{array} \right. & \begin{array}{l} \gamma\gamma D \\ \gamma\gamma \end{array} \end{array} \right.$$

If the initial word is abcc, then there exists a nondeterministic interactive derivation chain as follows:

$$\begin{tikzcd}
abcc \arrow[rr, Mapsto] \arrow[dr, dash, dashed] & & a^2b^3c^8 \arrow[rr, Mapsto] \arrow[dr, dash, dashed] & & \cdots \arrow[rr, Mapsto] \arrow[dr, dash, dashed] & & a^{2^i}b^{3^i}c^{2\cdot 4^i} \arrow[rr, Mapsto] \arrow[dr, dash, dashed] & & \cdots . \\
& \alpha^2\beta^3\gamma^4 \arrow[ur, dash, dashed] & & \alpha^4\beta^9\gamma^{16} \arrow[ur, dash, dashed] & & \alpha^{2^i}\beta^{3^i}\gamma^{4^i} \arrow[ur, dash, dashed] & & &
\end{tikzcd}$$

Consequently this nondeterministic interactive language is $\mathrm{NL}(G_3, G_4; abcc) = \{a^{2^n} b^{3^n} c^{2\cdot 4^n} \mid n \geqq 0\}$.

Theorem 4.1. The family of nondeternimistic interactive languages properly includes the family of deterministic interactive languages.

Proof. Immediately from the Lemma 2.11 and the following Example 4.3. Q. E. D.

Example 4.3. Let $G_5 = (\{S\}, \{a\}, P_5, S)$ and $G_6 = (\{X\}, \{1, 2\}, P_6, X)$, where

$$P_5 = \begin{cases} (1)\ \ S \longrightarrow aS \\ \\ (2)\ \ S \longrightarrow a \end{cases}, \qquad P_6 = (a)\left(\begin{array}{l} A \longrightarrow 11A \\ \\ A \longrightarrow 2 \end{array} \right.$$

If the initial word is aa, then the nondeterministic interactive language is $NL(G_5, G_6; aa) = \{a^{2^n+1} \mid n \geqq 0\}$.

Theorem 4.2. For any phrase-structure grammar G_0, there exists a grammar G_a such that $NL(G_0, G_a; w_0) = L(G_0)$ if and only if $L(G_0)$ is simple.

Proof. ($\Leftarrow$) It is clear from the technique given in the previous Example 4.3.

($\Rightarrow$) Similar to the deterministic case, Lemma 2.12 holds for the nondeterministic interactive language. So, this theorem is the direct consequence of Definition 4.4 and Lemma 2.12. Q. E. D.

4.3 Interactive Web Languages

Usually we exchange our ideas not simply by words but by figures; for example well-defined program structures are often represented by means of flow charts. Therefore, it will be very interesting to introduce web grammars to interactive systems.

Now we shall investigate the interaction between a web grammar and a string grammar in which the basis grammar is a web grammar and the associate grammar is ordinary string grammar.

Definition 4.6. A parallel direction-sensitive web grammar (pdswg) is a triplet WG = (V, I, R), where every production in R is labelled by a set of labels R_ℓ, $h_{WG} : R \longrightarrow R_\ell$. Moreover, every embedding function of the rewriting rule in R is assumed to be direction-sensitive, and in every derivation stage each rewriting rule should be applied in parallel to the intermediate webs.

Definition 4.7. Let WG = $(V_{T_W} \cup V_{N_W}, I, R)$ be a pdswg, and $G_a = (V_{T_a}, V_{N_a}, P_a, S_a)$ be a string grammar, where $V_{T_a} \subset R_\ell$, $V_{T_W} \subset P_\ell$ (R_ℓ and P_ℓ are sets of labels of R and P respectively). Let

$$W_1 \underset{WG,G_a}{\Longrightarrow} W_2$$

be a relation between the webs W_1 and W_2 satisfying the following conditions.

(1) W_1 and W_2 are in L(WG).

(2) There exists a directed path p in W_1 from a node whose in-degree is zero to a node whose out-degree is zero, and there exists a word y in L(G_a) such that $g_G(p)$ = y and $g_{WG}(y) = W_2$.

Example 4.5. The following is an example of an interactive web language where WG_1 = ({S, X, Y} U {a, b, c, d, e}, { •S }, R_1) and G_1 = ({A}, {1, 2, 3, 4, 5}, P_1, A).

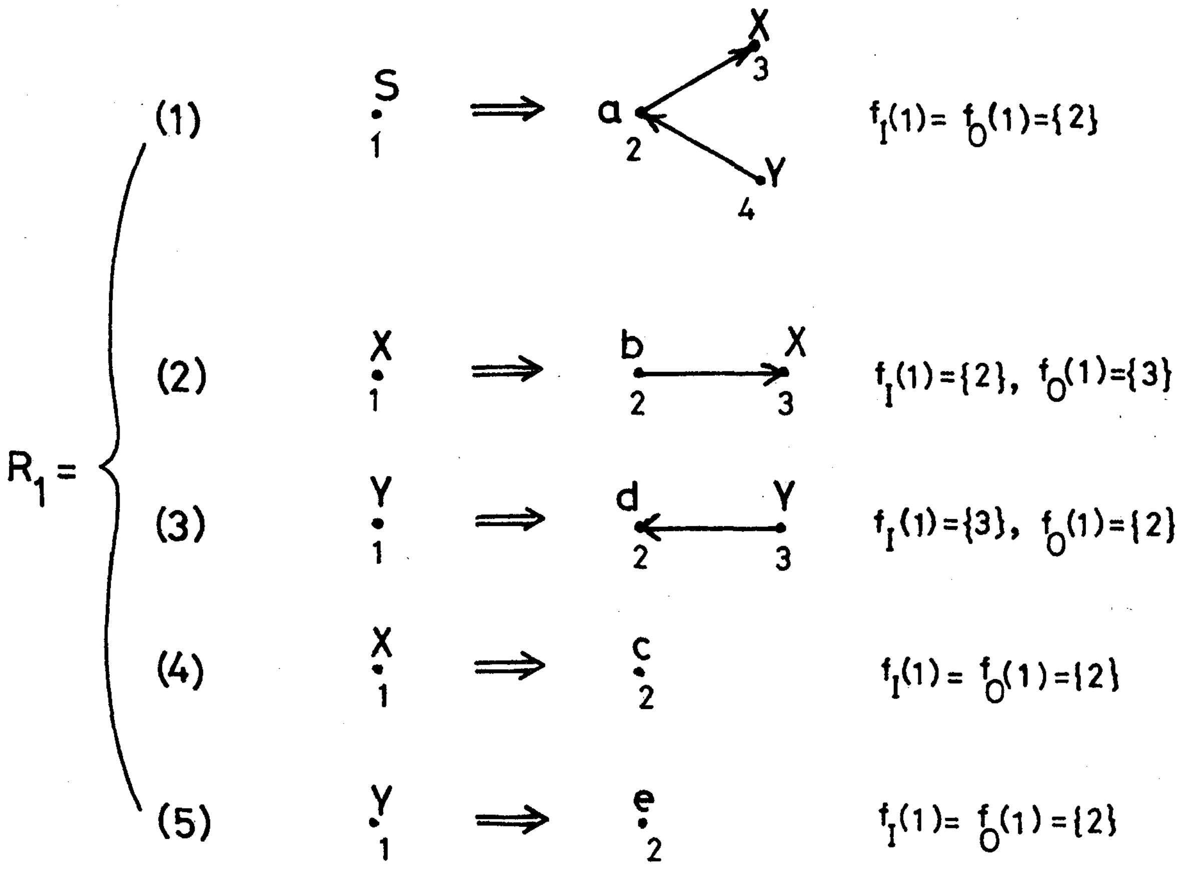

R_1 =
(1) S 1 ⟹ a 2, X 3, Y 4 $f_I(1) = f_O(1) = \{2\}$
(2) X 1 ⟹ b 2, X 3 $f_I(1) = \{2\}$, $f_O(1) = \{3\}$
(3) Y 1 ⟹ d 2, Y 3 $f_I(1) = \{3\}$, $f_O(1) = \{2\}$
(4) X 1 ⟹ c 2 $f_I(1) = f_O(1) = \{2\}$
(5) Y 1 ⟹ e 2 $f_I(1) = f_O(1) = \{2\}$

Fig. 4.1 The rewriting rules of WG_1.

$$P_1 = \left\{ \begin{array}{llll} (a) & A \longrightarrow 5A & (d) & A \longrightarrow 33A \\ (b) & A \longrightarrow 222A & (e) & A \longrightarrow 1A \\ (c) & A \longrightarrow 4 & & \end{array} \right.$$

Figure 4.2. The rewriting rules of G_1.

If a web shown in Fig. 4.3 is an initial web, then there exists an interactive derivation chain for the interactive web system (WG_1, G_1) as shown in Fig. 4.4. Then, the interactive web language is shown in Fig. 4.5 and this is not a context-free web language although WG_1 is a context-free web grammar.

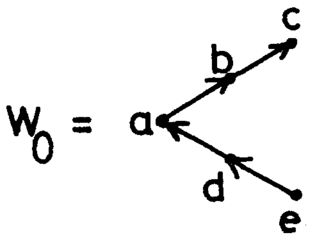


Figure 4.3. Initial web in Example 4.5.

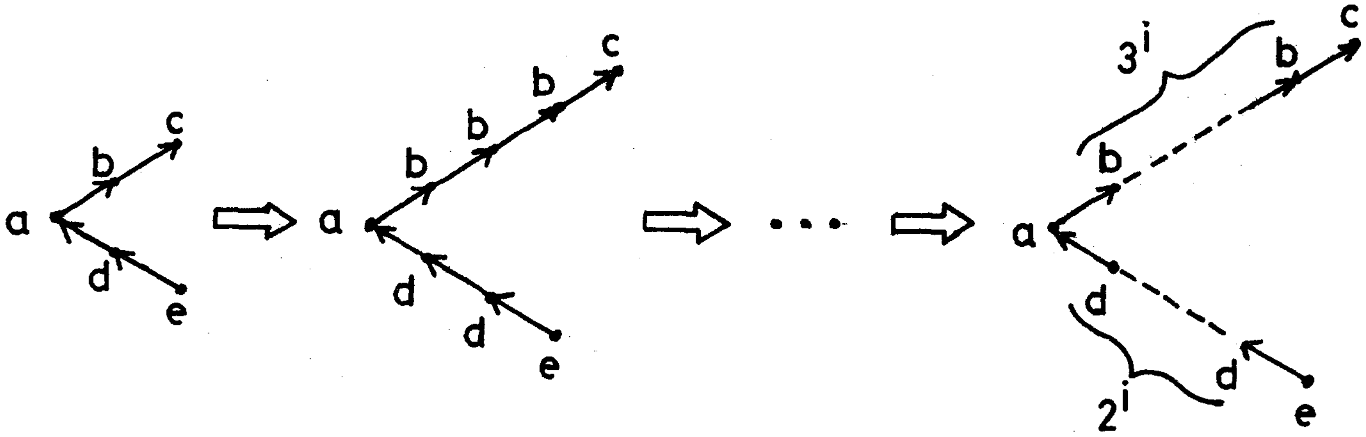


Figure 4.4. Interactive derivation chain in Example 4.5.

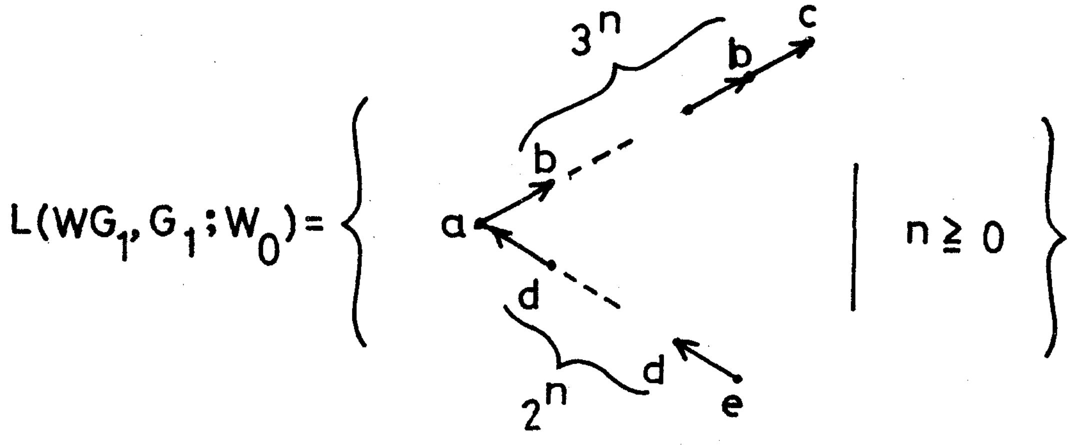


Figure 4.5. Interactive web language in Example 4.5.

Example 4.6. The following interactive web language is a web representation of a well-known context-free language (so called Knuth-language [2.10]), which is a set of all words consisting of equal numbers of a's and b's. Let $WG_2 = (\{S, A, B\} \cup \{a, b, c, d\},$ $\{ c\bullet\!\longrightarrow \bullet \xrightarrow[S]{} \bullet d \}, R_2)$ and $G_2 = (\{X_0, X_1\}, \{S, A, B, A', B'\}, P_2, X_0)$.

Thus, if the initial web is shown in Fig. 4.8, then there exists an interactive web derivation chain as shown in Fig. 4.9. On the web which is derived in the n-th stage of the above derivation, all words which have n a's and b's are presented as pathes from c to d.

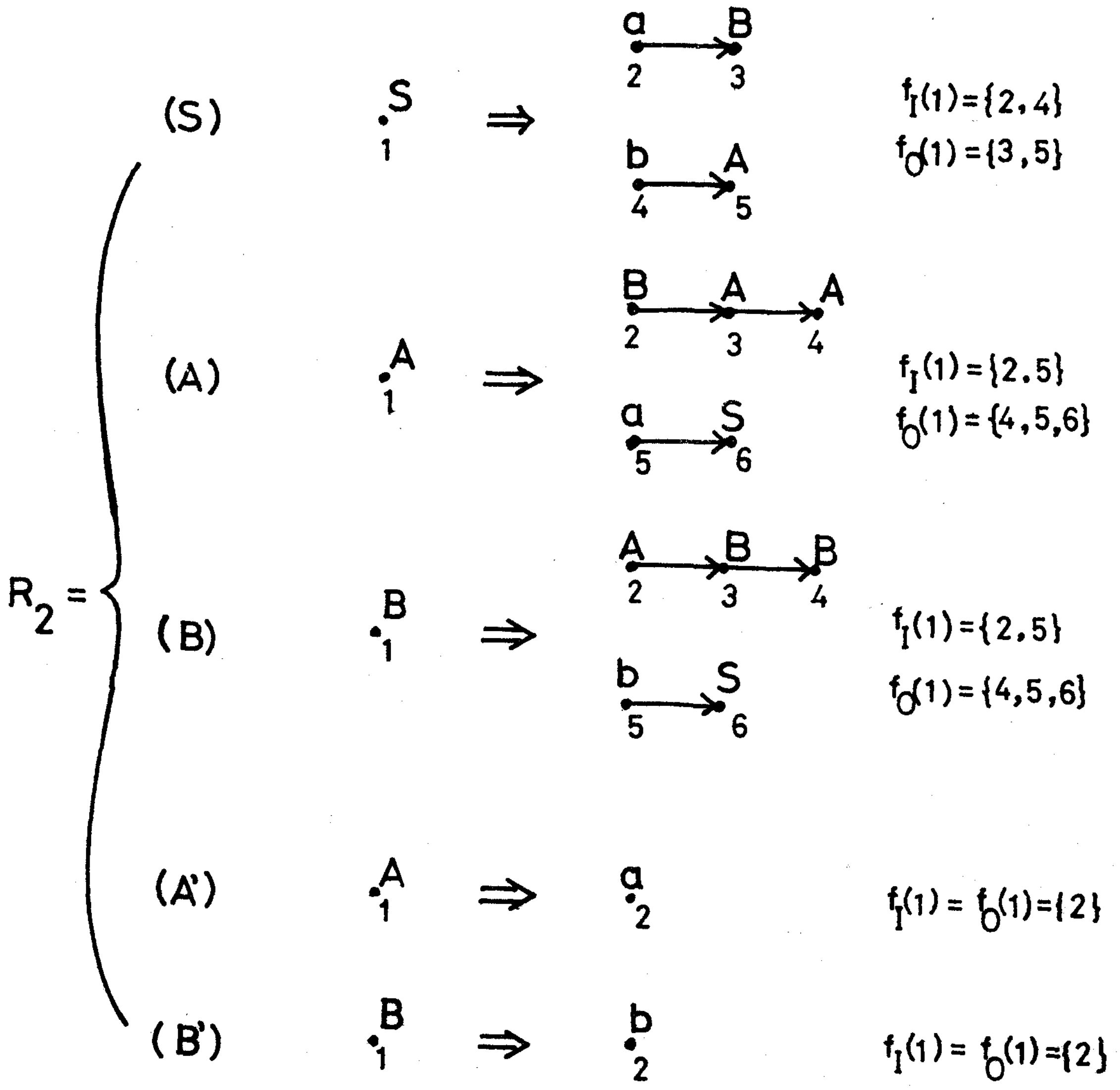


Figure 4.6. The rewriting rules of WG_2.

$$P_2 = \begin{cases} (a)\quad X_1 \longrightarrow SABX_1 & (c)\quad X_0 \longrightarrow X_1 \\ (b)\quad X_1 \longrightarrow X_1 & (d)\quad X_1 \longrightarrow SA'B' \end{cases}$$

Figure 4.7. The rewriting rules of G_2.

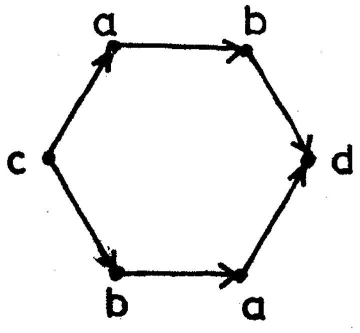


Figure 4.8. Initial web in Example 4.6.

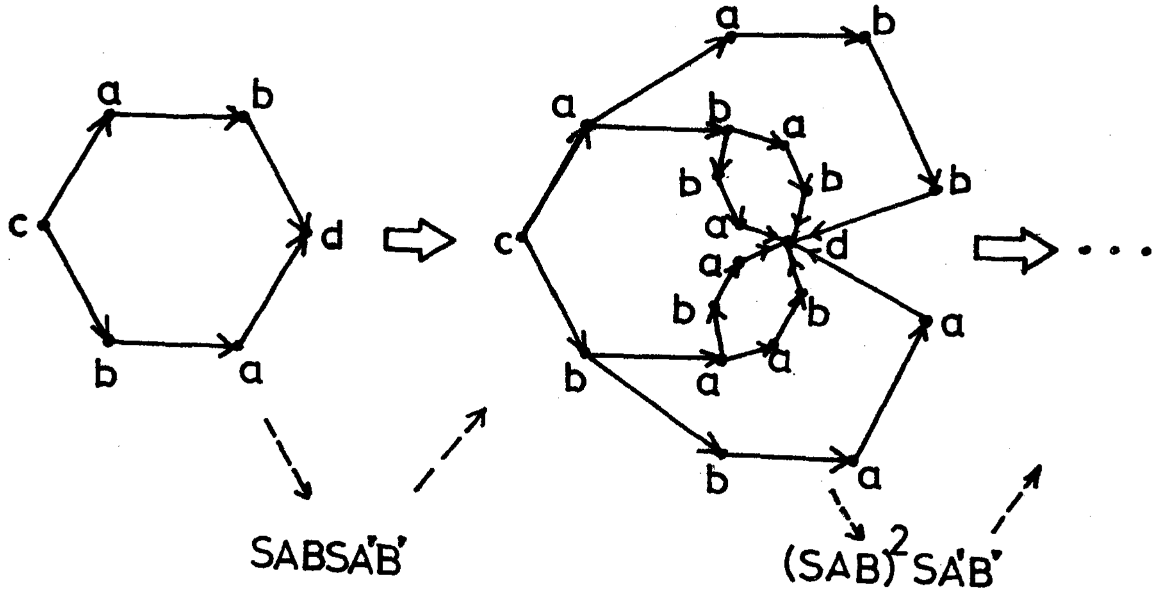


Figure 4.9. Interactive web derivation chain in Example 4.6.

Example 4.7. This example shows an interactive web language which has a cycle on it. Let $WG_3 = (\{S, A, B\} \cup \{o, a, b, \alpha, \beta\}, \{ \cdot S \}, R_3)$ and $G_3 = (\{X_0\}, \{1, 2, 3, 4, 5\}, P_3, X_0)$.

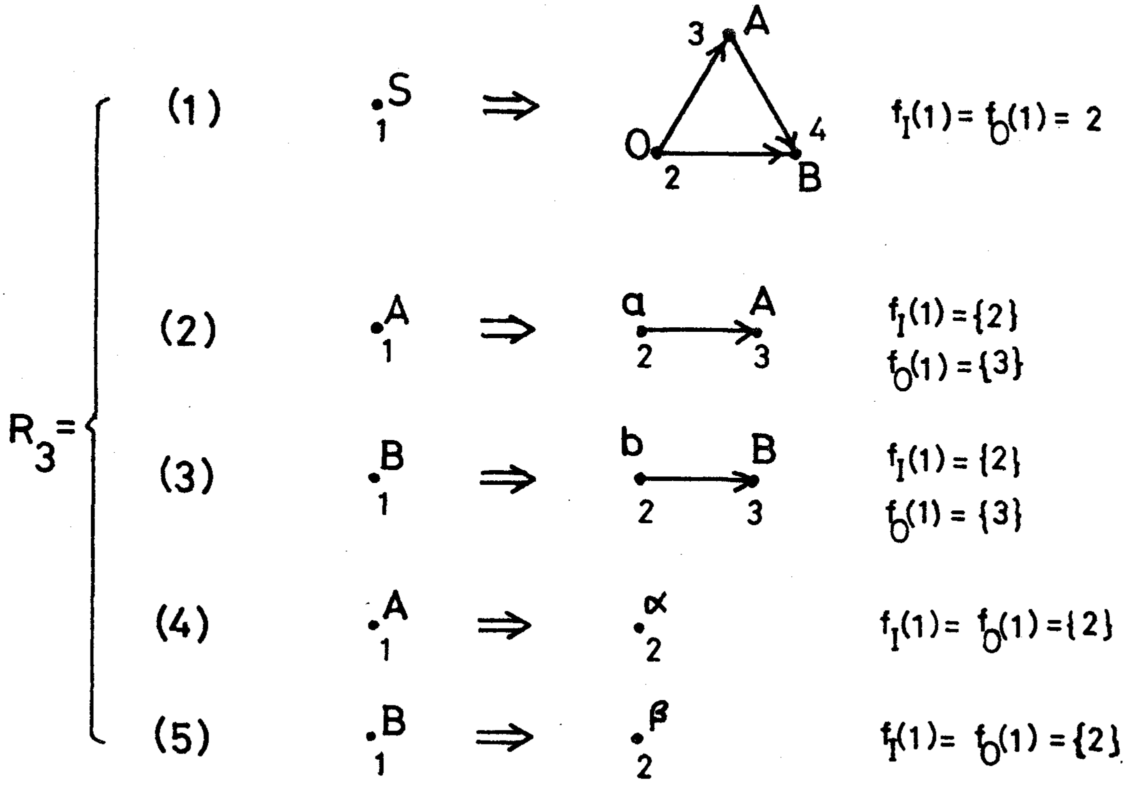


Figure 4.10. The rewriting rules of WG_3.

$$P_3 = \begin{cases} (o)\ X_0 \longrightarrow 1X_0 \qquad (\beta)\ X_0 \longrightarrow 2345 \\ (b)\ X_0 \longrightarrow 23X_0 \end{cases}$$

Figure 4.11. The rewriting rules of G_3.

Thus, there exists an interactive web derivation chain as shown in Fig. 4.12. The associate words for the n-th stage are expressed as a regular expression $o(a^n\alpha)^* b^n\beta$.

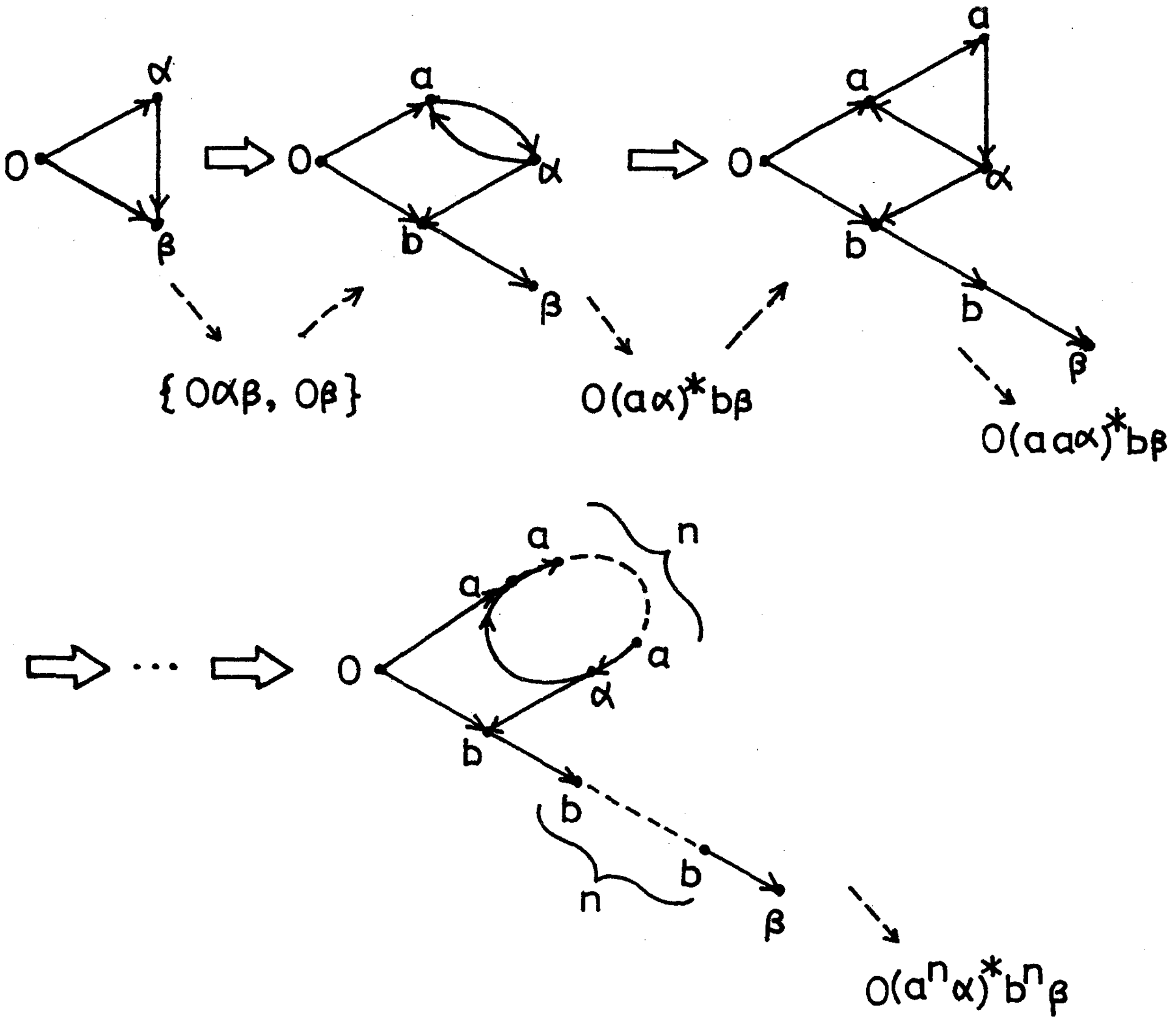


Figure 4.12. The interactive web derivation in Example 4.7.

Consider the case that the following productions are added to P_3,

$$P_3' = \begin{cases} (a) & X_0 \longrightarrow 2X_0 \\ \\ (\alpha) & X_0 \longrightarrow X_0 \end{cases},$$

that is, let the associate grammar G_3' be $(\{X_0\}, \{1, 2, 3, 4, 5\}, P_3 \cup P_3', X_0)$. In this case, the web language interactively derived at the second stage has infinite webs, such that many of them have an arbitrary large cycle as shown in Fig. 4.13-b.

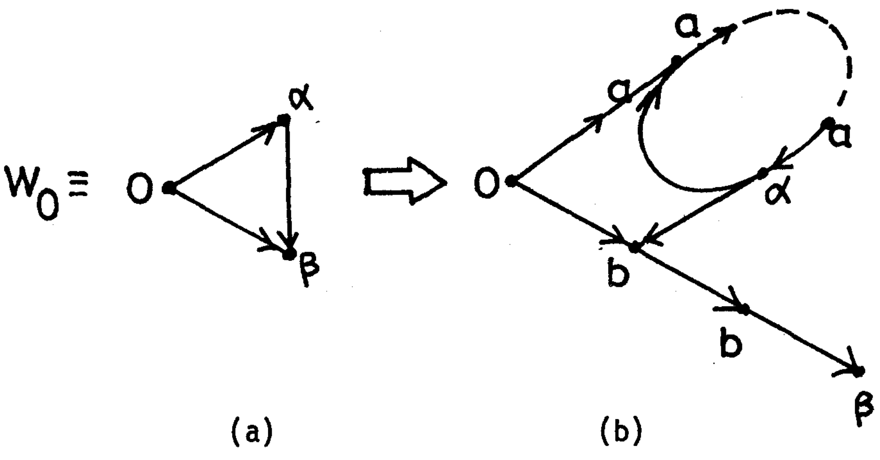


Figure 4.13. Interactive web derivation in the interactive system (WG_3, G_3').

So, the interactive web language $IWL(WG_3, G_3'; W_0)$ is as shown in Fig. 4.14.

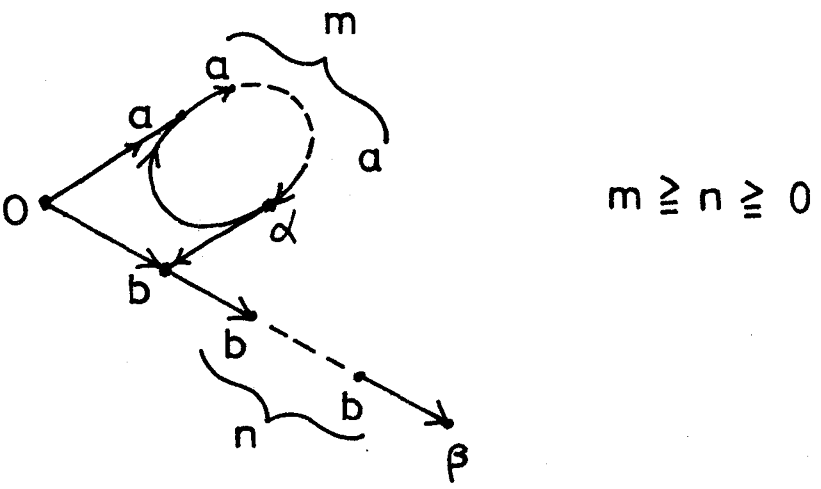


Figure 4.14. The interactive web language IWL(WG_3, G_3; W_0).

Theorem 4.6. An interactive web system IWS = (WG, G) whose web grammar WG is a context-free web grammar and the associate grammar G is a regular string grammar can generate a context-sensitive web language which is not a context-free web language.

Proof. See the Example 4.5 and Abe's Lemma in Section 2.2.

Q. E. D.

Theorem 4.7. For any nondeterministic interactive language NL(G_1, G_2; w_0) with G_1 and G_2 being context-free grammars, there exists an interactive web system which can generate the web representations of that language as directed series-parallel networks.

Proof. From the Example 4.6, it is easy to see the way how to construct an inteactive web system in question, for a given nondeterministic interactive system. Q. E. D.

4.4 Conclusions

The importance of having a method for the simple formal definition of interactions by means of webs and strings lies in the broad use of web representation of program structure. For example, in the formal definition of programming languages some form of directed graphs are often the basic representation of programs for interactive execution and formal analysis.

CHAPTER 5

CONCLUSIONS

The major concern of this part has been the problem of formal definition of interactions among several formal systems by means of strings and webs. We have shown that an interactive system provides simple and elegant means for defining such interactions without going out far from the well-known concepts of interactive problem solving using electronic computers.

The concept of interactive systems is a simple one which may readily be applied to interactions of grammars of many different types. We have chosen here a particular form of "context-free grammar" as the basic grammar form of which interactive systems are consisted.

The classes of interactive languages are discussed and compared with so-called Chomsky's hierarchy. It is shown that the family of interactive languages is not closed under usual operations. Furthermore, the sequential and parallel cyclic interactions among n grammars are also discussed. We have also proposed the notion of interactive web languages. The importance of having a method for the simple formal definition of interactions using webs lies in the widespread use of web representation of structure (especially program structure), both in formal analysis and in computer applications.

In summary, interactive languages should be of interest to those concerned with the more practical aspects of languages and utilizations of computers in interactive modes.

For a future study, the characterization of the cyclic interactions among n grammars of any types (for example, context-sensitive grammars), the reasonable different definitions of the interactions among many grammars, and the interaction among several web grammars will be the interesting topics.

LIST OF REFERENCES

[1. 1] Abe, N., Studies on structural description formal grammars, Ph. D. Thesis, Osaka University, Feb. 1974 (in Japanese).

[1. 2] Abe, N., Mizumoto, M., Toyoda, J., Tanaka, K., Web grammars and several graphs, Trans. Inst. Elect. Commun. Engineers of Japan, 54-c, 12, 1971, pp. 1149-1155 (in Japanese).

[1. 3] Abe, N., Ezawa, Y., Mizumoto, M., Toyoda, J., Tanaka, K., An example of a web grammar generating a set of all separable webs, Trans. Inst. Elect. Commun. Engineers of Japan, 55-D, 4, 1972, pp. 294-295 (in Japanese).

[1. 4] Abe, N., Mizumoto, M., Toyoda, J., Tanaka, K., A description of graphs represented by web grammars, Information Processing, 13, 7, 1972, pp. 452-459 (in Japanese).

[1. 5] Abe, N., Mizumoto, M., Toyoda, J., Tanaka, K., Web grammars and several graphs, J. Comp. System Sciences, 7, 1, 1973, pp. 37-65.

[1. 6] Cook, C. R., First order graph grammars, SIAM J. Computing, 3, 1, 1974, pp. 90-99.

[1. 7] Earley, J., Toward an understanding of data structures, C. of ACM, 14, 10, 1971, pp. 617-627.

[1. 8] Ezawa, Y., Abe, N., Mizumoto, M., Toyoda, J., Tanaka, K., Web automata, Papers of Technical Groups on Automata and Languages, AL-72-35, 1972, (in Japanese).

[1. 9] Ezawa, Y., Abe, N., Mizumoto, M., Toyoda, J., Tanaka, K., Web grammars and web automata, Trans. Inst. Elect. Commun. Engrs. of Japan, 56-A, 4, 1973, pp. 234-241 (in Japanese).

[1.10] Ezawa, Y., Abe, N., Mizumoto, M., Toyoda, J., Tanaka, K., Operations on web languages, 1973 National Convention Records of the Inst. Elect. Commun. Engineers of Japan, Spring 1973, No. 1186 (in Japanese).

[1.11] Ezawa, Y., Mizumoto, M., Toyoda, J., Tanaka, K., Attempt of interactive web languages, 1975 National Convention Records of the Inst. Elect. Commun. Engineers of Japan, Spring 1975 (in Japanese).

[1.12] Harary, F., Graph theory, Addison-Wesley, 1969.

[1.13] Hoare, C. A. R., Notes on data structuring, in Structured Programming, O.-J. Dahl, E. W. Dijkstra, C. A. R. Hoare, Eds., Academic Press, 1972, pp. 83-174.

[1.14] Hopcroft, J. E., Ullman, J. D., Formal languages and their relation to automata, Addison-Wesley, 1969.

[1.15] Khabbaz, N. A., Control sets on linear grammars, Information and Control, 25, 1974, pp. 206-221.

[1.16] Kuroda, S. Y., Classes of languages and linear-bounded automata, Information and Control, 7, 1964, pp.207-223.

[1.17] Lindenmayer, A., Mathematical models for cellular interactions in development, Part I, II, J. of Theoretical Biology, 18, 1968, pp. 280-315.

[1.18] Milgram, D. L., Web automata, University of Maryland Computer Science Center Technical Report 181, 1972.

[1.19] Montanari, U. G., Separable graphs, planar graphs, and web grammars, Information and Control, 16, 1970, pp. 243-267.

[1.20] Mylopoulos, J., On the relation of graph automata and graph grammars, University of Toronto, Dept. of Computer Science, Technical Report 34, 1971.

[1.21] Pavlidis, T., Linear and context-free graph grammars, Journal of ACM, 19, 1972, pp. 11-22.

[1.22] Pfaltz, J. L., Web grammars and picture description, Computer Graphics and Image Processing, 1, 1972, pp. 193-220.

[1.23] Pfaltz, J. L., Rosenfeld, A., Web grammars, Proc. 1st Inter. Joint Conf. on Artificial Intelligence, May 1969, pp. 609-619.

[1.24] Rosenfeld, A., Isotonic grammars, parallel grammars, and picture grammars, Machine Intelligence, 6, 1971, pp. 281-294.

[1.25] Rosenfeld, A., SURVEY, Picture processing 1973, Computer Graphics and Image Processing, 3, 1974, pp. 178-194.

[1.26] Rosenfeld, A., Strong, J. P., A grammar for maps, in Software Engineering, 2, J. T. Tou,Ed., Academic Press, 1971, pp.227-239.

[1.27] Rosenfeld, A., Milgram, D. L., Web automata and web grammars, Machine Intelligence, 7, 1972, pp. 307-324.

[1.28] Rosenkrantz, D. J., Programmed grammars and classes of formal languages, J. of ACM, 16, 1969, pp. 107-131.

[1.29] Salomaa, A., Formal languages, Academic press, 1973.

[1.30] Salomaa, A., Rozenberg, G., L Systems, Lecture Notes in Computer Science, 15, Springer-Verlag, New York, 1974.

[1.31] Shank, H. S., Graph property recognition machine, Mathematical Systems Theory, 5, 1971, pp. 45-49.

[1.32] Shave, M. J. R., Data-structures, Computer Aided Design, 6, 4, 1974, pp. 256-261.

[1.33] Shaw, A. C., A formal picture description scheme as a basis for picture processing systems, Information and Control, 14, 1969, pp. 9-52.

[1.34] Torii, K., Arisawa, M., Phrase structure languages by a concurrent derivation, Trans. of Inst. Elect. Commun. Engineers of Japan, 54-C, 2, 1971, pp. 124-131 (in Japanese).

[1.35] Torii, K., Sugito, Y., Mano, Y., GMS/I: An interactive graph manipulation system, 1st USA-Japan Computer Conference, Proc. 1972, p.593.

[1.36] Torii, K., Sugito, Y., Mano, Y., GMS/II-A graph manipulation system for solving problems interactively, Papers of Technical Groups on Automata and Languages, IECE of Japan, AL-74-33, 1974, (in Japanese).

[1.37] Underwood, W. E., Kanal, L. N., Structural descriptions, transformational rules, and pattern analysis, 1st Int. Conf. on Pattern Recognition, 1973.

[1.38] Yamamoto, N., Abe, N., Toyoda, J., Tanaka, K., On the data structure manipulating system with web grammars, Information Processing of Japan (in press), (in Japanese).

****

[2.1] Abraham, A., Compound and serial grammars, Information and Control, 20, 1972, pp. 432-438.

[2.2] Ezawa, Y., Yamauchi, H., Mizumoto, M., Toyoda, J., Tanaka, K., On interactive languages, Trans. on Inst. Elect. Commun. Engrs. of Japan, 56-D, 11, 1973, pp. 670-671 (in Japanese).

[2.3] Ezawa, Y., Mizumoto, M., Toyoda, J., Tanaka, K., Interactive languages, Journal of Comp. System Sciences, (in press).

[2.4] Ezawa, Y., Mizumoto, M., Toyoda, J., Tanaka, K., Non-deterministic interactive languages, 1974 National Convention Records of the Inst. Elect. Commun. Engineers of Japan, Summer 1974, No. 1478, (in Japanese).

[2.5] Ezawa, Y., Mizumoto, M., Toyoda, J., Tanaka, K., On interactive generating systems and their languages, Trans. on Inst. Elect. Commun. Engineers of Japan, submitted, (in Japanese).

[2.6] Ezawa, Y., Mizumoto, M., Toyoda, J., Tanaka, K., Attempt of interactive web languages, 1975 National Convention Records of the Inst. Elect. Commun. Engineers of Japan, Spring 1975, (in Japanese).

[2.7] Gabrielian, A., The theory of interactive local automata, Information and Control, 16, 1970, pp. 360-377.

[2.8] Ginsburg, S., and Spanier, E. H., Control sets on grammars, Mathematical Systems Theory, 2, 1968, pp. 159-177.

[2.9] Hanakata, K., A methodalogy for interactive systems, in Learning Systems and Intelligent Robots, Ed. by Fu, K. S. and Tou, J. T., Plenum Press, New York, 1974, pp. 317-324.

[2.10] Hopcroft, J. E. and Ullman, J. D., Formal languages and their relation to automata, Addison-Wesley, 1969.

[2.11] Khabbaz, N. A., Control sets on linear grammars, Information and Control, 25, 1974, pp. 206-221.

[2.12] Kupka, I., and Wilsing, N., Functions describing interactive programming, International Computing Symposium 1973, pp. 41-45.

[2.13] Lindenmayer, A., Mathematical models for cellular interaction in development, I-II, Journal of Theoretical Biology, 18, 1968, pp. 280-315.

[2.14] Mayoh, B. H., Multidimensional Lindenmayer organism, Lecture Notes in Computer Science, 15, Springer-Verlag, 1974, pp. 302-326.

[2.15] Moriya, E., Associate languages and derivational complexity of formal grammars and languages, Information and Control, 22, 1973, pp. 139-162.

[2.16] Pratt, T. W., Pair grammars, graph languages and string-to-graph translations, J. Comp. System Sciences, 5, 1971, pp. 560-595.

[2.17] Rozenberg, G and Doucet, P. G., On OL-languages, Information and Control, 19, 1971, pp. 302-318.

[2.18] Salomaa, A., Formal languages, Academic Press, New York, 1973.

[2.19] Salomaa, A., On sentential forms of context-free grammars, Acta Informatica, 2, 1973, pp. 40-49.

[2.20] Salomaa, A., and Rozenberg, G., L Systems, Lecture Notes in Computer Science, 15, Springer-Verlag, 1974.

[2.21] Torii, K., Sugito, Y. and Mano, Y., GMS/II- A graph manipulation system for solving problems interactively, Papers of Technical Groups on Automata and Languages of IECE of Japan, AL74-33, 1974 (in Japanese).

****